КОМПЛЕКС ВЫЧИСЛИТЕЛЬНЫЙ СМ I700

Заводской № 0212 Год выпуска 1989

СМДО СМ I700

Подсистема процессора СМ 2700.2400

Микропрограммные тесты контроллера ОЗУ

Текст программы

00076-0I I2 04

Книга I2

ерждeн

00076-01 12 04-ЛУ

СИСТЕМА МИКРОДИАГНОСТИЧЕСКОГО ОБЕСПЕЧЕНИЯ
ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА СМ1700

СМДО СМ 1700

ПОДСИСТЕМА ПРОЦЕССОРА СМ2700.2400
Микропрограммные тесты контроллера ОЗУ

Текст программы

00076-01 12 04

Листов 387

1987

Инв. № подл. | Подпись и дата | Взам. инв. № | Инв. № дубл. | Подпись и дата

Литера

АННОТАЦИЯ

Настоящий документ содержит тексты программной секции ENKCC, входящей в систему микродиагностического обеспечения вычислительного комплекса СМ 1700 - СМДО СМ1700 и реализующей микропрограммные тесты контроллера ОЗУ.

Текст программы представлен в загрузочном формате и в форме листинга трансляции, содержащего исходные тексты микропрограмм на языке МИАСС .

Магнитная лента с текстом программы предназначена для создания рабочих кодов микродиагностики на носителе устройства ввода консоли и получения твердой копии документа с исходными текстами, используемого при локализации неисправности, обнаруженной микропрограммными тестами во время выполнения.

Подробные сведения о структуре микропрограммных тестов содержатся в документе 00076-01 13 01. СИСТЕМА МИКРОДИАГНОСТИЧЕСКОГО ОБЕСПЕЧЕНИЯ ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА СМДО СМ1700. Описание программы.

Загрузочный модуль на магнитной ленте имеет метку ENKCC, листинг трансляции - ENKCC.MCR.

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ

```
;4       .PAGE "ФОРМАТЫ МИКРОИНСТРУКЦИЙ"
;5       .HEXADECIMAL
;6       .LIST
;7       .NOCREF
;8       ;
;9       ;
;10      ;          Следующая таблица изображает 24-битовое микрослово
;11      ;        процессора и соответствует формату слова в управляющей
;12      ;        памяти (WCS). Бит 23 (не показан) представляет собой
;13      ;        бит контроля по паритету (нечет).
;14      ;
;15      ;
;16      ;                  22 21       16 15          9 8 7 6 5 4       0
;17      ;                  +-----+-------+-------+-------+-------+-------+
;18      ;1.BASIC           :1:     DP    :     D ADRS    : B :CC :  SCTL   :
;19      ;  (базовый)       +-+-----------+---------------+---+---+-------+
;20      ;
;21      ;                  22  20 19 17 16            9 8 7 6 5 4       0
;22      ;                  +-----+-------+-------+-------+-------+-------+
;23      ;2.MOVE            :0 1 1: MDP :     XD ADRS     : B :CC :  SCTL   :
;24      ;  (пересылки)     +-----+-----+---------------+---+---+-------+
;25      ;
;26      ;                  22  20 19       14 13 11   9 8 7 6 5 4       0
;27      ;                  +-----+-------+-------+-------+-------+-------+
;28      ;3.EXTENDED        :0 1 0:    XDP    : SI  : A : B :CC :  SCTL   :
;29      ;  (расширенный)   +-----+---------+-----+---+---+---+-------+
;30      ;
;31      ;                  22    19 18 16 15          9 8 7 6 5 4       0
;32      ;                  +-----+--+----+-------+-------+-------+-------+
;33      ;4.MEM.REQ         :0 0 1 1| MF1 |     D ADRS    :MF2:DT :  SCTL   :
;34      ;  (запроса        +-------+-----+---------------+---+---+-------+
;35      ;      памяти)
;36      ;                  22    19 18 16 15   12 11  9 8 7 6 5 4       0
;37      ;                  +-----+-------+-------+-------+-------+-------+
;38      ;5.MISC            :0 0 1 0:P:0 0:   M1  :  M2   :0:R:0 0:  SCTL   :
;39      ;  (разных         +-------+-+---+-------+-------+-+-+---+-------+
;40      ;      операций)
;41      ;                  22    19 18 17                        4 3       0
;42      ;                  +-----+--+----+-------+-------+-------+-------+
;43      ;6.JUMP            :0 0 0|1:OS:    JUMP.ADRS                : JCTL  :
;44      ;  (перехода)      +-------+--+-------------------------------+-------+
;45      ;
;46      ;                  22    19 18 17      12 11  9 8 7 6 5 4 3       0
;47      ;                  +-----+-------+-------+-------+-------+-------+
;48      ;7.DECODE          :0 0 0 0: : DEC.ADRS  :IFUNC: : : : : : JCTL  :
;49      ;  (дешифрации     +-------++++-----------------+++++++++-------+
;50      ;    инструкций)          :                     : : : : :
;51      ;                  BACK UP PC          IB REA----+ : : :. +--OPC/SPEC
;52      ;                                                  : : :
;53      ;                            SEL CM HI IB BYTE-----+ : +--LOAD RDEST
;54      ;                                                    :
;55      ;                                       LOAD OS-----+
;56      ;
```

```
;57       .PAGE "ОПРЕДЕЛЕНИЯ ПОЛЕЙ ОСНОВНОГО МИКРОСЛОВА"
;58       ;
;59       ;    Следующие выражения используются при контроле истинности комбинаций полей
;60       ;
;61       .SET/A.VAL=<.OR[<.EQL[<OPC2/>,4]>,<.EQL[<OPC2/>,5]>]>
;62       .SET/B.VAL=<.AND[<.NEQ[<OPC2/>,0]>,<.NEQ[<OPC2/>,1]>,<.NEQ[<OPC2/>,2]>,
;63       <.NEQ[<OPC2/>,3]>]>
;64       .SET/D.VAL=<.AND[<.NEQ[<OPC2/>,0]>,<.NEQ[<OPC2/>,1]>,<.NEQ[<OPC2/>,2]>,
;65       <.NEQ[<OPC2/>,4]>,<.NEQ[<OPC2/>,5]>,<.NEQ[<OPC2/>,6]>,
;66       <.NEQ[<OPC2/>,7]>]>
;67       .SET/XD.VAL=<.EQL[<OPC1/>,<OPC1/MOVE>]>
;68       .SET/CC.VAL=<.AND[<.NEQ[<OPC2/>,0]>,<.NEQ[<OPC2/>,2]>,<.NEQ[<OPC2/>,3]>]>
;69        SET/DT.VAL=<.EQL[<OPC2/>,<OPC2/MEM.REQ>]>
;70       .SET/SCTL.VAL=<.AND[<.NEQ[<OPC2/>,0]>,<.NEQ[<OPC2/>,1]>]>
;71       .SET/JCTL.VAL=<.OR[<.EQL[<OPC2/>,0]>,<.EQL[<OPC2/>,1]>]>
;72       .SET/JUMP.VAL=<.EQL[<OPC2/>,<OPC2/JUMP>]>
;73       .SET/DECODE.VAL=<.EQL[<OPC2/>,<OPC2/DECODE>]>
;74       .SET/MISC.VAL=<.EQL[<OPC2/>,<OPC2/MISC>]>
;75       .SET/DP.VAL=<.EQL[<OPC/>,<OPC/BASIC>]>
;76       .SET/PAGE.PROB=<.EQL[<.AND[<7FF>,<.>]>,<7FE>]>
;77       .SET/SKIP.LEGAL2=<.OR[<.EQL[<SCTL/>,<SCTL/RET.NOERR>]>,<.EQL[<SCTL/>,
;78       <SCTL/POP.USTACK>]>]>
;79       .SET/SKIP.LEGAL=<.OR[<.EQL[<SCTL/>,<SCTL/NO.SKIP>]>,<.EQL[<SCTL/>,
,80       <SCTL/RETURN>]>,<.EQL[<SCTL/>,<SCTL/RETURN+1>]>,<SKIP.LEGAL2>]>
;81        SET/SKIP.PAGE.VAL=<.SELECT[<.EQL[<PAGE.PROB>,0]>,<1>,<.EQL[<PAGE.PROB>,1]>,
;82       <SKIP.LEGAL>]>
;83       .SET/JUMP.CHECK=<.OR[<.EQL[<JCTL/>,<JCTL/JSR>]>,
;84       <.EQL[<JCTL/>,
;85       <JCTL/JSR.VALID>]>]>
;86       .SET/JUMP.PAGE.VAL=<.SELECT[<.EQL[<PAGE.PROB>,0]>,<1>,
;87       <.EQL[<PAGE.PROB>,1]>,<.XOR[1,<JUMP.CHECK>]>]>
;88       ;
;89       ;   Определения кодов операций микроинструкций
;90       ;
;91       OPC/=<22>,.DEFAULT=<OPC/BASIC>
;92               BASIC=1            ; микроинструкция BASIC
;93
;94       OPC1/=<22:20>
;95               EXTENDED=2         ; микроинструкция EXTENDED
;96               MOVE=3             ; микроинструкция MOVE
;97
;98       OPC2/=<22:19>
;99               MEM.REQ=3          ; микроинструкция MEM.REQ
;100              MISC=2             ; микроинструкция MISC
;101              JUMP=1             ; микроинструкция JUMP
;102              DECODE=0           ; микроинструкция DECODE
;103      ;
;104      ;        Определения адресных полей в микроинструкциях:
;105      ;
;106      ;   A.ADRS - обращение к внутренней озу микропроцессора K1804BC1 через порт A
;107      ;   B.ADRS - обращение к внутренней озу микропроцессора K1804BC1 через порт B
;108      ;  XB.ADRS        "           "             "          "          "
;109      ;
;110      A.ADRS/=<10:9>,.VALIDITY=<A.VAL>
;111              MASK=3             ; маска резервной копии регистров
```

```
;112    B.ADRS/=<8:7>,.VALIDITY=<B.VAL>,.DEFAULT=0
;113            MASK=3          ; маска резервной копии регистров
;114    XB.ADRS/=<8:7>,.VALIDITY=<.EQL[<OPC1/>,<OPC1/MOVE>]>
;115            BPC=0           ; начальное значение счетчика команд (PC)(WR[4])
;116            VA=1            ; адрес обращения к памяти (WR[5])
;117            VAR=1           ; адрес последнего обращения к памяти (WR[5])
;118
;119    ;
;120    ;       Поля управления путями данных
;121    ;
;122    ;  Управляющее поле путей данных микроинструкции BASIC
;123    ;
;124    DP/=<21:16>,.VALIDITY=<DP.VAL>,.DEFAULT=<.SHIFT[.AND[<SDP/Q.CMP.LS>,0FF],-2]>
;125    ;
;126    ;  DP.SMALL применяется в микроинструкциях, к которым можно присоединить XCHG
;127    ;
;128    DP.SMALL/=<20:16>,.VALIDITY=<DP.VAL>
;129    XCHG.BIT/=<21>,.DEFAULT=<0>
;130            YES=1           ; взаимная замена исходного значения рабочего регистра
;131                            ; и ячейки LS
;132            NO=0
;133    ;
;134       Управляющее поле путей данных микроинструкции EXTENDED
;135    ;
;136    XDP/=<19:14>,.VALIDITY=<A.VAL>
;137
;138       Управляющее поле путей данных микроинструкции MOVE
;139    ;
;140    MDP/=<19:17>,.VALIDITY=<XD.VAL>
;141    ;
;142    ;  Поле кодов условий
;143    ;
;144    CC/=<6:5>,.VALIDITY=<CC.VAL>,.DEFAULT=<CC/DT(LONG)&HOLD.ALU.CC>
;145            DT(LONG)&HOLD.ALU.CC=0  ; использование контекста длинных слов
;146                                    ; (32 бита) и сохранение кодов условий АЛУ
;147            DT(LONG)&SET.ALU.CC=1   ; использование контекста длинных слов
;148                                    ; (32.бита) и установка кодов условий АЛУ
;149            DT(SIZE)&SET.ALU.CC=2   ; использование контекста согласно регистру
;150                                    ; размера и установка кодов условий АЛУ
;151            DT(SIZE)&MAP.ALU.CC.TO.PSL=3 ; аппаратно-зависимая пересылка кодов
;152                                    ; условий АЛУ в коды условий PSL
;153    ;
;154    ;      Поле типа данных для памяти
;155    ;
;156    MDT/=<6:5>,  VALIDITY=<DT.VAL>
;157                    BYTE=0          ; размер = байт
;158                    WORD=1          ; размер = слово
;159                    SIZE=2          ; размер = содержимое регистра размера
;160                    LONG=3          ; размер = длинное слово
;161    ;
;162    ;      Поле входа сдвигов
;163    ;
;164    ;      Это управляющее поле определяет входы сдвигов для следующих операций.
;165    ;      Направление сдвига определяется кодом приемника К1804ВС1
;166    ;
```

```
;167     SI/=<13:11>,.VALIDITY=<A.VAL>
;168                     ROT.WR=0            ; циклический сдвиг рабочего регистра (WR)
;169                     ROT.WR.Q=1          ; циклический сдвиг WR и Q
;170                     ASH.WR=2            ; арифметический сдвиг WR
;171                     ASH.WR.Q=3          ; арифметический сдвиг WR и Q
;172                     MUL.SHF=4           ; операция сдвига для умножения с учетом знака
;173                     UMUL.SHF=5          ; операция сдвига для беззнакового умножения
;174                     SHF.WR.Q=6          ; логический сдвиг WR и Q
;175     ;
;176     ;       Поле управления пропуском микроинструкций
;177     ;
;178     ;       Это поле действительно для всех микроинструкций, за исключением
;179     ;               микроинструкции JUMP
;180     ;               микроинструкции DECODE
;181     ,
;182     SCTL/=<4:0>,.VALIDITY=<.AND[<SCTL.VAL>,<SKIP.PAGE.VAL>]>,.DEFAULT=<SCTL/NO.SKIP>
;183                     N.CLR=0             ; бит N АЛУ равен нулю
;184                     NEQ=1               ; бит Z АЛУ равен нулю
;185                     BIT.SET=1           ; бит Z АЛУ равен нулю
;186                     V.CLR=2             ; бит V АЛУ равен нулю
;187                     C.SET=3             ; бит C АЛУ равен единице
;188                     GEQU=3              ; бит C АЛУ равен единице
;189                     NOT.PSL.C=4         ; бит C PSL равен нулю
;190                     BR.FALSE=5          ; ветвление в команде условного перехода ложно
;191                     R.DST=6             ; признак регистрового режима адресации
;192                     NO.INTERRUPT=7      ; ожидающих прерываний нет
;193                     N.SET=8             ; бит N АЛУ равен единице
;194                     EQL=9               ; бит Z АЛУ равен единице
;195                     BITS.CLR=9          ; бит Z АЛУ равен единице
;196                     V.SET=0A            ; бит V АЛУ равен единице
;197                     C.CLR=0B            ; бит C АЛУ равен нулю
;198                     LSSU=0B             ; бит C АЛУ равен нулю
;199                     STATE.ZERO.SET=0C   ; бит 0 регистра STATE истинный
;200                     STATE.ONE.SET=0D    ; бит 1 регистра STATE истинный
;201                     STATE.ZERO.CLR=0E   ; бит 0 регистра STATE ложный
;202                     STATE.ONE.CLR=0F    ; бит 1 регистра STATE ложный
;203                     RET.NOERR=10        ; возврат+1, если отсутствует ошибка системы памяти
;204                     JMP.Z.CLR=11        ; переход (по стеку), если бит Z АЛУ равен 0
;205                     POP.USTACK=12       ; сбрасывание верхней записи стека
;206                     JMP.C.SET=13        ; переход (по стеку), если бит C АЛУ равен единице
;207                     RETURN=14           ; возврат к (по микростеку)
;208                     NO.SKIP=15          ; выполнение следующей микроинструкции
;209                     RETURN+1=16         ; возврат (по микростеку)+1
;210                     LOOP.IF.NO.I.AND.SYNC=17 ; переход (по стеку), если отсутствуют
;211                                         ; прерывания и требование синхронизации ускорителя
;212                     CONSOLE.ATTN=18     ; "внимание" (ATTN) консоли
;213                     CONSOLE.ACK=19      ; "подтверждение" (ACK) консоли
;214                     NO.FAST.INTERRUPT=1A; ожидающих быстрых прерываний нет
;215                     BACKUP.MASK.VALID=1B; маска для резервной копии регистров истинная
;216                     LSS=1C              ; меньше, чем (с учетом знака)
;217                     MEM.REF.OK=1D       ; ошибки системы памяти нет
;218                     SKIP=1E             ; выполнение микронструкции по адресу плюс один
;219                     SYNC=1F             ; синхронизация ускорителя
;220     ;
;221     ;       Поле управления микроинструкцией JUMP
```

```
;222    ;
;223    ;       Это поле истинно для следующих инструкций:
;224    ;               микроинструкция JUMP
;225    ;               микроинструкция DECODE
;226    ;
;227    JCTL/=<3:0>,.VALIDITY=<.AND[<JCTL.VAL>,<JUMP.PAGE.VAL>]>,.DEFAULT=<JCTL/JUMP.VALID>
;228                    N.CLR=0             ; бит N АЛУ равен нулю
;229                    NEQ=1               ; бит Z АЛУ равен нулю
;230                    BIT.SET=1           ; бит Z АЛУ равен нулю
;231                    V.CLR=2             ; бит V АЛУ равен нулю
;232                    C.SET=3             ; бит C АЛУ равен единице
;233                    GEQU=3              ; бит C АЛУ равен единице
;234                    JUMP=4              ; безусловный переход
;235                    BR.FALSE=5          ; ветвление в команде условного перехода ложно
;236                    R.DST=6             ; признак регистрового режима
;237                    NO.INTERRUPT=7      ; ожидающих прерываний нет
;238                    N.SET=8             ; бит N АЛУ равен единице
;239                    EQL=9               ; бит Z АЛУ равен единице
;240                    BITS.CLR=9          ; бит Z АЛУ равен единице
;241                    V.SET=0A            ; бит V АЛУ равен единице
;242                    C.CLR=0B            ; бит C АЛУ равен нулю
;243                    LSSU=0B             ; бит C АЛУ равен нулю
;244                    JSR=0C              ; безусловный переход и занесение в микростек
;245                    JSR.VALID=0D        ; переход и занесение в микростек,если
;246                                        ; IB VALID=1
;247                    NO.JUMP.TST=0E      ; нет перехода и пропуск,если IB VALID=0
;248                    JUMP.VALID=0F       ; переход,если IB VALID=1
;249    ;
;250    ; Поле адреса перехода
;251    ;
;252    JUMP.ADRS/=<17:4>,.ADDRESS,.VALIDITY=<JUMP.VAL>
;253    ;
;254    ; Разрешение для регистра OS
;255    ;
;256    OS/=<18>,.VALIDITY=<JUMP.VAL>
;257                    NOP=0
;258                    WITH.OS=1           присоединение регистра OS к адресу перехода
;259    ;
;260    ; Резервная копия счетчика команд (PC)
;261    ;
;262    BPC/=<18>,.VALIDITY=<DECODE.VAL>,.DEFAULT=<.CASE[<.EQL[<OPC2/>,<OPC2/DECODE>]>]OF[0,1]>
;263                    NOP=1
;264                    SAVE.PC=0           ; запись данных АЛУ в рабочий регистр по адресу
;265                                        ; порта B (PC+1)
;266    ;
;267    ; Адресное поле микроинструкции DECODE
;268    ;
;269    DEC.ADRS/=<17:12>,.VALIDITY=<DECODE.VAL>
;270    ;
;271    ; Функция регистра предвыборки команд (PFR)
;272    ;
;273    IFUNC/=<11:9>,.VALIDITY=<DECODE.VAL>
;274                    CM.EXEC=0           ; начальное ветвление на выполнение в режиме совместимости,
;275                                        ; если старший байт=нулю
;276                    SPEC=0              ; начальное ветвление по спецификатору
```

```
;277                    CM.IRD=1          ; начальное ветвление по дешифрации команды в режиме
;278                                      ; совместимости
;279                    FLOAT=1           ; начальное ветвление по спецификатору плавающего типа
;280                    VAX.EXEC=2        ; начальное ветвление на выполнение в собственном режиме
;281                    ASRC=2            ; начальное ветвление по спецификатору адреса
;282                    VAX.IRD=3         ; начальное ветвление по коду операции в собственном режиме
;283                    VSRC=4            ; начальное ветвление по спецификатору адреса для команд
;284                                      ; битовых полей
;285                    ESRC=5            ; начальное ветвление по спецификатору в индексном режиме
;286                    CM.DST=6          ; начальное ветвление по приемнику в режиме совместимости
;287                    CM.SINGLE=7       ; начальное ветвление в режиме совместимости при одном операнде
;288    ;
;289    ; Запрос заполнения регистра предвыборки команд
;290    ;
;291    IB.REQ/=<6>,.VALIDITY=<DECODE.VAL>,.DEFAULT=0
;292                    NOP=0
;293                    IB.REQ=1          ; разрешение аппаратно-зависимого запроса памяти
;294    ;
;295    ; Выбор кода операции режима совместимости
;296    ;
;297    CM.HI/=<7>,.VALIDITY=<DECODE.VAL>,.DEFAULT=0
;298                    LOW.BYTE=0        ; дешифрация битов <7:0> регистра предвыборки инструкций (PFR)
;299                    HI.BYTE=1         ; дешифрация битов <15:6> регистра предвыборки инструкций (PFR)
;300    ;
;301    ; Загрузка регистра OS
;302    ;
;303    LD.OS/=<6>, VALIDITY=<DECODE.VAL>,.DEFAULT=0
;304                    NOP=0
;305                    LOAD.OS=1         ; загрузка регистра OS из регистра предвыборки инструкций
;306    ;
;307    ; Разрешение установки признака приемника типа регистра
;308    ;
;309    R.DST/=<5>,.VALIDITY=<DECODE.VAL>,.DEFAULT=0
;310                    NOP=0
;311                    ENAB=1
;312    ;
;313    ; Бит кода операции/спецификатора
;314    ;
;315    OPC.SPEC/=<4>,.VALIDITY=<DECODE.VAL>
;316                    CM.EXEC=1         ; выход на выполнение в режиме совместимости
;317                    SPEC=0            ; выход по спецификатору
;318                    CM.IRD=1          ; выход по дешифрации команд в режиме
;319                                      ; совместимости
;320                    FLOAT=0           ; выход по спецификатору плавающего типа
;321                    VAX.EXEC=1        ; выход на выполнение в собственном режиме
;322                    ASRC=0            ; выход по адресу спецификатора
;323                    VAX.IRD=1         ; выход по коду операций в собственном режиме
;324                    VSRC=0            ; выход по спецификатору адреса для команд
;325                                      ; битовых полей
;326                    ESRC=0            ; выход по спецификатору в индексном режиме
;327                    CM.DST=0          ; выход по приемнику в режиме совместимости
;328                    CM.SINGLE=0       ; выход в режиме совместимости при одном операнде
;329    ;
;330    ; Разные функции
;331
```

```
;332    ; Это поле представляет собой признак для возбуждения интерпретации микрослова
;333    ; в качестве инструкции разного назначения или в качестве инструкции порта
;334    ;
;335    MISC.PORT/=<18>
;336                    MISC=0
;337                    PORT=1
;338    ;
;339    MISC.0/=<17>,.VALIDITY=<MISC.VAL>
;340                    NOP=0               ; операции в АЛУ и с кодами условий отсутствуют
;341    ;
;342    MISC.1/=<15:12>;.VALIDITY=<MISC.VAL>
;343                    NOP=0F              ; отсутствие операции
;344                    SET.CP.ATTN=0E      ; установка для консоли сигнала ATTN (внимание)
;345                                        ; из DAP
;346                    SET.CP.ACK=0D       ; установка для консоли сигнала ACK (подтверждение)
;347                                        ; из DAP
;348                    CLR.CP.ATTN.AND.ACK=0C  ; очистка сигналов DAP ATTN и ACK
;349                    SET.HIGH.PAGE=08 ; установка бита для доступа к старшей половине
;350                                        ; управляющей памяти (WCS)
;351                    CLR.HIGH.PAGE=0A ; очистка бита для доступа к младшей половине
;352                                        ; WCS
;353                    SET.PORT.I.O=9      ; установка режима ввода/вывода порта
;354                    SET.ACC.I.O=8       ; установка режима ввода/вывода ускорителя
;355                    SET.BACKUP.MASK.VALID=7  ; установка признака маски для резервной
;356                                        ; копии регистров
;357                    SET.S1=6            ; возбуждение бита 1 регистра STATE
;358                    SET.S0=5            ; возбуждение бита 0 регистра STATE
;359                    CLR.S1=4            ; очистка бита 1 регистра STATE
;360                    CLR.S0=3            ; очистка бита 0 регистра STATE
;361                    SET.S1&CLR.S0=2     ; возбуждение бита 1 регистра STATE и очистка
;362                                        ; бита 0 регистра STATE
;363                    CLR.S1&SET.S0=1     ; очистка бита 1 регистра STATE и возбуждение
;364                                        ; бита 0 регистра STATE
;365                    CLR=0               ; очистка битов регистра STATE 0 и 1
;366    ;
;367    MISC.2/=<11:9>,.VALIDITY=<MISC.VAL>
;368                    NOP=0               ; отсутствие операции
;369                    MASK.INTERRUPTS=1; маскирование всех прерываний (для дешифрации
;370                                        ; в следующем цикле)
;371                    ASSERT.DA.AND.PASS.WR=2; выдача сигнала наличия данных и пересылка
;372                                        ; WR[0] ускорителю или порту
;373                    TRAP.ACC=3          ; прерывание ускорителя
;374                    READ.ACC.UPC=4      ; чтение счетчика микрокоманд ускорителя
;375                    TRANSFER.GRANT=5 ; опознавание запроса порта
;376                    MASK.HALT.AND.T.BIT=6; маскирование прерываний по останову (HALT)
;377                                        ; и Т-биту (для дешифрации двумя циклами позже)
;378                    WRITE.WR.TO.CONSOLE.REGISTER=7 ; пересылка битов <7:0> для
;379                                                   ; микропроцессора консоли
;380    ;
;381    MISC.WR/=<8:7>
;382    MISC.WRL/=<8>
;383    MISC.WRR/=<7>
;384    ;
;385    ;  Выборка устройства порта
;386    ;
```

```
;387    PORT.SELECT/=<17:15>
;388                IDC=7               ; адресуется контроллер дисков IDC
;389    ;
;390    ;  Функция порта
;391    ;
;392    PORT.FUNC/=<14:13>
;393                READ=0              ; чтение
;394                WRITE=1             ; запись
;395                CONTROL=2           ; управление
;396    ;
;397    ;  Команда для обмена данными между IDC и DAP
;398    ;
;399    R.W.FUNC/=<12:10>
;400                CSR=0               ; чтение/загрузка управляющего слова IDC
;401                DAR=1               ; чтение/загрузка адреса данных на диске или
;402                                    ; загрузка команды
;403                DATA.BYTE=2         ; чтение/загрузка одного байта данных
;404                DATA.WORD=3         ; чтение/загрузка одного слова данных
 405                PATTERN=4           ; чтение в DAP информации об ошибке (данные)
 406                POSITION=5          ; чтение в DAP информации об ошибке (адрес)
 407    ;
 408    ;  Функции управления
 409
 410    CNTL.FUNC/=<12:10>
 411                CLEAR.FIFO.CNTR=0 ; очистка счетчика управления буферами данных
 412                RESET.BR=1          ; очистка запроса прерывания BR5
 413                CLEAR.IDC=3         ; начальная установка IDC
 414                SET.AUTO=4          ; установка режима автоматической передачи слов
 415                                    ; данных в DAP
 416                CLEAR.AUTO=5          снятие режима автоматической передачи
 417                SEL.FIFO.A=6          выборка буфера A
 418                SEL.FIFO.B=7          выборка буфера B
 419
;420       Поле запроса памяти
 421
 422       Это поле используется для указания нужной операции для управления памяти.
;423       Поле является фиктивным. В действительности оно образовано из двух полей
;424       M.FUNC1 и M.FUNC2
 425
 426       Особенности некоторых функций памяти:
;427
;428    ;  ROTATE.BYTE.RIGHT - выполняет девятибитовый циклический сдвиг, поэтому ответ
;429    ;                      должен корректироваться действием ROL WRD ]
 430    ;
 431       WORD.SWAP         - выполняет 15-битовый циклический сдвиг, поэтому ответ должен
 432                           корректироваться действием ROL WRD ]
 433
;434    ;  OCTA.WRITE.P      - адрес должен быть выравнен по длинному слову
;435
 436       OCTA.WR.V.WCHK    - адрес должен быть выравнен по длинному слову. Эти данные могут
;437                           пересекать границы страниц
 438
;439    MEM.FUNC/=<7>,.VALIDITY=0
;440            PREFETCH=0              ; запрос потока команд. вызывает увеличение (+1)
;441                                    ; виртуального адреса перед обращением и заполнение
```

```
;442                                    ; буфера команд
;443             WRITE.TB=1             ; запись в буфер трансляции
;444             TEST.V.RCHK=2          ; проверка защиты по чтению. Возвращает на шину
;445                                    ; МС содержимое буфера трансляции
;446             READ.P=3               ; чтение по физическому адресу
;447             WRITE.P=4              ; запись по физическому адресу
;448             TEST.V.WCHK=5          ; проверка защиты по записи. Возвращает на шину
;449                                    ; МС содержимое буфера трансляции
;450             ROTATE.BYTE.RIGHT=6    ; циклический сдвиг адресных данных на 9 битов
;451                                    ; вправо и возвращение их на шину МС
;452             WORD.SWAP=7            ; циклический сдвиг адресных данных на 15 битов
;453                                    ; влево и возвращение их на шину МС
;454             READ.V.NOCHK=8         ; чтение по виртуальному адресу без проверки
;455                                    ; защиты
;456             READ.V.WCHK=9          ; чтение по виртуальному адресу с проверкой защиты
;457                                    ; по записи
;458             READ.V.RCHK=0A         ; чтение по виртуальному адресу с проверкой защиты
;459                                    ; по чтению
;460             READ.MAINT.VAR.INC=0B  ; проверка инкрементирования виртуального адреса
;461             WRITE.V.NOCHK=0C       ; запись по виртуальному адресу без проверки
;462                                    ; защиты
;463             WRITE.V.WCHK=0D        ; запись по виртуальному адресу с проверкой защиты
;464                                    ; по записи
;465             IB.FILL=0E             ; чтение по виртуальному адресу с проверкой защиты
;466                                    ; по чтению и заполнение регистра предвыборки
;467                                    ; инструкций
;468             READ.V.RCHK.IFILL=0E   ; чтение по виртуальному адресу с проверкой защиты
;469                                    ; по чтению и заполнение регистра предвыборки
;470                                    ; инструкций
;471             WRITE.UBS.MAP=0F       ; запись в буфер трансляции общей шины
;472             OCTA.WRITE.P=10        ; запись восьмикратных слов данных по физическому
;473                                    ; адресу
;474             WRITE.TB.STEP=11       ; запись в две последовательные ячейки буфера
;475                                    ; трансляции
;476             READ.UBS.MAP=12        ; чтение из буфера трансляции общей шины
;477             READ.TB=13             ; чтение из буфера трансляции
;478             READ.MAINT.ADRS=14     ; выдача виртуального адреса и чтение его обратно
;479                                    ; в качестве данных (режим отладки)
;480             MAINT.ECC.DATA=15      ; проверка всех функций коррекции ошибки (ECC)
;481             READ.CSR=16            ; чтение регистра управления
;482             READ.CNTRL.REG=16      ; чтение регистра управления
;483             WRITE.CNTRL.REG=17     ; запись в регистр управления
;484             WRITE.CSR=17           ; запись в регистр управления
;485             OCTA.READ.P=18         ; чтение восьмикратных слов данных по физическому
;486                                    ; адресу
;487             READ.V.WCHK.LOCKED=19  ; чтение по виртуальному адресу и проверка защиты
;488                                    ; по записи с блокировкой
;489             OCTA.READ.V.RCHK=1A    ; чтение восьмикратных слов данных по виртуальному
;490                                    ; адресу с проверкой защиты по чтению
;491             READ.MAINT.BR.CHECK=1B; проверка функций ветвления
;492             ISSUE.BG=1C            ; выдача подтверждения для шины
;493             OCTA.WR.V.WCHK=1D      ; запись восьмикратных слов данных по
;494                                    ; виртуальному адресу с проверкой защиты по записи
;495             QUAD.READ.V.RCHK=1E    ; чтение четырехкратных слов данных по
;496                                    ; виртуальному адресу с проверкой защиты по чтению
```

```
;497            READ.MAINT.UBS=1F       ; выдача виртуального адреса на общую шину
;498                                    ; и чтение его в качестве данных (режим отладки)
;499    ;
;500    M.FUNC2/=<18:16>,.VALIDITY=<DT.VAL>
;501    M.FUNC1/=<8:7>,.VALIDITY=<DT.VAL>
;502    ;
;503    PAR/=<23>,.DEFAULT=<.PARITY[<OPC2/>,<OS/>,<JUMP.ADRS/>,<JCTL/>3>
;504    ;
```

```
;505    .PAGE "ОПРЕДЕЛЕНИЯ АДРЕСНЫХ ПОЛЕЙ МЕСТНОЙ ПАМЯТИ"
;506    ;
;507    ;  Это поле задает обращение к младшим 128 ячейкам LS
;508    ;
;509    D.ADRS/=<15:9>,.VALIDITY=<D.VAL>,.DEFAULT=0
;510    ;
;511            SAV.WR0=1                       ; память для WR0
;512            SAV.WR1=2                       ; память для WR1
;513            SAV.WR2=3                       ; память для WR2
;514            SAV.WR3=4                       ; память для WR3
;515            T5.SUB=5                        ; резервировано для общих подпрограмм
;516            T6.SUB=6                        ; резервировано для общих подпрограмм
;517            T7.SUB=7                        ; резервировано для общих подпрограмм
;518            TRANSFER.POINTER=7              ; указатель для программы переноса данных
;519            MEMORY.SIZE=7                   ; запоминание размера памяти в блоках по 256K байт
;520                                            ; после "прощупывания"
;521            FPA.FOUND=7                     ; место запоминания результата проверки
;522                                            ; наличия ускорителя плавающей запятой (FPA)
;523            UBF.FOUND=7                     ; место запоминания результата проверки
;524                                            ; наличия тестера шины (UBE)
;525            T8=8                            ; ячейка для временного хранения 8
;526            T9=9                            ; ячейка для временного хранения 9
;527            T10=0A                          ; ячейка для временного хранения 10
;528            T11=0B                          ; ячейка для временного хранения 11
;529            T12=0C                          ; ячейка для временного хранения 12
;530            T13=0D                          ; ячейка для временного хранения 13
;531            T14=0E                          ; ячейка для временного хранения 14
;532            T15=0F                          ; ячейка для временного хранения 15
;533            PC=10                           ; счетчик инструкций макроуровня
;534            WR.BACKUP=11                    ; резервная копия WR для MOV MEM.DATA TO WRI 3
;535            MM.LASTADDR+1=12                ; место запоминания последнего адреса основной
;536                                            ; памяти плюс 1 (первый несуществующий адрес памяти)
;537            #FF=13                          ; маска
;538            #FFFF=14                        ; маска
;539            #FF000000=15                    ; маска
;540            #FFFFFF00=16                    ; маска
;541            CSR0.MASK=16                    ; маска
;542            CSR1.MASK=17                    ; маска (01003FFF)
;543            CSR2.MASK=18                    ; маска (7FFE3FFF)
;544            #FE7FFFFF=19                    ; маска
;545            UBS.SET=1A                      ; константа (7FF80000),используемая тестом адреса
;546                                            ; общей шины
;547            UBS.DATA=1B                     ; маска (7FFF8000),используемая в качестве маски
;548                                            ; ошибок для теста данных общей шины
;549            ERR.CON.NA=1E                     онстанта 800500 (16-ричная) с установленными
;550                                            ;  тами 23,10,8 для загрузки регистра управления и
;551                                            ; состояния в начальных тестах
;552            ERR.CON=1F                      ; константа 500 (16-ричная). Биты 10 и 8
;553                                            ;  тановлены для загрузки регистра управления
;554                                            ; и состояния в начальных тестах
;555    ;
;556    ;  Следующие ячейки используются, главным образом, в качестве тестовых наборов
;557    ;  данных (1,.перемещаемая в поле нулей)
;558    ;
;559            #1=20                           ; набор 00000001(H)
```

```
;560            #2=21                   ; набор 00000002
;561            #4=22                   ; набор 00000004
;562            #8=23                   ; набор 00000008
;563            #10=24                  ; набор 00000010
;564            #20=25                  ; набор 00000020
;565            #40=26                  ; набор 00000040
;566            #80=27                  ; набор 00000080
;567            #100=28                 ; набор 00000100
;568            #200=29                 ; набор 00000200
;569            #400=2A                 ; набор 00000400
;570            #800=2B                 ; набор 00000800
;571            #1000=2C                ; набор 00001000
;572            #2000=2D                ; набор 00002000
;573            #4000=2E                ; набор 00004000
;574            #8000=2F                ; набор 00008000
;575            #10000=30               ; набор 00010000
;576            #20000=31               ; набор 00020000
;577            #40000=32               ; набор 00040000
;578            #80000=33               ; набор 00080000
;579            #100000=34              ; набор 00100000
;580            #200000=35              ; набор 00200000
;581            #400000=36              ; набор 00400000
;582            #800000=37              ; набор 00800000
;583            #1000000=38             ; набор 01000000
;584            #2000000=39             ; набор 02000000
;585            #4000000=3A             ; набор 04000000
;586            #8000000=3B             ; набор 08000000
;587            #10000000=3C            ; набор 10000000
;588            #20000000=3D            ; набор 20000000
;589            #40000000=3E            ; набор 40000000
;590            #80000000=3F            ; набор 80000000
;591            BIT0=20                 ; набор 00000001
;592            BIT1=21                 ; набор 00000002
;593            BIT2=22                 ; набор 00000004
;594            BIT3=23                 ; набор 00000008
;595            BIT4=24                 ; набор 00000010
;596            BIT5=25                 ; набор 00000020
;597            BIT6=26                 ; набор 00000040
;598            BIT7=27                 ; набор 00000080
;599            BIT8=28                 ; набор 00000100
;600            BIT9=29                 ; набор 00000200
;601            BIT10=2A                ; набор 00000400
;602            BIT11=2B                ; набор 00000800
;603            BIT12=2C                ; набор 00001000
;604            BIT13=2D                ; набор 00002000
;605            BIT14=2E                ; набор 00004000
;606            BIT15=2F                ; набор 00008000
;607            BIT16=30                ; набор 00010000
;608            BIT17=31                ; набор 00020000
;609            BIT18=32                ; набор 00040000
;610            BIT19=33                ; набор 00080000
;611            BIT20=34                ; набор 00100000
;612            BIT21=35                ; набор 00200000
;613            BIT22=36                ; набор 00400000
;614            BIT23=37                ; набор 00800000
```

```
;615             BIT24=38                        ; набор 01000000
;616             BIT25=39                        ; набор 02000000
;617             BIT26=3A                        ; набор 04000000
;618             BIT27=3B                        ; набор 08000000
;619             BIT28=3C                        ; набор 10000000
;620             BIT29=3D                        ; набор 20000000
;621             BIT30=3E                        ; набор 40000000
;622             BIT31=3F                        ; набор 80000000
;623     ;
;624     ;  Мнемоника адресов CSR контроллера памяти
;625     ;
;626             CSR0=4E                 ; для доступа к CSR0 все биты очищены
;627             CSR1=22                 ; для доступа к CSR1 установлен бит 2
;628             CSR2=23                 ; для доступа к CSR2 установлен бит 3
;629     ;
;630     ;  Биты слова управления и состояния (информация для микромонитора во время
;631     ;  выполнения тестов). Слово управления и состояния доступно микромонитору
;632     ;  по адресу LS 40
;633     ;
;634             PRINT.UBE=26            ; печать,имеется или нет тестер шины UBE (бит 6)
;635                                     ; (индикатор в LS7)
;636             PRINT.R80=27            ; печать,имеется или нет R80 и на каком
;637                                     ; устройстве (бит 7).Индикатор в LS7. Номер
;638                                     ; накопителя в LS6
;639             ERROR=28                ; индикация ошибки теста (бит 8)
;640             SET.PA.ERR=29           ; указывает консольному процессору,что следует
;641                                     ; генерировать ошибку паритета по адресу
;642                                     ; управляющей памяти WCS,указанному в LS7 (бит 9)
;643             CONSOLE.LOE=2A          ; указывает,что консольный процессор выполняет
;644                                     ; зацикливание при ошибке (бит 10) (для начальных
;645                                     ; тестов)
;646             PRINT.MEM.SIZE=2B       ; печать размера памяти в блоках по 256 K
;647                                     ; (бит 11) (размер в LS7)
;648             PRINT.FPA=2C            ; печать, имеется или нет ускоритель плавающей
;649                                     ; запятой FPA (бит 12) (индикатор в LS7)
;650             XFER.DATA=2D            ; указывает перенос данных в местную (LS) или
;651                                     ; основную (MM) память (бит 13)
;652             EOS=2E                  ; конец сегмента (бит 14)
;653             BEGIN.TEST=2F           ; начало теста (бит 15)
;654             INTERUPT.EN=30          ; разрешение прерывания или сигнала, заданного
;655                                     ; битами от 17 до 22 и от 28 до 30 (бит 16)
;656             CONSOLE.HALT=31         ; прерывание по останову от консоли (бит 17)
;657             PWR.FAIL=32             ; прерывание по аварии питания (бит 18)
;658             INTERVAL.TIM=33         ; прерывание от интервального таймера (бит 19)
;659             CONSOLE.ATTN=34         ; прерывание по биту "внимание" (ATTN) консоли
;660                                     ; (бит 20)
;661             CONSOLE.ACK=35          ; прерывание по биту "подтверждение" (ACK) консоли
;662                                     ; (бит 21)
;663             DISABLE.MEM.REF=36      ; запрет обращений к памяти (бит 22)
;664             CINIT=3C                ; сигнал общей шины INIT для контроллера памяти
;665                                     ; (бит 28)
;666             UBS.DCLO=3D             ; индикация общей шины DC LO для контроллера
;667                                     ; памяти (бит 29)
;668             UBS.BBSY=3E             ; индикация общей шины BUS BUSY для контроллера
;669                                     ; памяти (бит 30)
```

```
;670             NA.EXP.REC=37           ; печать N/A под EXP и REC (бит 23)
;671             OTHER.DATA=38           ; печать значений под OTHER (бит 24)
;672             CONSOLE.LOOP=39         ; консоль выполняет зацикливание в соответствии со
;673                                     ; счетчиком циклов в LS (бит 25)
;674             PRINT.CRD=3A            ; печать числа ошибок в одиночных битах (бит 26)
;675             CLR.PA.ERR=3B           ; указывает консоли, что следует очистить ошибку
;676                                     ; паритета в управляющей памяти WCS по адресу,
;677                                     ; указанному в LS7 (бит 27)
;678             PRINT.IDC=3F            ; печать,имеется или нет встроенный контроллер
;679                                     ; дисков IDC (бит 31)(индикатор в LS7)
;680     ;
;681     ;  Коды модулей
;682     ;
;683             WCS=20                  ; код модуля для платы управляющей памяти WCS
;684                                     ; (установлен бит 0)
;685             CPU=21                  ; код модуля для платы путей данных DAP
;686                                     ; (установлен бит 1)
;687             MCT=22                  ; код модуля для платы контроллера памяти MCT
;688                                     ; (установлен бит 2)
;689             FPA=23                  ; код модуля для платы ускорителя плавающей
;690                                     ; запятой FPA (установлен бит 3)
;691             ARRAY=24                ; код модуля для платы ОЗУ (установлен бит 4)
;692             IDC=25                  ; код модуля для платы встроенного контроллера
;693                                     ; дисков IDC (установлен бит 5)
;694     ;
;695     ;  Биты ошибок CSR1 контроллера памяти
;696     ;
;697             VALID.ERR=2E            ; не установлен бит действительности данных (VALID),
;698                                     ; если 1 (бит 14)
;699             TB.PAR.ERR=2F           ; ошибка паритета буфера трансляции (бит 15)
;700             NXM=30                  ; ошибка - несуществующая память (бит 16)
;701             UB.BSY=31               ; общая шина занята (бит 17)
;702             ADP.REG=32              ; выбор регистра адаптера общей шины (бит 18)
;703             WR.ACROSS.PG=33         ; ошибка записи за пределами страницы (бит 19)
;704             OP.ERR=34               ; ошибка операции (бит 20)
;705             TB.MISS=35              ; промах в буфере трансляции (транслированный
;706                                     ; виртуальный адрес в буфере отсутствует)
;707                                     ; (бит 21)
;708             ACCESS.REFUSED=36       ; ошибка защиты при обращении (бит 22)
;709             MODIFY.REFUSED=37       ; ошибка защиты при записи (бит 23)
;710             CRD=3E                  ; исправлена ошибка в одиночном бите (бит 30)
;711             RDS=3F                  ; неисправимая ошибка данных (бит 31)
;712     ;
;713     ;  Управляющие биты CSR1 контроллера памяти
;714     ;
;715             ECC.DIS=39              ; бит запрета коррекции (ECC) в CSR1 (бит 25)
;716             DIAG.CHK=3A             ; бит диагностического контроля CSR1 (бит 26)
;717             MME=3B                  ; бит разрешения диспетчера памяти в CSR1
;718                                     ; (бит 27)
;719             INH.CRD=3C              ; бит запрета сообщения о корректируемых ошибках
;720                                     ; данных (CRD) (бит 28)
;721             TB.PAR.DIAG=3D          ; бит диагностирования паритета буфера трансляции
;722                                     ; (принудительная ошибка паритета TB) (бит 29)
;723     ;
;724     ;  Биты регистров управления тестера шины (UBE)
```

```
;725    ;
;726    ;       Регистр управления 1
;727    ;
;728            GO=20                   ; запуск тестера шины (UBE) (бит 0)
;729            BR4=21                  ; выдача запроса шины на уровне 4 (бит 1)
;730            BR5=22                  ; выдача запроса шины на уровне 5 (бит 2)
;731            BR6=23                  ; выдача запроса шины на уровне 6 (бит 3)
;732            BR7=24                  ; выдача запроса шины на уровне 7 (бит 4)
;733            NPR=25                  ; выдача запроса прямого доступа (NPR) (бит 5)
;734            INT.ODNE=26             ; прерывание по завершению (при переполнении
;735                                    ; счетчика циклов) (бит 6)
;736            RDY=27                  ; бит "готово", устанавливаемый, если есть
;737                                    ; готовность начать функционирование (бит 7)
;738            C00=28                  ; бит C0 операции шины (бит 8)
;739            C01=29                  ; бит C1 операции шины (бит 9)
;740            FUNA=2A                 ; бит функции A тестера шины (бит 10)
;741            FUNB=2B                 ; бит функции B тестера шины (бит 11)
;742            CC+DAT=2C               ; данные из счетчика циклов, если установлен,
;743                                    ; и из регистра данных, если очищен
;744            INH.DATIP.ROL=2D        ; запрет циклического сдвига при чтении с паузой
;745                                    ; (DATIP) (бит 13)
;746            DATOB.ON.DATIP=2E       ; выполнение записи байта (DATOB) после цикла
;747                                    ; чтения с паузой (DATIP) (бит 14)
;748            ERR=2F                  ; ошибка общей шины или тестера (бит 15)
;749    ;
;750    ;   Регистр управления 2
;751    ;
;752            EXT.MEM0=20             ; бит 0 расширения памяти (бит 0)
;753            EXT.MEM1=21             ; бит 1 расширения памяти (бит 1)
;754            INH.INC.ADDR&CC=22      ; запрет наращивания счетчика циклов и регистра
;755                                    ; адреса (бит 2)
;756            INH.SACK=23             ; запрет выдачи BUS SACK по BUS GRANT (бит 3)
;757            PWR.DWN.SEQ=24          ; установка ACLO (бит 4)
;758            WNG.GNT.BCK=25          ; не получен BUS GRANT в ответ на запрос
;759                                    ; высшего приоритета (бит 5)
;760            MAX.LATE.ERR=26         ; превышено время задержки для NPR и BR (бит 6)
;761            NO.SACK.ERR=27          ; центральный процессор не зафиксировал таймаута
;762                                    ; при отсутствии SACK (бит 7)
;763            NO.SSYN.ERR=28          ; после выдачи MSYN не получен SSYN (бит 8)
;764            WRG.A.LINES=29          ; ошибка адреса в переданном или принятом
;765                                    ; адресе (бит 9)
;766            NO.GNT+NOT.ONE.GNT=2A   ; отсутствует GRANT или более одного сигнала
;767                                    ; GRANT после запроса (бит 10)
;768            INTR.SSYN.ERR=2B        ; не получен SSYN после прерывания шины (бит 11)
;769            ENB.PB=2C               ; разрешение для бита паритета B (бит 12)
;770            CCOVF=2D                ; переполнение счетчика циклов (бит 13)
;771            TM.DLY=2E               ; запрос для шины каждые 10 мкс (бит 14)
;772    ;
;773    ;   Мнемоника BG6
;774    ;
;775            BG6=23                  ; для адреса BG6 установлен бит 3
;776    ;
;777    ;   Информация об ошибках и управляющая информация для микромонитора
;778    ;
;779            CONTROL.STATUS=40       ; слово управления и состояния
```

```
;780            ERROR.NUMBER=41         ; номер ошибки в текущем тексте
;781            EXPECTED.DATA=42        ; ожидаемый результат текущего теста
;782            RECEIVED.DATA=43        ; действительный результат текущего теста
;783            ADDRESS.DATA=44         ; другие актуальные данные
;784            ERROR.MASK=45           ; биты,подлежащие проверке в результате
;785            MODULE.NUM=46           ; место хранения номера модуля (кодированного)
;786            LOOP.CNT=47             ; место запоминания счета циклов для управления
;787                                    ; зацикливанием из консольного процессора
;788            ERROR.CONTROL=48        ; место запоминания информации управления
;789                                    ; ошибками (зацикливание по ошибке и т.д.)
;790            LOE=20                  ; если установлен,зацикливание на ошибке (бит 0)
;791            NER=21                  ; если установлен,не выдаются сообщения об
;792                                    ; ошибках (бит 1)
;793            BELL=22                 ; если установлен,звуковой сигнал при ошибке
;794                                    ; (бит 2)
;795            HOE=23                  ; если установлен, останов при ошибке (бит 3)
;796            SER=24                  ; если установлен, разрешение сообщений об
;797                                    ; исправимых ошибках данных
;798            APT.PRESENT=25          ; если установлен, имеется APT (бит 5)
;799            LOOP.COMMAND=26         ; если установлен, зацикливание напечатанной
;800                                    ; команды (бит 6)
;801            SPECIAL=27              ; специальный бит для зацикливания в тестах
;802                                    ; "прощупывания" (бит 7)
;803            APT.PARAM=49            ; регистры текущих параметров APT (размер памяти,
;804                                    ; конфигурация аппаратуры,конфигурация програм-
;805                                    ; много обеспечения в байтах 0,1 и 2
;806                                    ; соответственно. Байт 3 для использования в
;807                                    ; будущем)
;808            APT.RESERVED=4A         ; резервировано для будущего использования в APT
;809    ;
;810    ;  Разные константы и ячейки для запоминания
;811    ;
;812            #AAAAAAAA=4C            ; константа
;813            ALTER.ODD=4C            ; константа
;814            #55555555=4D            ; константа
;815            ALTER.EVEN=4D           ; константа
;816            #0=4E                   ; константа
;817            ZERO=4E                 ; константа
;818            #FFFFFFFF=4F            ; константа
;819            ONES=4F                 ; константа
;820            PREVIOUS.ERROR=50       ; номер последней ошибки
;821            DATA.1=51               ; место запоминания данных для ускорителя FPA
;822            DATA.2=52               ; место запоминания данных для ускорителя FPA
;823            DATA.3=53               ; место запоминания данных для ускорителя FPA
;824            FIRST.FLT=54            ; место запоминания данных для ускорителя FPA
;825            SEC.FLT=55              ; место запоминания данных для ускорителя FPA
;826            THIRD.FLT=56            ; место запоминания данных для ускорителя FPA
;827            T57=57                  ; ячейка для временного хранения 57
;828            T58=58                  ; ячейка для временного хранения 58
;829            T59=59                  ; ячейка для временного хранения 59
;830            BEDB=5A                 ; регистр буфера данных тестера шины (UBE)
;831            BECC=5B                 ; регистр счетчика циклов тестера шины (UBE)
;832            BEBA=5C                 ; регистр адреса тестера шины (UBE)
;833            BECR1=5D                ; регистр управления 1 тестера шины (UBE)
;834            CLEAR.ERR.ADDR=5E       ; адрес регистра очистки ошибок тестера шины
```

```
;835                                    ; (UBE)
;836            BECR2=5F                ; регистр управления 2 тестера шины (UBE)
;837            #3(H)=60                ; константа
;838            #12(H)=61               ; константа
;839            #33333333=62            ; константа
;840            #0F0F0F0F=63            ; константа
;841            #00FF00FF=64            ; константа
;842            #162(H)=65              ; константа для указателя переноса данных в LS
;843            BR7.IDENT=66            ; идентификатор BR7 (1010 (двоичн.) в битах 5-2)
;844            BG7=67                  ; адрес BG7 (установлены биты 2 и 3)
;845    ;
;846    ;  Ячейки временного хранения для тестов центрального процессора
;847    ;
;848            L30=30                  ; ячейка временного хранения LS 30 (после
;849                                    ; использования восстанавливается значением
;850                                    ; 10000)
;851            L31=31                  ; ячейка временного хранения LS 31 (после
;852                                    ; использования восстанавливается значением
;853                                    ; 20000)
;854            L53=53                  ; ячейка временного хранения LS 53
;855            L73=73                  ; ячейка временного хранения LS 73
;856    ;
;857    ;  Адреса регистров
;858    ;
;859            CONTROL.OS=78           ; аппаратный индекс для управляющих слов (LS 40-4F)
;860            SHIFT.OS(3-0)=79        ; аппаратный индекс на 16 ячеек для наборов
;861                                    ; тестовых данных со сдвигом (LS 20-2F)
;862            SHIFT.OS(4-0)=7B        ; аппаратный индекс на 32 ячейки для наборов
;863                                    ; тестовых данных со сдвигом (LS 20-3F)
;864            OS=7C                   ; регистр OS
;865            DEC.CON=7D              ; десятичные переносы
;866            SIZE=7E                 ; регистр размера
;867            MDT=7E                  ; размер данных для обращений к памяти
;868            INTERRUPT.VEC=7E        ; аппаратный вектор прерывания
;869    ;
;870    ;  Это слово содержит аппаратные коды условий (CC). Они могут считываться или
;871    ;  записываться сюда. На другие биты PSL не отражается.
;872    ;
;873            PSL.CC=7F               ; коды условий PSL
;874    ;
;875    ;  Это поле допускает обращение ко всем 256 ячейкам LS
;876    ;
;877    XD.ADRS/=<16:9>,.VALIDITY=<XD.VAL>
;878    ;
;879            SAV.WR0=1                       ; память для WR0
;880            SAV.WR1=2                       ; память для WR1
;881            SAV.WR2=3                       ; память для WR2
;882            SAV.WR3=4                       ; память для WR3
;883            T5.SUB=5                        ; резервировано для общих подпрограмм
;884            T6.SUB=6                        ; резервировано для общих подпрограмм
;885            T7.SUB=7                        ; резервировано для общих подпрограмм
;886            TRANSFER.POINTER=7              ; указатель для программы переноса данных
;887            MEMORY_SIZE=7                   ; запоминание размера памяти в блоках по 256K байт
;888                                            ; после "прощупывания"
;889            FPA.FOUND=7                     ; место запоминания результата проверки
```

```
;890                                     ; наличия ускорителя плавающей запятой (FPA)
;891             UBE.FOUND=7             ; место запоминания результата проверки
;892                                     ; наличия тестера шины (UBE)
;893             T8=8                    ; ячейка для временного хранения 8
;894             T9=9                    ; ячейка для временного хранения 9
;895             T10=0A                  ; ячейка для временного хранения 10
;896             T11=0B                  ; ячейка для временного хранения 11
;897             T12=0C                  ; ячейка для временного хранения 12
;898             T13=0D                  ; ячейка для временного хранения 13
;899             T14=0E                  ; ячейка для временного хранения 14
;900             T15=0F                  ; ячейка для временного хранения 15
;901             PC=10                   ; счетчик инструкций макроуровня
;902             WR.BACKUP=11            ; резервная копия WR для MOV MEM.DATA TO WR[ ]
;903             MM.LASTADDR+1=12        ; место запоминания последнего адреса основной
;904                                     ; памяти плюс 1 (первый несуществующий адрес памяти)
;905             #FF=13                  ; маска
;906             #FFFF=14                ; маска
;907             #FF000000=15            ; маска
;908             #FFFFFF00=16            ; маска
;909             CSR0.MASK=16            ; маска
;910             CSR1.MASK=17            ; маска (01003FFF)
;911             CSR2.MASK=18            ; маска (7FFE3FFF)
;912             #FE7FFFFF=19            ; маска
;913             UBS.SET=1A              ; константа (7FF80000),используемая тестом адреса
;914                                     ; общей шины
;915             UBS.DATA=1B             ; маска (7FFF8000),используемая в качестве маски
;916                                     ; ошибок для теста данных общей шины
;917             ERR.CON.NA=1E           ; константа 800500 (16-ричная) с установленными
;918                                     ; битами 23,10,8 для загрузки регистра управления и
;919                                     ; состояния в начальных тестах
;920             ERR.CON=1F              ; константа 500 (16-ричная). биты 10 и 8
;921                                     ; установлены для загрузки регистра управления
;922                                     ; и состояния в начальных тестах
;923     ;
;924     ;  Следующие ячейки используются, главным образом, в качестве тестовых наборов
;925     ;  данных (1, перемещаемая в поле нулей)
;926     ;
;927             #1=20                   ; набор 00000001(H)
;928             #2=21                   ; набор 00000002
;929             #4=22                   ; набор 00000004
;930             #8=23                   ; набор 00000008
;931             #10=24                  ; набор 00000010
;932             #20=25                  ; набор 00000020
;933             #40=26                  ; набор 00000040
;934             #80=27                  ; набор 00000080
;935             #100=28                 ; набор 00000100
;936             #200=29                 ; набор 00000200
;937             #400=2A                 ; набор 00000400
;938             #800=2B                 ; набор 00000800
;939             #1000=2C                ; набор 00001000
;940             #2000=2D                ; набор 00002000
;941             #4000=2E                ; набор 00004000
;942             #8000=2F                ; набор 0000B000
;943             #10000=30               ; набор 00010000
;944             #20000=31               ; набор 00020000
```

```
;945             #40000=32                       ; набор 00040000
;946             #80000=33                       ; набор 00080000
;947             #100000=34                      ; набор 00100000
;948             #200000=35                      ; набор 00200000
;949             #400000=36                      ; набор 00400000
;950             #800000=37                      ; набор 00800000
;951             #1000000=38                     ; набор 01000000
;952             #2000000=39                     ; набор 02000000
;953             #4000000=3A                     ; набор 04000000
;954             #8000000=3B                     ; набор 08000000
;955             #10000000=3C                    ; набор 10000000
;956             #20000000=3D                    ; набор 20000000
;957             #40000000=3E                    ; набор 40000000
;958             #80000000=3F                    ; набор 80000000
;959             BIT0=20                         ; набор 00000001
;960             BIT1=21                         ; набор 00000002
;961             BIT2=22                         ; набор 00000004
;962             BIT3=23                         ; набор 00000008
;963             BIT4=24                         ; набор 00000010
;964             BIT5=25                         ; набор 00000020
;965             BIT6=26                         ; набор 00000040
;966             BIT7=27                         ; набор 00000080
;967             BIT8=28                         ; набор 00000100
;968             BIT9=29                         ; набор 00000200
;969             BIT10=2A                        ; набор 00000400
;970             BIT11=2B                        ; набор 00000800
;971             BIT12=2C                        ; набор 00001000
;972             BIT13=2D                        ; набор 00002000
;973             BIT14=2E                        ; набор 00004000
;974             BIT15=2F                        ; набор 00008000
;975             BIT16=30                        ; набор 00010000
;976             BIT17=31                        ; набор 00020000
;977             BIT18=32                        ; набор 00040000
;978             BIT19=33                        ; набор 00080000
;979             BIT20=34                        ; набор 00100000
;980             BIT21=35                        ; набор 00200000
;981             BIT22=36                        ; набор 00400000
;982             BIT23=37                        ; набор 00800000
;983             BIT24=38                        ; набор 01000000
;984             BIT25=39                        ; набор 02000000
;985             BIT26=3A                        ; набор 04000000
;986             BIT27=3B                        ; набор 08000000
;987             BIT28=3C                        ; набор 10000000
;988             BIT29=3D                        ; набор 20000000
;989             BIT30=3E                        ; набор 40000000
;990             BIT31=3F                        ; набор 80000000
;991     ;
;992     ;  Мнемоника адресов CSR контроллера памяти
;993     ;
;994             CSR0=4E                 ; для доступа к CSR0 все биты очищены
;995             CSR1=22                 ; для доступа к CSR1 установлен бит 2
;996             CSR2=23                 ; для доступа к CSR2 установлен бит 3
;997     ;
;998     ;  Биты слова управления и состояния (информация для микромонитора во время
;999     ;  выполнения тестов). Слово управления и состояния доступно микромонитору
```

```
;1000    ;  по адресу LS 40
;1001    ;
;1002            PRINT.UBE=26            ; печать,имеется или нет тестер шины UBE (бит 6)
;1003                                    ; (индикатор в LS7)
;1004            PRINT.RB0=27            ; печать,имеется или нет RB0 и на каком
;1005                                    ; устройстве (бит 7).индикатор в LS7. номер
;1006                                    ; накопителя в LS6
;1007            ERROR=28                ; индикация ошибки теста (бит 8)
;1008            SET.PA.ERR=29           ; указывает консольному процессору,что следует
;1009                                    ; генерировать ошибку паритета по адресу
;1010                                    ; управляющей памяти WCS,указанному в LS7 (бит 9)
;1011            CONSOLE.LOE=2A          ; указывает,что консольный процессор выполняет
;1012                                    ; зацикливание при ошибке (бит 10) (для начальных
;1013                                    ; тестов)
;1014            PRINT.MEM.SIZE=2B       ; печать размера памяти в блоках по 256 K
;1015                                    ; (бит 11) (размер в LS7)
;1016            PRINT.FPA=2C            ; печать, имеется или нет ускоритель плавающей
;1017                                    ; запятой FPA (бит 12) (индикатор в LS7)
;1018            XFER.DATA=2D            ; указывает перенос данных в местную (LS) или
;1019                                    ; основную (мм) память (бит 13)
;1020            EOS=2E                  ; конец сегмента (бит 14)
;1021            BEGIN.TEST=2F           ; начало теста (бит 15)
;1022            INTERUPT.EN=30          ; разрешение прерывания или сигнала, заданного
;1023                                    ; битами от 17 до 22 и от 28 до 30 (бит 16)
;1024            CONSOLE.HALT=31         ; прерывание по останову от консоли (бит 17)
;1025            PWR.FAIL=32             ; прерывание по аварии питания (бит 18)
;1026            INTERVAL.TIM=33         ; прерывание от интервального таймера (бит 19)
;1027            CONSOLE.ATTN=34         ; прерывание по биту "внимание" (ATTN) консоли
;1028                                    ; (бит 20)
;1029            CONSOLE.ACK=35          ; прерывание по биту "подтверждение" (ACK) консоли
;1030                                    ; (бит 21)
;1031            DISABLE.MEM.REF=36      ; запрет обращений к памяти (бит 22)
;1032            CINIT=3C                ; сигнал общей шины INIT для контроллера памяти
;1033                                    ; (бит 28)
;1034            UBS.DCLO=3D             ; индикация общей шины DC LO для контроллера
;1035                                    ; памяти (бит 29)
;1036            UBS.BBSY=3E             ; индикация общей шины BUS BUSY для контроллера
;1037                                    ; памяти (бит 30)
;1038            NA.EXP.REC=37           ; печать N/A под EXP и REC (бит 23)
;1039            OTHER.DATA=38           ; печать значений под OTHER (бит 24)
;1040            CONSOLE.LOOP=39         ; консоль выполняет зацикливание в соответствии со
;1041                                    ; счетчиком циклов в LS (бит 25)
;1042            PRINT.CRD=3A            ; печать числа ошибок в одиночных битах (бит 26)
;1043            CLR.PA.ERR=3B           ; указывает консоли, что следует очистить ошибку
;1044                                    ; паритета в управляющей памяти WCS по адресу,
;1045                                    ; указанному в LS7 (бит 27)
;1046            PRINT.IDC=3F            ; печать,имеется или нет встроенный контроллер
;1047                                    ; дисков IDC (бит 31)(индикатор в LS7)
;1048    ;
;1049    ;  Коды модулей
;1050    ;
;1051            WCS=20                  ; код модуля для платы управляющей памяти WCS
;1052                                    ; (установлен бит 0)
;1053            CPU=21                  ; код модуля для платы путей данных DAP
;1054                                    ; (установлен бит 1)
```

```
;1055            MCT=22                  ; код модуля для платы контроллера памяти MCT
;1056                                    ; (установлен бит 2)
;1057            FPA=23                  ; код модуля для платы ускорителя плавающей
;1058                                    ; запятой FPA (установлен бит 3)
;1059            ARRAY=24                ; код модуля для платы ОЗУ (установлен бит 4)
;1060            IDC=25                  ; код модуля для платы встроенного контроллера
;1061                                    ; дисков IDC (установлен бит 5)
;1062    ;
;1063    ;   Биты ошибок CSR1 контроллера памяти
;1064    ;
;1065            VALID.ERR=2E            ; не установлен бит действительности данных (VALID),
;1066                                    ; если 1 (бит 14)
;1067            TB.PAR.ERR=2F           ; ошибка паритета буфера трансляции (бит 15)
;1068            NXM=30                  ; ошибка - несуществующая память (бит 16)
;1069            UB.BSY=31               ; общая шина занята (бит 17)
;1070            ADP.REG=32              ; выбор регистра адаптера общей шины (бит 18)
;1071            WR.ACROSS.PG=33         ; ошибка записи за пределами страницы (бит 19)
;1072            OP.ERR=34               ; ошибка операции (бит 20)
;1073            TB.MISS=35              ; промах в буфере трансляции (транслированный
;1074                                    ; виртуальный адрес в буфере отсутствует)
;1075                                    ; (бит 21)
;1076            ACCESS.REFUSED=36       ; ошибка защиты при обращении (бит 22)
;1077            MODIFY.REFUSED=37       ; ошибка защиты при записи (бит 23)
;1078            CRD=3E                  ; исправлена ошибка в одиночном бите (бит 30)
;1079            RDS=3F                  ; неисправимая ошибка данных (бит 31)
;1080    ;
;1081    ;   Управляющие биты CSR1 контроллера памяти
;1082    ;
;1083            ECC.DIS=39              ; бит запрета коррекции (ECC) в CSR1 (бит 25)
;1084            DIAG.CHK=3A             ; бит диагностического контроля CSR1 (бит 26)
;1085            MME=3B                  ; бит разрешения диспетчера памяти в CSR1
;1086                                    ; (бит 27)
;1087            INH.CRD=3C              ; бит запрета сообщения о корректируемых ошибках
;1088                                    ; данных (CRD) (бит 28)
;1089            TB.PAR.DIAG=3D          ; бит диагностирования паритета буфера трансляции
;1090                                    ; (принудительная ошибка паритета TB) (бит 29)
;1091    ;
;1092    ;   Биты регистров управления тестера шины (UBE)
;1093    ;
;1094    ;       Регистр управления 1
;1095    ;
;1096            GO=20                   ; запуск тестера шины (UBE) (бит 0)
;1097            BR4=21                  ; выдача запроса шины на уровне 4 (бит 1)
;1098            BR5=22                  ; выдача запроса шины на уровне 5 (бит 2)
;1099            BR6=23                  ; выдача запроса шины на уровне 6 (бит 3)
;1100            BR7=24                  ; выдача запроса шины на уровне 7 (бит 4)
;1101            NPR=25                  ; выдача запроса прямого доступа (NPR) (бит 5)
;1102            INT.ODNE=26             ; прерывание по завершению (при переполнении
;1103                                    ; счетчика циклов) (бит 6)
;1104            RDY=27                  ; бит "готово", устанавливаемый, если есть
;1105                                    ; готовность начать функционирование (бит 7)
;1106            C00=28                  ; бит C0 операции шины (бит 8)
;1107            C01=29                  ; бит C1 операции шины (бит 9)
;1108            FUNA=2A                 ; бит функции A тестера шины (бит 10)
;1109            FUNB=2B                 ; бит функции B тестера шины (бит 11)
```

```
;1110            CC+DAT=2C                ; данные из счетчика циклов, если установлен,
;1111                                     ; и из регистра данных, если очищен
;1112            INH.DATIP.ROL=2D         ; запрет циклического сдвига при чтении с паузой
;1113                                     ; (DATIP) (бит 13)
;1114            DATOB.ON.DATIP=2E        ; выполнение записи байта (DATOB) после цикла
;1115                                     ; чтения с паузой (DATIP) (бит 14)
;1116            ERR=2F                   ; ошибка общей шины или тестера (бит 15)
;1117    ;
;1118    ;    Регистр управления 2
;1119    ;
;1120            EXT.MEM0=20              ; бит 0 расширения памяти (бит 0)
;1121            EXT.MEM1=21              ; бит 1 расширения памяти (бит 1)
;1122            INH.INC.ADDR&CC=22       ; запрет наращивания счетчика циклов и регистра
;1123                                     ; адреса (бит 2)
;1124            INH.SACK=23              ; запрет выдачи BUS SACK по BUS GRANT (бит 3)
;1125            PWR.DWN.SEQ=24           ; установка ACLO (бит 4)
;1126            WNG.GNT.BCK=25           ; не получен BUS GRANT в ответ на запрос
;1127                                     ; высшего приоритета (бит 5)
;1128            MAX.LATE.ERR=26          ; превышено время задержки для NPR и BR (бит 6)
;1129            NO.SACK.ERR=27           ; центральный процессор не зафиксировал таймаута
;1130                                     ; при отсутствии SACK (бит 7)
;1131            NO.SSYN.ERR=28           ; после выдачи MSYN не получен SSYN (бит 8)
;1132            WRG.A.LINES=29           ; ошибка адреса в переданном или принятом
;1133                                     ; адресе (бит 9)
;1134            NO.GNT+NOT.ONE.GNT=2A    ; отсутствует GRANT или более одного сигнала
;1135                                     ; GRANT после запроса (бит 10)
;1136            INTR.SSYN.ERR=2B         ; не получен SSYN после прерывания шины (бит 11)
;1137            ENB.PB=2C                ; разрешение для бита паритета B (бит 12)
;1138            CCOVF=2D                 ; переполнение счетчика циклов (бит 13)
;1139            TM.DLY=2E                ; запрос для шины каждые 10 мкс (бит 14)
;1140    ;
;1141    ;   Мнемоника BG6
;1142    ;
;1143            BG6=23                   ; для адреса BG6 установлен бит 3
;1144    ;
;1145    ;   Информация об ошибках и управляющая информация для микромонитора
;1146    ;
;1147            CONTROL.STATUS=40        ; слово управления и состояния
;1148            ERROR.NUMBER=41          ; номер ошибки в текущем тексте
;1149            EXPECTED.DATA=42         ; ожидаемый результат текущего теста
;1150            RECEIVED.DATA=43         ; действительный результат текущего теста
;1151            ADDRESS.DATA=44          ; другие актуальные данные
;1152            ERROR.MASK=45            ; биты,подлежащие проверке в результате
;1153            MODULE.NUM=46            ; место хранения номера модуля (кодированного)
;1154            LOOP.CNT=47              ; место запоминания счета циклов для управления
;1155                                     ; зацикливанием из консольного процессора
;1156            ERROR.CONTROL=48         ; место запоминания информации управления
;1157                                     ; ошибками (зацикливание по ошибке и т.д.)
;1158            LOE=20                   ; если установлен,зацикливание на ошибке (бит 0)
;1159            NER=21                   ; если установлен,не выдаются сообщения об
;1160                                     ; ошибках (бит 1)
;1161            BELL=22                  ; если установлен,звуковой сигнал при ошибке
;1162                                     ; (бит 2)
;1163            HOE=23                   ; если установлен, останов при ошибке (бит 3)
;1164            SER=24                   ; если установлен, разрешение сообщений об
```

```
;1165                                   ; исправимых ошибках данных
;1166            APT.PRESENT=25         ; если установлен, имеется APT (бит 5)
;1167            LOOP.COMMAND=26        ; если установлен, зацикливание напечатанной
;1168                                   ; команды (бит 6)
;1169            SPECIAL=27             ; специальный бит для зацикливания в тестах
;1170                                   ; "прощупывания" (бит 7)
;1171            APT.PARAM=49           ; регистры текущих параметров APT (размер памяти,
;1172                                   ; конфигурация аппаратуры, конфигурация програм-
;1173                                   ; много обеспечения в байтах 0,1 и 2
;1174                                   ; соответственно. байт 3 для использования в
;1175                                   ; будущем)
;1176            APT.RESERVED=4A        ; резервировано для будущего использования в APT
;1177    ;
;1178    ;  Разные константы и ячейки для запоминания
;1179    ;
;1180            #AAAAAAAA=4C           ; константа
;1181            ALTER.ODD=4C           ; константа
;1182            #55555555=4D           ; константа
;1183            ALTER.EVEN=4D          ; константа
;1184            #0=4E                  ; константа
;1185            ZERO=4E                ; константа
;1186            #FFFFFFFF=4F           ; константа
;1187            ONES=4F                ; константа
;1188            PREVIOUS.ERROR=50      ; номер последней ошибки
;1189            DATA.1=51              ; место запоминания данных для ускорителя FPA
;1190            DATA.2=52              ; место запоминания данных для ускорителя FPA
;1191            DATA.3=53              ; место запоминания данных для ускорителя FPA
;1192            FIRST.FLT=54           ; место запоминания данных для ускорителя FPA
;1193            SEC.FLT=55             ; место запоминания данных для ускорителя FPA
;1194            THIRD.FLT=56           ; место запоминания данных для ускорителя FPA
;1195            T57=57                 ; ячейка для временного хранения 57
;1196            T58=58                 ; ячейка для временного хранения 58
;1197            T59=59                 ; ячейка для временного хранения 59
;1198            BEDB=5A                ; регистр буфера данных тестера шины (UBE)
;1199            BECC=5B                ; регистр счетчика циклов тестера шины (UBE)
;1200            BEBA=5C                ; регистр адреса тестера шины (UBE)
;1201            BECR1=5D               ; регистр управления 1 тестера шины (UBE)
;1202            CLEAR.ERR.ADDR=5E      ; адрес регистра очистки ошибок тестера шины
;1203                                   ; (UBE)
;1204            BECR2=5F               ; регистр управления 2 тестера шины (UBE)
;1205            #3(H)=60               ; константа
;1206            #12(H)=61              ; константа
;1207            #33333333=62           ; константа
;1208            #0F0F0F0F=63           ; константа
;1209            #00FF00FF=64           ; константа
;1210            #162(H)=65             ; константа для указателя переноса данных в LS
;1211            BR7.IDENT=66           ; идентификатор BR7 (1010 (двоичн.) в битах 5-2)
;1212            BG7=67                 ; адрес BG7 (установлены биты 2 и 3)
;1213    ;
;1214    ;  Ячейки временного хранения для тестов центрального процессора
;1215    ;
;1216            L30=30                 ; ячейка временного хранения LS 30 (после
;1217                                   ; использования восстанавливается значением
;1218                                   ; 10000)
;1219            L31=31                 ; ячейка временного хранения LS 31 (после
```

```
;1220                                   ; использования восстанавливается аппаратурой
;1221                                   ; 20000)
;1222           L53=53                  ; ячейка временного хранения LS 53
;1223           L73=73                  ; ячейка временного хранения LS 73
;1224   ;
;1225   ;  Адреса регистров
;1226   ;
;1227           CONTROL.OS=7B           ; аппаратный индекс для управляющих слов (LS 40-4F)
;1228           SHIFT.OS(3-0)=79        ; аппаратный индекс на 16 ячеек для наборов
;1229                                   ; тестовых данных со сдвигом (LS 20-2F)
;1230           SHIFT.OS(4-0)=7B        ; аппаратный индекс на 32 ячейки для наборов
;1231                                   ; тестовых данных со сдвигом (LS 20-3F)
;1232           OS=7C                   ; регистр OS
;1233           DEC.CON=7D              ; десятичные переносы
;1234           SIZE=7E                 ; регистр размера
;1235           MDT=7E                  ; размер данных для обращений к памяти
;1236           INTERRUPT.VEC=7E        ; аппаратный вектор прерывания
;1237   ;
;1238   ;  Это слово содержит аппаратные коды условий (CC). Они могут считываться или
;1239   ;  записываться сюда. На другие биты PSL не отражается.
;1240   ;
;1241           PSL.CC=7F               ; коды условий PSL
;1242   ;
;1243   ;     ОБЩЕЕ ПРИСВОЕНИЕ ЯЧЕЕК LS (регистры)
;1244   ;
;1245   ;  Верхняя половина LS
;1246   ;
;1247           XCPR.OS=0F8             ; аппаратный индекс для регистров общего
;1248                                   ; назначения
;1249           USER.INDEX(3-0)=0F9     ; аппаратный индекс на 16 ячеек для области
;1250                                   ; диагностических данных (ячейки LS от 0A0 до 0AF)
;1251           USER.INDEX(4-0)=0FB     ; аппаратный индекс на 32 ячейки для области
;1252                                   ; диагностических данных (ячейки LS от 0A0 до 0BF)
;1253           CCR=0FC                 ; регистр чтения консоли
;1254           TIMER=0FD               ; значение интервального таймера
;1255           ALU.CC=0FE              ; коды условий АЛУ
;1256   ;
;1257   ;  Эта ячейка представляет собой регистр, предназначенный только для записи.
;1258   ;  Оказывается воздействие на следующие биты PSL:
;1259   ;
;1260   ;     режим совместимости - бит <31>
;1261   ;     текущий режим - биты <25:24>
;1262   ;     уровень приоритета прерываний (IPL) - биты <20:16>
;1263   ;     разрешение прерываний по слежению (T или TP) - бит <4>
;1264   ;     код отрицательного условия - бит <3>
;1265   ;     код условия нуля - бит <2>
;1266   ;     код условия переполнения - бит <1>
;1267   ;     код условия переноса - бит <0>
;1268   ;
;1269           PSL.HW=0FF              ; биты аппаратного PSL
;1270   ;
;1271   ;       СПЕЦИФИЧЕСКОЕ ДЛЯ ПРОГРАММЫ ПРИСВОЕНИЕ ЯЧЕЕК LS (LS 80-F7)
;1272   ;
;1273   ;  Эти адреса доступны только для микроинструкции MOV, начиная с адреса 80. Они не
;1274   ;  используются всеми модулями программного обеспечения, но специально предназна-
```

```
;1275      ; чены для этого модуля.
;1276      ;
;1277      ;
;1278
;1279            CKBTS.0111100=80        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1280                                    ; контрольных битов
;1281            CKBTS.1000011=81        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1282                                    ; контрольных битов
;1283            CKBTS.1100100=82        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1284                                    ; контрольных битов
;1285            CKBTS.0100101=83        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1286                                    ; контрольных битов
;1287            CKBTS.0100110=84        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1288                                    ; контрольных битов
;1289            CKBTS.0100000=85        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1290                                    ; контрольных битов
;1291            CKBTS.1010100=86        ; инверсия ожидаемых контрольных битов (CKBT) для теста генерации
;1292                                    ; контрольных битов
;1293            CKBTS.1111101=87        ; инверсия ожидаемых контрольных битов для теста схемы ECC
;1294            IDENT.MASK=88           ; маска для идентификатора BR (FFFFFFC3)
;1295            #FFFFFF80=89            ; маска ошибок для битов 6-0 (маска CSR 0)
;1296            #30000=9A               ; эталон данных для двойной ошибки
;1297            #7FFF8000=9B            ; маска для теста поля признаков (устанавливаемые биты 15-30)
;1298            #540000=9C              ; адрес для использования в тесте несуществующей памяти
;1299            #F00000=9D              ; адрес для использования в тесте несуществующей памяти
;1300            #FC0000=9E              ; адрес для использования в тесте несуществующей памяти
;1301            #3F003FFF=9F            ; маска для битов ошибок в CSR1
;1302            ACCESS.RCHK.KERNAL=0A0  ; ожидаемые данные для теста защиты
;1303            MODIFY.RCHK.KERNEL=0A1  ; ожидаемые данные для теста защиты
;1304            ACCESS.RCHK.EXEC=0A2    ; ожидаемые данные для теста защиты
;1305            MODIFY.RCHK.EXEC=0A3    ; ожидаемые данные для теста защиты
;1306            ACCESS.RCHK.SUPER=0A4   ; ожидаемые данные для теста защиты
;1307            MODIFY.RCHK.SUPER=0A5   ; ожидаемые данные для теста защиты
;1308            ACCESS.RCHK.USER=0A6    ; ожидаемые данные для теста защиты
;1309            MODIFY.RCHK.USER=0A7    ; ожидаемые данные для теста защиты
;1310            ACCESS.WCHK.KERNAL=0A8  ; ожидаемые данные для теста защиты
;1311            MODIFY.WCHK.KERNAL=0A9  ; ожидаемые данные для теста защиты
;1312            ACCESS.WCHK.EXEC=0AA    ; ожидаемые данные для теста защиты
;1313            MODIFY.WCHK.EXEC=0AB    ; ожидаемые данные для теста защиты
;1314            ACCESS.WCHK.SUPER=0AC   ; ожидаемые данные для теста защиты
;1315            MODIFY.WCHK.SUPER=0AD   ; ожидаемые данные для теста защиты
;1316            ACCESS.WCHK.USER=0AE    ; ожидаемые данные для теста защиты
;1317            MODIFY.WCHK.USER=0AF    ; ожидаемые данные для теста защиты
;1318    ;
```

```
;1319    .PAGE  "МАКРООПРЕДЕЛЕНИЯ "
;1320    .UCODE
;1321    .CREF
;1322    ;
;1323    ;  Эта группа имеет двухадресный формат с модификацией приемника.
;1324    ;  Первый операнд комбинируется со вторым операндом и результат записывается по
;1325    ;  второму операнду. Инструкции CMP и BIT содержат только считываемые операнды.
;1326    ;
;1327    ;  Во время всех арифметических и логических инструкций коды условий (CC) АЛУ
;1328    ;  можно загрузить добавлением к микрослову макрооператора
;1329    ;
;1330    ;     DT(SIZE)&SET.ALU.CC
;1331    ;
;1332    ;  Этим указывается, что регистр ALU CC загружается условиями с K1804BC1. Никакие
;1333    ;  внешние манипуляции не происходят. Если операция производится над байтами дан-
;1334    ;  ных, CC загружается в соответствии с битами 7-0. Слова используют биты 15-0, а
;1335    ;  длинные слова используют биты 31-0. Далее следует описание значений CC АЛУ для
;1336    ;  каждой  функции:
;1337    ;
;1338    ;  ADD - двоичное сложение двух операндов
;1339    ;
;1340    ;     N - бит знака
;1341    ;     Z - установлен, если результат равен нулю
;1342    ;     V - арифметическое переполнение
;1343    ;     C - перенос
;1344    ;
;1345    ;  SUB - двоичное сложение первого операнда с дополнением до двух второго операнда
;1346    ;
;1347    ;     N - бит знака
;1348    ;     Z - установлен, если результат равен нулю
;1349    ;     V - арифметическое переполнение
;1350    ;     C - инверсия займа
;1351    ;
;1352    ;  BIS - логическое "ИЛИ" для двух операндов
;1353    ;
;1354    ;     N - бит знака
;1355    ;     Z - установлен, если результат равен нулю
;1356    ;     V - случайное значение (не присвоена никакая функция)
;1357    ;     C - случайное значение (не присвоена никакая функция)
;1358    ;
;1359    ;  BIC - функция "И" для дополнения до единицы (инверсии) первого операнда
;1360    ;        и второго операнда
;1361    ;
;1362    ;     N - бит знака
;1363    ;     Z - установлен, если результат равен нулю
;1364    ;     V - случайное значение (не присвоена никакая функция)
;1365    ;     C - случайное значение (не присвоена никакая функция)
;1366    ;
;1367    ;  AND - логическое "И" для двух операндов
;1368    ;
;1369    ;     N - бит знака
;1370    ;     Z - установлен, если результат равен нулю
;1371    ;     V - случайное значение (не присвоена никакая функция)
;1372    ;     C - случайное значение (не присвоена никакая функция)
;1373    ;
```

```
;1374    ;  XOR - логическое исключающее "ИЛИ" для двух операндов
;1375    ;
;1376    ;     N - бит знака
;1377    ;     Z - установлен, если результат равен нулю
;1378    ;     V - случайное значение (не присвоена никакая функция)
;1379    ;     C - случайное значение (не присвоена никакая функция)
;1380    ;
;1381    ;  CMP - выполняет двоичное сложение первого операнда с дополнением до двух
;1382    ;        второго операнда без запоминания результата
;1383    ;
;1384    ;     N - бит знака
;1385    ;     Z - установлен, если результат равен нулю
;1386    ;     V - арифметическое переполнение
;1387    ;     C - перенос
;1388    ;
;1389    ;  BIT - выполняет логическое умножение (И) для двух операндов без запоминания
;1390    ;        результата
;1391    ;
;1392    ;     N - бит знака
;1393    ;     Z - установлен, если результат равен нулю
;1394    ;     V - случайное значение (не присвоена никакая функция)
;1395    ;     C - случайное значение (не присвоена никакая функция)
;1396    ;
;1397    ;  SWAP - взаимная замена значений двух операндов
;1398    ;
;1399    ;     N - бит знака для значения в рабочем регистре
;1400    ;     Z - установлен, если рабочий регистр в результате имеет значение нуль
;1401    ;     V - случайное значение (не присвоена никакая функция)
;1402    ;     C - случайное значение (не присвоена никакая функция)
;1403    ;
;1404    ;  Макрооператоры формата  WR[B] = WR[B]  операция  LS[D]
;1405    ;
;1406    ADD LS[] TO   WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP.SMALL/<.SHIFT[<.AND[<SDP/LS.PLUS.WR>,07F]>,-2]>"
;1407    SUB LS[] FROM WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP.SMALL/<.SHIFT[<.AND[<SDP/LS.FROM.WR>,07F]>,-2]>"
;1408    BIS LS[] TO   WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.OR.WR>,0FF]>,-2]>"
;1409    BIC LS[] TO   WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.MASK.WR>,0FF]>,-2]>"
;1410    AND LS[] TO   WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP.SMALL/<.SHIFT[<.AND[<SDP/LS.AND.WR>,07F]>,-2]>"
;1411    XOR LS[] TO   WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP.SMALL/<.SHIFT[<.AND[<SDP/LS.XOR.WR>,07F]>,-2]>"
;1412    CMP LS[] WITH WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.CMP.WR>,0FF]>,-2]>"
;1413    BIT LS[] WITH WR[]  "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.BIT.LS>,0FF]>,-2]>"
;1414    SWAP LS[] WITH WR[] "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/SWAP.LS.WR>,0FF]>,-2]>"
;1415    ;
;1416    ;  Следующий макрооператор используется с ADD, SUB, AND, XOR LS[] TO WR[].
;1417    ;  Он берет исходное содержимое WR[] и заносит его в LS[].
;1418    ;
;1419    XCHG                "XCHG.BIT/YES"
;1420    ;
;1421    ;  Макрооператоры формата  LS[D] = LS[D]  операция  Q-регистр
;1422    ;
;1423    ADD  Q TO    LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.PLUS.LS>,0FF]>,-2]>"
;1424    SUB  Q FROM  LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.FROM.LS>,0FF]>,-2]>"
;1425    BIS  Q TO    LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.OR.LS>,0FF]>,-2]>"
;1426    AND  Q TO    LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.AND.LS>,0FF]>,-2]>"
;1427    XOR  Q TO    LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.XOR.LS>,0FF]>,-2]>"
;1428    CMP  Q WITH  LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.CMP.LS>,0FF]>,-2]>"
```

```
;1429    BIT  Q WITH  LS[]   "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.BIT.Q>;0FF]>,-2]>"
;1430    ;
;1431    ;  Макрооператоры формата  Q-регистр = Q-регистр  операция  LS[D]
;1432    ;
;1433    ADD  LS[] TO   Q     "OPC/BASIC,D.ADRS/@1;DP/<.SHIFT[<.AND[<SDP/LS.PLUS.Q>,0FF]>,-2]>"
;1434    SUB  LS[] FROM Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.FROM.Q>,0FF]>,-2]>"
;1435    BIS  LS[] TO   Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.OR.Q>,0FF]>,-2]>"
;1436    BIC  LS[] TO   Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.MASK.Q>,0FF]>,-2]>"
;1437    AND  LS[] TO   Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.AND.Q>,0FF]>,-2]>"
;1438    XOR  LS[] TO   Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.XOR.Q>,0FF]>,-2]>"
;1439    CMP  LS[] WITH Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.CMP.Q>,0FF]>,-2]>"
;1440    BIT  LS[] WITH Q     "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.BIT.Q>,0FF]>,-2]>"
;1441    ;
;1442    ;  Макрооператоры формата  LS[D] = LS[D]  операция  WR[B]
;1443    ;
;1444    ADD  WR[] TO   LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.PLUS.LS>,0FF]>,-2]>"
;1445    SUB  WR[] FROM LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.FROM.LS>,0FF]>,-2]>"
;1446    BIS  WR[] TO   LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.OR.LS>,0FF]>,-2]>"
;1447    AND  WR[] TO   LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.AND.LS>,0FF]>,-2]>"
;1448    XOR  WR[] TO   LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.XOR.LS>,0FF]>,-2]>"
;1449    CMP  WR[] WITH LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.CMP.LS>,0FF]>,-2]>"
;1450    BIT  WR[] WITH LS[]  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.BIT.LS>,0FF]>,-2]>"
;1451    SWAP WR[] WITH LS[]  "SWAP LS[@2] WITH WR[@1]"
;1452    ;
;1453    ;  Макрооператоры формата  WR[B] = WR[B]  операция  WR[A]
;1454    ;
;1455    ADD  WR[] TO   WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.PLUS.WR>,03F]>"
;1456    SUB  WR[] FROM WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.FROM.WR>,03F]>"
;1457    BIS  WR[] TO   WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.OR.WR>,03F]>"
;1458    BIC  WR[] TO   WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.MASK.WR>,03F]>"
;1459    AND  WR[] TO   WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.AND.WR>,03F]>"
;1460    XOR  WR[] TO   WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.XOR.WR>,03F]>"
;1461    CMP  WR[] WITH WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.CMP.WR>,03F]>"
;1462    BIT  WR[] WITH WR[]  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.BIT.WR>,03F]>"
;1463    ;
;1464    ;  Макрооператоры формата  WR[B] = WR[B]  операция  Q-регистр
;1465    ;
;1466    ADD  Q TO     WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.PLUS.WR>,03F]>"
;1467    SUB  Q FROM   WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.FROM.WR>,03F]>"
;1468    BIS  Q TO     WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.OR.WR>,03F]>"
;1469    AND  Q TO     WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.AND.WR>,03F]>"
;1470    XOR  Q TO     WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.XOR.WR>,03F]>"
;1471    CMP  Q WITH   WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.CMP.WR>,03F]>"
;1472    BIT  Q WITH   WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/Q.BIT.WR>,03F]>"
;1473    ;
;1474    ;  Макрооператоры формата  Q-регистр = Q-регистр  операция  WR[B]
;1475    ;
;1476    ADD  WR[] TO   Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.PLUS.Q>,03F]>"
;1477    SUB  WR[] FROM Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.FROM.Q>,03F]>"
;1478    BIS  WR[] TO   Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.OR.Q>,03F]>"
;1479    BIC  WR[] TO   Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.MASK.Q>,03F]>"
;1480    AND  WR[] TO   Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.AND.Q>,03F]>"
;1481    XOR  WR[] TO   Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.XOR.Q>,03F]>"
;1482    CMP  WR[] WITH Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/WR.CMP.Q>,03F]>"
;1483    BIT  WR[] WITH Q     "OPC1/EXTENDED,A.ADRS/@1,XDP/<.AND[<SDP/Q.BIT.WR>,03F]>"
```

```
;1484   ;
;1485   ;  Форматы этой группы трехадресного типа. Первый и второй операнды
;1486   ;  считываются, а результат записывается по третьему операнду
;1487   ;
;1488   ;  Макрооператоры формата  Q-регистр = WR[B]  операция  WR[A]
;1489   ;
;1490   ADD  WR[] PLUS WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.PLUS.WR.Q>,03F]>"
;1491   SUB  WR[] FROM WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.FROM.WR.Q>,03F]>"
;1492   BIS  WR[] WITH WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.OR.WR.Q>,03F]>"
;1493   BIC  WR[] WITH WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.MASK.WR.Q>,03F]>"
;1494   AND  WR[] WITH WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.AND.WR.Q>,03F]>"
;1495   XOR  WR[] WITH WR[] TO Q  "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.XOR.WR.Q>,03F]>"
;1496   ;
;1497   ;   Макрооператоры формата  Q-регистр = WR[B]  операция  LS[D]
;1498   ;
;1499   ADD  WR[] PLUS LS[] TO Q  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.PLUS.LS.Q>,0FF]>,-2]>"
;1500   SUB  WR[] FROM LS[] TO Q  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.FROM.LS.Q>,0FF]>,-2]>"
;1501   BIS  WR[] WITH LS[] TO Q  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.OR.LS.Q>,0FF]>,-2]>"
;1502   AND  WR[] WITH LS[] TO Q  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.AND.LS.Q>,0FF]>,-2]>"
;1503   XOR  WR[] WITH LS[] TO Q  "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.XOR.LS.Q>,0FF]>,-2]>"
;1504   ;
;1505   ;   Макрооператоры формата  Q-регистр = LS[D]  операция  WR[B]
;1506   ;
;1507   ADD LS[] PLUS WR[] TO Q    "ADD WR[@2] PLUS LS[@1] TO Q"
;1508   SUB LS[] FROM WR[] TO Q    "OPC/BASIC,B.ADRS/@2,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.FROM.WR.Q>,0FF]>,-2]>"
;1509   BIS LS[] WITH WR[] TO Q    "BIS WR[@2] WITH LS[@1] TO Q"
;1510   AND LS[] WITH WR[] TO Q    "AND WR[@2] WITH LS[@1] TO Q"
;1511   XOR LS[] WITH WR[] TO Q    "XOR WR[@2] WITH LS[@1] TO Q"
;1512   ;
;1513   ;   Операции  ADD/SUB  специального назначения
;1514   ;
;1515   ADD (WR[] TO WR[])+1       "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.PLUS.WR+1>,03F]>"
;1516   ADD (LS[] TO WR[])+1       "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.PLUS.WR+1>,0FF]>,-2]>"
;1517   ADD (WR[] TO LS[])+1       "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.PLUS.LS+1>,0FF]>,-2]>"
;1518   ADD OS PLUS WR[] TO LS[]   "OPC1/MOVE,B.ADRS/@1,XD.ADRS/@2,MDP/<.SHIFT[<.AND[<SDP/WR.PLUS.OS>,3F]>,-31>"
;1519   SUB (WR[] FROM WR[])-1     "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.FROM.WR-1>,03F]>"
;1520   SUB (LS[] FROM WR[])-1     "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.FROM.WR-1>,0FF]>,-2]>"
;1521   SUB (WR[] FROM LS[])-1     "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.FROM.LS-1>,0FF]>,-2]>"
;1522   SUB WRB[] FROM WRA[] TO WRB[]  "OPC1/EXTENDED,B.ADRS/@1,A.ADRS/@2,XDP/<.AND[<SDP/WR.FROM.WR.WR>,03F]>"
;1523   SUB LS[] FROM WR[] TO LS       "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.FROM.WR.LS>,0FF]>,-2]
;1524   SUB WR[] FROM LS[] TO WR       "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WRA.FROM.LS.WRB>,0FF]>,-
;1525   SUB WRA[] FROM Q TO WRB[]      "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.FROM.Q.WR>,03F]>"
;1526   SUB LS[] FROM Q TO LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/LS.FROM.Q.LS>,0FF]>,-2]>"
;1527   SUB Q FROM WR[] TO Q         "OPC1/EXTENDED,B.ADRS/@1,XDP/<.AND[<SDP/Q.FROM.WR.Q>,03F]>"
;1528   SUB Q FROM LS[] TO Q         "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.FROM.LS.Q>,0FF]>,-2]>"
;1529   ADD Q PLUS WR[] TO LS[]      "OPC/BASIC,D.ADRS/@2,B.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.PLUS.WR.TO.LS>,0FF]>,-2]>
;1530   SUB Q FROM WR[] TO LS[]      "OPC/BASIC,D.ADRS/@2,B.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.FROM.WR.TO.LS>,0FF]>,-2]>
;1531   ADD Q PLUS LS[] TO WR[]      "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/Q.PLUS.LS.TO.WR>,0FF]>,-2]>
;1532   ADD Q TO WR[] XCHG TO LS[]   "OPC/BASIC,D.ADRS/@2,B.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/WR.PLUS.Q.XCHG>,0FF]>,-2]>"
;1533   ;
;1534   ;  Макроинструкции пересылки
;1535   ;
;1536   ;  Во время макроинструкций пересылки можно загружать коды условий (CC) АЛУ
;1537   ;  путем добавления к микрослову макрооператора
;1538   ;
```

```
;1539    ;       DT(SIZE)&SET.ALU.CC
;1540    ;
;1541    ;   Далее следует описание значений кодов условий АЛУ для каждой функций:
;1542    ;
;1543    ;   MOV - занесение первого операнда на место второго операнда
;1544    ;
;1545    ;       N - бит знака
;1546    ;       Z - установлен, если результат равен нулю
;1547    ;       V - случайное значение (не присвоена никакая функция)
;1548    ;       C - случайное значение (не присвоена никакая функция)
;1549    ;
;1550    ;   MCOM - занесение дополнения до единицы (инверсии) первого операнда
;1551    ;          на место второго операнда
;1552    ;
;1553    ;       N - бит знака
;1554    ;       Z - установлен, если результат равен нулю
;1555    ;       V - случайное значение (не присвоена никакая функция)
;1556    ;       C - случайное значение (не присвоена никакая функция)
;1557    ;
;1558    ;   MNEG - занесение дополнения до двух первого операнда на место второго
;1559    ;          операнда
;1560    ;
;1561    ;       N - бит знака
;1562    ;       Z - установлен, если результат нулевой
;1563    ;       V - арифметическое переполнение
;1564    ;       C - перенос
;1565    ;
;1566    MOV Q  TO LS[]              "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/MOV.Q.LS>,0FF]>,-2]>"
;1567    MOV Q  TO WR[]              "OPC1/EXTENDED,B.ADRS/@1,XDP/<.AND[<SDP/MOV.Q.WR>,03F]>"
;1568    MOV LS[] TO Q               "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/MOV.LS.Q>,0FF]>,-2]>"
;1569    MOV Q TO WR[] XCHG TO LS[]  "OPC/BASIC,D.ADRS/@2,B.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/MOV.Q.WR&WR.LS>,0FF]>,-2]>"
;1570    MOV IB.DATA TO OS           "OPC2/DECODE,IFUNC/SPEC,R.DST/NOP,IB.REQ/IB.REQ,JCTL/NO.JUMP.TST,LD.OS/LOAD.OS"
;1571    MOV CM.IB.DATA TO OS        "OPC2/DECODE,IFUNC/SPEC,R.DST/NOP,IB.REQ/NOP,JCTL/NO.JUMP.TST,LD.OS/LOAD.OS"
;1572    MOV MEM.DATA TO WR[]        "OPC1/MOVE,B.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.MEM.WR>,3F]>,-3]>,XD.ADRS/WR.BAC
;1573    MOV MEM.DATA TO WR[] XCHG TO LS[]    "OPC1/MOVE,B.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.MEM.WR>,3F]>,-3]>,XD.AD
;1574    MOV MEM.DATA TO LS[]        "OPC1/MOVE,XD.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.MEM.LS>,3F]>,-3]>"
;1575    MOV LS[] TO WR[]    "OPC1/MOVE,XD.ADRS/@1,B.ADRS/@2,MDP/<.SHIFT[<.AND[<SDP/MOV.LS.WR>,3F]>,-3]>"
;1576    MOV WR[] TO LS[]    "OPC1/MOVE,B.ADRS/@1,XD.ADRS/@2,MDP/<.SHIFT[<.AND[<SDP/MOV.WR.LS>,3F]>,-3]>"
;1577    MOV WR[] TO WR[]    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.PLUS.0.TO.WR>,03F]>"
;1578    MOV WR[] TO Q       "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.OR.WR.Q>,03F]>"
;1579    MOV XWR[] TO Q      "OPC1/MOVE,XB.ADRS/@1,XD.ADRS/0,MDP/<.SHIFT[<.AND[<SDP/MOV.XWR.Q>,3F]>,-3]>"
;1580    MOV FPA TO LS[]     "OPC1/MOVE,XD.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.ACC.LS>,3F]>,-3]>"
;1581    MOV PORT TO LS[]    "OPC1/MOVE,XD.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.ACC.LS>,3F]>,-3]>,CC/DT(LONG)&HOLD.ALU.
;1582    MCOM WR[] TO LS[]   "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.COM.LS>,0FF]>,-2]>"
;1583    MCOM LS[] TO WR[]   "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.COM.WR>,0FF]>,-2]>"
;1584    MNEG WR[] TO LS[]   "OPC/BASIC,B.ADRS/@1,D.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/WR.NEG.LS>,0FF]>,-2]>"
;1585    MNEG LS[] TO WR[]   "OPC/BASIC,D.ADRS/@1,B.ADRS/@2,DP/<.SHIFT[<.AND[<SDP/LS.NEG.WR>,0FF]>,-2]>"
;1586    MNEG WR[] TO WR[]   "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.NEG.WR>,03F]>"
;1587    MCOM WR[] TO WR[]   "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.COM.WR>,03F]>"
;1588    MINC WR[] TO WR[]   "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.INC.WR>,03F]>"
;1589    MDEC WR[] TO WR[]   "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,XDP/<.AND[<SDP/WR.DEC.WR>,03F]>"
;1590    MNEG Q TO LS[]      "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/Q.NEG.LS>,0FF]>,-2]>"
;1591    ;
;1592    ;
;1593    ;       Операции с одним операндом
```

```
;1594    ;
;1595    ;  Инструкции этого формата выполняют требуемую операцию над заданным
;1596    ;  операндом:
;1597    ;                очистку
;1598    ;                отрицание
;1599    ;                инкрементирование
;1600    ;                декрементирование
;1601    ;                дополнение
;1602    ;                проверку
;1603    ;
;1604    ;   Во время инструкций с одним операндом коды условий (CC) АЛУ можно
;1605    ;   загрузить с использованием в микрослове макрооператора
;1606    ;
;1607    ;              DT(SIZE)&SET.ALU.CC
;1608    ;
;1609    ;   Далее следует описание кодов условий АЛУ каждой функции:
;1610    ;
;1611    ;  CLR - запоминание нуля в операнде
;1612    ;
;1613    ;      N - бит знака
;1614    ;      Z - установлен, если результат равен нулю
;1615    ;      V - случайное значение (не присвоена никакая функция)
;1616    ;      C - случайное значение (не присвоена никакая функция)
;1617    ;
;1618    ;  NEG - выполнение для операнда дополнения до двух
;1619    ;
;1620    ;      N - бит знака
;1621    ;      Z - установлен, если результат равен нулю
;1622    ;      V - арифметическое переполнение
;1623    ;      C - перенос
;1624    ;
;1625    ;  INC - прибавление единицы к операнду
;1626    ;
;1627    ;      N - бит знака
;1628    ;      Z - установлен, если результат равен нулю
;1629    ;      V - арифметическое переполнение
;1630    ;      C - перенос
;1631    ;
;1632    ;  DEC - вычитание единицы из операнда
;1633    ;
;1634    ;      N - бит знака
;1635    ;      Z - установлен, если результат равен нулю
;1636    ;      V - арифметическое переполнение
;1637    ;      C - перенос
;1638    ;
;1639    ;  COM - выполнение для операнда дополнения до единицы (XNOR с нулем)
;1640    ;
;1641    ;      N - бит знака
;1642    ;      Z - установлен, если результат равен нулю
;1643    ;      V - случайное значение (не присвоена никакая функция)
;1644    ;      C - случайное значение (не присвоена никакая функция)
;1645    ;
;1646    ;  TST - прибавление нуля к операнду с целью получения кодов условий
;1647    ;
;1648    ;      N - бит знака
```

```
;1649    ;       Z - установлен,если результат равен нулю
;1650    ;       V - арифметическое переполнение (в данном случае всегда нуль)
;1651    ;       C - перенос (в данном случае нуль)
;1652    ;
;1653    CLR Q           "OPC1/EXTENDED,A.ADRS/0,B.ADRS/0,XDP/<.AND[<SDP/WR.XOR.WR.Q>,03F]>"
;1654    CLR LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/CLR.LS>,0FF]>,-2]>"
;1655    CLR WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.XOR.WR>,03F]>"
;1656    NEG Q           "OPC1/EXTENDED,XDP/<.AND[<SDP/NEG.Q>,03F]>"
;1657    NEG LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/NEG.LS>,0FF]>,-2]>"
;1658    NEG WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.NEG.WR>,03F]>"
;1659    INC Q           "OPC1/EXTENDED,XDP/<.AND[<SDP/INC.Q>,03F]>"
;1660    INC LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/INC.LS>,0FF]>,-2]>"
;1661    INC WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.INC.WR>,03F]>"
;1662    DEC Q           "OPC1/EXTENDED,XDP/<.AND[<SDP/DEC.Q>,03F]>"
;1663    DEC LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/DEC.LS>,0FF]>,-2]>"
;1664    DEC WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.DEC.WR>,03F]>"
;1665    COM Q           "OPC1/EXTENDED,XDP/<.AND[<SDP/COM.Q>,03F]>"
;1666    COM LS[]        "OPC/BASIC,D.ADRS/@1,DP/<.SHIFT[<.AND[<SDP/COM.LS>,0FF]>,-2]>"
;1667    COM WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.COM.WR>,03F]>"
;1668    TST WR[]        "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,XDP/<.AND[<SDP/WR.PLUS.0.TO.WR>,03F]>"
;1669    TST Q           "OPC1/EXTENDED,XDP/<.AND[<SDP/Q.PLUS.0>,03F]>"
;1670    NOP             "OPC/BASIC,D.ADRS/0,B.ADRS/0,DP/<.SHIFT[<.AND[<SDP/Q.CMP.LS>,0FF]>,-2]>"
;1671    ;
;1672    ;    Макрооператоры для операций сдвига WR[B] и/или Q-регистра
;1673    ;
;1674    ;  Во время операций сдвига можно загрузить коды условий (CC) АЛУ использованием
;1675    ;  в микрослове макрооператора
;1676    ;
;1677    ;       DT(SIZE)&SET.ALU.CC
;1678    ;
;1679    ;  Далее следует описание кодов условий АЛУ для каждой функции:
;1680    ;
;1681    ;  ROR WR[] - циклический сдвиг 32-битового значения на одну позицию вправо
;1682    ;
;1683    ;       N - знак входных данных
;1684    ;       Z - установлен, если входные данные равны нулю
;1685    ;       V - случайное значение (не присвоена никакая функция)
;1686    ;       C - случайное значение (не присвоена никакая функция)
;1687    ;
;1688    ;  ROL WR[] - циклический сдвиг 32-битового значения на одну позицию влево
;1689    ;
;1690    ;       N - бит знака входных данных
;1691    ;       Z - установлен,если входные данные равны нулю
;1692    ;       V - случайное значение (не присвоена никакая функция)
;1693    ;       C - случайное значение (не присвоена никакая функция)
;1694    ;
;1695    ;  ASHR WR[] - сдвиг 32-битового значения на одну позицию вправо с расширением знака
;1696    ;
;1697    ;       N - знак входных данных
;1698    ;       Z - установлен, если входные данные равны нулю
;1699    ;       V - случайное значение (не присвоена никакая функция)
;1700    ;       C - случайное значение (не присвоена никакая функция)
;1701    ;
;1702    ;  ASHL WR[] - сдвиг 32-битового значения на одну позицию влево с установкой нуля в
;1703    ;              младшем бите
```

```
;1704    ;
;1705    ;       N - знак входных данных
;1706    ;       Z - установлен, если входные данные нулевые
;1707    ;       V - случайное значение (не присвоена никакая функция)
;1708    ;       C - случайное значение (не присвоена никакая функция)
;1709    ;
;1710    ;   ASHRC WR[].Q - сдвиг WR и Q, как 64-битовых данных, на одну позицию влево с
;1711    ;                  расширением знака
;1712    ;
;1713    ;       N - знак входных данных в WR[]
;1714    ;       Z - установлен, если входные данные в WR[] равны нулю
;1715    ;       V - случайное значение (не присвоена никакая функция)
;1716    ;       C - случайное значение (не присвоена никакая функция)
;1717    ;
;1718    ;   RORC WR[].Q - циклический сдвиг WR и Q, как 64-битовых данных, на одну
;1719    ;                 позицию вправо
;1720    ;
;1721    ;       N - знак входных данных в WR[]
;1722    ;       Z - установлен, если входные данные в WR[] равны нулю
;1723    ;       V - случайное значение (не присвоена никакая функция)
;1724    ;       C - случайное значение (не присвоена никакая функция)
;1725    ;
;1726    ;   ASHLC WR[].Q - сдвиг WR и Q, как 64-битовых данных, на одну позицию влево
;1727    ;                  с занесением нуля
;1728    ;
 1729    ;       N - знак входных данных в WR[]
;1730    ;       Z - установлен, если входные данные в WR[] равны нулю
;1731    ;       V - случайное значение (не присвоена никакая функция)
;1732    ;       C - случайное значение (не присвоена никакая функция)
;1733    ;
;1734    ;   ROLC WR[].Q - циклический сдвиг WR и Q, как 64-битовых данных, на одну
;1735    ;                 позицию влево
;1736    ;
;1737    ;       N - знак входных данных в WR[]
;1738    ;       Z - установлен, если входные данные в WR[] нулевые
;1739    ;       V - случайное значение (никакая функция не присвоена)
;1740    ;       C - случайное значение (никакая функция не присвоена)
;1741    ;
;1742    ;   SHL2 WR[] - сдвиг 32-битового значения на 2 позиции влево с занесением нулей в
;1743    ;               младшие биты
;1744    ;
;1745    ;       N - знак входных данных WR[]+WR[]
;1746    ;       Z - установлен , если входные данные WR[]+WR[] равны нулю
;1747    ;       V - переполнение по входным данным WR[]+WR[]
;1748    ;       C - перенос из входных данных WR[]+WR[]
;1749    ;
;1750    ROR WR[]        "OPC1/EXTENDED,B.ADRS/@1,SI/ROT.WR,XDP/<.AND[<SDP/ASHR>,03F]>"
;1751    ROL WR[]        "OPC1/EXTENDED,B.ADRS/@1,SI/ROT.WR,XDP/<.AND[<SDP/ASHL>,03F]>"
;1752    ASHR WR[]       "OPC1/EXTENDED,B.ADRS/@1,SI/ASH.WR,XDP/<.AND[<SDP/ASHR>,03F]>"
;1753    ASHL WR[]       "OPC1/EXTENDED,B.ADRS/@1,SI/ASH.WR,XDP/<.AND[<SDP/ASHL>,03F]>"
;1754    ASHRC WR[].Q    "OPC1/EXTENDED,B.ADRS/@1,SI/ASH.WR.Q,XDP/<.AND[<SDP/ASHRC>,03F]>"
;1755    RORC WR[].Q     "OPC1/EXTENDED,B.ADRS/@1,SI/ROT.WR.Q,XDP/<.AND[<SDP/ASHRC>,03F]>"
;1756    ASHLC WR[].Q    "OPC1/EXTENDED,B.ADRS/@1,SI/ASH.WR.Q,XDP/<.AND[<SDP/ASHLC>,03F]>"
;1757    ROLC WR[].Q     "OPC1/EXTENDED,B.ADRS/@1,SI/ROT.WR.Q,XDP/<.AND[<SDP/ASHLC>,03F]>"
;1758    SHL2 WR[]       "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@1,SI/2,XDP/<.AND[<SDP/SHL2>,03F]>"
```

```
;1759   COND.ADD&SHIFT WR[] TO WR[].Q    "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,SI/MUL:SHF,XDP/<.AND[<SDP/COND.ADD&SHIFT>
;1760   COND.SUB&SHIFT WR[] FROM WR[].Q "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,SI/MUL.SHF,
;1761                                    XDP/<.AND[<SDP/COND.SUB&SHIFT>,03F]>"
;1762   UNSIGNED.MUL WR[] TO WR[].Q      "OPC1/EXTENDED,A.ADRS/@1,B.ADRS/@2,SI/UMUL.SHF,
;1763                                    XDP/<.AND[<SDP/COND.ADD&SHIFT>,03F]>"
;1764   SHR WR[]          "OPC1/EXTENDED,B.ADRS/@1,SI/SHF.WR.Q,XDP/<.AND[<SDP/ASHR>,03F]>"
;1765   SHL WR[]          "ASHL WR[@1]"
;1766   SHRC WR[].Q       "OPC1/EXTENDED,B.ADRS/@1,SI/SHF.WR.Q,XDP/<.AND[<SDP/ASHRC>,03F]>"
;1767   SHLC WR[].Q       "ASHLC WR[].Q"
;1768   ;
;1769   ;   Макрооператор запроса памяти
;1770   ;
;1771   ; *  Этот макрооператор не предназначен для общего использования.        *
;1772   ; *  Он предназначен только для использования в следующем макрооператоре *
;1773   ;
;1774   MEM.FUNC[]                "M.FUNC2/<.SHIFT[<MEM.FUNC/@1>,-2]>,M.FUNC1/<.AND[<MEM.FUNC/@1>,3]>"
;1775   MEM.REQ[] ADRS[] DT[]     "OPC2/MEM.REQ,MEM.FUNC[@1],D.ADRS/@2,MDT/@3"
;1776   ;
;1777   WRITE.MEM LS[]            "OPC1/MOVE,XD.ADRS/@1,MDP/<.SHIFT[<.AND[<SDP/MOV.LS.MEM>,3F]>,-3]>"
;1778   ;
;1779   ;
;1780   ;  Макрооператоры для разных операций
;1781   ;
;1782   ;  В языке микропрограммирования на уровне схем существует необходимость в раз-
;1783   ;  личных уникальных действиях для функционирования. Они задаются инструкциями
;1784   ;  MISC. Большинство функций выполняются указанием запроса внутри макрооперато-
;1785   ;  ров MISC[] и MISC2[].
;1786   ;
;1787   MISC []                   "OPC2/MISC,MISC.1/@1,MISC.2/NOP,MISC.PORT/MISC"
;1788   MISC2 []                  "OPC2/MISC,MISC.1/NOP,MISC.2/@1,MISC.PORT/MISC"
;1789   MISC [] WR[]              "MISC [@1],MISC.WRL/0,MISC.WRR/@2"
;1790   MISC2 [] WR[]             "MISC2 [@1],MISC.WRL/0,MISC.WRR/@2"
;1791   PORT.CONTROL []           "OPC2/MISC,MISC.PORT/PORT,CNTL.FUNC/@1,PORT.SELECT/IDC,PORT.FUNC/CONTROL"
;1792   PORT.WRITE PORT.ADRS[] WITH WR[]   "OPC2/MISC,MISC.PORT/PORT,R.W.FUNC/@1,
;1793                                       PORT.SELECT/IDC,PORT.FUNC/WRITE,MISC.WRL/0,MISC.WRR/@2"
;1794   PORT.READ PORT.ADRS[]     "OPC2/MISC,MISC.PORT/PORT,R.W.FUNC/@1,PORT.SELECT/IDC,PORT.FUNC/READ"
;1795   ;
;1796   ;  В нескольких случаях разные функции вызываются как уникальные функции набора
;1797   ;  языка микропрограммирования. Они перечислены ниже.
;1798   ;
;1799   ;  Следующие макрооператоры пересылают данные во внешние ускорители.
;1800   ;
;1801   ASSERT.DA.AND.PASS.WR0          "MISC2 [ASSERT.DA.AND.PASS.WR] WR[0]"
;1802   ASSERT.DA.AND.PASS.WR1          "MISC2 [ASSERT.DA.AND.PASS.WR] WR[1]"
;1803   ;
;1804   ;  Макрооператоры регистра STATE - возможные изменения битов STATE задаются как
;1805   ;  индивидуальные функции.
;1806   ;
;1807   CLR.STATE.ZERO                   "OPC2/MISC,MISC.1/CLR.S0,MISC.2/NOP,MISC.PORT/MISC"
;1808   CLR.STATE.ONE                    "OPC2/MISC,MISC.1/CLR.S1,MISC.2/NOP,MISC.PORT/MISC"
;1809   SET.STATE.ZERO                   "OPC2/MISC,MISC.1/SET.S0,MISC.2/NOP,MISC.PORT/MISC"
;1810   SET.STATE.ONE                    "OPC2/MISC,MISC.1/SET.S1,MISC.2/NOP,MISC.PORT/MISC"
;1811   SET.STATE.ZERO&CLR.STATE.ONE     "OPC2/MISC,MISC.1/CLR.S1&SET.S0,MISC.2/NOP,MISC.PORT/MISC"
;1812   SET.STATE.ONE&CLR.STATE.ZERO     "OPC2/MISC,MISC.1/SET.S1&CLR.S0,MISC.2/NOP,MISC.PORT/MISC"
;1813   CLR.STATE.ONE&CLR.STATE.ZERO     "OPC2/MISC,MISC.1/CLR,MISC.2/NOP,MISC.PORT/MISC"
```

```
;1814   SET.BACKUP.MASK.VALID          "OPC2/MISC,MISC.1/SET.BACKUP.MASK.VALID,MISC.2/NOP,MISC.PORT/MISC"
;1815   ;
;1816   ;  Макрооператоры размера контекста и кодов условий
;1817   ;
;1818   DT(SIZE)&MAP.ALU.CC.TO.PSL     "CC/DT(SIZE)&MAP.ALU.CC.TO.PSL"
;1819   DT(LONG)&SET.ALU.CC            "CC/DT(LONG)&SET.ALU.CC"
;1820   DT(SIZE)&SET.ALU.CC            "CC/DT(SIZE)&SET.ALU.CC"
;1821   ; .
;1822   ;  Макрооператоры управления переходами и микросеквенсером
;1823   ;
;1824   ;  Следующие макрооператоры используются для управления прохождением программ
;1825   ;
;1826   ;  JMP []          - безусловный переход к новому адресу
;1827   ;  JMP.OS []       - безусловный переход к адресу, полученному в результате логического
;1828   ;                    обьединения по "ИЛИ" адреса в инструкции и регистра OS
;1829   ;  JMP.IF[] TO []  - переход к новому адресу только если выполнено условие перехода
;1830   ;  JSR []          - безусловный переход к новому адресу и занесение адреса возврата в стек
;1831   ;                    секвенсера
;1832   ;  JSR.OS []       - переход к адресу, полученному в результате логического "ИЛИ" адреса
;1833   ;                    в инструкции и регистра OS. Адрес возврата заносится также в стек
;1834   ;                    секвенсера
;1835   ;  RETURN          - переход к адресу из вершины стека секвенсера и продвижение стека
;1836   ;                    вперед
;1837   ;  RETURN+1        - переход к адресу из вершины стека секвенсера плюс один и продвижение
;1838   ;                    стека вперед
;1839   ;  RETURN+1.IF(MEM.REF.OK) - если в самом последнем запросе памяти не обнаружено
;1840   ;                    ошибок, тогда переход к адресу из вершины стека секвенсера плюс один
;1841   ;                    и продвижение стека вперед. Если была ошибка, тогда пропускается
;1842   ;                    следующая инструкция
;1843   ;  LOOP.IF(NEQ)    - если бит Z АЛУ равен нулю (последнее значение, вычисленное с
;1844   ;                    занесением в ALU.CC, не показывает результата, равного нулю), то
;1845   ;                    переход к адресу из вершины стека секвенсера. Стек не продвигается.
;1846   ;                    Если бит Z АЛУ равен единице, тогда продолжением является следующая
;1847   ;                    инструкция (UPC+1) и стек продвигается вперед
;1848   ;  LOOP.IF(GEQU)   - если бит "C" АЛУ равен единице (последнее значение, вычисленное с
;1849   ;                    занесением в ALU.CC, было больше или равно нулю), то переход к
;1850   ;                    адресу из вершины стека секвенсера, стек не продвигается. Если бит "C"
;1851   ;                    АЛУ равен нулю, то продолжением является следующая инструкция (UPC+1)
;1852   ;                    и стек продвигается вперед
;1853   ;  LOOP.IF(NO INTERRUPT AND NO SYNC) - если условие прерывания не является истинным
;1854   ;                    и условие синхронизации с ускорителем не является истинным, тогда
;1855   ;                    переход к адресу из вершины стека секвенсера, стек не продвигается.
;1856   ;                    Если условие прерывания истинно, то продолжением является следующая
;1857   ;                    инструкция (UPC+1) и стек секвенсера не продвигается. Если является
;1858   ;                    истинным условие синхронизации с ускорителем (SYNC), то продолжением
;1859   ;                    является следующая инструкция и стек секвенсера продвигается вперед
;1860   ;  SKIP.IF[]       - если условие пропуска удовлетворено, то переход к UPC+2, иначе
;1861   ;                    переход к UPC+1
;1862   ;
;1863   JMP []                         "OPC2/JUMP,JUMP.ADRS/@1,JCTL/JUMP"
;1864   JMP.OS []                      "OPC2/JUMP,JUMP.ADRS/@1,OS/WITH.OS,JCTL/JUMP"
;1865   JMP.IF[] TO []                 "OPC2/JUMP,JCTL/@1,JUMP.ADRS/@2"
;1866   JSR []                         "OPC2/JUMP,JCTL/JSR,JUMP.ADRS/@1"
;1867   JSR.OS []                      "OPC2/JUMP,JCTL/JSR,JUMP.ADRS/@1,OS/WITH.OS"
;1868   RETURN                         "SCTL/RETURN"
```

```
;1869   RETURN+1                          "SCTL/RETURN+1"
;1870   RETURN+1.IF(MEM.REF.OK)           "SCTL/RET.NOERR"
;1871   LOOP.IF(NEQ)                      "SCTL/JMP.Z.CLR"
;1872   LOOP.IF(GEQU)                     "SCTL/JMP.C.SET"
;1873   LOOP.IF(NO INTERRUPT AND NO SYNC) ELSE SKIP.IF(SYNC)   "SCTL/LOOP.IF.NO.I.AND.SYNC"
;1874   SKIP.IF[]                         "SCTL/@1"
;1875   SKIP                              "SCTL/SKIP"
;1876   ;
;1877   ;  Макрооператоры потока инструкций
;1878   ;
;1879   ; * Этот макрооператор не предназначен для общего использования.          *
;1880   ; * Он предназначен только для использования в следующих макрооператорах *
;1881   ;
;1882   DEC[ ]                    "OPC2/DECODE,DEC.ADRS/<.SHIFT[<JUMP.ADRS/@1>,-8]>"
;1883   ;
;1884   DECODE.SPEC ADRS[]        "DEC[@1],IFUNC/SPEC,BPC/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,LD.OS/LOAD.OS,R.DST/ENAB,OPC
;1885   DECODE.ASRC ADRS[]        "DEC[@1],IFUNC/ASRC,BPC/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,LD.OS/LOAD.OS,RDST/ENAB,OPC.
;1886   DECODE.ESRC ADRS[]        "DEC[@1],IFUNC/ESRC,BPC/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,LD.OS/LOAD.OS,R.DST/ENAB,OPC
;1887   DECODE.VSRC ADRS[]        "DEC[@1],IFUNC/VSRC,BPC/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,LD.OS/LOAD.OS,R.DST/ENAB,OPC
;1888   DECODE.FLOAT ADRS[]       "DEC[@1],IFUNC/FLOAT,BPC/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,LD.OS/LOAD.OS,R.DST/ENAB,OP
;1889   DECODE.OPC1 ADRS[]        "DEC[@1],IFUNC/VAX.IRD,R.DST/NOP,CM.HI/LOW.BYTE,IB.REQ/IB.REQ,OPC.SPEC/VAX.IRD"
;1890   EXEC.DISPATCH ADRS[]      "DEC[@1],IFUNC/VAX.EXEC,IB.REQ/NOP,R.DST/NOP,CM.HI/LOW.BYTE,JCTL/JSR,
;1891                              OPC.SPEC/VAX.EXEC"
;1892   DECODE.CM.OPC ADRS[]      "DEC[@1],IFUNC/CM.IRD,CM.HI/HI.BYTE,OPC.SPEC/CM.IRD,LD.OS/LOAD.OS"
;1893   DECODE.CM.DST ADRS[]      "DEC[@1],IFUNC/CM.DST,OPC.SPEC/CM.DST,LD.OS/LOAD.OS,R.DST/ENAB"
;1894   DECODE.CM.SINGLE ADRS[ ]      "DEC[@1],IFUNC/CM.SINGLE,OPC.SPEC/CM.SINGLE"
;1895   CM.EXEC ADRS[]            "DEC[@1],IFUNC/CM.EXEC,JCTL/JUMP,OPC.SPEC/CM.EXEC,LD.OS/LOAD.OS"
;1896   SAVE.PC WR0               "BPC/SAVE.PC"
;1897   SAVE.PC WR4               "BPC/SAVE.PC"
;1898   SAVE.CM.PC WR0            "BPC/SAVE.PC"
;1899   SAVE.CM.PC WR4            "BPC/SAVE.PC"
;1900   ;
;1901   ;  Эти условия пропуска используются микроинструкцией DECODE
;1902   ;
;1903   SKIP.IF(IB VALID)         "JCTL/NO.JUMP.TST"
;1904   JMP.IF(IB VALID)          "JCTL/JUMP.VALID"
;1905   JSR.IF(IB VALID)          "JCTL/JSR.VALID"
;1906   ;
;1907   ;  Эти условия пропуска должны использоваться с макрооператорами режима
;1908   ;  совместимости
;1909   ;
;1910   JMP.TO []                 "JCTL/JUMP,DEC[@1]"
;1911   JSR.TO []                 "JCTL/JSR,DEC[@1]"
;1912
;1913   ;       МАКРООПРЕДЕЛЕНИЯ ДЛЯ ДИАГНОСТИКИ
;1914   ;
;1915   LS.DATA.WORD/=<15:0>
;1916   UPPER.BYTE/=<23:16>
;1917   ;
;1918   WORD[]          "UPPER.BYTE/0,LS.DATA.WORD/@1"
;1919   COUNT.LS[]      "UPPER.BYTE/0,LS.DATA.WORD/@1"
;1920   COUNT.MM[]      "UPPER.BYTE/1,LS.DATA.WORD/@1"
;1921   START.ADR[]     "UPPER.BYTE/0,LS.DATA.WORD/@1"
;1922   ;
;1923   ASCII.LIT/=<15:0>
```

```
                ;1924           CA=4341
                ;1925           CB=4342
                ;1926           CC=4343
                ;1927           CD=4344
                ;1928           CE=4345
                ;1929           CF=4346
                ;1930           CG=4347
                ;1931           CH=4348
                ;1932     ;
                ;1933     ASCII []      "UPPER.BYTE/0,ASCII.LIT/@1"
                ;1934     ;
```

```
;1935   .PAGE "ОПРЕДЕЛЕНИЯ ПОЛЕЙ УПРАВЛЯЮЩЕГО МИКРОСЛОВА ПУТЕЙ ДАННЫХ "
;1936   .BIN
;1937   ;
;1938   ;  Этот раздел относится к ПЗУ и ПМЛ управления операцией (функция, источник (R),
;1939   ;  приемник (S), входной перенос) микропроцессоров К1804ВС1 платы DAP. ПЗУ УПР.
;1940   ;  ИСТ.ПРИЕМН.АЛУ и ПМЛ УПР. ФУНКЦИЙ АЛУ рассматриваются как одна управляющая
;1941   ;  память на 512 адресов с шириной слова 10 битов
;1942   ;
;1943   .CCODE
;1944   SDP/=<8:0>,.ADDRESS,.VALIDITY=0
;1945   ;
;1946   ; Это поле соответствует входам управления функцией АЛУ IN5,IN4 и IN3
;1947   ;
;1948   ALU/=<2:0>,.DEFAULT=<ALU/R.OR.S>
;1949   ;
;1950         R.PLUS.S=0                ; сложение R и S
;1951         S.MINUS.R=1               ; вычитание R из S
;1952         R.MINUS.S=2               ; вычитание S из R
;1953         R.OR.S=3                  ; "ИЛИ" для R и S
;1954         R.AND.S=4                 ; "И" для R и S
;1955         NOTR.AND.S=5              ; инвертирование R и затем "И" с S
;1956         R.XOR.S=6                 ; исключающее "ИЛИ" для R и S
;1957         R.XNOR.S=7                ; исключающее "ИЛИ" для R и S, а затем
;1958                                   ; инвертирование результата
;1959   ;
;1960   ;  Это поле соответствует входам управления приемником АЛУ
;1961   ;  IN8,IN7 и IN6
;1962   ;
;1963   DST/=<5:3>,.DEFAULT=<DST/NOP>
;1964   ;
;1965         LOAD.Q=0                  ; загрузка Q-регистра
;1966         NOP=1                     ; сохранение всех регистров
;1967         WRITE.B.A=2               ; запись во внутренее ЗУ по адресу на B-порте
;1968                                   ; и вывод из внутреннего ЗУ по адресу на A-порте
;1969         WRITE.B.F=3               ; запись во внутреннюю память по B-порту и
;1970                                   ; вывод АЛУ
;1971         RSHF.RAM.Q=4              ; сдвиг вправо внутренней памяти и Q
;1972         RSHF.RAM=5                ; сдвиг вправо внутренней памяти
;1973         LSHF.RAM.Q=6              ; сдвиг влево внутренней памяти и Q
;1974         LSHF.RAM=7                ; сдвиг влево внутренней памяти
;1975   ;
;1976   ;  Это поле соответствует входам управления источником IN2, IN1 и IN0
;1977   ;
;1978   SRC/=<8:6>,.DEFAULT=<SRC/D.0>
;1979   ;
;1980         0.Q=2                     ; R=0    S=Q
;1981         0.B=3                     ; R=0    S=B
;1982         A.Q=0                     ; R=A    S=Q
;1983         A.B=01                    ; R=A    S=B
;1984         D.Q=06                    ; R=D    S=Q
;1985         D.0=7                     ; R=D    S=0
;1986         0.A=4                     ; R=0    S=A
;1987         D.A=5                     ; R=D    S=A
;1988   ;
;1989   ;  Входной перенос
```

```
;1990   ;
;1991   CIN/=<9>,.DEFAULT=0
;1992   ;
;1993         NO.CIN=0                  ; входной перенос АЛУ отсутствует
;1994         CIN=1                     ; имеется входной перенос в АЛУ
;1995   ;
;1996   ;
;1997   ; .
```

```
                    ;1998   .PAGE  "СОДЕРЖИМОЕ УПРАВЛЯЮЩЕЙ ПАМЯТИ ПУТЕЙ ДАННЫХ "
                    ;1999   .SEQUENTIAL
                    ;2000   ;
                    ;2001   ;  Здесь представлено содержимое управляющей памяти путей данных.
                    ;2002   ;
                    ;2003   ;  Следующие ячейки используются микроинструкцией DECODE.IS. Эта микроинструкция выпол-
                    ;2004   ;  няет операцию PC+1 и резервирует копию счетчика команд PC во внутренней
                    ;2005   ;  памяти
                    ;2006   ;
                    ;2007   000:
                    ;2008   DECODE.IS:
C 0000,  3D8        ;2009           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0001,  3D8        ;2010           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0002,  3D8        ;2011           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0003,  3D8        ;2012           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0004,  3D8        ;2013           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0005,  3D8        ;2014           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0006,  3D8        ;2015           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0007,  3D8        ;2016           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0008,  3D8        ;2017           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0009,  3D8        ;2018           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 000A,  3D8        ;2019           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0008,  3D8        ;2020           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 000C,  3D8        ;2021           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 000D,  3D8        ;2022           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 000E,  3D8        ;2023           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 000F,  3D8        ;2024           SRC/D.0,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
                    ;2025
                    ;2026   ;  PC не резервируется
                    ;2027   ;
C 0010,  3C8        ;2028           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0011,  3C8        ;2029           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0012,  3C8        ;2030           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0013,  3C8        ;2031           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0014,  3C8        ;2032           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0015,  3C8        ;2033           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0016,  3C8        ;2034           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0017,  3C8        ;2035           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0018,  3C8        ;2036           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 0019,  3C8        ;2037           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001A,  3C8        ;2038           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001B,  3C8        ;2039           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001C,  3C8        ;2040           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001D,  3C8        ;2041           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001E,  3C8        ;2042           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 001F,  3C8        ;2043           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
                    ;2044   ;
                    ;2045   ;
                    ;2046   ;  Эти адреса используются микроинструкцией JUMP
                    ;2047   ;
                    ;2048   ;  Следующие управляющие коды гарантируют, что во время операций JUMP или JSR
                    ;2049   ;  не будут разрушены никакие данные внутри К1804ВС1
                    ;2050   ;
                    ;2051   020:
                    ;2052   JUMP:
```

```
C 0020, 3CB      ;2053          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0021, 3CB      ;2054          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0022, 3CB      ;2055          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0023, 3CB      ;2056          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0024, 3CB      ;2057          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0025, 3CB      ;2058          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0026, 3CB      ;2059          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0027, 3CB      ;2060          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0028, 3CB      ;2061          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0029, 3CB      ;2062          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002A, 3CB      ;2063          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002B, 3CB      ;2064          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002C, 3CB      ;2065          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002D, 3CB      ;2066          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002E, 3CB      ;2067          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 002F, 3CB      ;2068          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0030, 3CB      ;2069          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0031, 3CB      ;2070          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0032, 3CB      ;2071          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0033, 3CB      ;2072          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0034, 3CB      ;2073          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0035, 3CB      ;2074          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0036, 3CB      ;2075          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0037, 3CB      ;2076          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0038, 3CB      ;2077          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 0039, 3CB      ;2078          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003A, 3CB      ;2079          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003B, 3CB      ;2080          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003C, 3CB      ;2081          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003D, 3CB      ;2082          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003E, 3CB      ;2083          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
C 003F, 3CB      ;2084          SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/CIN
                 ;2085    ;
                 ;2086    ;
                 ;2087    ;  Эти ячейки используются микроинструкцией MISC
                 ;2088    ;
                 ;2089    ;  Следующие управляющие коды гарантируют, что во время выполнения инструкций
                 ;2090    ;  MISC не будут разрушены никакие данные внутри K1804BC1
                 ;2091    ;
                 ;2092    040:
                 ;2093    MISC:
C 0040, 313      ;2094          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0041, 313      ;2095          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0042, 313      ;2096          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0043, 313      ;2097          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0044, 313      ;2098          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0045, 313      ;2099          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0046, 313      ;2100          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0047, 313      ;2101          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0048, 313      ;2102          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0049, 313      ;2103          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 004A, 313      ;2104          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 004B, 313      ;2105          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 004C, 313      ;2106          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 004D, 313      ;2107          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
```

```
C 004E, 313     ;2108          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 004F, 313     ;2109          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0050, 313     ;2110          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0051, 313     ;2111          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0052, 313     ;2112          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0053, 313     ;2113          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0054, 313     ;2114          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0055, 313     ;2115          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0056, 313     ;2116          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0057, 313     ;2117          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0058, 313     ;2118          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0059, 313     ;2119          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005A, 313     ;2120          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005B, 313     ;2121          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005C, 313     ;2122          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005D, 313     ;2123          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005E, 313     ;2124          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 005F, 313     ;2125          SRC/0.A,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
                ;2126   ;
                ;2127   ;
                ;2128   ;  Эти адреса используются микроинструкцией MEM.REQ
                ;2129   ;
                ;2130   ;  Этой инструкцией вызывается загрузка WR[5] виртуальным адресом обращения к
                ;2131   ;  памяти.
                ;2132   ;
                ;2133   060:
                ;2134   MEM.REQ:
C 0060, 3DB     ;2135          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0061, 3DB     ;2136          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0062, 3DB     ;2137          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0063, 3DB     ;2138          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0064, 3DB     ;2139          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0065, 3DB     ;2140          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0066, 3DB     ;2141          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0067, 3DB     ;2142          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0068, 3DB     ;2143          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0069, 3DB     ;2144          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006A, 3DB     ;2145          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006B, 3DB     ;2146          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006C, 3DB     ;2147          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006D, 3DB     ;2148          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006E, 3DB     ;2149          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 006F, 3DB     ;2150          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0070, 3DB     ;2151          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0071, 3DB     ;2152          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0072, 3DB     ;2153          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0073, 3DB     ;2154          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0074, 3DB     ;2155          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0075, 3DB     ;2156          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0076, 3DB     ;2157          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0077, 3DB     ;2158          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0078, 3DB     ;2159          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 0079, 3DB     ;2160          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 007A, 3DB     ;2161          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 007B, 3DB     ;2162          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
```

```
C 007C, 3DB      ;2163          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 007D, 3DB      ;2164          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 007E, 3DB      ;2165          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 007F, 3DB      ;2166          SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
                 ;2167  ;
                 ;2168  ;  Эти адреса, начиная от 80 до BF, являются адресами для микроинструкций EXTENDED
                 ;2169  ;
                 ;2170  080:
                 ;2171  WR.PLUS.0.TO.WR:
C 00_0, 118      ;2172          SRC/0.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
                 ;2173  WR.INC.WR:
C 0081, 318      ;2174          SRC/0.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
                 ;2175  WR.NEG.WR:
C 0082, 31A      ;2176          SRC/0.A,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
                 ;2177  WR.COM.WR:
C 0083, 31F      ;2178          SRC/0.A,ALU/R.XNOR.S,DST/WRITE.B.F,CIN/CIN
                 ;2179  WR.DEC.WR:
C 0084, 119      ;2180          SRC/0.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
                 ;2181  FILL85:
C 0085, 10B      ;2182          SRC/0.A
                 ;2183  MOV.Q.WR:
C 0086, 29B      ;2184          SRC/0.Q,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
                 ;2185  FILL87:
C 0087, 08B      ;2186          SRC/0.Q
                 ;2187  COND.ADD&SHIFT:
C 0088, 060      ;2188          SRC/A.B,ALU/R.PLUS.S,DST/RSHF.RAM.Q,CIN/NO.CIN
                 ;2189  COND.SUB&SHIFT:
C 0089, 261      ;2190          SRC/A.B,ALU/S.MINUS.R,DST/RSHF.RAM.Q,CIN/CIN
                 ;2191  FILL8A:
C 008A, 04B      ;2192          SRC/A.B
                 ;2193  SHL2:
C 008B, 078      ;2194          SRC/A.B,ALU/R.PLUS.S,DST/LSHF.RAM,CIN/NO.CIN
                 ;2195  ASHRC:
C 008C, 0E3      ;2196          SRC/0.B,ALU/R.OR.S,DST/RSHF.RAM.Q,CIN/NO.CIN
                 ;2197  ASHR:
C 008D, 2EB      ;2198          SRC/0.B,ALU/R.OR.S,DST/RSHF.RAM,CIN/CIN
                 ;2199  ASHLC:
C 008E, 2F3      ;2200          SRC/0.B,ALU/R.OR.S,DST/LSHF.RAM.Q,CIN/CIN
                 ;2201  ASHL:
C 008F, 2FB      ;2202          SRC/0.B,ALU/R.OR.S,DST/LSHF.RAM,CIN/CIN
                 ;2203  Q.PLUS.0:
C 0090, 08B      ;2204          SRC/0.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
                 ;2205  FILL91:
C 0091, 08B      ;2206          SRC/0.Q
                 ;2207  WR.CMP.Q:
C 0092, 20A      ;2208          SRC/A.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                 ;2209  Q.CMP.WR:
C 0093, 209      ;2210          SRC/A.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                 ;2211  DEC.Q:
C 0094, 081      ;2212          SRC/0.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/NO.CIN
                 ;2213  INC.Q:
C 0095, 280      ;2214          SRC/0.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/CIN
                 ;2215  NEG.Q:
C 0096, 282      ;2216          SRC/0.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
                 ;2217  COM.Q:
```

```
C 0097, 287     ;2218           SRC/0.Q,ALU/R.XNOR.S,DST/LOAD.Q,CIN/CIN
                ;2219   WR.BIT.WR:
C 0098, 04C     ;2220           SRC/A.B,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
                ;2221   WR.CMP.WR:
C 0099, 24A     ;2222           SRC/A.B,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                ;2223   FILL9A:
C 009A, 04B     ;2224           SRC/A.B
                ;2225   WR.PLUS.WR+1:
C 009B, 258     ;2226           SRC/A.B,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
                ;2227   FILL9C:
C 009C, 04B     ;2228           SRC/A.B
                ;2229   FILL9D:
C 009D, 04B     ;2230           SRC/A.B
                ;2231   FILL9E:
C 009E, 0CB     ;2232           SRC/0.B
                ;2233   FILL9F:
C 009F, 0CB     ;2234           SRC/0.B
                ;2235   WR.PLUS.Q:
C 00A0, 000     ;2236           SRC/A.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
                ;2237   WR.FROM.Q:
C 00A1, 201     ;2238           SRC/A.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
                ;2239   Q.FROM.WR.Q:
C 00A2, 202     ;2240           SRC/A.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
                ;2241   WR.OR.Q:
C 00A3, 203     ;2242           SRC/A.Q,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
                ;2243   WR.AND.Q:
C 00A4, 004     ;2244           SRC/A.Q,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
                ;2245   WR.MASK.Q:
C 00A5, 205     ;2246           SRC/A.Q,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/CIN
                ;2247   WR.XOR.Q:
C 00A6, 206     ;2248           SRC/A.Q,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
                ;2249   FILLA7:
C 00A7, 00B     ;2250           SRC/A.Q
                ;2251   WR.PLUS.WR.Q:
C 00A8, 040     ;2252           SRC/A.B,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
                ;2253   WR.FROM.WR.Q:
C 00A9, 241     ;2254           SRC/A.B,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
                ;2255   FILLAA:
C 00AA, 04B     ;2256           SRC/A.B
                ;2257   WR.OR.WR.Q:
C 00AB, 243     ;2258           SRC/A.B,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
                ;2259   WR.AND.WR.Q:
C 00AC, 044     ;2260           SRC/A.B,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
                ;2261   WR.MASK.WR.Q:
C 00AD, 245     ;2262           SRC/A.B,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/CIN
                ;2263   WR.XOR.WR.Q:
C 00AE, 246     ;2264           SRC/A.B,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
                ;2265   FILLAF:
C 00AF, 04B     ;2266           SRC/A.B
                ;2267   Q.PLUS.WR:
C 00B0, 01B     ;2268           SRC/A.Q,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
                ;2269   WR.FROM.Q.WR:
C 00B1, 219     ;2270           SRC/A.Q,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
                ;2271   Q.FROM.WR:
C 00B2, 21A     ;2272           SRC/A.Q,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
```

```
                ;2273   Q.OR.WR:
C 00B3, 21B     ;2274           SRC/A.Q,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
                ;2275   Q.AND.WR:
C 00B4, 01C     ;2276           SRC/A.Q,ALU/R.AND.S,DST/WRITE.B.F,CIN/NO.CIN
                ;2277   FILLB5:
C 00B5, 00B     ;2278           SRC/A.Q
                ;2279   Q.XOR.WR:
C 00B6, 21E     ;2280           SRC/A.Q,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
                ;2281   Q.BIT.WR:
C 00B7, 20C     ;2282           SRC/A.Q,ALU/R.AND.S,DST/NOP,CIN/CIN
                ;2283   WR.PLUS.WR:
C 00B8, 058     ;2284           SRC/A.B,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
                ;2285   WR.FROM.WR:
C 00B9, 259     ;2286           SRC/A.B,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
                ;2287   WR.FROM.WR.WR:
C 00BA, 25A     ;2288           SRC/A.B,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
                ;2289   WR.OR.WR:
C 00BB, 25B     ;2290           SRC/A.B,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
                ;2291   WR.FROM.WR-1:
C 00BC, 059     ;2292           SRC/A.B,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
                ;2293   WR.MASK.WR:
C 00BD, 25D     ;2294           SRC/A.B,ALU/NOTR.AND.S,DST/WRITE.B.F,CIN/CIN
                ;2295   WR.XOR.WR:
C 00BE, 25E     ;2296           SRC/A.B,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
                ;2297   WR.AND.WR:
C 00BF, 25C     ;2298           SRC/A.B,ALU/R.AND.S,DST/WRITE.B.F,CIN/CIN
                ;2299   ;
                ;2300   ;
                ;2301   ;  Здесь представлены управляющие коды путей данных для микроинструкций MOVE
                ;2302   ;
                ;2303   0C0:
                ;2304   MOV.MEM.WR:
C 00C0, 1D3     ;2305           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C1, 1D3     ;2306           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C2, 1D3     ;2307           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C3, 1D3     ;2308           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C4, 1D3     ;2309           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C5, 1D3     ;2310           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C6, 1D3     ;2311           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 00C7, 1D3     ;2312           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
                ;2313   MOV.LS.MEM:
C 00C8, 1CB     ;2314           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00C9, 1CB     ;2315           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CA, 1CB     ;2316           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CB, 1CB     ;2317           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CC, 1CB     ;2318           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CD, 1CB     ;2319           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CE, 1CB     ;2320           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00CF, 1CB     ;2321           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
                ;2322   MOV.XWR.Q:
C 00D0, 0C3     ;2323           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D1, 0C3     ;2324           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D2, 0C3     ;2325           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D3, 0C3     ;2326           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D4, 0C3     ;2327           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
```

```
C 00D5,  0C3      ;2328           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D6,  0C3      ;2329           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 00D7,  0C3      ;2330           SRC/0.B,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
                  ;2331   MOV.LS.WR:
C 00D8,  1DB      ;2332           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00D9,  1DB      ;2333           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DA,  1DB      ;2334           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DB,  1DB      ;2335           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DC,  1DB      ;2336           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DD,  1DB      ;2337           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DE,  1DB      ;2338           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
C 00DF,  1DB      ;2339           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.F,CIN/NO.CIN
                  ;2340   MOV.MEM.LS:
C 00E0,  1CB      ;2341           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E1,  1CB      ;2342           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E2,  1CB      ;2343           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E3,  1CB      ;2344           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E4,  1CB      ;2345           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E5,  1CB      ;2346           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E6,  1CB      ;2347           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E7,  1CB      ;2348           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
                  ;2349   MOV.ACC.LS:
C 00E8,  1CB      ;2350           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00E9,  1CB      ;2351           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00EA,  1CB      ;2352           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00EB,  1CB      ;2353           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00EC,  1CB      ;2354           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00ED,  1CB      ;2355           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00EE,  1CB      ;2356           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00EF,  1CB      ;2357           SRC/D.0,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
                  ;2358   WR.PLUS.OS:
C 00F0,  14B      ;2359           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F1,  14B      ;2360           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F2,  14B      ;2361           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F3,  14B      ;2362           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F4,  14B      ;2363           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F5,  14B      ;2364           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F6,  14B      ;2365           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 00F7,  14B      ;2366           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
                  ;2367   MOV.WR.LS:
C 00F8,  10B      ;2368           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00F9,  10B      ;2369           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FA,  10B      ;2370           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FB,  10B      ;2371           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FC,  10B      ;2372           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FD,  10B      ;2373           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FE,  10B      ;2374           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
C 00FF,  10B      ;2375           SRC/0.A,ALU/R.OR.S,DST/NOP,CIN/NO.CIN
                  ;2376   ;
                  ;2377   ;
                  ;2378   ;  Эта группа ячеек предназначена для микроинструкций BASIC.
                  ;2379   ;  Она использует адреса от 100 до 1FF
                  ;2380   ;
                  ;2381   100:
                  ;2382   LS.PLUS.WR:
```

```
C 0100,  158      ;2383          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0101,  158      ;2384          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0102,  158      ;2385          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0103,  158      ;2386          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
                  ;2387    LS.FROM.WR:
C 0104,  359      ;2388          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 0105,  359      ;2389          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 0106,  359      ;2390          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 0107,  359      ;2391          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
                  ;2392    LS.AND.WR:
C 0108,  35C      ;2393          SRC/D.A,ALU/R.AND.S,DST/WRITE.B.F,CIN/CIN
C 0109,  35C      ;2394          SRC/D.A,ALU/R.AND.S,DST/WRITE.B.F,CIN/CIN
C 010A,  35C      ;2395          SRC/D.A,ALU/R.AND.S,DST/WRITE.B.F,CIN/CIN
C 010B,  35C      ;2396          SRC/D.A,ALU/R.AND.S,DST/WRITE.B.F,CIN/CIN
                  ;2397    LS.XOR.WR:
C 010C,  35E      ;2398          SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
C 010D,  35E      ;2399          SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
C 010E,  35E      ;2400          SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
C 010F,  35E      ;2401          SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.F,CIN/CIN
                  ;2402    LS.FROM.WR-1:
C 0110,  159      ;2403          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
C 0111,  159      ;2404          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
C 0112,  159      ;2405          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
C 0113,  159      ;2406          SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/NO.CIN
                  ;2407    LS.MASK.WR:
C 0114,  35D      ;2408          SRC/D.A,ALU/NOTR.AND.S,DST/WRITE.B.F,CIN/CIN
C 0115,  35D      ;2409          SRC/D.A,ALU/NOTR.AND.S,DST/WRITE.B.F,CIN/CIN
C 0116,  35D      ;2410          SRC/D.A,ALU/NOTR.AND.S,DST/WRITE.B.F,CIN/CIN
C 0117,  35D      ;2411          SRC/D.A,ALU/NOTR.AND.S,DST/WRITE.B.F,CIN/CIN
                  ;2412    LS.PLUS.WR+1:
C 0118,  358      ;2413          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 0119,  358      ;2414          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 011A,  358      ;2415          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
C 011B,  358      ;2416          SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/CIN
                  ;2417    LS.OR.WR:
C 011C,  358      ;2418          SRC/D.A,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 011D,  358      ;2419          SRC/D.A,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 011E,  358      ;2420          SRC/D.A,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
C 011F,  358      ;2421          SRC/D.A,ALU/R.OR.S,DST/WRITE.B.F,CIN/CIN
                  ;2422    WR.PLUS.LS.Q:
C 0120,  140      ;2423          SRC/D.A,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0121,  140      ;2424          SRC/D.A,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0122,  140      ;2425          SRC/D.A,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0123,  140      ;2426          SRC/D.A,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
                  ;2427    LS.FROM.WR.Q:
C 0124,  341      ;2428          SRC/D.A,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0125,  341      ;2429          SRC/D.A,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0126,  341      ;2430          SRC/D.A,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0127,  341      ;2431          SRC/D.A,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
                  ;2432    WR.FROM.LS.Q:
C 0128,  342      ;2433          SRC/D.A,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 0129,  342      ;2434          SRC/D.A,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 012A,  342      ;2435          SRC/D.A,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 012B,  342      ;2436          SRC/D.A,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
                  ;2437    WR.OR.LS.Q:
```

```
C 012C, 343      ;2438          SRC/D.A,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 012D, 343      ;2439          SRC/D.A,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 012E, 343      ;2440          SRC/D.A,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 012F, 343      ;2441          SRC/D.A,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
                 ;2442    WR.AND.LS.Q:
C 0130, 144      ;2443          SRC/D.A,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0131, 144      ;2444          SRC/D.A,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0132, 144      ;2445          SRC/D.A,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0133, 144      ;2446          SRC/D.A,ALU/R.AND.S,DST/LOAD.Q,CIN/NO.CIN
                 ;2447    WR.XOR.LS.Q:
C 0134, 346      ;2448          SRC/D.A,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0135, 346      ;2449          SRC/D.A,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0136, 346      ;2450          SRC/D.A,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0137, 346      ;2451          SRC/D.A,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
                 ;2452    LS.CMP.WR:
C 0138, 34A      ;2453          SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 0139, 34A      ;2454          SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 013A, 34A      ;2455          SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 013B, 34A      ;2456          SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                 ;2457    WR.CMP.LS:
C 013C, 349      ;2458          SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 013D, 349      ;2459          SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 013E, 349      ;2460          SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 013F, 349      ;2461          SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                 ;2462    LS.PLUS.Q:
C 0140, 180      ;2463          SRC/D.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0141, 180      ;2464          SRC/D.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0142, 180      ;2465          SRC/D.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
C 0143, 180      ;2466          SRC/D.Q,ALU/R.PLUS.S,DST/LOAD.Q,CIN/NO.CIN
                 ;2467    LS.FROM.Q:
C 0144, 381      ;2468          SRC/D.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0145, 381      ;2469          SRC/D.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0146, 381      ;2470          SRC/D.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
C 0147, 381      ;2471          SRC/D.Q,ALU/S.MINUS.R,DST/LOAD.Q,CIN/CIN
                 ;2472    Q.FROM.LS.Q:
C 0148, 382      ;2473          SRC/D.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 0149, 382      ;2474          SRC/D.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 014A, 382      ;2475          SRC/D.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
C 014B, 382      ;2476          SRC/D.Q,ALU/R.MINUS.S,DST/LOAD.Q,CIN/CIN
                 ;2477    LS.OR.Q:
C 014C, 383      ;2478          SRC/D.Q,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 014D, 383      ;2479          SRC/D.Q,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 014E, 383      ;2480          SRC/D.Q,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
C 014F, 383      ;2481          SRC/D.Q,ALU/R.OR.S,DST/LOAD.Q,CIN/CIN
                 ;2482    LS.MASK.Q:
C 0150, 185      ;2483          SRC/D.Q,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0151, 185      ;2484          SRC/D.Q,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0152, 185      ;2485          SRC/D.Q,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/NO.CIN
C 0153, 185      ;2486          SRC/D.Q,ALU/NOTR.AND.S,DST/LOAD.Q,CIN/NO.CIN
                 ;2487    LS.XOR.Q:
C 0154, 386      ;2488          SRC/D.Q,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0155, 386      ;2489          SRC/D.Q,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0156, 386      ;2490          SRC/D.Q,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
C 0157, 386      ;2491          SRC/D.Q,ALU/R.XOR.S,DST/LOAD.Q,CIN/CIN
                 ;2492    LS.AND.Q:
```

```
C 0158, 384     ;2493          SRC/D.Q,ALU/R.AND.S,DST/LOAD.Q,CIN/CIN
C 0159, 384     ;2494          SRC/D.Q,ALU/R.AND.S,DST/LOAD.Q,CIN/CIN
C 015A, 384     ;2495          SRC/D.Q,ALU/R.AND.S,DST/LOAD.Q,CIN/CIN
C 015B, 384     ;2496          SRC/D.Q,ALU/R.AND.S,DST/LOAD.Q,CIN/CIN
                ;2497   LS.BIT.Q:
C 015C, 38C     ;2498          SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/CIN
C 015D, 38C     ;2499          SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/CIN
C 015E, 38C     ;2500          SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/CIN
C 015F, 38C     ;2501          SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/CIN
                ;2502   MOV.LS.Q:
C 0160, 1C3     ;2503          SRC/D.0,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 0161, 1C3     ;2504          SRC/D.0,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 0162, 1C3     ;2505          SRC/D.0,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
C 0163, 1C3     ;2506          SRC/D.0,ALU/R.OR.S,DST/LOAD.Q,CIN/NO.CIN
                ;2507   WR.BIT.LS:
C 0164, 14C     ;2508          SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 0165, 14C     ;2509          SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 0166, 14C     ;2510          SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 0167, 14C     ;2511          SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
                ;2512   LS.CMP.Q:
C 0168, 38A     ;2513          SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 0169, 38A     ;2514          SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 016A, 38A     ;2515          SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 016B, 38A     ;2516          SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN.
                ;2517   Q.CMP.LS:
C 016C, 389     ;2518          SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 016D, 389     ;2519          SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 016E, 389     ;2520          SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 016F, 389     ;2521          SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                ;2522   Q.PLUS.LS.TO.WR:
C 0170, 198     ;2523          SRC/D.Q,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0171, 198     ;2524          SRC/D.Q,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0172, 198     ;2525          SRC/D.Q,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
C 0173, 198     ;2526          SRC/D.Q,ALU/R.PLUS.S,DST/WRITE.B.F,CIN/NO.CIN
                ;2527   WRA.FROM.LS.WRB:
C 0174, 35A     ;2528          SRC/D.A,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
C 0175, 35A     ;2529          SRC/D.A,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
C 0176, 35A     ;2530          SRC/D.A,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
C 0177, 35A     ;2531          SRC/D.A,ALU/R.MINUS.S,DST/WRITE.B.F,CIN/CIN
                ;2532   LS.NEG.WR:
C 0178, 3D9     ;2533          SRC/D.0,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 0179, 3D9     ;2534          SRC/D.0,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 017A, 3D9     ;2535          SRC/D.0,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
C 017B, 3D9     ;2536          SRC/D.0,ALU/S.MINUS.R,DST/WRITE.B.F,CIN/CIN
                ;2537   LS.COM.WR:
C 017C, 3DF     ;2538          SRC/D.0,ALU/R.XNOR.S,DST/WRITE.B.F,CIN/CIN
C 017D, 3DF     ;2539          SRC/D.0,ALU/R.XNOR.S,DST/WRITE.B.F,CIN/CIN
C 017E, 3DF     ;2540          SRC/D.0,ALU/R.XNOR.S,DST/WRITE.B.F,CIN/CIN
C 017F, 3DF     ;2541          SRC/D.0,ALU/R.XNOR.S,DST/WRITE.B.F,CIN/CIN
                ;2542   ;
                ;2543   ; Следующие ячейки задают в качестве приемника местную память.
                ;2544   ; Этот факт используется аппаратурой для генерации импульсов записи
                ;2545   ; для местной памяти
                ;2546   ;
                ;2547   180:
```

```
                    ;2548   LS.PLUS.WR.XCHG:
C 0180,  150        ;2549           SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 0181,  150        ;2550           SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 0182,  150        ;2551           SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 0183,  150        ;2552           SRC/D.A,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
                    ;2553   LS.FROM.WR.XCHG:
C 0184,  351        ;2554           SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.A,CIN/CIN
C 0185,  351        ;2555           SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.A,CIN/CIN
C 0186,  351        ;2556           SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.A,CIN/CIN
C 0187,  351        ;2557           SRC/D.A,ALU/S.MINUS.R,DST/WRITE.B.A,CIN/CIN
                    ;2558   LS.AND.WR.XCHG:
C 0188,  354        ;2559           SRC/D.A,ALU/R.AND.S,DST/WRITE.B.A,CIN/CIN
C 0189,  354        ;2560           SRC/D.A,ALU/R.AND.S,DST/WRITE.B.A,CIN/CIN
C 018A,  354        ;2561           SRC/D.A,ALU/R.AND.S,DST/WRITE.B.A,CIN/CIN
C 018B,  354        ;2562           SRC/D.A,ALU/R.AND.S,DST/WRITE.B.A,CIN/CIN
                    ;2563   LS.XOR.WR.XCHG:
C 018C,  356        ;2564           SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.A,CIN/CIN
C 018D,  356        ;2565           SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.A,CIN/CIN
C 018E,  356        ;2566           SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.A,CIN/CIN
C 018F,  356        ;2567           SRC/D.A,ALU/R.XOR.S,DST/WRITE.B.A,CIN/CIN
                    ;2568   SWAP.LS.WR:
C 0190,  1D3        ;2569           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 0191,  1D3        ;2570           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 0192,  1D3        ;2571           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
C 0193,  1D3        ;2572           SRC/D.0,ALU/R.OR.S,DST/WRITE.B.A,CIN/NO.CIN
                    ;2573   CLR.LS:
C 0194,  3CC        ;2574           SRC/D.0,ALU/R.AND.S,DST/NOP,CIN/CIN
C 0195,  3CC        ;2575           SRC/D.0,ALU/R.AND.S,DST/NOP,CIN/CIN
C 0196,  3CC        ;2576           SRC/D.0,ALU/R.AND.S,DST/NOP,CIN/CIN
C 0197,  3CC        ;2577           SRC/D.0,ALU/R.AND.S,DST/NOP,CIN/CIN
                    ;2578   MOV.Q.WR&WR.LS:
C 0198,  293        ;2579           SRC/0.Q,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 0199,  293        ;2580           SRC/0.Q,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 019A,  293        ;2581           SRC/0.Q,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
C 019B,  293        ;2582           SRC/0.Q,ALU/R.OR.S,DST/WRITE.B.A,CIN/CIN
                    ;2583   WR.PLUS.Q.XCHG:
C 019C,  010        ;2584           SRC/A.Q,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 019D,  010        ;2585           SRC/A.Q,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 019E,  010        ;2586           SRC/A.Q,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
C 019F,  010        ;2587           SRC/A.Q,ALU/R.PLUS.S,DST/WRITE.B.A,CIN/NO.CIN
                    ;2588   WR.FROM.LS-1:
C 01A0,  14A        ;2589           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01A1,  14A        ;2590           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01A2,  14A        ;2591           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01A3,  14A        ;2592           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
                    ;2593   LS.FROM.WR.LS:
C 01A4,  349        ;2594           SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01A5,  349        ;2595           SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01A6,  349        ;2596           SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01A7,  349        ;2597           SRC/D.A,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                    ;2598   WR.PLUS.LS+1:
C 01A8,  348        ;2599           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01A9;  348        ;2600           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01AA,  348        ;2601           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01AB,  348        ;2602           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/CIN
```

```
                    ;2603   WR.FROM.LS:
C 01AC, 34A         ;2604           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01AD, 34A         ;2605           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01AE, 34A         ;2606           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01AF, 34A         ;2607           SRC/D.A,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                    ;2608   WR.PLUS.LS:
C 01B0, 148         ;2609           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01B1, 148         ;2610           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01B2, 148         ;2611           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01B3, 148         ;2612           SRC/D.A,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
                    ;2613   WR.OR.LS:
C 01B4, 34B         ;2614           SRC/D.A,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01B5, 34B         ;2615           SRC/D.A,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01B6, 34B         ;2616           SRC/D.A,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01B7, 34B         ;2617           SRC/D.A,ALU/R.OR.S,DST/NOP,CIN/CIN
                    ;2618   WR.AND.LS:
C 01B8, 34C         ;2619           SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/CIN
C 01B9, 34C         ;2620           SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/CIN
C 01BA, 34C         ;2621           SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/CIN
C 01BB, 34C         ;2622           SRC/D.A,ALU/R.AND.S,DST/NOP,CIN/CIN
                    ;2623   WR.XOR.LS:
C 01BC, 34E         ;2624           SRC/D.A,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01BD, 34E         ;2625           SRC/D.A,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01BE, 34E         ;2626           SRC/D.A,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01BF, 34E         ;2627           SRC/D.A,ALU/R.XOR.S,DST/NOP,CIN/CIN
                    ;2628   Q.PLUS.LS:
C 01C0, 188         ;2629           SRC/D.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01C1, 188         ;2630           SRC/D.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01C2, 188         ;2631           SRC/D.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01C3, 188         ;2632           SRC/D.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
                    ;2633   LS.FROM.Q.LS:
C 01C4, 389         ;2634           SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01C5, 389         ;2635           SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01C6, 389         ;2636           SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01C7, 389         ;2637           SRC/D.Q,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                    ;2638   Q.FROM.LS:
C 01C8, 38A         ;2639           SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01C9, 38A         ;2640           SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01CA, 38A         ;2641           SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01CB, 38A         ;2642           SRC/D.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                    ;2643   Q.OR.LS:
C 01CC, 38B         ;2644           SRC/D.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01CD, 38B         ;2645           SRC/D.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01CE, 38B         ;2646           SRC/D.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01CF, 38B         ;2647           SRC/D.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
                    ;2648   Q.AND.LS:
C 01D0, 18C         ;2649           SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 01D1, 18C         ;2650           SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 01D2, 18C         ;2651           SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
C 01D3, 18C         ;2652           SRC/D.Q,ALU/R.AND.S,DST/NOP,CIN/NO.CIN
                    ;2653   Q.XOR.LS:
C 01D4, 38E         ;2654           SRC/D.Q,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01D5, 38E         ;2655           SRC/D.Q,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01D6, 38E         ;2656           SRC/D.Q,ALU/R.XOR.S,DST/NOP,CIN/CIN
C 01D7, 38E         ;2657           SRC/D.Q,ALU/R.XOR.S,DST/NOP,CIN/CIN
```

```
                ;2658   MOV.Q.LS:
C 01D8, 28B     ;2659           SRC/0.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01D9, 28B     ;2660           SRC/0.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01DA, 28B     ;2661           SRC/0.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
C 01DB, 28B     ;2662           SRC/0.Q,ALU/R.OR.S,DST/NOP,CIN/CIN
                ;2663   Q.NEG.LS:
C 01DC, 28A     ;2664           SRC/0.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01DD, 28A     ;2665           SRC/0.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01DE, 28A     ;2666           SRC/0.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01DF, 28A     ;2667           SRC/0.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                ;2668   WR.COM.LS:
C 01E0, 0CF     ;2669           SRC/0.B,ALU/R.XNOR.S,DST/NOP,CIN/NO.CIN
C 01E1, 0CF     ;2670           SRC/0.B,ALU/R.XNOR.S,DST/NOP,CIN/NO.CIN
C 01E2, 0CF     ;2671           SRC/0.B,ALU/R.XNOR.S,DST/NOP,CIN/NO.CIN
C 01E3, 0CF     ;2672           SRC/0.B,ALU/R.XNOR.S,DST/NOP,CIN/NO.CIN
                ;2673   WR.NEG.LS:
C 01E4, 2CA     ;2674           SRC/0.B,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01E5, 2CA     ;2675           SRC/0.B,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01E6, 2CA     ;2676           SRC/0.B,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01E7, 2CA     ;2677           SRC/0.B,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                ;2678   Q.PLUS.WR.TO.LS:
C 01E8, 008     ;2679           SRC/A.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01E9, 008     ;2680           SRC/A.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01EA, 008     ;2681           SRC/A.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
C 01EB, 008     ;2682           SRC/A.Q,ALU/R.PLUS.S,DST/NOP,CIN/NO.CIN
                ;2683   Q.FROM.WR.TO.LS:
C 01EC, 20A     ;2684           SRC/A.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01ED, 20A     ;2685           SRC/A.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01EE, 20A     ;2686           SRC/A.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
C 01EF, 20A     ;2687           SRC/A.Q,ALU/R.MINUS.S,DST/NOP,CIN/CIN
                ;2688   DEC.LS:
C 01F0, 1CA     ;2689           SRC/D.0,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01F1, 1CA     ;2690           SRC/D.0,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01F2, 1CA     ;2691           SRC/D.0,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
C 01F3, 1CA     ;2692           SRC/D.0,ALU/R.MINUS.S,DST/NOP,CIN/NO.CIN
                ;2693   COM.LS:
C 01F4, 3CF     ;2694           SRC/D.0,ALU/R.XNOR.S,DST/NOP,CIN/CIN
C 01F5, 3CF     ;2695           SRC/D.0,ALU/R.XNOR.S,DST/NOP,CIN/CIN
C 01F6, 3CF     ;2696           SRC/D.0,ALU/R.XNOR.S,DST/NOP,CIN/CIN
C 01F7, 3CF     ;2697           SRC/D.0,ALU/R.XNOR.S,DST/NOP,CIN/CIN
                ;2698   NEG.LS:
C 01F8, 3C9     ;2699           SRC/D.0,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01F9, 3C9     ;2700           SRC/D.0,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01FA, 3C9     ;2701           SRC/D.0,ALU/S.MINUS.R,DST/NOP,CIN/CIN
C 01FB, 3C9     ;2702           SRC/D.0,ALU/S.MINUS.R,DST/NOP,CIN/CIN
                ;2703   INC.LS:
C 01FC, 3C8     ;2704           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01FD, 3C8     ;2705           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01FE, 3C8     ;2706           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
C 01FF, 3C8     ;2707           SRC/D.0,ALU/R.PLUS.S,DST/NOP,CIN/CIN
                ;2708   ;
                ;2709   .UCODE
                ;2710   .SEQUENTIAL
```

```
                ;2711    .PAGE  "УКАЗАТЕЛИ ТЕСТОВ"
                ;2712      ; Эта часть содержит указатели для всех тестов данного сегмента. Адрес WCS для
                ;2713      ; инструкции JUMP соответствует номеру теста,т.е., WSC 5 содержит JUMP к тесту 5.
                ;2714      ; Все неиспользованные номера тестов содержат переход к программе обработки оши-
                ;2715      ; бок. Эта часть имеет длину 80 ячеек, что позволяет иметь до 80 тестов на сег-
                ;2716      ; мент, начиная от адреса 1 WSC до адреса 4F.
                ;2717      ;
                ;2718    1:                                          ; указатели тестов начинаются адресом WCS 1
U 0001, 0870,04 ;2719              JMP [T.1]                        ; переход к тесту 1
U 0002, 8873,24 ;2720              JMP [T.2]                        ; переход к тесту 2
U 0003, 0878,14 ;2721              JMP [T.3]                        ; переход к тесту 3
U 0004, 087A,F4 ;2722              JMP [T.4]                        ; переход к тесту 4
U 0005, 887D,C4 ;2723              JMP [T.5]                        ; переход к тесту 5
U 0006, 8880,14 ;2724              JMP [T.6]                        ; переход к тесту 6
U 0007, 0882,44 ;2725              JMP [T.7]                        ; переход к тесту 7
U 0008, 0883,94 ;2726              JMP [T.8]                        ; переход к тесту 8
U 0009, 0885,34 ;2727              JMP [T.9]                        ; переход к тесту 9
U 000A, 0886,64 ;2728              JMP [T.A]                        ; переход к тесту A
U 000B, 0888,74 ;2729              JMP [T.B]                        ; переход к тесту B
U 000C, 888C,E4 ;2730              JMP [T.C]                        ; переход к тесту C
U 000D, 8891,84 ;2731              JMP [T.D]                        ; переход к тесту D
U 000E, 889B,D4 ;2732              JMP [T.E]                        ; переход к тесту E
U 000F, 88A8,D4 ;2733              JMP [T.F]                        ; переход к тесту F
U 0010, 08A9,E4 ;2734              JMP [T.10]                       ; переход к тесту 10
U 0011, 88AA,F4 ;2735              JMP [T.11]                       ; переход к тесту 11
U 0012, 08BA,64 ;2736              JMP [T.12]                       ; переход к тесту 12
U 0013, 08C7,C4 ;2737              JMP [T.13]                       ; переход к тесту 13
U 0014, 88CA,C4 ;2738              JMP [T.14]                       ; переход к тесту 14
U 0015, 88D1,34 ;2739              JMP [T.15]                       ; переход к тесту 15
U 0016, 88E4,54 ;2740              JMP [T.16]                       ; переход к тесту 16
U 0017, 88EC,14 ;2741              JMP [T.17]                       ; переход к тесту 17
U 0018, 88ED,C4 ;2742              JMP [T.18]                       ; переход к тесту 18
U 0019, 88EF,84 ;2743              JMP [T.19]                       ; переход к тесту 19
U 001A, 88F7,24 ;2744              JMP [T.1A]                       ; переход к тесту 1A
U 001B, 88FB,24 ;2745              JMP [T.1B]                       ; переход к тесту 1B
U 001C, 0900,04 ;2746              JMP [T.1C]                       ; переход к тесту 1C
U 001D, 0906,64 ;2747              JMP [T.1D]                       ; переход к тесту 1D
U 001E, 890B,A4 ;2748              JMP [T.1E]                       ; переход к тесту 1E
U 001F, 0911,F4 ;2749              JMP [T.1F]                       ; переход к тесту 1F
U 0020, 8913,F4 ;2750              JMP [T.20]                       ; переход к тесту 20
U 0021, 091B,64 ;2751              JMP [T.21]                       ; переход к тесту 21
U 0022, 0927,94 ;2752              JMP [T.22]                       ; переход к тесту 22
U 0023, 0929,24 ;2753              JMP [T.23]                       ; переход к тесту 23
U 0024, 892C,F4 ;2754              JMP [T.24]                       ; переход к тесту 24
U 0025, 093C,64 ;2755              JMP [T.25]                       ; переход к тесту 25
U 0026, 8944,84 ;2756              JMP [T.26]                       ; переход к тесту 26
U 0027, 8947,84 ;2757              JMP [T.27]                       ; переход к тесту 27
U 0028, 094D,A4 ;2758              JMP [T.28]                       ; переход к тесту 28
U 0029, 0951,44 ;2759              JMP [T.29]                       ; переход к тесту 29
U 002A, 0956,C4 ;2760              JMP [T.2A]                       ; переход к тесту 2A
U 002B, 8958,A4 ;2761              JMP [T.2B]                       ; переход к тесту 2B
U 002C, 8960,84 ;2762              JMP [T.2C]                       ; переход к тесту 2C
U 002D, 096C,A4 ;2763              JMP [T.2D]                       ; переход к тесту 2D
U 002E, 0971,94 ;2764              JMP [T.2E]                       ; переход к тесту 2E
U 002F, 0976,74 ;2765              JMP [T.2F]                       ; переход к тесту 2F
```

```
U 0030, 8979,64 ;2766           JMP [T.30]                      ; переход к тесту 30
U 0031, 8985,64 ;2767           JMP [T.31]                      ; переход к тесту 31
U 0032, 0995,94 ;2768           JMP [T.32]                      ; переход к тесту 32
U 0033, 09AB,E4 ;2769           JMP [T.33]                      ; переход к тесту 33
U 0034, 09BC,D4 ;2770           JMP [T.34]                      ; переход к тесту 34
U 0035, 886B,E4 ;2771           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2772                                           ; в данном сегменте
U 0036, 886B,E4 ;2773           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2774                                           ; в данном сегменте
U 0037, 886B,E4 ;2775           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2776                                           ; в данном сегменте
U 0038, 886B,E4 ;2777           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2778                                           ; в данном сегменте
U 0039, 886B,E4 ;2779           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2780                                           ; в данном сегменте
U 003A, 886B,E4 ;2781           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2782                                           ; в данном сегменте
U 003B, 886B,E4 ;2783           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2784                                           ; в данном сегменте
U 003C, 886B,E4 ;2785           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2786                                           ; в данном сегменте
U 003D, 886B,E4 ;2787           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2788                                           ; в данном сегменте
U 003E, 886B,E4 ;2789           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2790                                           ; в данном сегменте
U 003F, 886B,E4 ;2791           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2792                                           ; в данном сегменте
U 0040, 886B,E4 ;2793           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2794                                           ; в данном сегменте
U 0041, 886B,E4 ;2795           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2796                                           ; в данном сегменте
U 0042, 886B,E4 ;2797           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2798                                           ; в данном сегменте
U 0043, 886B,E4 ;2799           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2800                                           ; в данном сегменте
U 0044, 886B,E4 ;2801           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2802                                           ; в данном сегменте
U 0045, 886B,E4 ;2803           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2804                                           ; в данном сегменте
U 0046, 886B,E4 ;2805           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2806                                           ; в данном сегменте
U 0047, 886B,E4 ;2807           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2808                                           ; в данном сегменте
U 0048, 886B,E4 ;2809           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2810                                           ; в данном сегменте
U 0049, 886B,E4 ;2811           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2812                                           ; в данном сегменте
U 004A, 886B,E4 ;2813           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2814                                           ; в данном сегменте
U 004B, 886B,E4 ;2815           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2816                                           ; в данном сегменте
U 004C, 886B,E4 ;2817           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2818                                           ; в данном сегменте
U 004D, 886B,E4 ;2819           JMP [ERR.NO.TEST]               ; переход к программе обработки ошибок. Тест отсутствует
                ;2820                                           ; в данном сегменте
```

```
U 004E, 886B,E4 ;2821            JMP [ERR.NO.TEST]                  ; переход к программе обработки ошибок. Тест отсутствует
                ;2822                                               ; в данном сегменте
U 004F, 886B,E4 ;2823            JMP [ERR.NO.TEST]                  ; переход к программе обработки ошибок. Тест отсутствует
                ;2824                                               ; в данном сегменте
```

```
                    ;2825      .PAGE  "ДИАГНОСТИЧЕСКИЕ УКАЗАТЕЛИ"
                    ;2826      ;
                    ;2827      ;   Этот раздел содержит все указатели, необходимые для данного диагностического
                    ;2828      ; сегмента. Первый указатель (по адресу WCS 0) указывает на программу переноса
                    ;2829      ; данных. Это первая ячейка, работающая после новой загрузки WCS. Инструкция, раз-
                    ;2830      ; мещенная здесь, является переходом (JUMP) к TRANSFER.POINT, где могут содержать-
                    ;2831      ; ся инструкции для переноса данных. Если отсутствуют данные, подлежащие переносу,
                    ;2832      ; или по завершению переноса данных, центральный процессор выставляет CPU ATTN,
                    ;2833      ; причем все биты слова управления и состояния очищены.
                    ;2834      ;   Следующие 80 ячеек (от ячейки WCS 1 до ячейки WCS 4F) содержат указатели для
                    ;2835      ; каждого теста в этом сегменте. Указателями являются инструкции JUMP к номеру
                    ;2836      ; теста, соответствующему номеру ячейки, т.е, ячейка 5 будет содержать переход к
                    ;2837      ; тесту 5. Все неиспользованные номера тестов содержат переход к программе обрабо-
                    ;2838      ; тки ошибок.
                    ;2839      ;   Наконец, следующие 32 ячейки (от ячейки WCS 50 до 6F) содержат указатели
                    ;2840      ; (инструкции JUMP) к подпрограммам WCS, которые подлежат использованию диагности-
                    ;2841      ; ческим монитором консольного процессора.
                    ;2842      ;
                    ;2843     0:
U 0000, 086C,24     ;2844           JMP [TRANSFER.POINT]                ; переход к подпрограмме, которая загружает LS 7
                    ;2845                                               ; указателем секции данных и выдает CPU ATTN с
                    ;2846                                               ; установленным битом переноса данных
                    ;2847     50:                                       ; начало указателей
                    ;2848                                               ; подпрограмм в ячейке WCS 50
                    ;2849      ;
                    ;2850      ;   указатели подпрограмм
                    ;2851      ;
U 0050, 0860,84     ;2852           JMP [DEPOSIT.CSR]                   ; переход к подпрограмме WCS, которая выполняет запись в
                    ;2853                                               ; регистры управления и состояния MCT по команде DEPOSIT
                    ;2854                                               ; CSR
U 0051, 0860,D4     ;2855           JMP [EXAMINE.CSR]                   ; переход к подпрограмме WCS, выполняющей чтение
                    ;2856                                               ; регистров управления и состояния MCT по команде
                    ;2857                                               ; EXAMINE CSR
U 0052, 0861,F4     ;2858           JMP [DEPOSIT.TB]                    ; переход к подпрограмме WCS, выполняющей запись в буфер
                    ;2859                                               ; трансляции MCT по команде DEPOSIT для буфера
                    ;2860                                               ; трансляции
U 0053, 8863,A4     ;2861           JMP [EXAMINE.TB]                    ; переход к подпрограмме WCS, выполняющей чтение буфера
                    ;2862                                               ; трансляции MCT по команде EXAMINE для буфера
                    ;2863                                               ; трансляции
U 0054, 8865,C4     ;2864           JMP [DEPOSIT.MM]                    ; переход к подпрограмме WCS, которая записывает в
                    ;2865                                               ; основную память по команде DEPOSIT для основной
                    ;2866                                               ; памяти. Используется также для пересылки данных из
                    ;2867                                               ; WCS в основную память
U 0055, 0866,24     ;2868           JMP [EXAMINE.MM]                    ; переход к подпрограмме WCS, которая читает из основной
                    ;2869                                               ; памяти по команде EXAMINE для основной памяти
U 0056, 0868,94     ;2870           JMP [SAVE.WR]                       ; переход к подпрограмме WCS, которая запоминает
                    ;2871                                               ; содержимое рабочих регистров WR0-WR3 в местной памяти
                    ;2872                                               ; LS 0-3
U 0057, 8868,E4     ;2873           JMP [RESTORE.WR]                    ; переход к подпрограмме WCS, которая восстанавливает
                    ;2874                                               ; содержимое WR0-WR3 из LS 0-3
U 0058, 8864,24     ;2875           JMP [DEPOSIT.UBS]                   ; переход к подпрограмме WCS, которая записывает в буфер
                    ;2876                                               ; трансляции общей шины контроллера памяти
U 0059, 0865,44     ;2877           JMP [EXAMINE.UBS]                   ; переход к подпрограмме WCS, которая читает из буфера
                    ;2878                                               ; трансляции общей шины контроллера памяти
U 005A, 0866,84     ;2879           JMP [EXAMINE.ICSR]                  ; переход к подпрограмме WCS, которая читает CSR IDC
```

```
U 005B, 0866,B4 ;2880            JMP [EXAMINE.IDAR]          ; переход к подпрограмме WCS, которая читает DAR IDC
U 005C, 0866,E4 ;2881            JMP [EXAMINE.DBUF]          ; переход к подпрограмме WCS, которая читает DBUF IDC
U 005D, 8867,14 ;2882            JMP [EXAMINE.PATT]          ; переход к подпрограмме WCS, которая читает код
                ;2883                                        ; коррекции ECC из IDC
U 005E, 8867,44 ;2884            JMP [EXAMINE.POSIT]         ; переход к подпрограмме WCS, которая читает адрес
                ;2885                                        ; ошибки ECC из IDC
U 005F, 0867,94 ;2886            JMP [DEPOSIT.ICSR]          ; переход к подпрограмме WCS, которая записывает CSR
                ;2887                                        ; IDC
U 0060, 0867,C4 ;2888            JMP [DEPOSIT.IDAR]          ; переход к подпрограмме WCS, которая записывает DAR
                ;2889                                        ; IDC
U 0061, 0867,F4 ;2890            JMP [DEPOSIT.DBUF]          ; переход к подпрограмме WCS, которая записывает DBUF
                ;2891                                        ; IDC
U 0062, 8868,24 ;2892            JMP [CLEAR.FIFO.ADDR]       ; переход к подпрограмме WCS, которая очищает адрес FIFO
                ;2893                                        ; IDC
U 0063, 8868,44 ;2894            JMP [SELECT.FIFO.A]         ; переход к подпрограмме WCS, которая выбирает FIFO A
U 0064, 0868,64 ;2895            JMP [SELECT.FIFO.B]         ; переход к подпрограмме WCS, которая выбирает FIFO B
U 0065, DB00,15 ;2896            NOP                         ; не используется
U 0066, DB00,15 ;2897            NOP                         ; не используется
U 0067, DB00,15 ;2898            NOP                         ; не используется
U 0068, DB00,15 ;2899            NOP                         ; не используется
U 0069, DB00,15 ;2900            NOP                         ; не используется
U 006A, DB00,15 ;2901            NOP                         ; не используется
U 006B, DB00,15 ;2902            NOP                         ; не используется
U 006C, DB00,15 ;2903            NOP                         ; не используется
U 006D, DB00,15 ;2904            NOP                         ; не используется
U 006E, DB00,15 ;2905            NOP                         ; не используется
U 006F, DB00,15 ;2906            NOP                         ; не используется
                ;2907
                ;2908
                ;2909    .REGION/70,5FF
```

```
                ;2910     .PAGE "СЕКЦИЯ ДАННЫХ МЕСТНОЙ ПАМЯТИ"
                ;2911     ;
                ;2912     ;***      Этот раздел перечисляет данные, которые будут переданы в LS консольным
                ;2913     ;       процессором в качестве части процедуры инициации, выполняемой тестом 0.
                ;2914     ;       Программа переноса данных ссылается на эту область данных. LS содержит
                ;2915     ;       константы, слова управления и состояния и ячейки временного хранения,
                ;2916     ;       используемые и подпрограммами микродиагностики.
                ;2917     ;          LS имеет ширину 32 бита. Каждое из слов здесь имеет ширину 16 битов,
                ;2918     ;.      так что два последовательных слова, перечисленные здесь, будут загру-
                ;2919     ;       жаться в каждую последовательную ячейку LS. Первое перечисленное слово
                ;2920     ;       будет занимать биты 0-15 ячейки LS. Второе перечисленное слово будет
                ;2921     ;       составлять биты 16-31 той же ячейки LS.
                ;2922     ;          Ячейки 78-7F первой половины LS и F8-FF второй половины LS не являются
                ;2923     ;       в действительности ячейками LS. Они задают доступ к одному из нескольких
                ;2924     ;       регистров центрального процессора или реализуют средство индексации
                ;2925     ;       других ячеек LS. По этой причине они не записываются посредством програ-
                ;2926     ;       ммы переноса данных.
                ;2927     ;          ПРИМЕЧАНИЕ: Эта таблица задана в шестнадцатеричном формате, т.е., каждый
                ;2928     ;       разряд представляется четырьмя битами, так что четыре разряда представ-
                ;2929     ;       ляют полное 16-битовое слово. Микроассемблер требует, чтобы шестнадцате-
                ;2930     ;       ричному значению, которое начинается буквой (например, FFFF), предшест-
                ;2931     ;       вовал 0 (0FFFF). Таким образом, некоторые значения в таблице содержат
                ;2932     ;       5 шестнадцатеричных разрядов. Пятый разряд в действительности не исполь-
                ;2933     ;       зуются.
                ;2934     ;
                ;2935     ;---
                ;2936     ;
                ;2937     TRANSFER.DATA.LS1:
                ;2938     ;
U 0070, 0000,78 ;2939           COUNT.LS[0078]                  ; счетчик длинных слов (число длинных слов, подлежащих
                ;2940                                           ; пересылке в LS, равно 120, десятичн.) приемником
                ;2941                                           ; является LS
U 0071, 0000,00 ;2942           START.ADR[0000]                 ; начальный адрес приемника (начинается при LS=0)
                ;2943     ;
U 0072, 0000,00 ;2944           WORD[0000]                      ; ячейка 0
U 0073, 0000,00 ;2945           WORD[0000]                      ;
                ;2946     ;
U 0074, 0000,00 ;2947           WORD[0000]                      ; ячейка 1
U 0075, 0000,00 ;2948           WORD[0000]                      ;
                ;2949     ;
U 0076, 0000,00 ;2950           WORD[0000]                      ; ячейка 2
U 0077, 0000,00 ;2951           WORD[0000]                      ;
                ;2952     ;
U 0078, 0000,00 ;2953           WORD[0000]                      ; ячейка 3
U 0079, 0000,00 ;2954           WORD[0000]                      ;
                ;2955     ;
U 007A, 0000,00 ;2956           WORD[0000]                      ; ячейка 4
U 007B, 0000,00 ;2957           WORD[0000]                      ;
                ;2958     ;
U 007C, 0000,00 ;2959           WORD[0000]                      ; ячейка 5
U 007D, 0000,00 ;2960           WORD[0000]                      ;
                ;2961     ;
U 007E, 0000,00 ;2962           WORD[0000]                      ; ячейка 6
U 007F, 0000,00 ;2963           WORD[0000]                      ;
                ;2964     ;
```

```
U 0080, 0000,00 ;2965           WORD[0000]                      ; ячейка 7
U 0081, 0000,00 ;2966           WORD[0000]                      ;
                ;2967   ;
U 0082, 0000,00 ;2968           WORD[0000]                      ; ячейка 8
U 0083, 0000,00 ;2969           WORD[0000]                      ;
                ;2970   ;
U 0084, 0000,00 ;2971           WORD[0000]                      ; ячейка 9
U 0085, 0000,00 ;2972           WORD[0000]                      ;
                ;2973   ;
U 0086, 0000,00 ;2974           WORD[0000]                      ; ячейка 0A
U 0087, 0000,00 ;2975           WORD[0000]                      ;
                ;2976   ;
U 0088, 0000,00 ;2977           WORD[0000]                      ; ячейка 0B
U 0089, 0000,00 ;2978           WORD[0000]                      ;
                ;2979   ;
U 008A, 0000,00 ;2980           WORD[0000]                      ; ячейка 0C
U 008B, 0000,00 ;2981           WORD[0000]                      ;
                ;2982   ;
U 008C, 0000,00 ;2983           WORD[0000]                      ; ячейка 0D
U 008D, 0000,00 ;2984           WORD[0000]                      ;
                ;2985   ;
U 008E, 0000,00 ;2986           WORD[0000]                      ; ячейка 0E
U 008F, 0000,00 ;2987           WORD[0000]                      ;
                ;2988   ;
U 0090, 0000,00 ;2989           WORD[0000]                      ; ячейка 0F
U 0091, 0000,00 ;2990           WORD[0000]                      ;
                ;2991   ;
U 0092, 0000,00 ;2992           WORD[0000]                      ; ячейка 10
U 0093, 0000,00 ;2993           WORD[0000]                      ;
                ;2994   ;
U 0094, 0000,00 ;2995           WORD[0000]                      ; ячейка 11
U 0095, 0000,00 ;2996           WORD[0000]                      ;
                ;2997   ;
U 0096, 0000,00 ;2998           WORD[0000]                      ; ячейка 12
U 0097, 0000,00 ;2999           WORD[0000]                      ;
                ;3000   ;
U 0098, 0000,FF ;3001           WORD[00FF]                      ; ячейка 13
U 0099, 0000,00 ;3002           WORD[0000]                      ;
                ;3003   ;
U 009A, 00FF,FF ;3004           WORD[0FFFF]                     ; ячейка 14
U 009B, 0000,00 ;3005           WORD[0000]                      ;
                ;3006   ;
U 009C, 0000,00 ;3007           WORD[0000]                      ; ячейка 15
U 009D, 00FF,00 ;3008           WORD[0FF00]                     ;
                ;3009   ;
U 009E, 00FF,00 ;3010           WORD[0FF00]                     ; ячейка 16
U 009F, 00FF,FF ;3011           WORD[0FFFF]                     ;
                ;3012   ;
U 00A0, 003F,FF ;3013           WORD[3FFF]                      ; ячейка 17
U 00A1, 0001,00 ;3014           WORD[0100]                      ;
                ;3015   ;
U 00A2, 003F,FF ;3016           WORD[3FFF]                      ; ячейка 18
U 00A3, 007F,FE ;3017           WORD[7FFE]                      ;
                ;3018   ;
U 00A4, 00FF,FF ;3019           WORD[0FFFF]                     ; ячейка 19
```

```
U 00A5, 00FE,7F ;3020           WORD[0FE7F]                             ;
                ;3021     ;
U 00A6, 0000,00 ;3022           WORD[0000]                              ; ячейка 1A
U 00A7, 007F,F8 ;3023           WORD[7FF8]                              ;
                ;3024     ;
U 00A8, 0080,00 ;3025           WORD[8000]                              ; ячейка 1B
U 00A9, 007F,FF ;3026           WORD[7FFF]                              ;
                ;3027     ;
U 00AA, 0000,00 ;3028           WORD[0000]                              ; ячейка 1C
U 00AB, 0000,00 ;3029           WORD[0000]                              ;
                ;3030     ;
U 00AC, 0000,00 ;3031           WORD[0000]                              ; ячейка 1D
U 00AD, 0000,00 ;3032           WORD[0000]                              ;
                ;3033     ;
U 00AE, 0005,00 ;3034           WORD[0500]                              ; ячейка 1E
U 00AF, 0000,80 ;3035           WORD[0080]                              ;
                ;3036     ;
U 00B0, 0005,00 ;3037           WORD[0500]                              ; ячейка 1F
U 00B1, 0000,00 ;3038           WORD[0000]                              ;
                ;3039     ;
U 00B2, 0000,01 ;3040           WORD[0001]                              ; ячейка 20
U 00B3, 0000,00 ;3041           WORD[0000]                              ;
                ;3042     ;
U 00B4, 0000,02 ;3043           WORD[0002]                              ; ячейка 21
U 00B5, 0000,00 ;3044           WORD[0000]                              ;
                ;3045     ;
U 00B6, 0000,04 ;3046           WORD[0004]                              ; ячейка 22
U 00B7, 0000,00 ;3047           WORD[0000]                              ;
                ;3048     ;
U 00B8, 0000,08 ;3049           WORD[0008]                              ; ячейка 23
U 00B9, 0000,00 ;3050           WORD[0000]                              ;
                ;3051     ;
U 00BA, 0000,10 ;3052           WORD[0010]                              ; ячейка 24
U 00BB, 0000,00 ;3053           WORD[0000]                              ;
                ;3054     ;
U 00BC, 0000,20 ;3055           WORD[0020]                              ; ячейка 25
U 00BD, 0000,00 ;3056           WORD[0000]                              ;
                ;3057     ;
U 00BE, 0000,40 ;3058           WORD[0040]                              ; ячейка 26
U 00BF, 0000,00 ;3059           WORD[0000]                              ;
                ;3060     ;
U 00C0, 0000,80 ;3061           WORD[0080]                              ; ячейка 27
U 00C1, 0000,00 ;3062           WORD[0000]                              ;
                ;3063     ;
U 00C2, 0001,00 ;3064           WORD[0100]                              ; ячейка 28
U 00C3, 0000,00 ;3065           WORD[0000]                              ;
                ;3066     ;
U 00C4, 0002,00 ;3067           WORD[0200]                              ; ячейка 29
U 00C5, 0000,00 ;3068           WORD[0000]                              ;
                ;3069     ;
U 00C6, 0004,00 ;3070           WORD[0400]                              ; ячейка 2A
U 00C7, 0000,00 ;3071           WORD[0000]                              ;
                ;3072     ;
U 00C8, 0008,00 ;3073           WORD[0800]                              ; ячейка 2B
U 00C9, 0000,00 ;3074           WORD[0000]                              ;
```

```
                    ;3075      ;
U 00CA,  0010,00    ;3076              WORD[1000]                          ; ячейка 2C
U 00CB,  0000,00    ;3077              WORD[0000]                          ;
                    ;3078      ;
U 00CC,  0020,00    ;3079              WORD[2000]                          ; ячейка 2D
U 00CD,  0000,00    ;3080              WORD[0000]                          ;
                    ;3081      ;
U 00CE,  0040,00    ;3082              WORD[4000]                          ; ячейка 2E
U 00CF,  0000,00    ;3083              WORD[0000]                          ;
                    ;3084      ;
U 00D0,  0080,00    ;3085              WORD[8000]                          ; ячейка 2F
U 00D1,  0000,00    ;3086              WORD[0000]                          ;
                    ;3087      ;
U 00D2,  0000,00    ;3088              WORD[0000]                          ; ячейка 30
U 00D3,  0000,01    ;3089              WORD[0001]                          ;
                    ;3090      ;
U 00D4,  0000,00    ;3091              WORD[0000]                          ; ячейка 31
U 00D5,  0000,02    ;3092              WORD[0002]                          ;
                    ;3093      ;
U 00D6,  0000,00    ;3094              WORD[0000]                          ; ячейка 32
U 00D7,  0000,04    ;3095              WORD[0004]                          ;
                    ;3096      ;
U 00D8,  0000,00    ;3097              WORD[0000]                          ; ячейка 33
U 00D9,  0000,08    ;3098              WORD[0008]                          ;
                    ;3099      ;
U 00DA,  0000,00    ;3100              WORD[0000]                          ; ячейка 34
U 00DB,  0000,10    ;3101              WORD[0010]                          ;
                    ;3102      ;
U 00DC,  0000,00    ;3103              WORD[0000]                          ; ячейка 35
U 00DD,  0000,20    ;3104              WORD[0020]                          ;
                    ;3105      ;
U 00DE,  0000,00    ;3106              WORD[0000]                          ; ячейка 36
U 00DF,  0000,40    ;3107              WORD[0040]                          ;
                    ;3108      ;
U 00E0,  0000,00    ;3109              WORD[0000]                          ; ячейка 37
U 00E1,  0000,80    ;3110              WORD[0080]                          ;
                    ;3111      ;
U 00E2,  0000,00    ;3112              WORD[0000]                          ; ячейка 38
U 00E3,  0001,00    ;3113              WORD[0100]                          ;
                    ;3114      ;
U 00E4,  0000,00    ;3115              WORD[0000]                          ; ячейка 39
U 00E5,  0002,00    ;3116              WORD[0200]                          ;
                    ;3117      ;
U 00E6,  0000,00    ;3118              WORD[0000]                          ; ячейка 3A
U 00E7,  0004,00    ;3119              WORD[0400]                          ;
                    ;3120      ;
U 00E8,  0000,00    ;3121              WORD[0000]                          ; ячейка 3B
U 00E9,  0008,00    ;3122              WORD[0800]                          ;
                    ;3123      ;
U 00EA,  0000,00    ;3124              WORD[0000]                          ; ячейка 3C
U 00EB,  0010,00    ;3125              WORD[1000]                          ;
                    ;3126      ;
U 00EC,  0000,00    ;3127              WORD[0000]                          ; ячейка 3D
U 00ED,  0020,00    ;3128              WORD[2000]                          ;
                    ;3129      ;
```

```
U 00EE, 0000,00 ;3130           WORD[0000]                          ; ячейка 3E
U 00EF, 0040,00 ;3131           WORD[4000]                          ;
                ;3132     ;
U 00F0, 0000,00 ;3133           WORD[0000]                          ; ячейка 3F
U 00F1, 0080,00 ;3134           WORD[8000]                          ;
                ;3135     ;
U 00F2, 0000,00 ;3136           WORD[0000]                          ; ячейка 40
U 00F3, 0000,00 ;3137           WORD[0000]                          ;
                ;3138     ;
U 00F4, 0000,00 ;3139           WORD[0000]                          ; ячейка 41
U 00F5, 0000,00 ;3140           WORD[0000]                          ;
                ;3141     ;
U 00F6, 0000,00 ;3142           WORD[0000]                          ; ячейка 42
U 00F7, 0000,00 ;3143           WORD[0000]                          ;
                ;3144     ;
U 00F8, 0000,00 ;3145           WORD[0000]                          ; ячейка 43
U 00F9, 0000,00 ;3146           WORD[0000]                          ;
                ;3147     ;
U 00FA, 0000,00 ;3148           WORD[0000]                          ; ячейка 44
U 00FB, 0000,00 ;3149           WORD[0000]                          ;
                ;3150     ;
U 00FC, 0000,00 ;3151           WORD[0000]                          ; ячейка 45
U 00FD, 0000,00 ;3152           WORD[0000]                          ;
                ;3153     ;
U 00FE, 0000,00 ;3154           WORD[0000]                          ; ячейка 46
U 00FF, 0000,00 ;3155           WORD[0000]                          ;
                ;3156     ;
U 0100, 0000,00 ;3157           WORD[0000]                          ; ячейка 47
U 0101, 0000,00 ;3158           WORD[0000]                          ;
                ;3159     ;
U 0102, 0000,00 ;3160           WORD[0000]                          ; ячейка 48
U 0103, 0000,00 ;3161           WORD[0000]                          ;
                ;3162     ;
U 0104, 0000,00 ;3163           WORD[0000]                          ; ячейка 49
U 0105, 0000,00 ;3164           WORD[0000]                          ;
                ;3165     ;
U 0106, 0000,00 ;3166           WORD[0000]                          ; ячейка 4A
U 0107, 0000,00 ;3167           WORD[0000]                          ;
                ;3168     ;
U 0108, 0000,00 ;3169           WORD[0000]                          ; ячейка 4B
U 0109, 0000,00 ;3170           WORD[0000]                          ;
                ;3171     ;
U 010A, 00AA,AA ;3172           WORD[0AAAA]                         ; ячейка 4C
U 010B, 00AA,AA ;3173           WORD[0AAAA]                         ;
                ;3174     ;
U 010C, 0055,55 ;3175           WORD[5555]                          ; ячейка 4D
U 010D, 0055,55 ;3176           WORD[5555]                          ;
                ;3177     ;
U 010E, 0000,00 ;3178           WORD[0000]                          ; ячейка 4E
U 010F, 0000,00 ;3179           WORD[0000]                          ;
                ;3180     ;
U 0110, 00FF,FF ;3181           WORD[0FFFF]                         ; ячейка 4F
U 0111, 00FF,FF ;3182           WORD[0FFFF]                         ;
                ;3183     ;
U 0112, 0000,00 ;3184           WORD[0000]                          ; ячейка 50
```

```
U 0113, 0000,00 ;3185           WORD[0000]                              ;
                ;3186     ;
U 0114, 0000,00 ;3187           WORD[0000]                              ; ячейка 51
U 0115, 0000,00 ;3188           WORD[0000]                              ;
                ;3189     ;
U 0116, 0000,00 ;3190           WORD[0000]                              ; ячейка 52
U 0117, 0000,00 ;3191           WORD[0000]                              ;
                ;3192     ;
U 0118, 0000,00 ;3193           WORD[0000]                              ; ячейка 53
U 0119, 0000,00 ;3194           WORD[0000]                              ;
                ;3195     ;
U 011A, 0000,00 ;3196           WORD[0000]                              ; ячейка 54
U 011B, 0000,00 ;3197           WORD[0000]                              ;
                ;3198     ;
U 011C, 0000,00 ;3199           WORD[0000]                              ; ячейка 55
U 011D, 0000,00 ;3200           WORD[0000]                              ;
                ;3201     ;
U 011E, 0000,00 ;3202           WORD[0000]                              ; ячейка 56
U 011F, 0000,00 ;3203           WORD[0000]                              ;
                ;3204     ;
U 0120, 0000,00 ;3205           WORD[0000]                              ; ячейка 57
U 0121, 0000,00 ;3206           WORD[0000]                              ;
U 0122, 0000,00 ;3207           WORD[0000]                              ; ячейка 58
U 0123, 0000,00 ;3208           WORD[0000]                              ;
U 0124, 0000,00 ;3209           WORD[0000]                              ; ячейка 59
U 0125, 0000,00 ;3210           WORD[0000]                              ;
U 0126, 00F0,00 ;3211           WORD[0F000]                             ; ячейка 5A
U 0127, 00FF,FF ;3212           WORD[0FFFF]                             ;
U 0128, 00F0,02 ;3213           WORD[0F002]                             ; ячейка 5B
U 0129, 00FF,FF ;3214           WORD[0FFFF]                             ;
U 012A, 00F0,04 ;3215           WORD[0F004]                             ; ячейка 5C
U 012B, 00FF,FF ;3216           WORD[0FFFF]                             ;
U 012C, 00F0,06 ;3217           WORD[0F006]                             ; ячейка 5D
U 012D, 00FF,FF ;3218           WORD[0FFFF]                             ;
U 012E, 00F0,08 ;3219           WORD[0F008]                             ; ячейка 5E
U 012F, 00FF,FF ;3220           WORD[0FFFF]                             ;
U 0130, 00F0,0E ;3221           WORD[0F00E]                             ; ячейка 5F
U 0131, 00FF,FF ;3222           WORD[0FFFF]                             ;
U 0132, 0000,03 ;3223           WORD[0003]                              ; ячейка 60
U 0133, 0000,00 ;3224           WORD[0000]                              ;
U 0134, 0000,12 ;3225           WORD[0012]                              ; ячейка 61
U 0135, 0000,00 ;3226           WORD[0000]                              ;
U 0136, 0033,33 ;3227           WORD[3333]                              ; ячейка 62
U 0137, 0033,33 ;3228           WORD[3333]                              ;
U 0138, 000F,0F ;3229           WORD[0F0F]                              ; ячейка 63
U 0139, 000F,0F ;3230           WORD[0F0F]                              ;
U 013A, 0000,FF ;3231           WORD[00FF]                              ; ячейка 64
U 013B, 0000,FF ;3232           WORD[00FF]                              ;
U 013C, 0001,62 ;3233           WORD[0162]                              ; ячейка 65
U 013D, 0000,00 ;3234           WORD[0000]                              ;
U 013E, 0000,28 ;3235           WORD[0028]                              ; ячейка 66
U 013F, 0000,00 ;3236           WORD[0000]                              ;
U 0140, 0000,0C ;3237           WORD[000C]                              ; ячейка 67
U 0141, 0000,00 ;3238           WORD[0000]                              ;
U 0142, 0000,00 ;3239           WORD[0000]                              ; ячейка 68
```

```
U 0143, 0000,00 ;3240            WORD[0000]                                ;
U 0144, 0000,00 ;3241            WORD[0000]                                ; ячейка 69
U 0145, 0000,00 ;3242            WORD[0000]                                ;
U 0146, 0000,00 ;3243            WORD[0000]                                ; ячейка 6A
U 0147, 0000,00 ;3244            WORD[0000]                                ;
U 0148, 0000,00 ;3245            WORD[0000]                                ; ячейка 6B
U 0149, 0000,00 ;3246            WORD[0000]                                ;
U 014A, 0000,00 ;3247            WORD[0000]                                ; ячейка 6C
U 014B, 0000,00 ;3248            WORD[0000]                                ;
U 014C, 0000,00 ;3249            WORD[0000]                                ; ячейка 6D
U 014D, 0000,00 ;3250            WORD[0000]                                ;
U 014E, 0000,00 ;3251            WORD[0000]                                ; ячейка 6E
U 014F, 0000,00 ;3252            WORD[0000]                                ;
U 0150, 0000,00 ;3253            WORD[0000]                                ; ячейка 6F
U 0151, 0000,00 ;3254            WORD[0000]                                ;
U 0152, 0000,00 ;3255            WORD[0000]                                ; ячейка 70
U 0153, 0000,00 ;3256            WORD[0000]                                ;
U 0154, 0000,00 ;3257            WORD[0000]                                ; ячейка 71
U 0155, 0000,00 ;3258            WORD[0000]                                ;
U 0156, 0000,00 ;3259            WORD[0000]                                ; ячейка 72
U 0157, 0000,00 ;3260            WORD[0000]                                ;
U 0158, 0000,00 ;3261            WORD[0000]                                ; ячейка 73
U 0159, 0000,00 ;3262            WORD[0000]                                ;
U 015A, 0000,00 ;3263            WORD[0000]                                ; ячейка 74
U 015B, 0000,00 ;3264            WORD[0000]                                ;
U 015C, 0000,00 ;3265            WORD[0000]                                ; ячейка 75
U 015D, 0000,00 ;3266            WORD[0000]                                ;
U 015E, 0000,00 ;3267            WORD[0000]                                ; ячейка 76
U 015F, 0000,00 ;3268            WORD[0000]                                ;
U 0160, 0000,00 ;3269            WORD[0000]                                ; ячейка 77
U 0161, 0000,00 ;3270            WORD[0000]                                ;
                ;3271     ;
                ;3272     ;       СЕКЦИЯ ДАННЫХ, СПЕЦИФИЧЕСКИХ ДЛЯ ПРОГРАММЫ (LS B0-F7)
                ;3273     ;
                ;3274     ;       Этот раздел задает вторую половину LS. Она доступна только для инструк-
                ;3275     ;       ции MOV или такой, которая использует формат микроинструкции MOVE.
                ;3276     ;       Эта секция загружается отдельной пересылкой.
                ;3277     ;
                ;3278   TRANSFER.DATA.LS2:
U 0162, 0000,78 ;3279            COUNT.LS[0078]                            ; счетчик длинных слов (число длинных слов, подлежащих
                ;3280                                                      ; пересылке в LS, равно 120 десятичн.). Приемником
                ;3281                                                      ; является LS
U 0163, 0000,80 ;3282            START.ADR[0080]                           ; начальный адрес приемника (начинается с LS B0
                ;3283                                                      ; (шестнадцатеричн.))
U 0164, 0000,3C ;3284            WORD[003C]                                ; ячейка B0
U 0165, 0000,00 ;3285            WORD[0000]                                ;
U 0166, 0000,43 ;3286            WORD[0043]                                ; ячейка B1
U 0167, 0000,00 ;3287            WORD[0000]                                ;
U 0168, 0000,64 ;3288            WORD[0064]                                ; ячейка B2
U 0169, 0000,00 ;3289            WORD[0000]                                ;
U 016A, 0000,25 ;3290            WORD[0025]                                ; ячейка B3
U 016B, 0000,00 ;3291            WORD[0000]                                ;
U 016C, 0000,26 ;3292            WORD[0026]                                ; ячейка B4
U 016D, 0000,00 ;3293            WORD[0000]                                ;
U 016E, 0000,20 ;3294            WORD[0020]                                ; ячейка B5
```

```
U 016F, 0000,00 ;3295          WORD[0000]                    ;
U 0170, 0000,54 ;3296          WORD[0054]                    ; ячейка 86
U 0171, 0000,00 ;3297          WORD[0000]                    ;
U 0172, 0000,7D ;3298          WORD[007D]                    ; ячейка 87
U 0173, 0000,00 ;3299          WORD[0000]                    ;
U 0174, 00FF,C3 ;3300          WORD[0FFC3]                   ; ячейка 88
U 0175, 00FF,FF ;3301          WORD[0FFFF]                   ;
U 0176, 00FF,80 ;3302          WORD[0FF80]                   ; ячейка 89
U 0177, 00FF,FF ;3303          WORD[0FFFF]                   ;
U 0178, 0000,00 ;3304          WORD[0000]                    ; ячейка 8A
U 0179, 0000,00 ;3305          WORD[0000]                    ;
U 017A, 0000,00 ;3306          WORD[0000]                    ; ячейка 8B
U 017B, 0000,00 ;3307          WORD[0000]                    ;
U 017C, 0000,00 ;3308          WORD[0000]                    ; ячейка 8C
U 017D, 0000,00 ;3309          WORD[0000]                    ;
U 017E, 0000,00 ;3310          WORD[0000]                    ; ячейка 8D
U 017F, 0000,00 ;3311          WORD[0000]                    ;
U 0180, 0000,00 ;3312          WORD[0000]                    ; ячейка 8E
U 0181, 0000,00 ;3313          WORD[0000]                    ;
U 0182, 0000,00 ;3314          WORD[0000]                    ; ячейка 8F
U 0183, 0000,00 ;3315          WORD[0000]                    ;
U 0184, 0000,00 ;3316          WORD[0000]                    ; ячейка 90
U 0185, 0000,00 ;3317          WORD[0000]                    ;
U 0186, 0000,00 ;3318          WORD[0000]                    ; ячейка 91
U 0187, 0000,00 ;3319          WORD[0000]                    ;
U 0188, 0000,00 ;3320          WORD[0000]                    ; ячейка 92
U 0189, 0000,00 ;3321          WORD[0000]                    ;
U 018A, 0000,00 ;3322          WORD[0000]                    ; ячейка 93
U 018B, 0000,00 ;3323          WORD[0000]                    ;
U 018C, 0000,00 ;3324          WORD[0000]                    ; ячейкА 94
U 018D, 0000,00 ;3325          WORD[0000]                    ;
U 018E, 0000,00 ;3326          WORD[0000]                    ; ячейка 95
U 018F, 0000,00 ;3327          WORD[0000]                    ;
U 0190, 0000,00 ;3328          WORD[0000]                    ; ячейка 96
U 0191, 0000,00 ;3329          WORD[0000]                    ;
U 0192, 0000,00 ;3330          WORD[0000]                    ; ячейка 97
U 0193, 0000,00 ;3331          WORD[0000]                    ;
U 0194, 0000,00 ;3332          WORD[0000]                    ; ячейка 98
U 0195, 0000,00 ;3333          WORD[0000]                    ;
U 0196, 0000,00 ;3334          WORD[0000]                    ; ячейка 99
U 0197, 0000,00 ;3335          WORD[0000]                    ;
U 0198, 0000,00 ;3336          WORD[0000]                    ; ячейка 9A
U 0199, 0000,03 ;3337          WORD[0003]                    ;
U 019A, 0080,00 ;3338          WORD[8000]                    ; ячейка 9B
U 019B, 007F,FF ;3339          WORD[7FFF]                    ;
U 019C, 0000,00 ;3340          WORD[0000]                    ; ячейка 9C
U 019D, 0000,54 ;3341          WORD[0054]                    ;
U 019E, 0000,00 ;3342          WORD[0000]                    ; ячейка 9D
U 019F, 0000,F0 ;3343          WORD[00F0]                    ;
U 01A0, 0000,00 ;3344          WORD[0000]                    ; ячейка 9E
U 01A1, 0000,FC ;3345          WORD[00FC]                    ;
U 01A2, 003F,FF ;3346          WORD[3FFF]                    ; ячейка 9F
U 01A3, 003F,00 ;3347          WORD[3F00]                    ;
U 01A4, 0000,03 ;3348          WORD[0003]                    ; ячейка 0A0
U 01A5, 0000,03 ;3349          WORD[0003]                    ;
```

```
U 01A6,  0000,00 ;3350            WORD[0000]                          ; ячейка 0A1
U 01A7,  0000,00 ;3351            WORD[0000]                          ;
U 01A8,  0000,0F ;3352            WORD[000F]                          ; ячейка 0A2
U 01A9,  0000,0F ;3353            WORD[000F]                          ;
U 01AA,  0000,00 ;3354            WORD[0000]                          ; ячейка 0A3
U 01AB,  0000,00 ;3355            WORD[0000]                          ;
U 01AC,  0000,EF ;3356            WORD[00EF]                          ; ячейка 0A4
U 01AD,  0000,EF ;3357            WORD[00EF]                          ;
U 01AE,  0000,00 ;3358            WORD[0000]                          ; ячейка 0A5
U 01AF,  0000,00 ;3359            WORD[0000]                          ;
U 01B0,  000F,EF ;3360            WORD[0FEF]                          ; ячейка 0A6
U 01B1,  000F,EF ;3361            WORD[0FEF]                          ;
U 01B2,  0000,00 ;3362            WORD[0000]                          ; ячейка 0A7
U 01B3,  0000,00 ;3363            WORD[0000]                          ;
!! 01B4, 0088,8B ;3364            WORD[8888]                          ; ячейка 0A8
U 01B5,  0088,8B ;3365            WORD[8888]                          ;
U 01B6,  0077,74 ;3366            WORD[7774]                          ; ячейка 0A9
U 01B7,  0000,00 ;3367            WORD[0000]                          ;
U 01B8,  00CC,CF ;3368            WORD[0CCCF]                         ; ячейка 0AA
U 01B9,  00CC,CF ;3369            WORD[0CCCF]                         ;
U 01BA,  0033,30 ;3370            WORD[3330]                          ; ячейка 0AB
U 01BB,  0000,00 ;3371            WORD[0000]                          ;
U 01BC,  00EE,EF ;3372            WORD[0EEEF]                         ; ячейка 0AC
U 01BD,  00EE,EF ;3373            WORD[0EEEF]                         ;
U 01BE,  0011,10 ;3374            WORD[1110]                          ; ячейка 0AD
U 01BF,  0000,00 ;3375            WORD[0000]                          ;
U 01C0,  00FF,EF ;3376            WORD[0FFEF]                         ; ячейка 0AE
U 01C1,  00FF,EF ;3377            WORD[0FFEF]                         ;
U 01C2,  0000,10 ;3378            WORD[0010]                          ; ячейка 0AF
U 01C3,  0000,00 ;3379            WORD[0000]                          ;
U 01C4,  0000,00 ;3380            WORD[0000]                          ; ячейка 0B0
U 01C5,  0000,00 ;3381            WORD[0000]                          ;
U 01C6,  0000,00 ;3382            WORD[0000]                          ; ячейка 0B1
U 01C7,  0000,00 ;3383            WORD[0000]                          ;
U 01C8,  0000,00 ;3384            WORD[0000]                          ; ячейка 0B2
U 01C9,  0000,00 ;3385            WORD[0000]                          ;
U 01CA,  0000,00 ;3386            WORD[0000]                          ; ячейка 0B3
U 01CB,  0000,00 ;3387            WORD[0000]                          ;
U 01CC,  0000,00 ;3388            WORD[0000]                          ; ячейка 0B4
U 01CD,  0000,00 ;3389            WORD[0000]                          ;
U 01CE,  0000,00 ;3390            WORD[0000]                          ; ячейка 0B5
U 01CF,  0000,00 ;3391            WORD[0000]                          ;
U 01D0,  0000,00 ;3392            WORD[0000]                          ; ячейка 0B6
U 01D1,  0000,00 ;3393            WORD[0000]                          ;
U 01D2,  0000,00 ;3394            WORD[0000]                          ; ячейка 0B7
U 01D3,  0000,00 ;3395            WORD[0000]                          ;
U 01D4,  0000,00 ;3396            WORD[0000]                          ; ячейка 0B8
U 01D5,  0000,00 ;3397            WORD[0000]                          ;
U 01D6,  0000,00 ;3398            WORD[0000]                          ; ячейка 0B9
U 01D7,  0000,00 ;3399            WORD[0000]                          ;
U 01D8,  0000,00 ;3400            WORD[0000]                          ; ячейка 0BA
U 01D9,  0000,00 ;3401            WORD[0000]                          ;
U 01DA,  0000,00 ;3402            WORD[0000]                          ; ячейка 0BB
U 01DB,  0000,00 ;3403            WORD[0000]                          ;
U 01DC,  0000,00 ;3404            WORD[0000]                          ; ячейка 0BC
```

```
U 01DD, 0000,00 ;3405           WORD[0000]                              ;
U 01DE, 0000,00 ;3406           WORD[0000]                              ; ячейка 0BD
U 01DF, 0000,00 ;3407           WORD[0000]                              ;
U 01E0, 0000,00 ;3408           WORD[0000]                              ; ячейка 0BE
U 01E1, 0000,00 ;3409           WORD[0000]                              ;
U 01E2, 0000,00 ;3410           WORD[0000]                              ; ячейка 0BF
U 01E3, 0000,00 ;3411           WORD[0000]                              ;
U 01E4, 0000,00 ;3412           WORD[0000]                              ; ячейка 0C0
U 01E5, 0000,00 ;3413           WORD[0000]                              ;
U 01E6, 0000,00 ;3414           WORD[0000]                              ; ячейка 0C1
U 01E7, 0000,00 ;3415           WORD[0000]                              ;
U 01E8, 0000,00 ;3416           WORD[0000]                              ; ячейка 0C2
U 01E9, 0000,00 ;3417           WORD[0000]                              ;
U 01EA, 0000,00 ;3418           WORD[0000]                              ; ячейка 0C3
U 01EB, 0000,00 ;3419           WORD[0000]                              ;
U 01EC, 0000,00 ;3420           WORD[0000]                              ; ячейка 0C4
U 01ED, 0000,00 ;3421           WORD[0000]                              ;
U 01EE, 0000,00 ;3422           WORD[0000]                              ; ячейка 0C5
U 01EF, 0000,00 ;3423           WORD[0000]                              ;
U 01F0, 0000,00 ;3424           WORD[0000]                              ; ячейка 0C6
U 01F1, 0000,00 ;3425           WORD[0000]                              ;
U 01F2, 0000,00 ;3426           WORD[0000]                              ; ячейка 0C7
U 01F3, 0000,00 ;3427           WORD[0000]                              ;
U 01F4, 0000,00 ;3428           WORD[0000]                              ; ячейка 0C8
U 01F5, 0000,00 ;3429           WORD[0000]                              ;
U 01F6, 0000,00 ;3430           WORD[0000]                              ; ячейка 0C9
U 01F7, 0000,00 ;3431           WORD[0000]                              ;
U 01F8, 0000,00 ;3432           WORD[0000]                              ; ячейка 0CA
U 01F9, 0000,00 ;3433           WORD[0000]                              ;
U 01FA, 0000,00 ;3434           WORD[0000]                              ; ячейка 0CB
U 01FB, 0000,00 ;3435           WORD[0000]                              ;
U 01FC, 0000,00 ;3436           WORD[0000]                              ; ячейка 0CC
U 01FD, 0000,00 ;3437           WORD[0000]                              ;
U 01FE, 0000,00 ;3438           WORD[0000]                              ; ячейка 0CD
U 01FF, 0000,00 ;3439           WORD[0000]                              ;
U 0200, 0000,00 ;3440           WORD[0000]                              ; ячейка 0CE
U 0201, 0000,00 ;3441           WORD[0000]                              ;
U 0202, 0000,00 ;3442           WORD[0000]                              ; ячейка 0CF
U 0203, 0000,00 ;3443           WORD[0000]                              ;
U 0204, 0000,00 ;3444           WORD[0000]                              ; ячейка 0D0
U 0205, 0000,00 ;3445           WORD[0000]                              ;
U 0206, 0000,00 ;3446           WORD[0000]                              ; ячейка 0D1
U 0207, 0000,00 ;3447           WORD[0000]                              ;
U 0208, 0000,00 ;3448           WORD[0000]                              ; ячейка 0D2
U 0209, 0000,00 ;3449           WORD[0000]                              ;
U 020A, 0000,00 ;3450           WORD[0000]                              ; ячейка 0D3
U 020B, 0000,00 ;3451           WORD[0000]                              ;
U 020C, 0000,00 ;3452           WORD[0000]                              ; ячейка 0D4
U 020D, 0000,00 ;3453           WORD[0000]                              ;
U 020E, 0000,00 ;3454           WORD[0000]                              ; ячейка 0D5
U 020F, 0000,00 ;3455           WORD[0000]                              ;
U 0210, 0000,00 ;3456           WORD[0000]                              ; ячейка 0D6
U 0211, 0000,00 ;3457           WORD[0000]                              ;
U 0212, 0000,00 ;3458           WORD[0000]                              ; ячейка 0D7
U 0213, 0000,00 ;3459           WORD[0000]                              ;
```

```
U 0214,  0000,00 ;3460          WORD[0000]                          ; ячейка 0D8
U 0215,  0000,00 ;3461          WORD[0000]                          ;
U 0216,  0000,00 ;3462          WORD[0000]                          ; ячейка 0D9
U 0217,  0000,00 ;3463          WORD[0000]                          ;
U 0218,  0000,00 ;3464          WORD[0000]                          ; ячейка 0DA
U 0219,  0000,00 ;3465          WORD[0000]                          ;
U 021A,  0000,00 ;3466          WORD[0000]                          ; ячейка 0DB
U 021B,  0000,00 ;3467          WORD[0000]                          ;
U 021C,  0000,00 ;3468          WORD[0000]                          ; ячейка 0DC
U 021D,  0000,00 ;3469          WORD[0000]                          ;
U 021E,  0000,00 ;3470          WORD[0000]                          ; ячейка 0DD
U 021F,  0000,00 ;3471          WORD[0000]                          ;
U 0220,  0000,00 ;3472          WORD[0000]                          ; ячейка 0DE
U 0221,  0000,00 ;3473          WORD[0000]                          ;
U 0222,  0000,00 ;3474          WORD[0000]                          ; ячейка 0DF
U 0223,  0000,00 ;3475          WORD[0000]                          ;
U 0224,  0000,00 ;3476          WORD[0000]                          ; ячейка 0E0
U 0225,  0000,00 ;3477          WORD[0000]                          ;
U 0226,  0000,00 ;3478          WORD[0000]                          ; ячейка 0E1
U 0227,  0000,00 ;3479          WORD[0000]                          ;
U 0228,  0000,00 ;3480          WORD[0000]                          ; ячейка 0E2
U 0229,  0000,00 ;3481          WORD[0000]                          ;
U 022A,  0000,00 ;3482          WORD[0000]                          ; ячейка 0E3
U 022B,  0000,00 ;3483          WORD[0000]                          ;
U 022C,  0000,00 ;3484          WORD[0000]                          ; ячейка 0E4
U 022D,  0000,00 ;3485          WORD[0000]                          ;
U 022E,  0000,00 ;3486          WORD[0000]                          ; ячейка 0E5
U 022F,  0000,00 ;3487          WORD[0000]                          ;
U 0230,  0000,00 ;3488          WORD[0000]                          ; ячейка 0E6
U 0231,  0000,00 ;3489          WORD[0000]                          ;
U 0232,  0000,00 ;3490          WORD[0000]                          ; ячейка 0E7
U 0233,  0000,00 ;3491          WORD[0000]                          ;
U 0234,  0000,00 ;3492          WORD[0000]                          ; ячейка 0E8
U 0235,  0000,00 ;3493          WORD[0000]                          ;
U 0236,  0000,00 ;3494          WORD[0000]                          ; ячейка 0E9
U 0237,  0000,00 ;3495          WORD[0000]                          ;
U 0238,  0000,00 ;3496          WORD[0000]                          ; ячейка 0EA
U 0239,  0000,00 ;3497          WORD[0000]                          ;
U 023A,  0000,00 ;3498          WORD[0000]                          ; ячейка 0EB
U 023B,  0000,00 ;3499          WORD[0000]                          ;
U 023C,  0000,00 ;3500          WORD[0000]                          ; ячейка 0EC
U 023D,  0000,00 ;3501          WORD[0000]                          ;
U 023E,  0000,00 ;3502          WORD[0000]                          ; ячейка 0ED
U 023F,  0000,00 ;3503          WORD[0000]                          ;
U 0240,  0000,00 ;3504          WORD[0000]                          ; ячейка 0EE
U 0241,  0000,00 ;3505          WORD[0000]                          ;
U 0242,  0000,00 ;3506          WORD[0000]                          ; ячейка 0EF
U 0243,  0000,00 ;3507          WORD[0000]                          ;
U 0244,  0000,00 ;3508          WORD[0000]                          ; ячейка 0F0
U 0245,  0000,00 ;3509          WORD[0000]                          ;
U 0246,  0000,00 ;3510          WORD[0000]                          ; ячейка 0F1
U 0247,  0000,00 ;3511          WORD[0000]                          ;
U 0248,  0000,00 ;3512          WORD[0000]                          ; ячейка 0F2
U 0249,  0000,00 ;3513          WORD[0000]                          ;
U 024A,  0000,00 ;3514          WORD[0000]                          ; ячейка 0F3
```

```
U 024B, 0000,00 ;3515          WORD[0000]                          ;
U 024C, 0000,00 ;3516          WORD[0000]                          ; ячейка 0F4
U 024D, 0000,00 ;3517          WORD[0000]                          ;
U 024E, 0000,00 ;3518          WORD[0000]                          ; ячейка 0F5
U 024F, 0000,00 ;3519          WORD[0000]                          ;
U 0250, 0000,00 ;3520          WORD[0000]                          ; ячейка 0F6
U 0251, 0000,00 ;3521          WORD[0000]                          ;
U 0252, 0000,00 ;3522          WORD[0000]                          ; ячейка 0F7
U 0253, 0000,00 ;3523          WORD[0000]                          ;
                ;3524    ;
                ;3525    ;
                ;3526    .REGION/600,6FF
                ;3527
```

```
                ;3528     .PAGE  "ПОДПРОГРАММЫ В УПРАВЛЯЮЩЕЙ ПАМЯТИ (WCS), ИСПОЛЬЗУЕМЫЕ МИКРОМОНИТОРОМ"
                ;3529     ;
                ;3530     ;                    Программа генерации данных
                ;3531     ;
                ;3532     ;
                ;3533     ;       Эта программа используется диагностическим монитором для загрузки рабо-
                ;3534     ;       чего регистра WR0 кодом данных из консольного процесора. Монитор устанав-
                ;3535     ;       ливает ATTN консоли, если должна генерироваться единица, или очищает ATTN
                ;3536     ;       консоли, если должен генерироваться 0. Затем он проходит эту программу
                ;3537     ;       один раз с целью вдвигания бита данных в WR0. Эта программа загружает 32-
                ;3538     ;       битовые значения намного быстрее, чем было бы при вдвигании каждой инст-
                ;3539     ;       рукции в CSR из монитора, и используется, главным образом, для формирования
                ;3540     ;       данных, подлежащих загрузке в LS в качестве констант.
                ;3541     ;
                ;3542     ;                       *** ПРЕДУПРЕЖДЕНИЕ ***
                ;3543     ;
                ;3544     ;       Эта программа должна начинаться адресом 600 и заканчиваться 605.
                ;3545     ;
                ;3546     ;
                ;3547     GEN.XFER.DATA:
U 0600, 2F80,15 ;3548             CLR WR[0]                          ; очистка рабочего регистра
U 0601, A0C0,15 ;3549             CON WR[0]                          ; рабочий регистр WR0 содержит все единицы
                ;3550     LOOP.XFER:
U 0602, 5B00,18 ;3551             SKIP.IF[CONSOLE.ATTN]              ; пропуск следующей инструкции, если установлен CONSOLE
                ;3552                                                ; ATTN (генерация единицы)
                ;3553               ASHL WR[0],                      ; вдвигание 0 в бит 0 WR0
U 0603, A3D0,1E ;3554             SKIP                               ; пропуск инструкции ROL
U 0604, A3C0,15 ;3555             ROL WR[0]                          ; вдвигание единицы в бит 0 WR0
U 0605, 0860,24 ;3556             JMP [LOOP.XFER]                    ; повторение
                ;3557     ;
                ;3558     ;       Здесь хранится номер версии диагностики
                ;3559     ;
                ;3560     ;                       *** ПРЕДУПРЕЖДЕНИЕ ***
                ;3561     ;
                ;3562     ;       Эти слова должны быть в ячейках 606 и 607.
                ;3563     ;
                ;3564     ;
U 0606, 0003,00 ;3565             WORD[0300]                         ; версия 03.00
U 0607, 0043,43 ;3566             ASCII [CC]                         ; последние две буквы имени файла [ENKCC]
                ;3567     ;
                ;3568     ;           Программа записи в регистр управления и состояния MCT
                ;3569     ;***
                ;3570     ;
                ;3571     ; Описание функционирования:
                ;3572     ;
                ;3573     ;          Программа записи CSR используется для записи в регистры управления и
                ;3574     ;       состояния контроллера памяти. Контроллер памяти имеет три регистра уп-
                ;3575     ;       равления и состояния, идентифицируемые как CSR0, CSR1 и CSR2.
                ;3576     ;         CSR0 содержит контрольные биты или биты синдрома, полученные из
                ;3577     ;       логических схем корректирующего кода (ECC) после определенных программ
                ;3578     ;       диагностирования. Его нельзя непосредственно записывать при помощи этой
                ;3579     ;       программы.
                ;3580     ;         CSR1 содержит биты ошибок, установленные, если произошла ошибка во
                ;3581     ;       время обращения к памяти. Он содержит 5 битов управления (29-25),
                ;3582     ;       допускающих запись. имеется также 7 проверочных битов (биты 6-0),
```

```
                    ;3583       ;       которые допускают запись, но не считываются. Эти биты используются для
                    ;3584       ;       записи контрольных битов в микросхемы кодов коррекции (ECC).
                    ;3585       ;         CSR2 содержит биты ошибок, установленные, если произошла ошибка во
                    ;3586       ;       время обращения к общей шине. Эти биты только считываются и не могут
                    ;3587       ;       записываться данной программой.
                    ;3588       ;         Поэтому CSR1 является единственным регистром управления и состояния,
                    ;3589       ;       допускающим запись, и только биты 25-29 могут как записываться, так и
                    ;3590       ;       считываться. Таким образом, эта программа при записи CSR вынуждает
                    ;3591       ;       запись в CSR1.
                    ;3592       ;         Эта программа заносит в LS5 данные для обращения к CSR1, затем запи-
                    ;3593       ;       сывает в CSR данные, хранящиеся в LS6. Затем она выдает CPU ATTN с целью
                    ;3594       ;       информирования монитора, что она завершила свою функцию.
                    ;3595       ;         ПРИМЕЧАНИЕ: биты ошибок в CSR1 очищаются при любой записи в CSR1. Эти-
                    ;3596       ;       ми битами являются 31,30, и 23-14. Биты 13-0 и бит 24 не используются.
                    ;3597       ;
                    ;3598       ;  Процедура вызова:
                    ;3599       ;       Консольный процессор загружает адрес указателя этой программы (инструк-
                    ;3600       ;       ции JMP в ячейке WCS 50) в счетчик микроинструкций (UPC) и запускает
                    ;3601       ;       центральный процессор.
                    ;3602       ;
                    ;3603       ;  Входные параметры:
                    ;3604       ;       В ячейку LS6 до выполнения этой программы должны быть записаны из кон-
                    ;3605       ;       соли требуемые данные, подлежащие занесению в CSR1.
                    ;3606       ;
                    ;3607       ;  Неявные входные данные:
                    ;3608       ;       Отсутствуют.
                    ;3609       ;
                    ;3610       ;  Параметры выхода:
                    ;3611       ;       В биты CSR 29-25 и 6-0 записаны данные из LS6.
                    ;3612       ;
                    ;3613       ;  Неявные выходы:
                    ;3614       ;       отсутствуют.
                    ;3615       ;
                    ;3616       ;  Побочный эффект:
                    ;3617       ;       Этой программой модифицируется WR0.
                    ;3618       ;       Этой программой модифицируется LS5.
                    ;3619       ;       Биты CSR1 31,30 и 23-14 очищаются.
                    ;3620       ;
                    ;3621       ;---
                    ;3622       DEPOSIT.CSR:
U 0608,  3644,15    ;3623               MOV LS[CSR1] TO WR[0]                   ; установка бита 2 в WR0 в качестве номера CSR,в который
                    ;3624                                                       ; будет производиться запись (биты 2 и 3 являются
                    ;3625                                                       ; номерами CSR)
U 0609,  BE0A,15    ;3626               MOV WR[0] TO LS[T5.SUB]                  ; занесение  номера CSR в ячейку LS5 (CSR1)
U 060A,  1D0B,F5    ;3627               MEM.REQ[WRITE.CSR] ADRS[T5.SUB] DT[LONG]    ; выдача запроса памяти для доступа к CSR1
U 060B,  B20C,15    ;3628               WRITE.MEM LS[T6.SUB]                    ; пересылка данных из ячейки 6 в CSR1
U 060C,  8868,74    ;3629               JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                    ;3630       ;
                    ;3631       ;         Программа индикации регистра управления и состояния МСТ
                    ;3632       ;***
                    ;3633       ;
                    ;3634       ;  Описание функционирования:
                    ;3635       ;
                    ;3636       ;           Программа индикации CSR читает регистр управления и состояния контрол-
                    ;3637       ;       лера памяти. Существуют 3 регистра управления и состояния, идентифициру-
```

```
                ;3638      ;       мые как CSR0, CSR1 и CSR2.
                ;3639      ;         CSR0 содержит контрольные биты или биты синдрома из логических схем
                ;3640      ;       корректирующего кода (ECC). Они содержатся в CSR в битах 6-0. Другие биты
                ;3641      ;       непредсказуемые. Биты 6-0 допускают только запись специальными диагнос-
                ;3642      ;       тическими программами (они не могут записываться командой DEPOSIT CSR).
                ;3643      ;         CSR1 содержит биты ошибок и биты управления. Битами ошибок являются
                ;3644      ;       31,30 и 23-14. Битами управления являются биты 29-25. другие биты не-
                ;3645      ;       предсказуемые. Биты ошибок записываются во время выполнения функций па-
                ;3646      ;       мяти и не могут записываться посредством команды DEPOSIT CSR. Биты управ-
                ;3647      ;       ления допускают запись. Заметим, что команда DEPOSIT CSR записывает биты
                ;3648      ;       управления и очищает все биты ошибок.
                ;3649      ;         CSR2 содержит 3 бита (биты 16-14), которые могут считываться. Они уста-
                ;3650      ;       навливаются, если произошла ошибка общей шины при обращении к общей шине.
                ;3651      ;       Они не могут записываться непосредственно командой DEPOSIT CSR.
                ;3652      ;         Консольный процессор загружает LS5 номером CSR (0,1 или 2), который
                ;3653      ;       подлежит считывать. Для правильного его расположения эта программа сдви-
                ;3654      ;       гает LS5 на два места влево. Затем она выполняет чтение CSR и помещает
                ;3655      ;       данные в LS6 для считывания с консоли.
                ;3656      ;
                ;3657      ; Процедура вызова:
                ;3658      ;
                ;3659      ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес
                ;3660      ;       указателя этой программы (инструкции JMP в ячейке WCS 51) и запускает
                ;3661      ;       центральный процессор.
                ;3662      ;
                ;3663      ; Входные параметры:
                ;3664      ;
                ;3665      ;       LS5 содержит номер CSR, подлежащего чтению.
                ;3666      ;
                ;3667      ; Неявные входные данные:
                ;3668      ;
                ;3669      ;       Отсутствуют.
                ;3670      ;
                ;3671      ; Выходные параметры:
                ;3672      ;
                ;3673      ;       Данные, считанные с выбранного CSR, в LS6.
                ;3674      ;
                ;3675      ; Неявные выходные данные:
                ;3676      ;
                ;3677      ;       Отсутствуют.
                ;3678      ;
                ;3679      ; Побочный эффект:
                ;3680      ;
                ;3681      ;        LS5 смещена на 2 места влево.
                ;3682      ;
                ;3683      ;---
                ;3684      EXAMINE.CSR:
U 060D, 360A,15 ;3685              MOV LS[T5.SUB] TO WR[0]                  ; выборка из LS5 номера CSR
U 060E, A2D0,15 ;3686              SHL2 WR[0]                               ; сдвиг числа на 2 места влево для получения номера CSR
                ;3687                                                       ; в битах 2 и 3
U 060F, BE0A,15 ;3688              MOV WR[0] TO LS[T5.SUB]                  ; занесение сдвинутого числа в LS5
U 0610, 9D0B,75 ;3689              MEM.REQ[READ.CSR] ADRS[T5.SUB] DT[LONG]  ; выдача запроса памяти для доступа к заданному CSR
U 0611, 3022,15 ;3690              MOV MEM.DATA TO WR[0]                    ; чтение выбранного CSR
                ;3691      CHECK.CSR2:
U 0612, B60A,95 ;3692              MOV LS[T5.SUB] TO WR[1]                  ; выборка сдвинутого номера CSR
```

```
                    ;3693                 CMP LS[#8] WITH WR[1],                ; проверка, был ли выбран CSR2
U 0613,  CE46,B5    ;3694               DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0614,  0861,61    ;3695               JMP.IF[NEQ] TO [CHECK.CSR1]             ; переход, если выбран не CSR2
U 0615,  C530,15    ;3696               BIC LS[CSR2.MASK] TO WR[0]              ; очистка неиспользуемых (непредсказуемых) битов в CSR2
                    ;3697     CHECK.CSR1:
                    ;3698                 CMP LS[#4] WITH WR[1],                ; проверка был ли выбран CSR1
U 0616,  4E44,B5    ;3699               DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0617,  0861,91    ;3700               JMP.IF[NEQ] TO [CHECK.CSR0]             ; переход, если выбран не CSR1
U 0618,  C52E,15    ;3701               BIC LS[CSR1.MASK] TO WR[0]              ; очистка неиспользуемых (непредсказуемых) битов в CSR1
                    ;3702     CHECK.CSR0:
                    ;3703                 CMP LS[#0] WITH WR[1],                  проверка был ли выбран CSR0
U 0619,  4E9C,B5    ;3704               DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 061A,  8861,D1    ;3705               JMP.IF[NEQ] TO [LOAD.LS6]               ; переход, если выбран не CSR0
U 061B,  452C,15    ;3706               BIC LS[CSR0.MASK] TO WR[0]              ; очистка неиспользуемых (непредсказуемых) битов в CSR0
U 061C,  C54E,15    ;3707               BIC LS[BIT7] TO WR[0]                   , бит 7 также не используется
                    ;3708     LOAD.LS6:
U 061D,  BE0C,15    ;3709               MOV WR[0] TO LS[IT6.SUB]                 загрузка данных CSR в LS6 для печати
U 061E,  8868,74    ;3710               JMP [WAIT.SUB]                        ; выдача CPU ATTN и ожидание ответа
                    ;3711     ;
                    ;3712     ;         Программа записи в буфер трансляции MCT
                    ;3713     ;***
                    ;3714     ;
                    ;3715     ; Описание функционирования:
                    ;3716     ;
                    ;3717     ;           Эта программа дает возможность записывать в ту часть контроллера памя-
                    ;3718     ;         ти, которая составляет буфер трансляции. Область буфера содержит 128 дес.
                    ;3719     ;         ячеек, адресуемых от 0 до 7F (шестнадцатер.). Остаток памяти буфера транс-
                    ;3720     ;         ляции (TB) либо не используется, либо содержит данные отображения адресов
                    ;3721     ;         общей шины. Для записи в ту часть ОЗУ TB, которая содержит отображение
                    ;3722     ;         общей шины, используется команда DEPOSIT UB. Адрес, заданный командой
                    ;3723     ;         DEPOSIT, является действительным адресом обращения к ОЗУ буфера. Эта прог-
                    ;3724     ;         рамма выполняет все необходимые сдвиги заданного адреса для правильного
                    ;3725     ;         его расположения. Заданный адрес загружен в ячейку LS5 до выполнения
                    ;3726     ;         этой программы. Адрес должен быть в шестнадцатеричном формате.
                    ;3727     ;           Требуемые данные должны размещаться консольным процессором в LS6 до вы-
                    ;3728     ;         полнения этой программы. Биты <0> являются битами VALID (бит действи-
                    ;3729     ;         тельности), PROT (биты защиты), MODIFY (бит модификации) и BYTE OFFSET
                    ;3730     ;         (бит смещения байта)(битов PRO имеется четыре). Биты 22-08 содержат би-
                    ;3731     ;         ты физического адреса (PA) от PA 04 до PA 09 соответственно. Биты от 31
                    ;3732     ;         до 23 и бит 0 не используются. Программа делает все сдвиги, необходи-
                    ;3733     ;         мые, чтобы записать биты в буфер трансляции так, как описано. Нужные
                    ;3734     ;         данные должны представляться в шестнадцатеричном формате.
                    ;3735     ;
                    ;3736     ; Процедура вызова:
                    ;3737     ;
                    ;3738     ;           Консольный процессор загружает в счетчик микроинструкций (UPC) адрес
                    ;3739     ;         указателя для данной программы (инструкции JMP в ячейке WCS 52) и запус-
                    ;3740     ;         кает центральный процессор.
                    ;3741     ;
                    ;3742     ; Входные параметры:
                    ;3743     ;
                    ;3744     ;         LS5 - адрес буфера трансляции, которому предстоит записывать (диапазон
                    ;3745     ;               от 0 до 7F)
                    ;3746     ;         LS6 - данные, подлежащие записи (22 бита, от 1 до 22). Биты 0 и 23-31
                    ;3747     ;               игнорируются.
```

```
                ;3748     ;
                ;3749     ; Неявные входные данные:
                ;3750     ;
                ;3751     ;       Отсутствуют.
                ;3752     ;
                ;3753     ; Выходные параметры:
                ;3754     ;
                ;3755     ;       В TB по заданному адресу будет записана заданная информация.
                ;3756     ;
                ;3757     ; Неявные выходные данные:
                ;3758     ;
                ;3759     ;       Отсутствуют.
                ;3760     ;
                ;3761     ; Побочный эффект:
                ;3762     ;
                ;3763     ;       WR0 - будет модифицирован
                ;3764     ;       WR1 - будет модифицирован
                ;3765     ;
                ;3766     ;---
                ;3767    DEPOSIT.TB:
U 061F, 8862,FC ;3768            JSR [SHIFT.TB.ADR]                      ; переход к подпрограмме правильного расположения адреса
                ;3769                                                    ; TB
U 0620, 360C,15 ;3770            MOV LS[T6.SUB] TO WR[0]                 ; выборка данных из LS6
U 0621, 4526,15 ;3771            BIC LS[#FF] TO WR[0]                    ; очистка младшего байта заданной информации. Младший
                ;3772                                                    ; байт будет позднее размещен в битах 24-31
U 0622, 8862,AC ;3773            JSR [SHIFT.LOOP]                        ; переход к подпрограмме для сдвига WR0 на 8 позиций
                ;3774                                                    ; вправо
U 0623, E40C,15 ;3775            SWAP LS[T6.SUB] WITH WR[0]              ; замена исходных данных модифицированными
                ;3776                                                    ; данными. Теперь LS6 содержит биты 8-22 в битах 0-14,
                ;3777                                                    ; как и требовалось. WR0 содержит заданные исходные
                ;3778                                                    ; данные
U 0624, 452C,15 ;3779            BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех заданных исходных данных, кроме младшего
                ;3780                                                    ; байта
U 0625, 8862,AC ;3781            JSR [SHIFT.LOOP]                        ; переход к подпрограмме сдвига WR0 на 8 позиций
                ;3782                                                    ; вправо. После возврата WR0 будет содержать биты 0-7 на
                ;3783                                                    ; месте битов 24-31. Другие биты очищены
U 0626, ED0C,15 ;3784            BIS WR[0] TO LS[T6.SUB]                 ; теперь LS6 содержит данные, подлежащие записи, в
                ;3785                                                    ; правильном формате, т.е., биты PA в битах 0-14, BYTE
                ;3786                                                    ; OFFSET, MODIFY, биты PROT от A до D и VALID в битах
                ;3787                                                    ; 25-31
U 0627, 980A,F5 ;3788            MEM.REQ[WRITE.TB] ADRS[T5.SUB] DT[LONG]  ; выдача запроса памяти для записи в TB
U 0628, B20C,15 ;3789            WRITE.MEM LS[T6.SUB]                    ; запись данных из LS6 в TB
U 0629, 8868,74 ;3790            JMP [WAIT.SUB]                          ; переход к выдаче CPU ATTN и ожиданию
                ;3791    ;
                ;3792    ;       Подпрограмма сдвига TB на 8 мест вправо
                ;3793    ;
                ;3794    SHIFT.LOOP:
U 062A, 3646,95 ;3795            MOV LS[#8] TO WR[1]                     ; в WR1 находится счетчик сдвига на 8 позиций
                ;3796    SHIFT.LOOP.1:
U 062B, 2340,15 ;3797            ROR WR[0]                               ; сдвиг WR0 на одну позицию вправо
                ;3798              DEC WR[1],                            ; уменьшение счетчика и установка
U 062C, A102,B5 ;3799            DT(LONG)&SET.ALU.CC                     ; кодов условий
U 062D, 8962,B1 ;3800            JMP.IF[NEQ] TO [SHIFT.LOOP.1]           ; повторение до тех пор, пока будет выполнено 8
                ;3801                                                    ; сдвигов, затем
U 062E, 5B00,14 ;3802            RETURN                                  ; возврат
```

```
                ;3803     ;
                ;3804     ;       Подпрограмма размещения адреса буфера TB в битах 9-14 и бите 31
                ;3805     ;       (бит 0 адреса TB должен находиться на месте бита 31)
                ;3806     ;
                ;3807     SHIFT.TB.ADR:
U 062F, 360A,15 ;3808             MOV LSI[T5.SUB] TO WR[0]                ; выборка адреса TB из LS5
U 0630, 452C,15 ;3809             BIC LSI[#FFFFFF00] TO WR[0]             ; очистка всех битов, кроме младшего байта (8-битовый
                ;3810                                                     ; адрес)
U 0631, A2D0,15 ;3811             SHL2 WR[0]                              ; сдвиг на 2 бита влево для расположения адреса
U 0632, A2D0,15 ;3812             SHL2 WR[0]                              ; сдвиг на 2 бита влево для расположения адреса
U 0633, A2D0,15 ;3813             SHL2 WR[0]                              ; сдвиг на 2 бита влево для расположения адреса
U 0634, A2D0,15 ;3814             SHL2 WR[0]                              ; сдвиг на 2 бита влево для расположения адреса
U 0635, 23D0,15 ;3815             ASHL WR[0]                              ; сдвиг еще на одну позицию. Теперь биты 0-6 находятся в
                ;3816                                                     ; битах 9-15
                ;3817               BIT LSI[BIT15] WITH WR[0],            ; проверка, установлен или очищен бит 15 (старший бит)
U 0636, 595E,35 ;3818             DI(LONG)&SET.ALU.CC                     ; для правильной адресации TB старший бит должен перейти
                ;3819                                                     ; в 31
U 0637, 5B00,09 ;3820             SKIP.IF[EQL]                            ; пропуск следующей инструкции, если старший бит равен
                ;3821                                                     ; нулю
U 0638, 477E,15 ;3822             BIS LSI[BIT31] TO WR[0]                 ; в противном случае поместить бит 31 в старший бит.
                ;3823               MOV WR[0] TO LSI[T5.SUB],             ; теперь адрес TB в правильной форме находится в ячейке
                ;3824                                                     ; LS5
U 0639, 3E0A,14 ;3825             RETURN                                  ; возврат к основному коду
                ;3826     ;
                ;3827     ;       Программа индикации  буфера трансляции MCT
                ;3828     ;***
                ;3829     ;
                ;3830     ; Описание функционирования:
                ;3831     ;
                ;3832     ;          Эта программа выполняет чтение из буфера трансляции (TB) контроллера
                ;3833     ;       памяти. Содержимое TB будет получено с битами BYTE OFFSET, MODIFY, PROT
                ;3834     ;       от A до D и VALID в битах 01-07 соответственно. Биты PA 09-23 поступают
                ;3835     ;       в битах 08-22 соответственно. Биты 23-31 и бит 0 не используются, поэтому
                ;3836     ;       будут очищены до печати результата. Результат для печати на консоли будет
                ;3837     ;       помещен в LS6.
                ;3838     ;          Программа выдает запрос к памяти с функцией памяти READ.TB и типом дан-
                ;3839     ;       ных LONGWORD (длинное слово). Консольный процессор загружает LS5 заданным
                ;3840     ;       адресом прежде, чем выполняется эта программа. Эта программа сдвигает ад-
                ;3841     ;       рес так, как требуется для адресации заданной ячейки TB. Заданный адрес
                ;3842     ;       должен быть в 16-ричном формате.
                ;3843     ;          TB состоит из 128 дес.ячеек (адреса от 0 до 7F). Оставшиеся ячейки ОЗУ
                ;3844     ;       буфера трансляции либо не используются, либо используются для отображения
                ;3845     ;       адресов общей шины. Адреса из области отображения общей шины считываются
                ;3846     ;       посредством программы EXAMINE UB.
                ;3847     ;
                ;3848     ; Процедура вызова:
                ;3849     ;
                ;3850     ;          Консольный процессор загружает в счетчик микроинструкций (UPC) адрес
                ;3851     ;       указателя этой программы (инструкции JMP в ячейке WCS 53).
                ;3852     ;
                ;3853     ; Входные параметры:
                ;3854     ;
                ;3855     ;       LS5 - адрес буфера трансляции (0-7F) в шестнадцатеричном формате.
                ;3856     ;
                ;3857     ; Неявные входные данные:
```

```
                    ;3858      ;
                    ;3859      ;       Отсутствуют.
                    ;3860      ;
                    ;3861      ; Выходные параметры:
                    ;3862      ;
                    ;3863      ;       LS6 - данные из TB по заданному адресу.
                    ;3864      ;
                    ;3865      ; Неявные выходные данные:
                    ;3866      ;
                    ;3867      ;       Отсутствуют.
                    ;3868      ;
                    ;3869      ; Побочный эффект:
                    ;3870      ;
                    ;3871      ;       LS5 - модифицируется
                    ;3872      ;       WR0 - модифицируется
                    ;3873      ;
                    ;3874      ;---
                    ;3875      EXAMINE.TB:
U 063A, 8862,FC     ;3876              JSR [SHIFT.TB.ADR]                          ; заданный в LS5 адрес сдвигается в правильное положение
                    ;3877                                                          ; и замещает исходный адрес в LS5
U 063B, 9C0B,F5     ;3878              MEM.REQ[READ.TB] ADRS[T5.SUB] DT[LONG]      ; выдача запроса к памяти для чтения TB по адресу,
                    ;3879                                                          ; содержащемуся в LS5
U 063C, 3022,15     ;3880              MOV MEM.DATA TO WR[0]                       ; занесение результата в WR0
U 063D, 452A,15     ;3881              BIC LS[#FF000000] TO WR[0]                  ; очистка старшего байта результата (старший байт не
                    ;3882                                                          ; является частью TB и является непредсказуемым)
U 063E, 456E,15     ;3883              BIC LS[BIT23] TO WR[0]                      ; то же действие для бита 23
U 063F, 4540,15     ;3884              BIC LS[BIT0] TO WR[0]                       ; то же действие для бита 0. WR0 теперь содержит
                    ;3885                                                          ; результат, готовый к печати
U 0640, BE0C,15     ;3886              MOV WR[0] TO LS[T6.SUB]                     ; результат занесен в LS6
U 0641, 8868,74     ;3887              JMP [WAIT.SUB]                              ; переход к установке ATTN и ожиданию
                    ;3888      ;
                    ;3889      ;       Программа записи в буфер трансляции адресов общей шины
                    ;3890      ;***
                    ;3891      ;
                    ;3892      ; Описание функционирования:
                    ;3893      ,
                    ;3894      ;          Эта программа позволяет записывать в ту часть буфера трансляции (TB)
                    ;3895      ;       контроллера памяти, которая содержит данные отображения адресов общей ши-
                    ;3896      ;       ны. Область отображения содержит 512 дес. ячеек, адресуемых от 200 до 3FF.
                    ;3897      ;       Оставшаяся часть ОЗУ TB либо не используется, либо содержит информацию бу-
                    ;3898      ;       фера трансляции. Для записи в ОЗУ буфера трансляции используется команда
                    ;3899      ;       DEPOSIT TB. Адрес, заданный командой DEPOSIT, является действительным ад-
                    ;3900      ;       ресом обращения к ОЗУ. Эта программа выполняет все сдвиги заданного адреса,
                    ;3901      ;       необходимые для правильного его расположения. Заданный адрес загружается
                    ;3902      ;       консольным процессором в ячейку LS5 прежде, чем выполняется эта программа.
                    ;3903      ;       Адрес должен иметь 16-ричный формат.
                    ;3904      ;          Требуемые данные размещаются консольным процессором в LS6 до выполне-
                    ;3905      ;       ния этой программы. Биты 07-01 являются битами VALID, PROT, MODIFY и BYTE
                    ;3906      ;       OFFSET (имеются четыре бита PROT). Биты 22-08 содержат биты PA от PA23 до
                    ;3907      ;       PA9 соответственно. Биты от 31 до 23 и бит 0 не используются. Эта програм-
                    ;3908      ;       ма выполняет все сдвиги, необходимые для записи этих битов в TB, как опи-
                    ;3909      ;       сано. Заданная информация должна быть в 16-ричном формате.
                    ;3910      ;
                    ;3911      ; Процедура вызова:
                    ;3912      ;
```

```
                ;3913      ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес ука-
                ;3914      ;       зателя этой программы (инструкции JMP в ячейке 58) и запускает централь-
                ;3915      ;       ный процессор.
                ;3916      ;
                ;3917      ; Входные параметры:
                ;3918      ;
                ;3919      ;       LS5 - адрес области отображения общей шины в ТВ (должен находиться в диа-
                ;3920      ;             пазоне 200-3FF шестнадцатер.).
                ;3921      ;       LS6 - в битах 1-22 содержит данные, подлежащие записи в область отобра-
                ;3922      ;             жения общей шины.
                ;3923      ;
                ;3924      ; Неявные входные данные:
                ;3925      ;
                ;3926      ;       Отсутствуют.
                ;3927      ;
                ;3928      ; Выходные параметры:
                ;3929      ;
                ;3930      ;       Часть ОЗУ ТВ, составляющая область отображения общей шины, записывается
                ;3931      ;       данными из LS6 по адресу, заданному в LS5.
                ;3932      ;
                ;3933      ; Неявные выходные данные:
                ;3934      ;
                ;3935      ;       Отсутствуют.
                ;3936      ;
                ;3937      ; Побочный эффект:
                ;3938      ;
                ;3939      ;       Будет модифицирован WR0.
                ;3940      ;       Будет модифицирован WR1.
                ;3941      ;
                ;3942      ;---
                ;3943      REPOSIT.UBS:
U 0642, 0B64,DC ;3944              JSR [SHIFT.UB.ADR]                      ; переход к подпрограмме для получения правильного
                ;3945                                                      ; адреса области отображения общей шины
U 0643, 360C,15 ;3946              MOV LS[T6.SUB] TO WR[0]                 ; выборка данных из LS6
U 0644, 4526,15 ;3947              BIC LS[#FF] TO WR[0]                    ; очистка младшего байта заданной информации. Младший
                ;3948                                                      ; байт будет позднее размещен в битах 24-31
U 0645, 8B62,AC ;3949              JSR [SHIFT.LOOP]                        ; переход к подпрограмме сдвига WR0 на 8 позиций вправо
U 0646, E40C,15 ;3950              SWAP LS[T6.SUB] WITH WR[0]              ; взаимная замена заданных исходных данных с
                ;3951                                                      ; модифицированными данными. LS6 теперь содержит биты
                ;3952                                                      ; 8-22 в битах 0-14, как и требовалось. WR0 содержит
                ;3953                                                      ; заданные исходные данные
U 0647, 452C,15 ;3954              BIC LS[#FFFFFF00] TO WR[0]               ; очистка всех битов за исключением младшего байта
                ;3955                                                      ; заданных исходных данных
U 0648, 8B62,AC ;3956              JSR [SHIFT.LOOP]                        ; переход к подпрограмме сдвига WR0 на 8 позиций
                ;3957                                                      ; вправо. После возврата WR0 будет содержать биты 0-7 на
                ;3958                                                      ; месте битов 24-31. Другие биты очищены
U 0649, ED0C,15 ;3959              BIS WR[0] TO LS[T6.SUB]                  ; теперь LS6 содержит данные, подлежащие записи в ТВ, в
                ;3960                                                      ; правильном формате, т.е. биты PA в битах 0-14 и BYTE
                ;3961                                                      ; OFFSET, MODIFY, PROT от A до D и VALID в битах 25-31
U 064A, 1B0B,F5 ;3962              MEM.REQ[WRITE.UBS.MAP] ADRS[T5.SUB] DT[LONG]   ; выдача запроса к памяти для записи в область
                ;3963                                                      ; отображения общей шины
U 064B, B20C,15 ;3964              WRITE.MEM LS[T6.SUB]                    ; запись данных из LS6 в область отображения общей шины
U 064C, 8B6B,74 ;3965              JMP BWAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;3966      ;
                ;3967      ;       Подпрограмма правильного позиционирования заданного адреса для
```

```
                        ;3968       ;       адресации области отображения адресов общей шины.
                        ;3969       ;
                        ;3970       SHIFT.UB.ADR:
U 064D, 360A,15         ;3971               MOV LS[TS.SUB] TO WR[0]                  ; выборка заданного адреса в WR0
U 064E, A2D0,15         ;3972               SHL2 WR[0]                              ; сдвиг адреса на 2 позиции влево
U 064F, A2D0,15         ;3973               SHL2 WR[0]                              ; сдвиг адреса на 2 позиции влево
U 0650, A2D0,15         ;3974               SHL2 WR[0]                              ; сдвиг адреса на 2 позиции влево
U 0651, A2D0,15         ;3975               SHL2 WR[0]                              ; сдвиг адреса на 2 позиции влево
U 0652, 72D0,15         ;3976               ASHL WR[0]                              ; сдвиг еще на 1 позицию. Теперь адрес находится в битах
                        ;3977                                                       ; 9-17 WR0
                        ;3978                MOV WR[0] TO LS[TS.SUB],               ; теперь LS5 содержит правильный адрес
U 0653, 3E0A,14         ;3979               RETURN                                  ; возврат к основной программе
                        ;3980       ;
                        ;3981       ;       Программа индикации буфера трансляции адресов общей шины
                        ;3982       ;***
                        ;3983       ;          Эта программа читает из той части контроллера памяти, которая составляет
                        ;3984       ;       область отображения общей шины. Содержимое области отображения общей шины
                        ;3985       ;       будет получено с битами BYTE OFFSET, MODIFY, PROT от A до D и VALID в битах
                        ;3986       ;       01-07 соответственно. Биты PA 09-23 поступают в битах 08-22 соответственно.
                        ;3987       ;       Биты 23-31 и бит 0 не используются, поэтому будут очищены до печати резуль-
                        ;3988       ;       тата. результат для печати на консоли будет помещен в LS6.
                        ;3989       ;          Эта программа выдает запрос к памяти с функцией памяти READ.UBS.MAP и
                        ;3990       ;       типом данных=LONGWORD (длинное слово). Консольный процессор загружает LS5
                        ;3991       ;       заданным адресом прежде, чем выполняется данная программа. Эта программа
                        ;3992       ;       сдвигает адрес так, как требуется для адресации заданной ячейки области
                        ;3993       ;       отображения. Заданный адрес должен быть в 16-ричном формате.
                        ;3994       ;          Область отображения ОШ состоит из 512 дес. ячеек (адреса от 200 до 3FF).
                        ;3995       ;       Оставшиеся ячейки в ОЗУ ТВ либо не используются, либо используются буфером
                        ;3996       ;       трансляции. Содержимое буфера трансляции считывается посредством программы
                        ;3997       ;       EXAMINE TB.
                        ;3998       ;
                        ;3999       ; Процедура вызова:
                        ;4000       ;
                        ;4001       ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес ука-
                        ;4002       ;       зателя этой программы (инструкции JUMP в ячейке WCS 59) и запускает цент-
                        ;4003       ;       ральный процессор.
                        ;4004       ;
                        ;4005       ; Входные параметры:
                        ;4006       ;
                        ;4007       ;       LS5 - адрес ТВ в шестнадцатеричном формате.
                        ;4008       ;
                        ;4009       ; Неявные входные данные:
                        ;4010       ;
                        ;4011       ;       Отсутствуют.
                        ;4012       ;
                        ;4013       ; Выходные параметры:
                        ;4014       ;
                        ;4015       ;       LS6 - данные из области отображения общей шины по заданному адресу.
                        ;4016       ;
                        ;4017       ; Неявные выходные данные:
                        ;4018       ;
                        ;4019       ;       Отсутствуют.
                        ;4020       ;
                        ;4021       ; Побочный эффект:
                        ;4022       ;
```

```
                    ;4023     ;       LS5 - будет модифицироваться.
                    ;4024     ;       WR0 - будет модифицироваться.
                    ;4025     ;
                    ;4026     ;---
                    ;4027     EXAMINE.UBS:
U 0654,  0864,DC    ;4028             JSR [SHIFT.UB.ADR]                    ; сдвиг адреса, заданного в LS5, в правильное положение и
                    ;4029                                                   ; замена LS5
U 0655,  1C08,75    ;4030             MEM.REQ[READ.UBS.MAP] ADRS[T5.SUB] DT[LONG]   ; выдача запроса к памяти для чтения области
                    ;4031                                                   ; отображения ОШ по адресу, содержащемуся в LS5
U 0656,  3022,15    ;4032             MOV MEM.DATA TO WR[0]                 ; занесение результата в WR0
U 0657,  452A,15    ;4033             BIC LS[#FF000000] TO WR[0]            ; очистка старшего байта результата (старший байт не
                    ;4034                                                   ; является частью ТВ и является непредсказуемым)
U 0658,  456E,15    ;4035             BIC LS[BIT23] TO WR[0]                ; то же для бита 23
U 0659,  4540,15    ;4036             BIC LS[BIT0] TO WR[0]                 ; то же для бита 0. WR0 теперь содержит результат, готовый
                    ;4037                                                   ; для печати
U 065A,  BE0C,15    ;4038             MOV WR[0] TO LS[T6.SUB]               ; результат находится в LS6
U 065B,  8868,74    ;4039             JMP [WAIT.SUB]                        ; переход к установке ATTN и ожиданию
                    ;4040     ;
                    ;4041     ;       Программа записи в основную память
                    ;4042     ;***
                    ;4043     ;
                    ;4044     ; Описание функционирования:
                    ;4045     ;
                    ;4046     ;          Эта программа записывает в основную память по заданному адресу длинное
                    ;4047     ;       слово данных. Адрес не обязательно должен быть на границе длинного слова.
                    ;4048     ;       Консольный процессор использует эту программу для пересылки данных в ос-
                    ;4049     ;       новную память, а также и для команды DEPOSIT.MM. Консольный процессор дол-
                    ;4050     ;       жен загружать адрес в LS5 и данные в LS6 до выполнения этой программы.
                    ;4051     ;          Эта программа выдает запрос для памяти с функцией памяти WRITE.P и ти-
                    ;4052     ;       пом данных=длинное слово. Затем она выдает инструкцию WRITE.MEM LS[6] для
                    ;4053     ;       записи данных в память. Для того, чтобы убедиться, что запись произошла
                    ;4054     ;       без ошибок, контролируется ERR.SUM. Если установлен сигнал ERR.SUM, то
                    ;4055     ;       центральный процессор, прежде, чем выставить CPU ATTN, устанавливает CPU
                    ;4056     ;       ACK для информирования консольного процессора, что произошла ошибка.
                    ;4057     ;
                    ;4058     ; Процедура вызова:
                    ;4059     ;
                    ;4060     ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес ука-
                    ;4061     ;       зателя этой программы (инструкции JMP в ячейке WCS 54) и запускает цент-
                    ;4062     ;       ральный процессор.
                    ;4063     ;
                    ;4064     ; Входные параметры:
                    ;4065     ;
                    ;4066     ;       LS5 - физический адрес основной памяти
                    ;4067     ;       LS6 - длинное слово данных для основной памяти
                    ;4068     ;
                    ;4069     ; Неявные входные данные:
                    ;4070     ;
                    ;4071     ;       Отсутствуют
                    ;4072     ;
                    ;4073     ; Выходные параметры:
                    ;4074     ;
                    ;4075     ;       В основной памяти по заданному адресу записаны данные из LS6
                    ;4076     ;
                    ;4077     ; Неявные выходные данные:
```

```
                    ;4078     ;
                    ;4079     ;       Ошибка памяти вызывает установку CPU ACK для информирования консольного
                    ;4080     ;       процессора об ошибке записи
                    ;4081     ;
                    ;4082     ; Побочный эффект:
                    ;4083     ;
                    ;4084     ;       Отсутствует
                    ;4085     ;
                    ;4086     ;---
                    ;4087     DEPOSIT.MM:
U 065C,  990A,75    ;4088             MEM.REQ[WRITE.P] ADRSIT5.SUB] DT[LONG]    ; инициирование записи в основную память с
                    ;4089                                                       ; использованием физического адреса, хранящегося в LS5
                    ;4090               WRITE.MEM LS[T6.SUB],                   ; запись данных из LS6 в основную память и
U 065D,  320C,1D    ;4091             SKIP.IF[MEM.REF.OK]                       ; пропуск следующей инструкции, если ошибки памяти
                    ;4092                                                       ; отсутствуют (нет ERR SUM)
U 065E,  10D0,15    ;4093             MISC [SET.CP.ACK]                         ; установка CPU ACK, если выставлен сигнал ERR.SUM
U 065F,  3644,15    ;4094             MOV LS[#4] TO WR[0]                       ; занесение числа 4 в WR0
U 0660,  6C0A,15    ;4095             ADD WR[0] TO LS[T5.SUB]                   ; увеличение LS5 для записи следующего длинного слова
U 0661,  8868,74    ;4096             JMP [WAIT.SUB]                            ; переход к установке ATTN и ожиданию
                    ;4097     ;
                    ;4098     ;       Программа индикации основной памяти
                    ;4099     ;***
                    ;4100     ;
                    ;4101     ; Описание функционирования:
                    ;4102     ;
                    ;4103     ;          Эта программа используется для чтения из основной памяти длинного сло-
                    ;4104     ;       ва по заданному физическому адресу. Адрес не обязательно должен быть на
                    ;4105     ;       границе длинного слова. Эта программа используется консольным процессором
                    ;4106     ;       при команде EXAMINE.MM. Прежде, чем выполняется эта программа, консольный
                    ;4107     ;       процессор должен загрузить в LS5 заданный физический адрес.
                    ;4108     ;          Эта программа производит вначале запись в CSR1 для установки бита INH
                    ;4109     ;       REP CRD (запрет сообщения об исправимых ошибках чтения), чтобы препятст-
                    ;4110     ;       вовать появлению ошибки системы памяти (ERR SUM) при одиночной ошибке, ко-
                    ;4111     ;       торая была исправлена. Затем программа выдает запрос к памяти с функцией
                    ;4112     ;       READ.P и типом данных=длинное слово. Затем она читает ячейку памяти и по-
                    ;4113     ;       мещает результат в LS6. Она также проверяет ERR SUM и, если произошла
                    ;4114     ;       ошибка, выдает CPU ACK (исправленные ошибки в одиночных битах не вызывают
                    ;4115     ;       ERR SUM). Если произошла ошибка, то бит CPU ACK будет установлен до выдачи
                    ;4116     ;       CPU ATTN для информирования консольного процессора, что имеется ошибка.
                    ;4117     ;
                    ;4118     ; Процедура вызова:
                    ;4119     ;
                    ;4120     ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес ука-
                    ;4121     ;       зателя этой программы (инструкция JMP в ячейке WCS 55) и запускает цент-
                    ;4122     ;       ральный процессор.
                    ;4123     ;
                    ;4124     ; Входные параметры:
                    ;4125     ;
                    ;4126     ;       LS5 - физический адрес основной памяти, по которому будет происходить
                    ;4127     ;             чтение
                    ;4128     ;
                    ;4129     ; Неявные входные данные:
                    ;4130     ;
                    ;4131     ;       LS[INH.CRD] - данные для записи в CSR1 с целью запрета ERR SUM для
                    ;4132     ;                     исправимых ошибок чтения (CRD)
```

```
                    ;4133      ;       LS[CSR1]    - адрес для записи CSR1
                    ;4134      ;
                    ;4135      ; Выходные параметры:
                    ;4136      ;
                    ;4137      ;       LS6 - длинное слово данных, считанное по заданному физическому адресу
                    ;4138      ;             памяти
                    ;4139      ;
                    ;4140      ; Неявные выходные данные:
                    ;4141      ;
                    ;4142      ;       CPU ACK, если зафиксирована ошибка
                    ;4143      ;
                    ;4144      ; Побочный эффект:
                    ;4145      ;
                    ;4146      ;       Отсутствует
                    ;4147      ;
                    ;4148      ;--
                    ;4149      EXAMINE.MM:
U 0662, 1D45,F5     ;4150              MEM.REQ[WRITE.CSR] ADRS[CSR1] DT[LONG]   ; выдача запроса к памяти для записи CSR1
U 0663, B278,15     ;4151              WRITE.MEM LS[INH.CRD]                    ; установка в CSR1 бита INH REP CRD и очистка запрета
                    ;4152                                                       ; коррекции ошибок (ECC)
U 0664, 180B,F5     ;4153              MEM.REQ[READ.P] ADRS[T5.SUB] DT[LONG]    ; выдача запроса к памяти для чтения длинного слова
                    ;4154                                                       ; данных из основной памяти. Физический адрес находится
                    ;4155                                                       ; в LS5
                    ;4156                MOV MEM.DATA TO LS[T6.SUB],            ; чтение данных и занесение в LS6
U 0665, 380C,1D     ;4157              SKIP.IF[MEM.REF.OK]                      ; пропуск следующей инструкции, если ошибки памяти
                    ;4158                                                       ; отсутствуют (нет ERR SUM)
U 0666, 10D0,15     ;4159              MISC [SET.CP.ACK]                        ; установка CPU ACK, если произошла ошибка
U 0667, 8B68,74     ;4160              JMP [WAIT.SUB]                           ; переход к установке ATTN и ожиданию
                    ;4161      ;
                    ;4162      ;       Программа индикации регистров IDC
                    ;4163      ;***
                    ;4164      ;
                    ;4165      ; Описание функционирования:
                    ;4166      ;
                    ;4167      ;       Следующие программы используются для чтения данных из регистров необяза-
                    ;4168      ;       тельного модуля IDC. Результат для печати на консоли заносится в LS6.
                    ;4169      ;
                    ;4170      ; Процедура вызова:
                    ;4171      ;
                    ;4172      ;       Консольный процессор загружает в счетчик микроинструкций (UPC) адрес ука-
                    ;4173      ;       зателя этой программы (инструкции JMP в ячейках WCS от 5A до 5E) и запус-
                    ;4174      ;       кает центральный процессор
                    ;4175      ;
                    ;4176      ; Входные параметры:
                    ;4177      ;
                    ;4178      ;       Отсутствуют
                    ;4179      ;
                    ;4180      ; Неявные входные данные:
                    ;4181      ;
                    ;4182      ;       Отсутствуют
                    ;4183      ;
                    ;4184      ; Выходные параметры:
                    ;4185      ;
                    ;4186      ;       LS6 - данные из заданного регистра IDC
                    ;4187      ;
```

```
                ;4188     ; Неявные выходные данные:
                ;4189     ;
                ;4190     ;       Отсутствуют
                ;4191     ;
                ;4192     ; Побочный эффект:
                ;4193     ;
                ;4194     ;       Отсутствует
                ;4195     ;
                ;4196     ;---
                ;4197     EXAMINE.ICSR:
U 0668, 9090,15 ;4198             MISC [SET.PORT.I.O]                     ; выборка порта на шину
U 0669, 9780,15 ;4199             PORT.READ PORT.ADRS[CSR]                ; пересылка команды чтения
U 066A, 0867,64 ;4200             JMP [READ.IDC]                          ; переход к чтению данных
                ;4201     EXAMINE.IDAR:
U 066B, 9090,15 ;4202             MISC [SET.PORT.I.O]                     ; выборка порта на шину
U 066C, 1784,15 ;4203             PORT.READ PORT.ADRS[DAR]                ; пересылка команды чтения
U 066D, 0867,64 ;4204             JMP [READ.IDC]                          ; переход к чтению данных
                ;4205     EXAMINE.DBUF:
U 066E, 9090,15 ;4206             MISC [SET.PORT.I.O]                     ; выборка порта на шину
U 066F, 978C,15 ;4207             PORT.READ PORT.ADRS[DATA.WORD]          ; пересылка команды чтения
U 0670, 0867,64 ;4208             JMP [READ.IDC]                          ; переход к чтению данных
                ;4209     EXAMINE.PATT:
U 0671, 9090,15 ;4210             MISC [SET.PORT.I.O]                     ; выборка порта на шину
U 0672, 1790,15 ;4211             PORT.READ PORT.ADRS[PATTERN]            ; пересылка команды чтения
U 0673, 0867,64 ;4212             JMP [READ.IDC]                          ; переход к чтению данных
                ;4213     EXAMINE.POSIT:
U 0674, 9090,15 ;4214             MISC [SET.PORT.I.O]                     ; выборка порта на шину
U 0675, 9794,15 ;4215             PORT.READ PORT.ADRS[POSITION]           ; пересылка команды чтения
                ;4216     READ.IDC:
U 0676, 3A0C,15 ;4217             MOV PORT TO LS[T6.SUB]                  ; чтение данных и занесение в LS6
U 0677, 1080,15 ;4218             MISC [SET.ACC.I.O]                      ; возврат к выборке ускорителя
U 0678, 8868,74 ;4219             JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4220     ;
                ;4221     ;       Программы записи в регистры IDC
                ;4222     ;***
                ;4223     ;
                ;4224     ; Описание функционирования:
                ;4225     ;
                ;4226     ;       Следующие программы используются для записи данных в регистры необяза-
                ;4227     ;       тельного модуля IDC. Данные берутся из LS6, которая предварительно заг-
                ;4228     ;       ружается монитором, если выполняется команда DEPOSIT.
                ;4229     ;
                ;4230     ; Процедура вызова:
                ;4231     ;
                ;4232     ;       Консольный процессор загружает в счетчик микроинструкций (UPC) указатель
                ;4233     ;       этой программы (инструкции JMP в ячейках WCS от 5F до 61) и запускает
                ;4234     ;       центральный процессор.
                ;4235     ;
                ;4236     ; Входные параметры:
                ;4237     ;
                ;4238     ;       LS6 - код данных, подлежащих записи
                ;4239     ;
                ;4240     ; Неявные входные данные:
                ;4241     ;
                ;4242     ;       Отсутствуют
```

```
                ;4243     ;
                ;4244     ; Выходные параметры:
                ;4245     ;
                ;4246     ;       Отсутствуют
                ;4247     ;
                ;4248     ; Неявные выходные параметры:
                ;4249     ;
                ;4250     ;       Отсутствуют
                ;4251     ;
                ;4252     ; Побочный эффект:
                ;4253     ;
                ;4254     ;       Отсутствует
                ;4255     ;
                ;4256     ;---
                ;4257    DEPOSIT.ICSR:
U 0679, 360C,15 ;4258            MOV LS[T6.SUB] TO WR[0]                 ; выборка данных, подлежащих записи
U 067A, 17A0,15 ;4259            PORT.WRITE PORT.ADRS[CSR] WITH WR[0]    ; запись в IDC
U 067B, 8868,74 ;4260            JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4261    DEPOSIT.IDAR:
U 067C, 360C,15 ;4262            MOV LS[T6.SUB] TO WR[0]                 ; выборка данных, подлежащих записи
U 067D, 97A4,15 ;4263            PORT.WRITE PORT.ADRS[IDAR] WITH WR[0]   ; запись в IDC
U 067E, 8868,74 ;4264            JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4265    DEPOSIT.DBUF:
U 067F, 360C,15 ;4266            MOV LS[T6.SUB] TO WR[0]                 ; выборка данных, подлежащих записи
U 0680, 17AC,15 ;4267            PORT.WRITE PORT.ADRS[DATA.WORD] WITH WR[0]  ; запись в IDC
U 0681, 8868,74 ;4268            JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4269    CLEAR.FIFO.ADDR:
U 0682, 17C0,15 ;4270            PORT.CONTROL [CLEAR.FIFO.CNTR]          ; очистка адреса FIFO A или FIFO B
U 0683, 8868,74 ;4271            JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4272    SELECT.FIFO.A:
U 0684, 17D8,15 ;4273            PORT.CONTROL [SEL.FIFO.A]               ; выборка FIFO A
U 0685, 8868,74 ;4274            JMP [WAIT.SUB]                          ; переход к установке ATTN и ожиданию
                ;4275    SELECT.FIFO.B:
U 0686, 97DC,15 ;4276            PORT.CONTROL [SEL.FIFO.B]               ; выборка FIFO B
                ;4277    WAIT.SUB:
U 0687, 10E0,15 ;4278            MISC [SET.CP.ATTN]                      ; выдача консольному процессору сигнала CPU ATTN
                ;4279    WAIT.DE.EX:
U 0688, 8868,84 ;4280            JMP [WAIT.DE.EX]                        ; зацикливание на себя до тех пор, пока консольный
                ;4281                                                    ; процессор возьмет управление
                ;4282     ;
                ;4283     ;       Программа запоминания рабочих регистров
                ;4284     ;***
                ;4285     ;
                ;4286     ; Описание функционирования:
                ;4287     ;
                ;4288     ;          Эта программа запоминает рабочие регистры 0-3 в ячейках местной памяти
                ;4289     ;       LS 1-4. Если консольный процессор забирает управление от центрального про-
                ;4290     ;       цессора, он вызывает эту программу для того, чтобы рабочие регистры могли
                ;4291     ;       использоваться консольным процессором. Рабочие регистры восстанавливаются
                ;4292     ;       другой программой, вызываемой консольным процессором прежде, чем централь-
                ;4293     ;       ный процессор возобновляет выполнение (например, при команде CONTINUE).
                ;4294     ;          Эта программа просто пересылает данные из WR 0-3 в ячейки LS 1-4 и за-
                ;4295     ;       тем выдает для консольного процессора CPU ATTN. Затем она зацикливается
                ;4296     ;       инструкцией JMP на себя до тех пор, пока консольный процессор возьмет
                ;4297     ;       управление.
```

```
                   ;
                   ; Процедура вызова:
                   ;
                   ;       Консольный процессор загружает в счетчик микроинструкций (UPC) указатель
                   ;       этой программы (инструкцию JMP в ячейке WCS 46) и запускает центральный
                   ;       процессор.
                   ;
                   ; Входные параметры:
                   ;.
                   ;       Отсутствуют
                   ;
                   ; Неявные входные данные:
                   ;
                   ;       Отсутствуют
                   ;
                   ; Выходные параметры:
                   ;
                   ;       LS1 - содержимое WR0
                   ;       LS2 - содержимое WR1
                   ;       LS3 - содержимое WR2
                   ;       LS4 - содержимое WR3
                   ;
                   ; Неявные выходные данные:
                   ;
                   ;       Отсутствуют
                   ;
                   ; Побочный эффект:
                   ;
                   ;       Отсутствует
                   ;
                   ;---
                  SAVE.WR:
U 0689, 3E02,15          MOV WR[0] TO LS[SAV.WR0]          ; запоминание WR0 в ячейке LS1
U 068A, BE04,95          MOV WR[1] TO LS[SAV.WR1]          ; запоминание WR1 в ячейке LS2
U 068B, 3E07,15          MOV WR[2] TO LS[SAV.WR2]          ; запоминание WR2 в ячейке LS3
U 068C, 3E09,95          MOV WR[3] TO LS[SAV.WR3]          ; запоминание WR3 в [illegible] LS4
U 068D, 8868,74          JMP [WAIT.SUB]                    ; переход к установке [illegible] ожидания
                   ;
                   ;       Программа восстановления рабочих регистров
                   ;***
                   ;
                   ; Описание функционирования:
                   ;
                   ;          Эта программа восстанавливает рабочие регистры К1804ВС1, хранящиеся в
                   ;       LS 1-4. Консольный процессор вызывает эту программу прежде, чем запускает
                   ;       ся центральный процессор после останова по инициативе [illegible]
                   ;       сора.
                   ;          Эта программа просто пересылает данные из ячеек LS 1-4 в рабочие регис-
                   ;       тры (WR) 0-3 и затем выдает для консольного процессора CPU [illegible] После
                   ;       этого выполняется инструкция JMP на себя, пока консольный процессор не
                   ;       возобновит управление.
                   ;
                   ; Процедура вызова:
                   ;
                   ;       Консольный процессор загружает в счетчик микроинструкций [illegible]
```

```
                ;4353     ;       зателя этой программы (инструкций JMP в ячейке WCS 47) и запускает цент-
                ;4354     ;       ральный процессор.
                ;4355     ;
                ;4356     ; Входные параметры:
                ;4357     ;
                ;4358     ;       LS1 содержит данные для восстановления WR0
                ;4359     ;       LS2 содержит данные для восстановления WR1
                ;4360     ;       LS3 содержит данные для восстановления WR2
                ;4361     ;       LS4 содержит данные для восстановления WR3
                ;4362     ;
                ;4363     ; Неявные входные данные:
                ;4364     ;
                ;4365     ;       Отсутствуют
                ;4366     ;
                ;4367     ; Выходные параметры:
                ;4368     ;
                ;4369     ;       WR0 получает данные из LS1
                ;4370     ;       WR1 получает данные из LS2
                ;4371     ;       WR2 получает данные из LS3
                ;4372     ;       WR3 получает данные из LS4
                ;4373     ;
                ;4374     ; Неявные выходные данные:
                ;4375     ;
                ;4376     ;       Отсутствуют
                ;4377     ;
                ;4378     ; Побочный эффект:
                ;4379     ;
                ;4380     ;       Отсутствует
                ;4381     ;
                ;4382     ;---
                ;4383     RESTORE.WR:
U 068E, B602,15 ;4384             MOV LS[SAV.WR0] TO WR[0]              ; восстановление WR0
U 068F, 3604,95 ;4385             MOV LS[SAV.WR1] TO WR[1]              ; восстановление WR1
U 0690, B607,15 ;4386             MOV LS[SAV.WR2] TO WR[2]              ; восстановление WR2
U 0691, B609,95 ;4387             MOV LS[SAV.WR3] TO WR[3]              ; восстановление WR3
U 0692, 8868,74 ;4388             JMP [WAIT.SUB]                       ; переход к установке ATTN и ожиданию
                ;4389     ;
```

```
;4390      .PAGE "ПОДПРОГРАММЫ WCS ДЛЯ НУЖД ЦЕНТРАЛЬНОГО ПРОЦЕССОРА"
;4391      ;
;4392      ;            Программа обработки ошибок
;4393      ;
;4394      ; Описание функционирования:
;4395      ;
;4396      ;     Эта программа используется для обнаружения ошибок и выдачи диагностических
;4397      ;   сообщений об ошибках, появляющихся во время выполнения микродиагностики, ба-
;4398      ;   зируемой на WCS. Микрокод WCS устанавливает параметры, перечисленные ниже, и
;4399      ;   вызывает эту программу. Эта программа сравнивает WR1 (ожидаемые данные) и
;4400      ;   WR0 (полученные данные) в соответствии с маской ошибок. Установленные биты
;4401      ;   маски указывают те биты, которые не проверяются на наличие ошибки.
;4402      ;     Если WR0 и WR1 равны (ошибок нет), программа выполняет возврат+1 к точке
;4403      ;   вызова. Единственным исключением является условие, когда установлено зацикли-
;4404      ;   вание на ошибке. В этом случае проверяется номер ошибки, чтобы убедиться,
;4405      ;   происходила ли эта ошибка ранее. Если да, то вынуждается цикл по ошибке, пу-
;4406      ;   тем выполнения возврата. Таким образом фиксируется цикл для неустойчивых
;4407      ;   ошибок.
;4408      ;     Если WR0 и WR1 не равны (ошибка), тогда в слове управления и состояния уста-
;4409      ;   навливается бит 8, ожидаемые и полученные данные запоминаются в LS для печа-
;4410      ;   ти и проверяются признаки в слове управления ошибками. Оно управляет запре-
;4411      ;   том печати ошибок, разрешением зацикливания на ошибке и т.д. После печати
;4412      ;   ошибки выполняется возврат+1 к точке вызова. Зацикливание на ошибке вместо
;4413      ;   возврата+1 вызывает возврат, чтобы вынудить зацикливание по ошибке. Также, в
;4414      ;   конце программы переключается CPU ACK для образования точки синхронизации
;4415      ;   осциллографа.
;4416      ;
;4417      ; Процедура вызова:
;4418      ;
;4419      ;   JSR [CHECK.RESULT]
;4420      ;
;4421      ; Входные параметры:
;4422      ;
;4423      ;   WR[0] = полученные данные, т.е. действительный результат теста
;4424      ;   WR[1] = ожидаемые данные, т.е. такой результат, каким он должен быть
;4425      ;
;4426      ; Неявные входные данные:
;4427      ;
;4428      ;   LS[ERROR.MASK]      = используется для маскирования битов, которые не подле-
;4429      ;                         жат проверке
;4430      ;   LS[ERROR.NUMBER]    = номер текущей ошибки
;4431      ;   LS[PREVIOUS.ERROR]  = номер ошибки предыдущих данных
;4432      ;   LS[CONTROL.STATUS]  = слово управления и состояния. Содержит информацию для
;4433      ;                         монитора о необходимых действиях, записанную централь-
;4434      ;                         ным процессором
;4435      ;   LS[ERROR.CONTROL]   = информация такого типа, как зацикливание по ошибке,
;4436      ;                         звуковой сигнал при ошибке, запрет сообщений об ошиб-
;4437      ;                         ках и останов при ошибке.
;4438      ;
;4439      ; Выходные параметры:
;4440      ;
;4441      ;   Отсутствуют
;4442      ;
;4443      ; Неявные выходные данные:
;4444      ;
```

```
                    ;4445      ;    LS[CONTROL.STATUS] = слово управления и состояния с новой информацией
                    ;4446      ;    LS[PREVIOUS.ERROR] = номер последней ошибки (модифицируется только при ошибке,
                    ;4447      ;                         если разрешено зацикливание при ошибке)
                    ;4448      ;    LS[EXPECTED.DATA]  = ожидаемый результат, если была обнаружена ошибка
                    ;4449      ;    LS[RECEIVED.DATA]  = действительный результат, если была обнаружена ошибка
                    ;4450      ;
                    ;4451      ; Побочный эффект:
                    ;4452      ;
                    ;4453      ;    WR0, WR1 и регистр Q модифицированы и не восстановлены перед возвратом.
                    ;4454      ;    Если разрешено зацикливание при ошибке и должен выполняться цикл по ошибке,
                    ;4455      ;    будет переключен CPU ACK для синхронизации осциллографа.
                    ;4456      ;
                    ;4457      CHECK.RESULT:
U 0693, 458A,15     ;4458              BIC LS[ERROR.MASK] TO WR[0]                ; маскирование непроверяемых битов (очистка) для
                    ;4459                                                         ; полученных данных
U 0694, C58A,95     ;4460              BIC LS[ERROR.MASK] TO WR[1]                ; то же для ожидаемых данных
                    ;4461                XOR WR[0] WITH WR[1] TO Q,                ; сравнение полученных данных в WR[0] с ожидаемыми
                    ;4462                                                         ; данными в WR[1] для обнаружения ошибки
U 0695, AB80,B5     ;4463              DT(LONG)&SET.ALU.CC                        ;
U 0696, 8869,F1     ;4464              JMP.IF[NEQ] TO [ERROR]                     ; переход, если ошибка
U 0697, 3690,15     ;4465              MOV LS[ERROR.CONTROL] TO WR[0]             ; выборка из LS слова управления ошибками
                    ;4466                BIT WR[0] WITH LS[LOE],                  ; проверка, разрешено ли зацикливание по ошибке и
U 0698, 5940,35     ;4467              DT(LONG)&SET.ALU.CC                        ; установка кодов условий
U 0699, DB00,01     ;4468              SKIP.IF[BIT.SET]                           ; пропуск следующей инструкции, если да
U 069A, D800,16     ;4469              RETURN+1                                   ; иначе возврат+1 (отсутствует ошибка и не разрешено
                    ;4470                                                         ; зацикливание)
U 069B, 36A0,15     ;4471              MOV LS[PREVIOUS.ERROR] TO WR[0]            ; если отсутствует ошибка, но зацикливание по ошибке
                    ;4472                                                         ; разрешено, проверка, была ли предыдущая ошибка
                    ;4473                XOR WR[0] WITH LS[ERROR.NUMBER] TO Q,    ;
U 069C, CD82,35     ;4474              DT(LONG)&SET.ALU.CC                        ; является ли это место тем же, в котором раньше была
                    ;4475                                                         ; ошибка (неустойчивая ошибка)?
U 069D, 0868,91     ;4476              JMP.IF[NEQ] TO [RET+1]                     ; если нет, переход к возврату без зацикливания по
                    ;4477                                                         ; ошибке (возврат+1)
U 069E, 0868,C4     ;4478              JMP [RET]                                  ; иначе цикл при ошибке (зацикливание будет происходить
                    ;4479                                                         ; даже если ошибка исчезла)
                    ;4480      ERROR:
U 069F, 3E86,15     ;4481              MOV WR[0] TO LS[RECEIVED.DATA]              ; занесение результата теста в LS для печати
U 06A0, 3690,15     ;4482              MOV LS[ERROR.CONTROL] TO WR[0]              ; выборка битов управления ошибками для проверки
                    ;4483                                                         ; разрешения зацикливания
                    ;4484                BIT WR[0] WITH LS[LOOP.COMMAND],         ; проверка, установлен ли бит 6
U 06A1, 594C,35     ;4485              DT(LONG)&SET.ALU.CC                        ; и установка кодов условий
U 06A2, 086B,C1     ;4486              JMP.IF[BIT.SET] TO [RET]                   ; переход на зацикливание, если установлен (быстрый
                    ;4487                                                         ; цикл)
U 06A3, 3E84,95     ;4488              MOV WR[1] TO LS[EXPECTED.DATA]              ; занесение в LS ожидаемых данных для печати
U 06A4, 3680,95     ;4489              MOV LS[CONTROL.STATUS] TO WR[1]             ; выборка из LS слова управления и состояния
U 06A5, C532,95     ;4490              BIC LS[#FE7FFFFF] TO WR[1]                  ; очистка всех битов слова управления и состояния, кроме
                    ;4491                                                         ; 23 и 24
U 06A6, C750,95     ;4492              BIS LS[ERROR] TO WR[1]                      ; установка в слове управления и состояния бита 8
                    ;4493                                                         ; (указывает, что обнаружена ошибка)
U 06A7, BE80,95     ;4494              MOV WR[1] TO LS[CONTROL.STATUS]             ; запись откорректированного слова управления и
                    ;4495                                                         ; состояния обратно в LS 40
                    ;4496                BIT WR[0] WITH LS[BELL],                 ; проверка, установлен ли бит звукового сигнала при
                    ;4497                                                         ; ошибке (WR0 все еще содержит слово управления
                    ;4498                                                         ; ошибками)
U 06AB, D944,35     ;4499              DT(LONG)&SET.ALU.CC                        ; установка кодов условий
```

```
U 06A9, 886A,D9 ;4500            JMP.IF[BITS.CLR] TO [CHECK.NER]        ; если бит "звуковой сигнал при ошибке" не установлен,
                ;4501                                                   ; переход для проверки, запрещаются или нет сообщения об
                ;4502                                                   ; ошибках
U 06AA, 10E0,15 ;4503            MISC [SET.CP.ATTN]                     ; иначе установка CPU ATTN (для звукового сигнала)
                ;4504    WAIT.1:
U 06AB, 086A,B4 ;4505            JMP [WAIT.1]                           ; ожидание ответа консольного процессора
U 06AC, 086B,64 ;4506            JMP [LOOP]                             ; переход к проверке цикла при ошибке после того, как
                ;4507                                                   ; консольный процессор закончил
                ;4508    CHECK.NER:
                ;4509            BIT WR[0] WITH LS[NER],                ; если не звуковой сигнал, проверка, запрещены ли
                ;4510                                                   ; сообщения об ошибках
U 06AD, D942,35 ;4511            DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 06AE, 886B,21 ;4512            JMP.IF[BIT.SET] TO [CHECK.HALT]        ; если сообщения об ошибках запрещены, переход к
                ;4513                                                   ; проверке останова по ошибке
U 06AF, 10E0,15 ;4514            MISC [SET.CP.ATTN]                     ; установка CPU ATTN (сообщение об ошибке)
                ;4515    WAIT.2:
U 06B0, 086B,04 ;4516            JMP [WAIT.2]                           ; ожидание ответа консольного процессора
U 06B1, 086B,64 ;4517            JMP [LOOP]                             ; переход к проверке зацикливания по ошибке после того,
                ;4518                                                   ; как консольный процессор закончил
                ;4519    CHECK.HALT:
                ;4520                BIT WR[0] WITH LS[HOE],            ; проверка, разрешен ли останов при ошибке и
U 06B2, 5946,35 ;4521            DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 06B3, 886B,69 ;4522            JMP.IF[BITS.CLR] TO [LOOP]             ; если нет, переход к проверке зацикливания при ошибке
U 06B4, 10E0,15 ;4523            MISC [SET.CP.ATTN]                     ; иначе установка CPU ATTN (останов при ошибке)
                ;4524    WAIT.3:
U 06B5, 086B,54 ;4525            JMP [WAIT.3]                           ; ожидание ответа консольного процессора
                ;4526    LOOP:
U 06B6, 3690,15 ;4527            MOV LS[ERROR.CONTROL] TO WR[0]         ; выборка битов управления ошибками (NER, HOE и т.д.)
                ;4528              BIT WR[0] WITH LS[LOE],              ; проверка, разрешено ли зацикливание при ошибке и
U 06B7, 5940,35 ;4529            DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 06B8, DB00,01 ;4530            SKIP.IF[BIT.SET]                       ; пропуск инструкции, если разрешено зацикливание при
                ;4531                                                   ; ошибке
                ;4532    RET+1:
U 06B9, DB00,16 ;4533            RETURN+1                               ; иначе возврат+1 в цикл по отсутствию ошибок
U 06BA, 3682,15 ;4534            MOV LS[ERROR.NUMBER] TO WR[0]          ; если зацикливание, выборка номера ошибки
U 06BB, BEA0,15 ;4535            MOV WR[0] TO LS[PREVIOUS.ERROR]        ; запоминание его по месту хранения предыдущей ошибки
                ;4536    RET:
U 06BC, 10D0,15 ;4537            MISC [SET.CP.ACK]                      ; установка CPU ACK для синхронизации осциллографа
                ;4538              MISC [CLR.CP.ATTN.AND.ACK],          ; и его очистка
U 06BD, 10C0,14 ;4539            RETURN                                 ; возврат к тесту и цикл при ошибке
                ;4540     ;
                ;4541     ;
                ;4542    ERR.NO.TEST:
U 06BE, DB00,15 ;4543            NOP                                    ; этот NOP вызывает ошибку в следовании тестов
U 06BF, B65E,15 ;4544            MOV LS[BEGIN.TEST] TO WR[0]            ; установка бита 15 в WR0 для слова управления и
                ;4545                                                   ; состояния
U 06C0, 3E80,15 ;4546            MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;4547                                                   ; 15 указывает конс.процессору начало теста
U 06C1, 886B,74 ;4548            JMP [WAIT.SUB]                         ; переход к установке ATTN и ожиданию
                ;4549     ;
                ;4550     ;
                ;4551     ;
                ;4552     ;
                ;4553     ;        Программа инициации переноса данных
                ;4554     ;
```

```
                    ;4555     ;   Эта программа загружает LS информацией, необходимой для выполнения тестов. За-
                    ;4556     ;   тем она выдает CPU ATTN с очищенным словом управления и состояния для ука-
                    ;4557     ;   зания того, что все пересылки завершены.
                    ;4558     ;
                    ;4559     ;   РЕЗУЛЬТАТ: LS загружена константами. Монитор информирован, что перенос дан-
                    ;4560     ;              ных завершен.
                    ;4561     ;
                    ;4562     TRANSFER.POINT:
U 06C2, 2F80,15     ;4563             CLR WR[0]                              ; начало. Очистка WR0
U 06C3, BFFE,15     ;4564             MOV WR[0] TO LS[PSL.HW]                 ; обеспечение очищенного бита режима совместимости
U 06C4, 2040,15     ;4565             INC WR[0]                              ; WR0 теперь содержит 1
U 06C5, 23D0,15     ;4566             ASHL WR[0]                             ; 10(B)
U 06C6, 2040,15     ;4567             INC WR[0]                              ; 11(B)
U 06C7, 23D0,15     ;4568             ASHL WR[0]                             ; 110(B)
U 06C8, 2040,15     ;4569             INC WR[0]                              ; 111(B)
U 06C9, 23D0,15     ;4570             ASHL WR[0]                             ; 1110(B)
U 06CA, 23D0,15     ;4571             ASHL WR[0]                             ; 1 1100(B)
U 06CB, 23D0,15     ;4572             ASHL WR[0]                             ; 11 1000(B)
U 06CC, 23D0,15     ;4573             ASHL WR[0]                             ; 111 0000(B) WR0 теперь содержит 70(H). Это правильный
                    ;4574                                                    ; начальный адрес секции данных (включая счетчик
                    ;4575                                                    ; длинных слов, приемник и адрес приемника)
U 06CD, 3E0E,15     ;4576             MOV WR[0] TO LS[TRANSFER.POINTER]       ; загрузка указателя секции данных, подлежащих переносу
                    ;4577                                                    ; в LS 7
U 06CE, 2F80,15     ;4578             CLR WR[0]                              ; 0 в WR0
U 06CF, 2040,15     ;4579             INC WR[0]                              ; 1 в WR0
U 06D0, 23D0,15     ;4580             ASHL WR[0]                             ; 10(B)
U 06D1, 23D0,15     ;4581             ASHL WR[0]                             ; 100(B)
U 06D2, 23D0,15     ;4582             ASHL WR[0]                             ; 1000(B)
U 06D3, 23D0,15     ;4583             ASHL WR[0]                             ; 1 0000(B)
U 06D4, 23D0,15     ;4584             ASHL WR[0]                             ; 10 0000(B)
U 06D5, 23D0,15     ;4585             ASHL WR[0]                             ; 100 0000(B)
U 06D6, 23D0,15     ;4586             ASHL WR[0]                             ; 1000 0000(B)
U 06D7, 23D0,15     ;4587             ASHL WR[0]                             ; 1 0000 0000(B)
U 06D8, 23D0,15     ;4588             ASHL WR[0]                             ; 10 0000 0000(B)
U 06D9, 23D0,15     ;4589             ASHL WR[0]                             ; 100 0000 0000(B)
U 06DA, 23D0,15     ;4590             ASHL WR[0]                             ; 1000 0000 0000(B)
U 06DB, 23D0,15     ;4591             ASHL WR[0]                             ; 1 0000 0000 0000(B)
U 06DC, 23D0,15     ;4592             ASHL WR[0]                             ; 10 0000 0000 0000(B)
U 06DD, 3E80,15     ;4593             MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 13 в слове управления и состояния в LS
                    ;4594                                                    ; (бит 13 указывает, что консольный процессор должен
                    ;4595                                                    ; выполнять перенос данных)
U 06DE, 10E0,15     ;4596             MISC [SET.CP.ATTN]                     ; выдача CPU ATTN для консольного процессора (выполнение
                    ;4597                                                    ; первой пересылки в LS в ячейки от 0 до 77(H))
                    ;4598     WAIT.XFER1:
U 06DF, 086D,F4     ;4599             JMP [WAIT.XFER1]                       ; зацикливание здесь в ожидании ответа консольного
                    ;4600                                                    ; процессора
U 06E0, 36CA,15     ;4601             MOV LS[#162(H)] TO WR[0]               ; выборка начального адреса второй половины данных в LS
U 06E1, 3E0E,15     ;4602             MOV WR[0] TO LS[TRANSFER.POINTER]       ; загрузка в LS начального адреса для консольного
                    ;4603                                                    ; процессора
U 06E2, 365A,15     ;4604             MOV LS[XFER.DATA] TO WR[0]              ; BIT 13 указывает на необходимость переноса данных
U 06E3, 3E80,15     ;4605             MOV WR[0] TO LS[CONTROL.STATUS]         ; установка слова управления и состояния
U 06E4, 10E0,15     ;4606             MISC [SET.CP.ATTN]                     ; выдача CPU ATTN для консольного процессора (выполнить
                    ;4607                                                    ; вторую пересылку в LS в ячейки от 80 до F7(H))
                    ;4608     WAIT.XFER2:
U 06E5, 086E,54     ;4609             JMP [WAIT.XFER2]                       ; зацикливание здесь в ожидании ответа консольного
```

```
                 ;4610                                                  ; процессора
U 06E6,  3022,15 ;4611          MOV MEM.DATA TO WR[0]                   ; эта инструкция используется для освобождения памяти,
                 ;4612                                                  ; если был приостанов в ожидании запроса данных из
                 ;4613                                                  ; центр.процессора в потоке данных восьмикратного
                 ;4614                                                  ; слова
U 06E7,  3022,15 ;4615          MOV MEM.DATA TO WR[0]                   ; требуются 4 таких инструкций
U 06E8,  3022,15 ;4616          MOV MEM.DATA TO WR[0]                   ;
U 06E9,  3022,15 ;4617          MOV MEM.DATA TO WR[0]                   ;
U 06EA,  17CC,15 ;4618          PORT.CONTROL [CLEAR.IDC]                ; сброс IDC в холостой цикл
U 06EB,  17CC,15 ;4619          PORT.CONTROL [CLEAR.IDC]                ;
U 06EC,  17CC,15 ;4620          PORT.CONTROL [CLEAR.IDC]                ;
U 06ED,  17CC,15 ;4621          PORT.CONTROL [CLEAR.IDC]                ; IDC сброшен
U 06EE,  B64E,95 ;4622          MOV LS[#80] TO WR[1]                    ; установка бита 7 в WR1
U 06EF,  97A0,95 ;4623          PORT.WRITE PORT.ADRS[CSR] WITH WR[1]    ; очистка CRDY В IDC
U 06F0,  17D4,15 ;4624          PORT.CONTROL [CLEAR.AUTO]               ; очистка автоматического режима в IDC
U 06F1,  97C4,15 ;4625          PORT.CONTROL [RESET.BR]                 ; очистка BR7 В IDC
U 06F2,  3660,15 ;4626          MOV LS[INTERUPT.EN] TO WR[0]            ; установка в WR0 бита 16
U 06F3,  3EB0,15 ;4627          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 16 для очистки любых прерываний и
                 ;4628                                                  ; информирования конс.процессора, что обмены данными
                 ;4629                                                  ; завершены
U 06F4,  10E0,15 ;4630          MISC [SET.CP.ATTN]                      ; вызов консольного процессора
                 ;4631  WAIT.XFER:
U 06F5,  886F,54 ;4632          JMP [WAIT.XFER]                         ; ожидание реакции консольного процессора
U 06F6,  5B00,14 ;4633          RETURN                                  ; используется только если перенос данных вызван
                 ;4634   .LIST                                          ; микродиагностикой
                 ;4635   .REGION/700,3FFF
                 ;4636
```

```
.PAGE "РАСПОЛОЖЕНИЕ БИТОВ РЕГИСТРОВ УПРАВЛЕНИЯ И СОСТОЯНИЯ МСТ"
;
; CSR0
;
;     CSR0 содержит 7 битов в позициях 6-0, допускающих только чтение и соответст-
;   вующих контрольным битам (CKBT) или битам синдрома (SYND) из микросхем кор-
;   рекции (ECC). Другие биты не используются, если выполняется EX MC 0 (индика-
;   ция CSR0 МСТ). Эти 7 битов расположены следующим образом:
;
;                Бит   6    5    4    3    2    1    0
;                    -----------------------------------
;   CKBT или SYND    : T  : 32 : 16 : 8  : 4  : 2  : 1 :
;                    -----------------------------------
;
; CSR1
;
;   CSR1 содержит биты диагностической проверки, индикации ошибок и управления,
;   используемые для управления или сообщения о ситуации, создавшейся в МСТ во
;   время работы. Биты ошибок могут только считываться, биты диагностической
;   проверки только записываются, а управляющие биты допускают чтение/запись.
;   Два управляющих бита (DIAG CHK и ECC DIS) используются также в качестве усло-
;   вий ветвления. Эти два бита используются только для проверки логических схем
;   МСТ и коррекции. Биты 24 и 13-07 не используются и маскируются, если выпол-
;   няется EX MC 1 (индикация CSR1 МСТ). Биты расположены следующим образом:
;
;
;    31          28             24            20               16
;   ---------------------------------------------------------------------
;   : R : C : T : I : M : D : E : : M : A : T : O : W : A : U : N : T : V :
;   : D : R : B : N : M : I : C : : O : C : B : P : R : D : B : X : B : A :
;   : S : D :   : H : E : A : C : : D : C :   :   :   : A : B : M :   : L :
;   :   :   : P :   :   : G :   : : I : E : M : E : A : P : S :   : P : I :
;   :   :   : A : R :   :   : D : : F : S : I : R : C : T : Y :   : A : D :
;   :   :   : R : E :   : C : I : : Y : S : S : R : R :   :   :   : R :   :
;   :   :   :   : P :   : H : S : :   :   : S :   : O : R :   :   :   :   :
;   :   :   : D :   :   : K :   : : R : R :   :   : S : E :   :   : E :   :
;   :   :   : I : C :   :   :   : : E : E :   :   : S : G :   :   : R :   :
;   :   :   : A : R :   :   :   : : F : F :   :   :   :   :   :   : R :   :
;   :   :   : G : D :   :   :   : :   :   :   :   : P : S :   :   :   :   :
;   :   :   :   :   :   :   :   : :   :   :   :   : G : E :   :   :   :   :
;   :   :   :   :   :   :   :   : :   :   :   :   :   : L :   :   :   :   :
;   :   :   :   :   :   :   :   : :   :   :   :   : E :   :   :   :   :   :
;   :   :   :   :   :   :   :   : :   :   :   :   : R :   :   :   :   :   :
;   :   :   :   :   :   :   :   : :   :   :   :   : R :   :   :   :   :   :
;   ---------------------------------------------------------------------
;
;     12       8   6        4                0
;   ---------------------------------------------
;   : : : : : : : : T : 32 : 16 : 8 : 4 : 2 : 1 :
;   : : : : : : : :   :    :    :   :   :   :   :
;   : : : : : : : :   :    :    :   :   :   :   :
;   ---------------------------------------------
;
; CSR2
;
```

```
;4692    ;       CSR 2 используется для сообщения об ошибках общей шины. Используются только
;4693    ;    4 бита. Другие биты маскируются, если выполняется EX MC 2 (индикация CSR2 МСТ.
;4694    ;    используемыми битами являются биты 16-14 и бит 31. Биты расположены следую-
;4695    ;    щим образом:
;4696    ;
;4697    ;        31  . . .  16      14  . . .  0
;4698    ;        ---------------------------------
;4699    ;        : U : ... : U : U : W : ... :
;4700    ;        : B : ... : B : B : R : ... :
;4701    ;        :   : ... :   :   :   : ... :
;4702    ;        : R : ... : N : T : N : ... :
;4703    ;        : D : ... : X : B : O : ... :
;4704    ;        : S : ... : M :   : T : ... :
;4705    ;        :   : ... :   : P :   : ... :
;4706    ;        :   : ... :   : A : V : ... :
;4707    ;        :   : ... :   : R : A : ... :
;4708    ;        :   : ... :   :   : L : ... :
;4709    ;        :   : ... :   : E : I : ... :
;4710    ;        :   : ... :   : R : D : ... :
;4711    ;        :   : ... :   : R :   : ... :
;4712    ;        ---------------------------------
;4713    ;
```

```
;4714    .PAGE "ИНИЦИАЛИЗАЦИЯ ПАМЯТИ"
;4715    ;
;4716    ;       Память требует еще некоторой инициализации, помимо сквозной записи кода дан-
;4717    ;   ных в матрицах основной памяти с целью установки контрольных битов для каж-
;4718    ;   дого длинного слова. В микродиагностике это делается перед тестами модулей
;4719    ;   памяти одновременно с определением объема памяти.
;4720    ;       В тесте 0 (который автоматически выполняется как только загружается новый
;4721    ;   сегмент) выполняется последовательность инструкций центр.процессора для ос-
;4722    ;   вобождения контроллера ОЗУ, если он "завис". Последовательность состоит
;4723    ;   из четырех инструкций MOV.MEM.DATA TO WR, за которыми следует инструкция
;4724    ;   READ.V.RCHK.IFILL. Инструкция READ.V.RCHK.IFILL вызывает сигнал DATA RCVD в
;4725    ;   случае, если память находится в цикле ожидания снятия CPU GRANT или получе-
;4726    ;   ния DATA.RCVD. Указанные четыре инструкции MOV.MEM.DATA выполняют эту функ-
;4727    ;   цию также в случае, если МСТ "завис" в процессе пересылки 8-кратных (оста)
;4728    ;   слов. Эта же последовательность может вызываться путем выполнения команды
;4729    ;   монитора INIT, если память "зависает" во время выполнения теста.
;4730    ;
;4731    ;       Описание начального ветвления микропрограммы памяти
;4732    ;
;4733    ;       В этом разделе описывается последовательность событий, которые происходят,
;4734    ;   когда центр.процессор выдает микроинструкцию MEM.REQ. Когда память не заня-
;4735    ;   та, она выполняет цикл на единственном микрослове, который стробирует в ре-
;4736    ;   гистр виртуального адреса (VAR) то, что имеется на адресных линиях общей ши-
;4737    ;   ны. Во время этого цикла сигнал CPU GRANT L является высоким (не возбужден).
;4738    ;   инструкция MEM.REQ генерирует в модуле центр.процессора MEMORY REQEST H и
;4739    ;   посылает его в модуль МСТ. Центр.процессор приостанавливается до тех пор,
;4740    ;   пока не будет установлено низкое значение CPU GRANT L (в действительности, ес-
;4741    ;   ли память не занята, разрешение устанавливается почти немедленно и центр.
;4742    ;   процессор не приостанавливается). В контроллере ОЗУ микросостояние холос-
;4743    ;   того цикла после холостого цикла стробирует в VAR данные с шины МС. Инструк-
;4744    ;   ция MEM.REQ заносит на шину D из заданной ячейки LS 32-битовый адрес. Геме-
;4745    ;   рируемый в МСТ сигнал GATE DIR H через ПМЛ УПР.РАСШИРЕНИЕМ знака формирует
;4746    ;   разрешение вывода на шину МС для приемников/передатчиков модуля центр.проце-
;4747    ;   ссора. Таким образом, после выхода из холостого цикла VAR содержит 32 бита
;4748    ;   данных, присутствующих на шине D.
;4749    ;       В МСТ ПМЛ АРБИТР после обнаружения MEMORY REQEST H возбуждает CPUG L (при
;4750    ;   условии, что память не занята). Это вызывает выход микрокода памяти из хо-
;4751    ;   лостого цикла. В источнике запроса (плате центр.процессора) сигнал CPU GRANT
;4752    ;   прекращает приостанов центр.процессора, но CSR не будет немедленно загружен
;4753    ;   новым микрословом. Следующее состояние в потоке микроинструкций все еще
;4754    ;   использует поле MF из CSR для перехода к конкретной программе. Переход управ-
;4755    ;   ляется битами 7-3 микроадреса МСТ (ADRS 7 L - ADRS 3 L), полученными из би-
;4756    ;   тов MF и ADRS 2 - ADRS 0, полученными из битов 2-0 поля NAD микрослова конт-
;4757    ;   роллера ОЗУ. ADRS 8 также задается полем NAD (бит 8), кроме случая, когда
;4758    ;   ADRS 8 принудительно устанавливается в низкий уровень (возбужден). VAR снова
;4759    ;   загружается данными из шины D центрального процессора и микрокод переходит к
;4760    ;   заданной программе.
```

```
;4761      .PAGE "ТЕСТЫ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА"
;4762    ;
;4763    .TOC  "ТЕСТ 1 - тест записи/чтения VAR (модули MCT и DAP)"
;4764    ;
;4765    ; ОПИСАНИЕ ТЕСТА:
;4766    ;
;4767    ;       Этот тест является первым тестом, использующим для каких-либо целей конт-
;4768    ;    роллер ОЗУ. Самым простым трактом данных является путь в регистр виртуаль-
;4769    ;    ного адреса (VAR) и обратно в центральный процессор, но он затрагивает много
;4770    ;    непроверенных схем. Путь данных является следующим: центральный процессор выдает
;4771    ;    инструкцию MEM.REQ с функцией в поле MF=14(H), указывающей чтение адреса в ре-
;4772    ;    жиме отладки (READ.MAINT.ADRS). Ветвление в памяти выполняет действия, описанные
;4773    ;    в начале этого листинга. После ветвления выполняется следующая последователь-
;4774    ;    ность: первое микрослово по адресу ветвления устанавливает сигнал MEMORY BUSY
;4775    ;    и разрешает передачу через два 8-канальных шинных формирователя LVA 09 H по
;4776    ;    LVA 24 H на шину PA (с PA 09 H по PA 24 H). Этим также открываются четыре пе-
;4777    ;    редатчика, которые помещают биты PA и оставшиеся биты регистра виртуального
;4778    ;    адреса на шину MC. Следующее микрослово повторяет предыдущие действия (за иск-
;4779    ;    лючением того, что не возбуждает сигнала MEMORY BUSY). Это микрослово также
;4780    ;    цикливается в ожидании снятия сигнала CPU GRANT L. В центральном процессоре
;4781    ;    сигнал DATA REQ H будет выдан, когда выполнится инструкция MOVE с полем MDP=0
;4782    ;    (данные из памяти передаются в рабочий регистр и местную память). Сигнал DATA
;4783    ;    REQ L, который формирует сигнал DATA REQ H, также формирует сигнал DATA RCVD H
;4784    ;    для модуля MCT. Сигнал DATA RCVD H в модуле MCT снимает сигнал CPU GRANT и вы-
;4785    ;    водит микропрограмму памяти из зацикливания. К этому времени центральный про-
;4786    ;    цессор имеет данные в WR0, так как инструкция MOVE изменила направление пере-
;4787    ;    дачи передатчиков данных в центральном процессоре и прочитала данные с шины MC.
;4788    ;    Следующее микрослово в контроллере ОЗУ выполняет переход опять в холостой
;4789    ;    цикл. Используемыми тестовыми данными являются чередующиеся единицы и нули,
;4790    ;    сдвигаемая единица и сдвигаемый нуль. Необходимо заметить, что данные, возв-
;4791    ;    ращаемые из памяти во время этого теста, будут сдвинуты на один бит вправо,
;4792    ;    где переданный младший бит вернется как бит 31.
;4793    ;
;4794    ; ПРЕДПОЛОЖЕНИЯ:
;4795    ;
;4796    ;       Предполагается, что диагностические микропрограммы центрального процессора
;4797    ;    без использования памяти выполнены успешно. Не предполагается, что сигналы,
;4798    ;    поступающие из центрального процессора в контроллер ОЗУ, работают правиль-
;4799    ;    но. Таким образом, неисправность может вызывать или модуль DAP или модуль MCT.
;4800    ;
;4801    ; ШАГИ ТЕСТА:
;4802    ;
;4803    ;    1)  Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;4804    ;        (для распечатки ошибок) и очистка в местной памяти номера предыдущей
;4805    ;        ошибки.
;4806    ;    2)  Загрузка WR1 тестовыми данными из местной памяти. Затем загрузка ячейки
;4807    ;        временного хранения местной памяти теми же тестовыми данными.
;4808    ;    3)  Выдача инструкции ROR для WR1 для получения сдвинутых вправо ожидаемых
;4809    ;        данных (данные возвращаются сдвинутые вправо на один бит с передачей
;4810    ;        младшего бита в бите 31 результата).
;4811    ;    4)  Выдача инструкции MEM.REQ с полем DT=длинное слово (11) и полем MF=чте-
;4812    ;        ние адреса в режиме отладки (14), используя ячейку временного хранения
;4813    ;        местной памяти в качестве источника данных.
;4814    ;    5)  Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата.
;4815    ;    6)  Повторение шагов с 2 по 5, пока все тестовые данные не будут проверены.
```

```
;4816    ;   7)  Повторение шагов с 2 по 5, используя чередующиеся тестовые данные С введе-
;4817    ;       нием задержки (инструкций NOP) между инструкциями MEM.REQ и MOV. Этим
;4818    ;       проверяется схема ветвления по сигналу CPU GRANT.
;4819    ;
;4820    ;   ОШИБКИ:
;4821    ;
;4822    ;    ошибка 1 - чтение адреса в режиме отладки работает неправильно с данными из
;4823    ;               чередующихся единиц и нулей.
;4824    ;    ошибка 2 - чтение адреса в режиме отладки работает неправильно с данными из
;4825    ;               чередующихся нулей и единиц.
;4826    ;    ошибка 3 - чтение адреса в режиме отладки работает неправильно с данными со
;4827    ;               сдвигаемой единицей.
;4828    ;    ошибка 4 - чтение адреса в режиме отладки работает неправильно с данными со
;4829    ;               сдвигаемым нулем.
;4830    ;    ошибка 5 - чтение адреса в режиме отладки работает неправильно с данными из
;4831    ;               чередующихся единиц и нулей с задержкой между инструкциями MEM.REQ
;4832    ;               и MOV.
;4833    ;
;4834    ;   НАЛАДКА:
;4835    ;
;4836    ;       Если центральный процессор завис, повидимому сигналы CPU GRANT L или MEMORY
;4837    ;   BUSY H вызвали приостанов центрального процессора и не разблокировывают его.
;4838    ;   Если центральный процессор завис в инструкции MEM.REQ и память находится в
;4839    ;   холостом цикле, необходимо проверить, что сигнал MEMORY REQ H поступает в
;4840    ;   модуль MCT на ножку 6 ПМЛ АРБИТР (консольный процессор будет печатать сооб-
;4841    ;   щение о таймауте, следовательно, возможно использование команд консоли для
;4842    ;   приближения к этой инструкции: пошагового режима или останова по микроадре-
;4843    ;   су). Если сигнал MEMORY REQ H низкий, необходимо проверить низкий уровень
;4844    ;   сигнала EN MEM REF L в модуле DAP. Этот сигнал управляется консольным проце-
;4845    ;   ссором. Этот сигнал на ножке 5 вентиля "И" также должен быть низким. Если
;4846    ;   этого нет, необходимо проверить высокий уровень на входах CSR 20 и CSR 19
;4847    ;   и низкий уровень на входах CSR 22 и CSR 21 мультиплексора УПР. ЗАПРОСОМ памя-
;4848    ;   ти. Если входы правильные, а выход на ножке 6 имеет высокий уровень, мульти-
;4849    ;   плексор неисправен.
;4850    ;
;4851    ;       Если сигнал MEMORY REQ H имеет высокий уровень на входе ПМЛ АРБИТР, не-
;4852    ;   обходимо проверить другие входы следующим образом: сигнал L MSYN H (ножка 13)
;4853    ;   должен быть низкий, сигнал LOCK L (ножка 14) должен быть высокий (высокий
;4854    ;   уровень устанавливает микрокод включения питания), сигнал CONT FUNC LAT L
;4855    ;   (ножка 7) должен быть высокий и сигнал DATA RCVD H (ножка 9) должен быть
;4856    ;   низкий. Также необходимо проверить низкий уровень разрешения на ножке 11 и
;4857    ;   наличие синхросигнала на ножке 1. Если какой-то сигнал неправильный, необхо-
;4858    ;   димо проследить его в обратном направлении. Если входы такие, как указано,
;4859    ;   сигнал CPU GRANT L должен быть низкий, в противном случае ПМЛ неисправна. До
;4860    ;   тех пор, пока сигнал CPU GRANT L не может быть выставлен, микропрограмма памя-
;4861    ;   ти не выйдет из холостого цикла. Если память не находится в холостом цикле,
;4862    ;   возможно, что сигнал CPU GRANT не возвращается в модуль DAP. Необходимо
;4863    ;   проверить изменение уровня ножки 10 микросхемы в модуле DAP, которая форми-
;4864    ;   рует сигнал CLOCK STALL L. Если изменения нет, лучший способ наладки - пока-
;4865    ;   говый режим контроллера ОЗУ от включения питания до холостого цикла, потом
;4866    ;   выполнение инструкции MEM.REQ и проверка, что сигнал CPU GRANT становится
;4867    ;   низким в модуле DAP.
;4868    ;
;4869    ;       Если центральный процессор завис в инструкции MOV, которая читает данные
;4870    ;   из памяти, повидимому, сигнал MEMORY BUSY H не становится низким для разблоки-
```

;4816 ; 7) Повторение шагов с 2 по 5, используя чередующиеся тестовые данные С введе-
;4817 ; нием задержки (инструкций NOP) между инструкциями MEM.REQ и MOV. Этим
;4818 ; проверяется схема ветвления по сигналу CPU GRANT.
;4819 ;
;4820 ; ОШИБКИ:
;4821 ;
;4822 ; ошибка 1 - чтение адреса в режиме отладки работает неправильно с данными из
;4823 ; чередующихся единиц и нулей.
;4824 ; ошибка 2 - чтение адреса в режиме отладки работает неправильно с данными из
;4825 ; чередующихся нулей и единиц.
;4826 ; ошибка 3 - чтение адреса в режиме отладки работает неправильно с данными со
;4827 ; сдвигаемой единицей.
;4828 ; ошибка 4 - чтение адреса в режиме отладки работает неправильно с данными со
;4829 ; сдвигаемым нулем.
;4830 ; ошибка 5 - чтение адреса в режиме отладки работает неправильно с данными из
;4831 ; чередующихся единиц и нулей с задержкой между инструкциями MEM.REQ
;4832 ; и MOV.
;4833 ;
;4834 ; НАЛАДКА:
;4835 ;
;4836 ; Если центральный процессор завис, повидимому сигналы CPU GRANT L или MEMORY
;4837 ; BUSY H вызвали приостанов центрального процессора и не разблокировывают его.
;4838 ; Если центральный процессор завис в инструкции MEM.REQ и память находится в
;4839 ; холостом цикле, необходимо проверить, что сигнал MEMORY REQ H поступает в
;4840 ; модуль MCT на ножку 6 ПМЛ АРБИТР (консольный процессор будет печатать сооб-
;4841 ; щение о таймауте, следовательно, возможно использование команд консоли для
;4842 ; приближения к этой инструкции: пошагового режима или останова по микроадре-
;4843 ; су). Если сигнал MEMORY REQ H низкий, необходимо проверить низкий уровень
;4844 ; сигнала EN MEM REF L в модуле DAP. Этот сигнал управляется консольным проце-
;4845 ; ссором. Этот сигнал на ножке 5 вентиля "И" также должен быть низким. Если
;4846 ; этого нет, необходимо проверить высокий уровень на входах CSR 20 и CSR 19
;4847 ; и низкий уровень на входах CSR 22 и CSR 21 мультиплексора УПР. ЗАПРОСОМ памя-
;4848 ; ти. Если входы правильные, а выход на ножке 6 имеет высокий уровень, мульти-
;4849 ; плексор неисправен.
;4850 ;
;4851 ; Если сигнал MEMORY REQ H имеет высокий уровень на входе ПМЛ АРБИТР, не-
;4852 ; обходимо проверить другие входы следующим образом: сигнал L MSYN H (ножка 13)
;4853 ; должен быть низкий, сигнал LOCK L (ножка 14) должен быть высокий (высокий
;4854 ; уровень устанавливает микрокод включения питания), сигнал CONT FUNC LAT L
;4855 ; (ножка 7) должен быть высокий и сигнал DATA RCVD H (ножка 9) должен быть
;4856 ; низкий. Также необходимо проверить низкий уровень разрешения на ножке 11 и
;4857 ; наличие синхросигнала на ножке 1. Если какой-то сигнал неправильный, необхо-
;4858 ; димо проследить его в обратном направлении. Если входы такие, как указано,
;4859 ; сигнал CPU GRANT L должен быть низкий, в противном случае ПМЛ неисправна. До
;4860 ; тех пор, пока сигнал CPU GRANT L не может быть выставлен, микропрограмма памя-
;4861 ; ти не выйдет из холостого цикла. Если память не находится в холостом цикле,
;4862 ; возможно, что сигнал CPU GRANT не возвращается в модуль DAP. Необходимо
;4863 ; проверить изменение уровня ножки 10 микросхемы в модуле DAP, которая форми-
;4864 ; рует сигнал CLOCK STALL L. Если изменения нет, лучший способ наладки - пошаговый режим контроллера ОЗУ от включения питания до холостого цикла, потом
;4865 ; выполнение инструкции MEM.REQ и проверка, что сигнал CPU GRANT становится
;4866 ; низким в модуле DAP.
;4867 ;
;4868 ; Если центральный процессор завис в инструкции MOV, которая читает данные
;4869 ; из памяти, повидимому, сигнал MEMORY BUSY H не становится низким для разблоки-
;4870 ;

```
                ;4926   ;   быть высокими. Эти сигналы вместе с высоким уровнем сигнала CSR 19 H (инструк-
                ;4927   ;   ция MEM.REQ) устанавливают низкий уровень сигнала OP ARY ADR L независимо от
                ;4928   ;   его предыдущего состояния. Этот сигнал не должен стать высоким, пока не будет
                ;4929   ;   выполняться инструкция DECODE центрального процессора. Загрузка регистра вир-
                ;4930   ;   туального адреса может быть проверена, пока память находится в холостом цикле.
                ;4931   ;   входы VAR LOAD H и VAR MUX SEL H должны быть низкими. Эта комбинация должна
                ;4932   ;   буферизировать в ПМЛ все, что находится на шине MC.
                ;4933   ;      Если выходы регистра виртуального адреса правильные, необходимо проверить
                ;4934   ;   входы формирователей модуля MCT и низкий уровень сигналов TB DATA DIR RD L и
                ;4935   ;   TB DATA EN+MAINT L. Правильные данные также могут быть проверены на выходе
                ;4936   ;   (шина MC). Если имеются неправильные биты на BUS MC 08 по 23, необходимо про-
                ;4937   ;   верить счетверенные буферы, которые выдают PA 09 по PA 24. Сигнал разрешения
                ;4938   ;   ADDR PH L поступает из ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. и должен быть низким. Входы
                ;4939   ;   этой ПМЛ должны быть низкими на SPF1 H и SPF2 H и высоким на SPF0 H.
                ;4940   ;      Другой причиной неправильной работы могут быть схемы микросеквенсера. Ес-
                ;4941   ;   ли контроллер ОЗУ не может работать в пошаговом режиме, это затруднит на-
                ;4942   ;   ладку этих схем. Если память зацикливается в правильных ячейках до выдачи
                ;4943   ;   инструкции MOV центральным процессором, по-видимому, микросеквенсер для это-
                ;4944   ;   го теста работает правильно.
                ;4945   ;
                ;4946   ;      ОШИБКИ С 2 ПО 4 - Если это первая ошибка, подозревается модуль MCT. Ве-
                ;4947   ;   роятнее всего, неправильно работает ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА, буфер, ко-
                ;4948   ;   торый выдает PA 09 по 24, или формирователи MCT.
                ;4949   ;
                ;4950   ;      ОШИБКА 5 - Эта ошибка указывает на неправильную работу схемы ветвления по
                ;4951   ;   сигналу CPU GRANT. Предполагается, что контроллер ОЗУ циклится, ожидая сиг-
                ;4952   ;   нала приема данных для сброса сигнала CPU GRANT. Если эти схемы неправильно вы-
                ;4953   ;   полняют зацикливание, контроллер ОЗУ может запретить выходы на шину MC до
                ;4954   ;   чтения данных в DAP. Остановите центральный процессор в пошаговом режиме на
                ;4955   ;   инструкции MEM.REQ и проверьте низкий уровень сигнала DATA RCVD H на входе
                ;4956   ;   ПМЛ АРБИТР модуля MCT. Если правильно, проверьте, что сигнал CPU GRANT L
                ;4957   ;   низкий на входе схемы ветвления (мультиплексор BEN 1).
                ;4958   T.1:
U 0700, B65E,15 ;4959           MOV LSIBEGIN.TEST] TO WR[0]                 ; установка бита 15 в WR[0] для слова управления и
                ;4960                                                       ; состояния
U 0701, 3E80,15 ;4961           MOV WR[0] TO LSICONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                ;4962                                                       ; 15 указывает начало теста для консольного процессора
U 0702, 10E0,15 ;4963           MISC [SET.CP.ATTN]                          ; выдача сигнала CPU ATTN для консольного процессора
                ;4964   WAIT.T1.0:
U 0703, 0870,34 ;4965           JMP [WAIT.T1.0]                             ; зацикливание для ожидания ответа консольного
                ;4966                                                       ; процессора
U 0704, 0A1A,AC ;4967           JSR [SETUP.1]                               ; установка масок, кода модуля и др.
U 0705, 4742,15 ;4968           BIS LSICPU] TO WR[0]                        ; добавление кода модуля центрального процессора
U 0706, 3E8C,15 ;4969           MOV WR[0] TO LSIMODULE.NUM]                 ; запоминание кода модуля в местной памяти для указания
                ;4970                                                       ; модулей DAP и MCT
                ;4971   LOOP.T1.1:
U 0707, B69A,95 ;4972           MOV LSIALTER.EVEN] TO WR[1]                 ; ожидаемые данные (данные сдвинуты на один бит влево)
U 0708, 9D98,75 ;4973           MEM.REQ[READ.MAINT.ADRS] ADRSIALTER.ODD] DT[LONG] ; запись эталона чередующихся тестовых данных в
                ;4974                                                       ; регистр виртуального адреса
U 0709, 3022,15 ;4975           MOV MEM.DATA TO WR[0]                       ; обратное чтение тестовых данных в WR[0]
U 070A, 0869,3C ;4976           JSR [CHECK.RESULT]                          ; проверка результата
U 070B, 8870,74 ;4977           JMP [LOOP.T1.1]                             ; зацикливание при ошибке, если разрешено
U 070C, FF82,15 ;4978           INC LSIERROR.NUMBER]                        ; номер ошибки = 2
                ;4979   LOOP.T1.2:
U 070D, 3698,95 ;4980           MOV LSIALTER.ODD] TO WR[1]                  ; следующий эталон ожидаемых данных
```

```
U 070E, 1D9A,75 ;4981            MEM.REQ[READ.MAINT.ADRS] ADRS[ALTER.EVEN] DT[LONG] ; запись чередующихся тестовых данных в
                ;4982                                                              ; регистр виртуального адреса
U 070F, 3022,15 ;4983            MOV MEM.DATA TO WR[0]                             ; обратное чтение тестовых данных в WR[0]
U 0710, 0869,3C ;4984            JSR [CHECK.RESULT]                                ; проверка результата
U 0711, 8870,D4 ;4985            JMP [LOOP.T1.2]                                   ; зацикливание при ошибке, если разрешено
U 0712, FF82,15 ;4986            INC LS[ERROR.NUMBER]                              ; номер ошибки = 3
U 0713, E5F8,15 ;4987            CLR LS[OS]                                        ; очистка регистра OS (используется для индексации)
                ;4988    LOOP.T1.3:
U 0714, B6F6,95 ;4989            MOV LS[SHIFT.OS(4-0)] TO WR[1]                    ; загрузка тестовых данных в WR[1]
U 0715, A340,95 ;4990            ROR WR[1]                                         ; сдвиг данных вправо для устранения аппаратного сдвига
                ;4991                                                              ; в модуле MCT
U 0716, 1DF6,75 ;4992            MEM.REQ[READ.MAINT.ADRS] ADRS[SHIFT.OS(4-0)] DT[LONG] ; запись тестовых данных со сдвигаемой
                ;4993                                                              ; единицей в регистр виртуального адреса
U 0717, 3022,15 ;4994            MOV MEM.DATA TO WR[0]                             ; выборка результата
U 0718, 0869,3C ;4995            JSR [CHECK.RESULT]                                ; проверка результата
U 0719, 0871,44 ;4996            JMP [LOOP.T1.3]                                   ; зацикливание при ошибке, если разрешено
U 071A, 7FF8,15 ;4997            INC LS[OS]                                        ; увеличение индекса для следующих сдвинутых тестовых данных
U 071B, 36F9,15 ;4998            MOV LS[OS] TO WR[2]                               ; подготовка для проверки, все ли тестовые данные
                ;4999                                                              ; использованы
                ;5000              BIT LS[BIT5] WITH WR[2],                        ; проверка, что содержимое регистра OS увеличено до
                ;5001                                                              ; 20(H)
U 071C, D948,35 ;5002            DT(LONG)&SET.ALU.CC                               ; установка кодов условий
U 071D, 8871,49 ;5003            JMP.IF[BITS.CLR] TO [LOOP.T1.3]                   ; повторение со следующими тестовыми данными, если не все
                ;5004                                                              ; использованы
U 071E, FF82,15 ;5005            INC LS[ERROR.NUMBER]                              ; номер ошибки = 4
U 071F, E5F8,15 ;5006            CLR LS[OS]                                        ; очистка регистра OS (используется для индексации)
                ;5007    LOOP.T1.4:
U 0720, DFF6,95 ;5008            MCOM LS[SHIFT.OS(4-0)] TO WR[1]                   ; загрузка тестовых данных со сдвигаемым нулем в WR[1]
U 0721, BE10,95 ;5009            MOV WR[1] TO LS[T8]                               ; сохранение эталона данных в ячейке местной
                ;5010                                                              ; памяти
U 0722, A340,95 ;5011            ROR WR[1]                                         ; сдвиг данных вправо для устранения аппаратного сдвига
                ;5012                                                              ; в модуле MCT
U 0723, 9D10,75 ;5013            MEM.REQ[READ.MAINT.ADRS] ADRS[T8] DT[LONG] ; запись тестовых данных со сдвигаемым нулем в регистр
                ;5014                                                              ; виртуального адреса
U 0724, 3022,15 ;5015            MOV MEM.DATA TO WR[0]                             ; выборка результата
U 0725, 0869,3C ;5016            JSR [CHECK.RESULT]                                ; проверка результата
U 0726, 8872,04 ;5017            JMP [LOOP.T1.4]                                   ; зацикливание при ошибке, если разрешено
U 0727, 7FF8,15 ;5018            INC LS[OS]                                        ; увеличение индекса для следующих сдвинутых тестовых данных
U 0728, 36F9,15 ;5019            MOV LS[OS] TO WR[2]                               ; подготовка для проверки, все ли тестовые данные
                ;5020                                                              ; использованы
                ;5021              BIT LS[BIT5] WITH WR[2],                        ; проверка, что содержимое регистра OS увеличено до
                ;5022                                                              ; 20(H)
U 0729, D948,35 ;5023            DT(LONG)&SET.ALU.CC                               ; установка кодов условий
U 072A, 0872,09 ;5024            JMP.IF[BITS.CLR] TO [LOOP.T1.4]                   ; повторение со следующими тестовыми данными, если не все
                ;5025                                                              ; использованы
U 072B, FF82,15 ;5026            INC LS[ERROR.NUMBER]                              ; номер ошибки = 5
                ;5027    LOOP.T1.5:
U 072C, B69A,95 ;5028            MOV LS[ALTER.EVEN] TO WR[1]                       ; ожидаемые данные (данные сдвинуты на 1 бит влево)
U 072D, 9D98,75 ;5029            MEM.REQ[READ.MAINT.ADRS] ADRS[ALTER.ODD] DT[LONG] ; запись чередующихся тестовых данных в
                ;5030                                                              ; регистр виртуального адреса
U 072E, 0A1B,4C ;5031            JSR [NOP.DELAY]                                   ; выполнение 5 циклов задержки ожидания ветвления памяти
                ;5032                                                              ; по сигналу CPU GRANT
U 072F, 3022,15 ;5033            MOV MEM.DATA TO WR[0]                             ; обратное чтение тестовых данных в WR[0]
U 0730, 0869,3C ;5034            JSR [CHECK.RESULT]                                ; проверка результата
U 0731, 8872,C4 ;5035            JMP [LOOP.T1.5]                                   ; зацикливание при ошибке, если разрешено
```

```
                ;5036    END.T1:
```

```
;5037   .PAGE "ТЕСТ 2 – тест увеличения VAR и приостанова по MEMORY BUSY (модуль MCT или DAP)"
;5038   ;
;5039   ; ОПИСАНИЕ ТЕСТА:
;5040   ;
;5041   ;       Этот тест проверяет функцию увеличения содержимого регистра виртуального
;5042   ;    адреса и приостанов центрального процессора, когда выполняется чтение и па-
;5043   ;    мять занята. Путь данных следующий: центральный процессор выдает микроинст-
;5044   ;    рукцию MEM.REQ с полем MF=READ.MAINT.VAR.INC (08(H)). Ветвление в памяти вы-
;5045   ;    полняет действия, описанные в начале этого листинга. По адресу ветвления ус-
;5046   ;    танавливается занятость памяти и разрешение физического адреса. Следующий
;5047   ;    микроцикл выполняет увеличение содержимого регистра виртуального адреса и под-
;5048   ;    держивает занятость памяти. За этим циклом следуют четыре цикла памяти, ко-
;5049   ;    торые просто поддерживают занятость памяти. Эта задержка необходима для про-
;5050   ;    верки, что приостанов центрального процессора при занятости памяти работает
;5051   ;    правильно. Если приостанов центр. процессора работает неправильно, централь-
;5052   ;    ный процессор пропустит данные, посылаемые обратно, и неисправность будет об-
;5053   ;    наружена. После задержки из 4 циклов, регистр виртуального адреса считывается
;5054   ;    разрешением регистра виртуального адреса через счетверенные буферы физическо-
;5055   ;    го адреса на счетверенные буферы и восьмиканальные шинные формирователи шины
;5056   ;    МС и помещается на шину МС. Также сбрасывается занятость памяти и микропрог-
;5057   ;    рамма памяти циклится в ожидании сброса CPU GRANT. Инструкция MOV центрально-
;5058   ;    го процессора была приостановлена до снятия занятости памяти, и теперь может
;5059   ;    прочитать данные с шины МС и проверить их. Память возвращается в холостой
;5060   ;    цикл после проверки, нет ли следующего запроса центрального процессора.
;5061   ;       Регистр виртуального адреса увеличивается на 4 (LVA00 и LVA01 не меняются).
;5062   ;    также не меняются LVA30 и LVA31. Используются следующие тестовые данные: все
;5063   ;    нули, все единицы, сдвигаемая единица и сдвигаемый нуль с дополнительным
;5064   ;    очищенным битом (сдвиг нуля повторяется 32 раза и каждый раз дополнительно
;5065   ;    к сдвигаемому биту сбрасывается другой бит, начиная с бита 0 до бита 31). До
;5066   ;    проверки каждого набора дополнительный код текущего набора записывается в
;5067   ;    регистр виртуального адреса.
;5068   ;
;5069   ; ПРЕДПОЛОЖЕНИЯ:
;5070   ;
;5071   ;       Предполагается, что предыдущий тест регистра виртуального адреса выполнен
;5072   ;    успешно. Небольшая схема модуля DAP (приостанов по занятости памяти) здесь
;5073   ;    проверяется впервые.
;5074   ;
;5075   ; ШАГИ ТЕСТА:
;5076   ;
;5077   ;    1) Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;5078   ;       (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;5079   ;       мяти.
;5080   ;    2) Очистка WR0 и пересылка этого значения в ячейку временного хранения мест-
;5081   ;       ной памяти.
;5082   ;    3) Загрузка 2(H) в WR1, как ожидаемого значения (результат сдвинут вправо на
;5083   ;       1 бит, когда выполняется чтение обратно).
;5084   ;    4) Запись дополнительного кода данных в регистр виртуального адреса, затем
;5085   ;       выдача инструкции MEM.REQ с DT=LONGWORD (длинное слово) и MF=READ.MAINT.VAR
;5086   ;       .INC (чтение в режиме наладки и увеличение регистра виртуального адреса),
;5087   ;       используя содержимое ячейки временного хранения местной памяти как данные
;5088   ;       (все нули).
;5089   ;    5) Выдача инструкции MOV MEM.DATA TO WR[0] и проверка результата.
;5090   ;    6) Повторение шагов с 2 по 5 с использованием данных из всех единиц. Результат
;5091   ;       должен быть все нули с единицами в битах 31, 30, 29 и 0 после сдвига.
```

```
;5092    ;    7) Повторение шагов с 2 по 5 с использованием данных со сдвигаемой единицей.
;5093    ;    8) Повторение шагов с 2 по 5 с использованием данных со сдвигаемым нулем и
;5094    ;       и одним дополнительно сбрасываемым битом в порядке их следования от млад-
;5095    ;       ших к старшим.
;5096    ;
;5097    ; ОШИБКИ:
;5098    ;
;5099    ;    ошибка 1 - неправильно работает INC VAR (увеличение содержимого регистра
;5100    ;               виртуального адреса) с начальными данными все нули.
;5101    ;    ошибка 2 - неправильно работает INC VAR с начальными данными все единицы.
;5102    ;    ошибка 3 - неправильно работает INC VAR с данными со сдвигаемой единицей.
;5103    ;    ошибка 4 - неправильно работает INC VAR с данными со сдвигаемым нулем и сбро-
;5104    ;               сом дополнительного бита.
;5105    ;
;5106    ; НАЛАДКА:
;5107    ;
;5108    ;       ОШИБКА 1 - Эта неисправность может быть вызвана или схемой контроллера ОЗУ для
;5109    ;    увеличения регистра виртуального адреса или схемой приостанова в DAP. Если
;5110    ;    фактический результат все единицы, в первую очередь подозревается схема
;5111    ;    приостанова в DAP. В первую очередь пошаговым режимом поставьте центральный
;5112    ;    процессор на инструкцию MOV MEM.DATA TO WR[0] и проверьте сигнал DATA REQ H,
;5113    ;    поступающий на схему приостанова синхронизации (ножка 3) в DAP. Этот сиг-
;5114    ;    нал должен быть высоким во время инструкции MOV. Если он правильный, сле-
;5115    ;    дует проверить сигнал MEMORY BUSY. Если MCT может работать в пошаговом режи-
;5116    ;    ме, остановите микропрограмму памяти на первой инструкции после адреса вет-
;5117    ;    вления (которая устанавливает физический адрес и занятость памяти) и проверь-
;5118    ;    те высокий уровень сигнала MEMORY BUSY H на схеме приостанова синхронизации
;5119    ;    (ножка 1) в DAP. Если центральный процессор остановлен на микроинструкции
;5120    ;    MOV, сигналы DATA REQ H и MEMORY BUSY H должны формировать низкий уровень
;5121    ;    сигнала CLOCK STALL L. Если сигнал MEMORY BUSY H низкий, необходимо прове-
;5122    ;    рить выход из модуля MCT. Этот сигнал можно проверить в обратном направле-
;5123    ;    нии на C38 микропрограммы MCT. Если пошаговый режим MCT невозможен, выпол-
;5124    ;    няется зацикливание при ошибке и сигналы проверяются в динамическом режиме.
;5125    ;       Если результат не все единицы, неисправность может быть в модуле MCT. Если
;5126    ;    пошаговый режим MCT возможен, необходимо проверить входы ПМЛ РЕГ.ВИРТУАЛЬНО-
;5127    ;    ГО АДРЕСА во время микроцикла памяти, который увеличивает регистр виртуаль-
;5128    ;    ного адреса, следующим образом: сигнал VAR MUX SEL H должен быть высокий и
;5129    ;    VAR LOAD H - низкий. Оба эти сигнала можно проверить в обратном направ-
;5130    ;    лении на микросхемах микропамяти MCT (C20 и C21). Если эти сигналы правиль-
;5131    ;    ные, необходимо проверить, что ножка 19 младших разрядов ПМЛ РЕГ.ВИРТУАЛЬНО-
;5132    ;    ГО АДРЕСА соединена с ПМЛ старших разрядов. Если нет, увеличение не может
;5133    ;    быть выполнено. Вход CIN (вход переноса) на другие ПМЛ РЕГ.ВИРТУАЛЬНОГО АД-
;5134    ;    РЕСА (ножка 19) должен быть низким для этих тестовых данных. Неисправная ПМЛ
;5135    ;    может быть обнаружена определением самого младшего неправильного бита. Не-
;5136    ;    обходимо помнить, что результат сдвинут влево на один бит. Таким образом,
;5137    ;    если в результате неправильный бит 9, тогда неправильным является бит 8 ре-
;5138    ;    гистра виртуального адреса. Если вход переноса и биты управления регистра
;5139    ;    виртуального адреса правильные, подозревается ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА,
;5140    ;    связанная с неправильным битом. Если пошаговый режим MCT невозможен, сигна-
;5141    ;    лы должны быть проверены во время зацикливания при ошибке.
;5142    ;
;5143    ;       ОШИБКА 2 - Если это первая ошибка, подозреваются сами ПМЛ РЕГ.ВИРТУАЛЬ-
;5144    ;    НОГО АДРЕСА. Неисправная ПМЛ может быть обнаружена определением первого непра-
;5145    ;    вильного бита (самого младшего). Необходимо помнить, что результат сдвинут на
;5146    ;    1 бит влево, так что если в результате неправильный 9-тый бит, тогда бит 8
```

```
                ;5147   ;   ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА вызывает ошибку. С этими тестовыми данными
                ;5148   ;   все сигналы входов переноса (ножка 19) должны быть высокими. Если нет, подоз-
                ;5149   ;   ревается предыдущая ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА.
                ;5150   ;
                ;5151   ;      ОШИБКИ 3 И 4 - Эти ошибки указывают на неисправность самих ПМЛ РЕГ.ВИР-
                ;5152   ;   ТУАЛЬНОГО АДРЕСА. Определение неправильного бита регистра виртуального ад-
                ;5153   ;   реса подозреваемой ПМЛ дано в описании ошибки 2.
                ;5154   ;
                ;5155    T.2:
U 0732, B65E,15 ;5156           MOV LS[BEGIN.TEST] TO WR[0]                     ; установка бита 15 в WR[0] для слова управления и
                ;5157                                                           ; состояния
U 0733, 3E80,15 ;5158           MOV WR[0] TO LS[CONTROL.STATUS]                 ; установка бита 15 в слове управления и состояния. Бит
                ;5159                                                           ; 15 указывает начало теста для консольного процессора
U 0734, 10E0,15 ;5160           MISC [SET.CP.ATTN]                              ; выдача сигнала CPU ATTN для конс.процессора
                ;5161    WAIT.T2.0:
U 0735, 0873,54 ;5162           JMP [WAIT.T2.0]                                 ; зацикливание для ожидания ответа конс.процессора
U 0736, 0A1A,AC ;5163           JSR [SETUP.1]                                   ; установка масок и др.
U 0737, 4742,15 ;5164           BIS LS[CPU] TO WR[0]                            ; добавление кода модуля DAP
U 0738, 3E8C,15 ;5165           MOV WR[0] TO LS[MODULE.NUM]                     ; запоминание кода модуля в местной памяти для указания
                ;5166                                                           ; модулей DAP и MCT
                ;5167    LOOP.T2.1:
U 0739, B642,95 ;5168           MOV LS[#2] TO WR[1]                             ; ожидаемые данные (биты 0-4 = 4, после сдвига
                ;5169                                                           ; вправо=2)
U 073A, 9D9E,75 ;5170           MEM.REQ[READ.MAINT.ADRS] ADRS[ONES] DT[LONG] ; запись дополнительного кода данных в регистр
                ;5171                                                           ; виртуального адреса
U 073B, 3022,15 ;5172           MOV MEM.DATA TO WR[0]                           ; выполнение пересылки для окончания цикла, но проверка
                ;5173                                                           ; не выполняется
U 073C, 9A9D,F5 ;5174           MEM.REQ[READ.MAINT.VAR.INC] ADRS[#0] DT[LONG] ; пересылка данных=0 в регистр виртуального
                ;5175                                                           ; адреса для увеличения
U 073D, 3022,15 ;5176           MOV MEM.DATA TO WR[0]                           ; выборка результата
U 073E, 0869,3C ;5177           JSR [CHECK.RESULT]                              ; проверка результата
U 073F, 0873,94 ;5178           JMP [LOOP.T2.1]                                 ; зацикливание при ошибке, если разрешено
U 0740, FF82,15 ;5179           INC LS[ERROR.NUMBER]                            ; ошибка 2
U 0741, 2F87,95 ;5180           CLR WR[3]                                       ; очистка ожидаемого результата
U 0742, 477F,95 ;5181           BIS LS[BIT31] TO WR[3]                          ; установка бита 31
U 0743, C77D,95 ;5182           BIS LS[BIT30] TO WR[3]                          ; установка бита 30
U 0744, C77B,95 ;5183           BIS LS[BIT29] TO WR[3]                          ; установка бита 29
U 0745, C741,95 ;5184           BIS LS[BIT0] TO WR[3]                           ; установка бита 0. Имеем ожидаемый результат
                ;5185    LOOP.T2.2:
U 0746, 2006,95 ;5186           MOV WR[3] TO WR[1]                              ; пересылка ожидаемого результата в WR1
U 0747, 1D9C,75 ;5187           MEM.REQ[READ.MAINT.ADRS] ADRS[ZERO] DT[LONG] ; запись дополнительного кода данных в регистр
                ;5188                                                           ; виртуального адреса
U 0748, 3022,15 ;5189           MOV MEM.DATA TO WR[0]                           ; выполнение пересылки для окончания цикла, но проверка
                ;5190                                                           ; не выполняется
U 0749, 1A9F,F5 ;5191           MEM.REQ[READ.MAINT.VAR.INC] ADRS[ONES] DT[LONG] ; пересылка данных из всех единиц в регистр
                ;5192                                                           ; виртуального адреса для увеличения
U 074A, 3022,15 ;5193           MOV MEM.DATA TO WR[0]                           ; выборка результата
U 074B, 0869,3C ;5194           JSR [CHECK.RESULT]                              ; проверка результата
U 074C, 8874,64 ;5195           JMP [LOOP.T2.2]                                 ; зацикливание при ошибке, если разрешено
U 074D, FF82,15 ;5196           INC LS[ERROR.NUMBER]                            ; ошибка 3
U 074E, E5F8,15 ;5197           CLR LS[OS]                                      ; очистка индекса
                ;5198    REPEAT.T2.3:
U 074F, 36F7,95 ;5199           MOV LS[SHIFT.OS(4-0)] TO WR[3]                  ; выборка тестовых данных
U 0750, 7815,95 ;5200           MCOM WR[3] TO LS[T10]                           ; пересылка дополнительного кода данных в местную
                ;5201                                                           ; память
```

```
U 0751, C045,95 ;5202            ADD LS[#4] TO WR[3]                        ; прибавление 4 к тестовым данным (местная память
                ;5203                                                       ; увеличивается на 4)
U 0752, 2341,95 ;5204            ROR WR[3]                                  ; сдвиг вправо, как делает аппаратура
                ;5205    LOOP.T2.3:
U 0753, 2006,95 ;5206            MOV WR[3] TO WR[1]                         ; установка ожидаемых данных
U 0754, 1D14,75 ;5207            MEM.REQ[READ.MAINT.ADRS] ADRS[T10] DT[LONG] ; запись дополнительного кода в регистр
                ;5208                                                       ; виртуального адреса
U 0755, 3022,15 ;5209            MOV MEM.DATA TO WR[0]                      ; выполнение пересылки для окончания цикла, но проверка
                ;5210                                                       ; не выполняется
U 0756, 9AF7,F5 ;5211            MEM.REQ[READ.MAINT.VAR.INC] ADRS[SHIFT.OS(4-0)] DT[LONG] ; пересылка тестовых данных в регистр
                ;5212                                                       ; виртуального адреса
U 0757, 3022,15 ;5213            MOV MEM.DATA TO WR[0]                      ; выборка результата
U 0758, 0869,3C ;5214            JSR [CHECK.RESULT]                         ; проверка результата
U 0759, 0875,34 ;5215            JMP [LOOP.T2.3]                            ; зацикливание при ошибке, если разрешено
U 075A, 7FF8,15 ;5216            INC LS[OS]                                 ; увеличение индекса
U 075B, 36F9,15 ;5217            MOV LS[OS] TO WR[2]                        ; подготовка для проверки, все ли тестовые данные
                ;5218                                                       ; использованы
                ;5219              BIT LS[BIT5] WITH WR[2],                 ; проверка, что содержимое регистра OS увеличено до
                ;5220                                                       ; 20(H)
U 075C, D94B,35 ;5221            DT(LONG)&SET.ALU.CC                        ; установка кодов условий
U 075D, 0874,F9 ;5222            JMP.IF[BITS.CLR] TO [REPEAT.T2.3]          ; повторение со следующими тестовыми данными, если не все
                ;5223                                                       ; использованы
U 075E, FF82,15 ;5224            INC LS[ERROR.NUMBER]                       ; номер ошибки=4
U 075F, E5F8,15 ;5225            CLR LS[OS]                                 ; очистка индекса
U 0760, DF41,95 ;5226            MCOM LS[BIT0] TO WR[3]                     ; WR[3] содержит тестовые данные со сдвигаемым
                ;5227                                                       ; нулем (FFFFFFFE)
U 0761, 3E11,95 ;5228            MOV WR[3] TO LS[T8]                        ; запоминание сдвигаемых тестовых данных в ячейке временного
                ;5229                                                       ; хранения местной памяти
                ;5230    REPEAT.T2.4:
U 0762, C5F7,95 ;5231            BIC LS[SHIFT.OS(4-0)] TO WR[3]             ; сброс дополнительного бита
U 0763, BE13,95 ;5232            MOV WR[3] TO LS[T9]                        ; ячейка T9 местной памяти будет использована как
                ;5233                                                       ; источник данных для операции MEM.REQ
U 0764, 7B15,95 ;5234            MCOM WR[3] TO LS[T10]                      ; пересылка дополнительного кода данных в местную
                ;5235                                                       ; память
U 0765, 2F85,15 ;5236            CLR WR[2]                                  ; WR[2] используется для запоминания состояния битов 30
                ;5237                                                       ; и 31
                ;5238              BIT LS[BIT30] WITH WR[3],                ; проверка, установлен ли бит 30
U 0766, 597D,B5 ;5239            DT(LONG)&SET.ALU.CC                        ; установка кодов условий
U 0767, 5B00,09 ;5240            SKIP.IF[BITS.CLR]                          ; пропуск следующей инструкции, если бит 30 сброшен
U 0768, 477D,15 ;5241            BIS LS[BIT30] TO WR[2]                     ; иначе установка бита в WR2
                ;5242              BIT LS[BIT31] WITH WR[3],                ; проверка, установлен ли бит 31
U 0769, D97F,B5 ;5243            DT(LONG)&SET.ALU.CC                        ; установка кодов условий
U 076A, 5B00,09 ;5244            SKIP.IF[BITS.CLR]                          ; пропуск следующей инструкции, если бит 31 сброшен
U 076B, C77F,15 ;5245            BIS LS[BIT31] TO WR[2]                     ; иначе установка бита в WR2
U 076C, C045,95 ;5246            ADD LS[#4] TO WR[3]                        ; прибавление 4 к тестовым данным (местная память
                ;5247                                                       ; увеличивается на 4)
U 076D, 457D,95 ;5248            BIC LS[BIT30] TO WR[3]                     ; сброс бита 30
U 076E, C57F,95 ;5249            BIC LS[BIT31] TO WR[3]                     ; сброс бита 31
U 076F, AEC5,95 ;5250            BIS WR[2] TO WR[3]                         ; восстановление битов 30 и 31 (в MCT не
                ;5251                                                       ; изменяются)
U 0770, 2341,95 ;5252            ROR WR[3]                                  ; сдвиг вправо, как выполняется аппаратурой
                ;5253    LOOP.T2.4:
U 0771, 2006,95 ;5254            MOV WR[3] TO WR[1]                         ; восстановление ожидаемых данных
U 0772, 1D14,75 ;5255            MEM.REQ[READ.MAINT.ADRS] ADRS[T10] DT[LONG] ; запись дополнительного кода данных в регистр
                ;5256                                                       ; виртуального адреса
```

```
U 0773,  3022,15 ;5257            MOV MEM.DATA TO WR[0]                          ; выполнение пересылки для окончания цикла, но проверка
                 ;5258                                                           ; не выполняется
U 0774,  9A13,F5 ;5259            MEM.REQ[READ.MAINT.VAR.INC] ADRS[T9] DT[LONG] ; пересылка тестовых данных в регистр виртуального
                 ;5260                                                           ; адреса
U 0775,  3022,15 ;5261            MOV MEM.DATA TO WR[0]                          ; выборка результата
U 0776,  0869,3C ;5262            JSR [CHECK.RESULT]                             ; проверка результата
U 0777,  0877,14 ;5263            JMP [LOOP.T2.4]                                ; зацикливание при ошибке, если разрешено
U 0778,  B611,95 ;5264            MOV LS[T8] TO WR[3]                            ; выборка старого набора со сдвигаемым нулем
U 0779,  A3C1,95 ;5265            ROL WR[3]                                      ; создание нового набора со сдвигаемым нулем
U 077A,  3E11,95 ;5266            MOV WR[3] TO LS[T8]                            ; сохранение нового набора в ячейке T8 местной памяти
                 ;5267              BIT LS[BIT0] WITH WR[3],                     ; проверка, выполнен ли сдвиг тестовых данных. Бит 0
                 ;5268                                                           ; должен быть сброшен, когда выполнено
U 077B,  5941,B5 ;5269            DT(LONG)&SET.ALU.CC                            ; установка кодов условий
U 077C,  8876,21 ;5270            JMP.IF[BIT.SET] TO [REPEAT.T2.4]               ; повторение сдвига тестовых данных, если бит 0
                 ;5271                                                           ; еще установлен
U 077D,  7FF8,15 ;5272            INC LS[OS]                                     ; иначе увеличение индекса для очистки другого бита
U 077E,  B6F7,15 ;5273            MOV LS[SHIFT.OS(4-0)] TO WR[2]                 ; выборка тестовых данных для проверки, закончен ли тест
                 ;5274              BIT LS[BIT0] WITH WR[2],                     ; закончен, если бит 0 установлен (как раз заканчивается
                 ;5275                                                           ; сбросом бита 31 тестовых данных)
U 077F,  D941,35 ;5276            DT(LONG)&SET.ALU.CC                            ; установка кодов условий
U 0780,  0876,29 ;5277            JMP.IF[BITS.CLR] TO [REPEAT.T2.4]              ; начинается снова сдвиг тестовых данных с различными
                 ;5278                                                           ; сброшенными дополнительными битами до окончания теста
                 ;5279    END.T2:
```

```
;5280    .PAGE "БАЗОВЫЙ ТЕСТ CSR1 И ТЕСТЫ ВЕТВЛЕНИЯ"
;5281    .TOC  "ТЕСТ 3 - тест чтения/записи CSR1 (частичный) (модули MCT и DAP)"
;5282    ;
;5283    ; ОПИСАНИЕ ТЕСТА:
;5284    ;
;5285    ;       Этот тест проверяет возможность чтения/записи битов 29-25 CSR1. Другие би-
;5286    ;    ты CSR1 будут проверяться в последующих тестах. Этот тест использует точки
;5287    ;    входа "ЗАПИСЬ РЕГИСЬРА УПРАВЛЕНИЯ И СОСТОЯНИЯ (CSR)" и "ЧТЕНИЕ РЕГИСТРА УПРАВ-
;5288    ;    ЛЕНИЯ И СОСТОЯНИЯ (CSR)", связанные с инструкцией MEM.REQ центрального процес-
;5289    ;    сора. Путь данных следующий: центральный процессор выдает иструкцию MEM.REQ
;5290    ;    со значением поля MF=17(H) (запись регистра управления и состояния). Ветвление
;5291    ;    в памяти выполняется так, как описано в начале этого листинга. Адрес, переда-
;5292    ;    ваемый в контроллер ОЗУ, содержит биты 3 и 2 установленные в 01(B). Они
;5293    ;    буферизируются в регистре виртуального адреса и будут использоваться для вы-
;5294    ;    борки CSR1. После ветвления выполняется следущая последовательность: по адре-
;5295    ;    су ветвления устанавливается занятость памяти и разрешаются формирователи из
;5296    ;    DAP. Следующий цикл памяти поддерживает разрешение формирователей из DAP, за-
;5297    ;    нятость памяти и разрешает сигнал сброса признаков ошибок в CSR1. Это микрос-
;5298    ;    лово зацикливается в ожидании запроса данных из DAP. Запрос данных формирует-
;5299    ;    ся инструкцией MOV центрального процессора, где память является приемником.
;5300    ;    Центральный процессор посылает тестовые данные инструкцией MOV, и когда конт-
;5301    ;    роллер ОЗУ получает сигнал DATA REQ, микрокод памяти переходит в следующий
;5302    ;    цикл. В это время данные из центрального процессора находятся на шине MC BUS
;5303    ;    и этот микроцикл поддерживает выставленный сигнал MEMORY для предотвращения
;5304    ;    перехода центрального процессора на следующую микроинструкцию (центральный
;5305    ;    процессор приостановлен до снятия занятости памяти). Кроме того, этот цикл
;5306    ;    выполняет запись в CSR возбуждением сигнала WR CSR L. Этот сигнал возбуждает
;5307    ;    сигнал WR CSR1 H, так как LVA03 и LVA02 содержат 01(B). WR CSR1 формируетоя
;5308    ;    в ПМЛ УПР.ПРЕДВЫБОРКОЙ. Данные из BUS MC D29 по BUS MC D25 будут записаны в
;5309    ;    биты CSR1 29 по 25. Следующий микроцикл снимает занятость памяти, подготавли-
;5310    ;    вается для другого запроса центрального процессора и переходит в холостой
;5311    ;    цикл. Снятие занятости памяти снимает приостанов центрального процессора.
;5312    ;    тогда выполняется чтение CSR. Инструкция MEM.REQ выдается с полем MF=16(H)
;5313    ;    (чтение CSR). Передаваемый адрес такой же, как и при записи. По адресу ветв-
;5314    ;    ления устанавливается MEMORY BUSY для приостанова центрального процессора,
;5315    ;    когда инструкция MOV центрального процессора будет выдана для чтения данных
;5316    ;    обратно. Следующий микроцикл памяти поддерживает выставленный сигнал MEMORY
;5317    ;    BYSY и выставляет сигнал RD CSR L. Этот сигнал совместно с LVA [03:02] выстав-
;5318    ;    ляет сигнал RD CSR L, который разрешает выдачу данных из CSR1 на шину MC BUS.
;5319    ;    Следующий микроцикл памяти снимает MEMORY BUSY и зацикливается в ожидании
;5320    ;    сигнала DATA RCVD из DAP для сброса сигнала CPU GRANT. Когда сигнал CPU GRANT
;5321    ;    сбрасывается, данные CSR уже могут считываться инструкцией MOV центрального
;5322    ;    процессора и могут быть проверены. Микрокод памяти подготовится для другого
;5323    ;    запроса центрального процессора и возвратится в холостой цикл. Тестовыми дан-
;5324    ;    ными являются все единицы, все нули и сдвигаемая единица.
;5325    ;
;5326    ; ПРЕДПОЛОЖЕНИЯ:
;5327    ;
;5328    ;       Предполагается, что тест регистра виртуального адреса выполнен успешно.
;5329    ;    Этот тест использует один новый сигнал из модуля DAP, DATA REQ H, который
;5330    ;    может быть причиной неисправности. Сигнал был проверен внутри модуля DAP,
;5331    ;    но выходной контакт до этого не использовался.
;5332    ;
;5333    ; ШАГИ ТЕСТА:
;5334    ;
```

```
;5335    ;    1) Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;5336    ;       (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;5337    ;       мяти.
;5338    ;    2) Загрузка в WR1 тестовых данных, которые будут использоваться.
;5339    ;    3) Загрузка битов 29-25 WR0 единицами (дополнительный код первого набора
;5340    ;       данных). Выполнение инструкции MEM.REQ с DT=LONGWORD и MF=WRITE.CSR. Ис-
;5341    ;       пользование ячейки местной памяти, содержащей 4(H) в качестве адреса.
;5342    ;    4) Выполнение нескольких задержек (NOP) чтобы убедиться, что контроллер
;5343    ;       ОЗУ ожидает запроса данных. Затем выполнение инструкции MOV из мес-
;5344    ;       тной памяти в память для загрузки CSR1 данными.
;5345    ;    5) Выполнение инструкции MEM.REQ с DT=LONGWORD и MF=READ.CSR. Использование
;5346    ;       ячейки местной памяти, содержащей константу 4(H) в качестве виртуального
;5347    ;       адреса (этим выбирается CSR1).
;5348    ;    6) Выполнение инструкции MOV из памяти в WR[0] для чтения CSR1 и проверки
;5349    ;       результата.
;5350    ;    7) Повторение шагов с 2 по 6 для каждого набора тестовых данных.
;5351    ;
;5352    ; ОШИБКИ:
;5353    ;
;5354    ;    ошибка 1 - запись или чтение CSR1 работает неправильно с данными - все нули.
;5355    ;    ошибка 2 - запись или чтение CSR1 работает неправильно с данными - все еди-
;5356    ;               ницы.
;5357    ;    ошибка 3 - запись или чтение CSR1 работает неправильно с данными - сдвигае-
;5358    ;               мая единица.
;5359    ;
;5360    ; НАЛАДКА:
;5361    ;
;5362    ;    ОШИБКА 1 - Эта ошибка показывает неправильную работу при чтении или записи
;5363    ;               битов 29-25 CSR1. Если один или два бита неправильные, необходимо
;5364    ;               проверить изменение уровня входов BUS MC D29 по BUS MC D25 регис-
;5365    ;               тра CSR1. Также необходимо проверить изменение уровня входов
;5366    ;               счетверенного буфера при чтении. Любой обрыв в этих точках должен
;5367    ;               установить 1 вместо 0. Если уровни не меняются, остановите цент-
;5368    ;               ральный процессор в пошаговом режиме на инструкции MEM.REQ.
;5369    ;               Микропрограмма памяти должна циклиться на инструкции, которая
;5370    ;               проверяет возбуждение сигнала центрального процессора DATA
;5371    ;               REQUEST (эта инструкция является первой после микрослова ветвле-
;5372    ;               ния).
;5373    ;                  Если микропрограмма памяти не циклится на этом микрослове, по-
;5374    ;               видимому, неправильно работают схемы следующего адреса или ветв-
;5375    ;               ления. Необходимо проверить сигнал L CPU DR H на входе мультиплек-
;5376    ;               сора, формирующего ADRS 1 L (микроадрес). Этот сигнал должен быть
;5377    ;               низким. Если нет, необходимо проследить этот сигнал в обратном на-
;5378    ;               правлении до регистра и дальше до контакта модуля MCT. Также необ-
;5379    ;               ходимо проверить строб и разрешение этого регистра. Если сигнал
;5380    ;               DATA REQ H неправильный, необходимо проверить контакт на модуле
;5381    ;               DAP. Если сигнал L CPU DR H правильный, наиболее легким способом
;5382    ;               проверки последовательности является пошаговый режим контроллера
;5383    ;               ОЗУ. Необходимо проверить наличие 6(H) на линиях выборки мульти-
;5384    ;               плексора ветвления (BEN1) и высокий уровень выходного сигнала
;5385    ;               ADRS 1 L.
;5386    ;                  Если память циклится на правильном микрослове, опять необходимо
;5387    ;               проверить правильность данных на входах регистра CSR1. Останови-
;5388    ;               те центральный процессор в пошаговом режиме на инструкции
;5389    ;               MOV WR[1] TO MEM и проверьте, что микропрограмма памяти выходит
```

```
                ;5390   ;               из цикла и возвращается в холостой цикл. Если нет, проверьте
                ;5391   ;               высокий уровень сигнала L CPU DR H, как описано выше. Если микро-
                ;5392   ;               программа памяти продвигается, необходимо проверить изменение
                ;5393   ;               сигнала WR CSR1 H с низкого уровня на высокий на регистре CSR1.
                ;5394   ;               Также необходимо проверить, что сигнал CLR CSR L остается высоким
                ;5395   ;               (легче всего это сделать в пошаговом режиме MCT, но можно прове-
                ;5396   ;               рить и при зацикливании по ошибке). Сигнал WR CSR1 H поступает из
                ;5397   ;               ПМЛ УПР.ПРЕДВЫБОРКОЙ и формируется из низкого уровня WR CSR L,
                ;5398   ;               LVA 03 H и высокого уровня LVA 02 H.
                ;5399   ;                  Выходы регистра CSR1 можно проверить после возврата памяти
                ;5400   ;               в холостой цикл. Также проверяются входы счетверенного буфера.
                ;5401   ;                  Чтение может быть проверено остановкой центрального процессора
                ;5402   ;               в пошаговом режиме на второй инструкции MEM.REQ (т.е., с полем
                ;5403   ;               MF=READ.CSR). Микропрограмма памяти должна циклиться, ожидая сиг-
                ;5404   ;               нала DATA RCVD H из центрального процессора для сброса сигнала CPU
                ;5405   ;               GRANT L. Во время этого цикла можно проверить низкий уровень
                ;5406   ;               сигнала разрешения RD CSR1 L на счетверенном буфере. Если этот
                ;5407   ;               сигнал высокий, необходимо проверить выход дешифратора, который
                ;5408   ;               формирует этот сигнал. Если здесь неправильно, необходимо про-
                ;5409   ;               верить низкий уровень на входах RD CSR L и LVA 03 H и высокий
                ;5410   ;               уровень на входе LVA 02 H. Если входы правильные, дешифратор не-
                ;5411   ;               исправен. Если сигнал RD CSR1 L низкий, необходимо проверить вхо-
                ;5412   ;               ды и выходы буфера. Подготовленные тестовые данные для инструкции
                ;5413   ;               MOV центрального процессора должны просматриваться на шине MC BUS.
                ;5414   ;
                ;5415   ;   ОШИБКА 2 - По существу то же самое, что и при ошибке 1, за исключением тестовых
                ;5416   ;               данных. Скорее всего, регистр CSR1 или счетверенный буфер являются
                ;5417   ;               причиной неисправности.
                ;5418   ;
                ;5419   ;   ОШИБКА 3 - Скорее всего, замыкание между связями регистра CSR1 и счетверенно-
                ;5420   ;               го буфера или внутри этих микросхем является причиной неисправности.
                ;5421   ;               При наладке следует пользоваться последним параграфом описания
                ;5422   ;               ошибки 1.
                ;5423   ;
                ;5424   T.3:
U 0781, B65E,15 ;5425           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;5426                                                   ; состояния
U 0782, 3EB0,15 ;5427           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;5428                                                   ; 15 указывает начало теста для конс.процессора
U 0783, 10E0,15 ;5429           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN для конс.процессора
                ;5430   WAIT.T3.0:
U 0784, 0878,44 ;5431           JMP [WAIT.T3.0]                         ; зацикливание для ожидания ответа конс.процессора
U 0785, 0A1A,AC ;5432           JSR [SETUP.1]                           ; установка масок, кода модуля и пр.
U 0786, 4742,15 ;5433           BIS LS[CPU] TO WR[0]                    ; добавление кода модуля DAP
U 0787, 3E8C,15 ;5434           MOV WR[0] TO LS[MODULE.NUM]             ; запоминание кода модуля в местной памяти для указания
                ;5435                                                   ; модулей DAP и MCT
U 0788, B69E,15 ;5436           MOV LS[ONES] TO WR[0]                   ; запрет проверки ошибок всех битов
U 0789, 457A,15 ;5437           BIC LS[BIT29] TO WR[0]                  ; сброс (проверка) бита 29
U 078A, C578,15 ;5438           BIC LS[BIT28] TO WR[0]                  ; сброс (проверка) бита 28
U 078B, 4576,15 ;5439           BIC LS[BIT27] TO WR[0]                  ; сброс (проверка) бита 27
U 078C, C574,15 ;5440           BIC LS[BIT26] TO WR[0]                  ; сброс (проверка) бита 26
U 078D, C572,15 ;5441           BIC LS[BIT25] TO WR[0]                  ; сброс (проверка) бита 25
U 078E, 3E8A,15 ;5442           MOV WR[0] TO LS[ERROR.MASK]             ; установка маски ошибки для проверки битов 29-25
U 078F, B69E,15 ;5443           MOV LS[ONES] TO WR[0]                   ; установка дополнительного кода ожидаемого результата
                ;5444   LOOP.T3.1:
```

```
U 0790,  2F82,95 ;5445            CLR WR[1]                                ; ожидаемый результат
U 0791,  1D45,F5 ;5446            MEM.REQ[WRITE.CSR] ADRS[CSR1] DT[LONG]   ; выполнение записи в CSR1
U 0792,  0A1B,4C ;5447            JSR [NOP.DELAY]                          ; выполнение 5 циклов задержки для проверки, что модуль
                 ;5448                                                     ; MCT ожидает сигнала DATA REQUEST
U 0793,  B29C,15 ;5449            WRITE.MEM LS[ZERO]                       ; пересылка тестового набора данных
U 0794,  9D45,75 ;5450            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; чтение данных из CSR
U 0795,  3022,15 ;5451            MOV MEM.DATA TO WR[0]                    ; выборка результата
U 0796,  0869,3C ;5452            JSR [CHECK.RESULT]                       ; проверка результата
U 0797,  0879,04 ;5453            JMP [LOOP.T3.1]                          ; зацикливание при ошибке, если разрешено
U 0798,  FF82,15 ;5454            INC LS[ERROR.NUMBER]                     ; номер ошибки=2
U 0799,  2F80,15 ;5455            CLR WR[0]                                ; установка дополнительного кода ожидаемого результата
                 ;5456    LOOP.T3.2:
U 079A,  369E,95 ;5457            MOV LS[ONES] TO WR[1]                    ; ожидаемый результат
U 079B,  1D45,F5 ;5458            MEM.REQ[WRITE.CSR] ADRS[CSR1] DT[LONG]   ; выполнение записи в CSR1
U 079C,  0A1B,4C ;5459            JSR [NOP.DELAY]                          ; выполнение 5 циклов задержки для проверки, что модуль
                 ;5460                                                     ; MCT ожидает сигнала DATA REQUEST
U 079D,  329E,15 ;5461            WRITE.MEM LS[ONES]                       ; пересылка тестовых данных
U 079E,  9D45,75 ;5462            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; чтение данных из CSR
U 079F,  3022,15 ;5463            MOV MEM.DATA TO WR[0]                    ; выборка результата
U 07A0,  0869,3C ;5464            JSR [CHECK.RESULT]                       ; проверка результата
U 07A1,  0879,A4 ;5465            JMP [LOOP.T3.2]                          ; зацикливание при ошибке, если разрешено
U 07A2,  FF82,15 ;5466            INC LS[ERROR.NUMBER]                     ; номер ошибки=3
U 07A3,  E5F8,15 ;5467            CLR LS[OS]                               ; очистка индекса
                 ;5468    LOOP.T3.3:
U 07A4,  B6F6,95 ;5469            MOV LS[SHIFT.OS(4-0)] TO WR[1]           ; ожидаемый результат
U 07A5,  1D45,F5 ;5470            MEM.REQ[WRITE.CSR] ADRS[CSR1] DT[LONG]   ; выполнение записи в CSR1
U 07A6,  B2F6,15 ;5471            WRITE.MEM LS[SHIFT.OS(4-0)]              ; пересылка тестовых данных
U 07A7,  9D45,75 ;5472            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; чтение данных из CSR
U 07A8,  3022,15 ;5473            MOV MEM.DATA TO WR[0]                    ; выборка результата
U 07A9,  0869,3C ;5474            JSR [CHECK.RESULT]                       ; проверка результата
U 07AA,  887A,44 ;5475            JMP [LOOP.T3.3]                          ; зацикливание при ошибке, если разрешено
U 07AB,  7FF8,15 ;5476            INC LS[OS]                               ; увеличение индекса для следующего набора
U 07AC,  36F9,15 ;5477            MOV LS[OS] TO WR[2]                      ; подготовка для проверки, все ли наборы использованы
                 ;5478              BIT LS[BIT5] WITH WR[2],               ; проверка, что содержимое регистра OS увеличено до
                 ;5479                                                     ; 20(H) и
U 07AD,  D948,35 ;5480            DT(LONG)&SET.ALU.CC                      ; установка кодов условий
U 07AE,  087A,49 ;5481            JMP.IF[BITS.CLR] TO [LOOP.T3.3]          ; повторение со следующим набором, если не все
                 ;5482                                                     ; использованы
                 ;5483    END.T3:
```

```
;5484    .PAGE  "ТЕСТ 4 – тест ветвления по LVA00, L ECC DIS и L DIAG CHK (модуль MCT)"
;5485    ;
;5486    ;  ОПИСАНИЕ ТЕСТА:
;5487    ;
;5488    ;          Этот тест проверяет схемы ветвления, используемые для управления бита-
;5489    ;       ми 0 и 1 (BEN0 и BEN1 соответственно) микроадреса через 3 входа: ветвление
;5490    ;       BEN0 по сигналам LVA00 и L DIAG CHK и ветвление BEN1 по сигналу L ECC DIS.
;5491    ;       сигналы L DIAG CHK и L ECC DIS поступают из CSR1 и непосредственно
;5492    ;       записываются инструкцией MEM.REQ с функцией WRITE.CSR (запись CSR). Сигнал
;5493    ;       LVA00 поступает непосредственно из регистра виртуального адреса. Путь
;5494    ;       данных следующий: центральный процессор выдает инструкцию MEM.REQ
;5495    ;       с полем MF=READ.MAINT.BR.CHECK (1B(H)) (чтение в режиме наладки с про-
;5496    ;       веркой ветвления). Ветвление в контроллере ОЗУ происходит так, как
;5497    ;       описано в начале этого листинга. После ветвления выполняется следующая
;5498    ;       последовательность: первое микрослово устанавливает занятость памяти и
;5499    ;       выполняет ветвление по сигналу LVA00 (BEN0, бит 0 микроадреса). Если сиг-
;5500    ;       нал LVA00 сброшен (0), тогда микропрограмма памяти поддерживает за-
;5501    ;       нятость памяти еще один цикл, затем возвращает содержимое регистра
;5502    ;       биртуального адреса на шину MC BUS (исползуя конец подпрограммы
;5503    ;       READ.MAINT.ADRS (чтение адреса в режиме наладки)). Центральный процессор
;5504    ;       тогда проверяет, что содержимое регистра виртуального адреса не изменено.
;5505    ;       Если сигнал LVA00 установлен (1), тогда занятость памяти поддерживается
;5506    ;       установленной и содержимое регистра виртуального адреса увеличивается.
;5507    ;       Кроме того, в этом микрослове выполняется ветвление в зависимости от
;5508    ;       значения сигналов L DIAG CHK и L ECC DIS. Сигналы L DIAG CHK и L ECC DIS
;5509    ;       уже были установлены предшествующей подпрограммой записи в CSR до начала
;5510    ;       этой проверки. Следующие микрослова увеличивают значение регистра вир-
;5511    ;       туального адреса, как описано ниже, и возвращают значение регистра вирту-
;5512    ;       ального адреса в центральный процессор для проверки: сигналы L DIAG
;5513    ;       CHK и L ECC DIS установлены – VAR не увеличивается; L DIAG CHK установлен,
;5514    ;       L ECC DIS сброшен – VAR увеличивается на 1; L DIAG CHK сброшен, L ECC DIS
;5515    ;       установлен – VAR увеличивается на 2; L DIAG CHK сброшен, L ECC DIS сброшен
;5516    ;       – VAR увеличивается на 3. Центральный процессор читает значение регистра
;5517    ;       виртуального адреса для определения, правильно ли выполнены ветвления.
;5518    ;
;5519    ;  ПРЕДПОЛОЖЕНИЯ:
;5520    ;
;5521    ;       Предполагается, что все предыдущие тесты выполнены успешно.
;5522    ;
;5523    ;  ШАГИ ТЕСТА:
;5524    ;
;5525    ;       1.Установка номера ошибки и номера модуля в местной памяти (для распе-
;5526    ;         чатки ошибок) и очистка предыдущего номера ошибки в местной памяти.
;5527    ;       2.Установка маски ошибки для проверки только младшего байта и очистка
;5528    ;         WR1.
;5529    ;       3.Выполнение инструкции MEM.REQ с DT=LONGWORD и MF=READ.MAINT.BR.CHECK
;5530    ;         с использованием ячейки местной памяти, содержащей нули, в качестве
;5531    ;         данных.
;5532    ;       4.Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка регистра вир-
;5533    ;         туального адреса на нуль.
;5534    ;       5.Запись в CSR1 для установки сигналов L DIAG.CHK и L ECC DIS. Запись
;5535    ;         2(H) в WR1.
;5536    ;       6.Выполнение инструкции MEM.REQ с DT=LONGWORD и MF=READ.MAINT.BR.CHECK
;5537    ;         с использованием ячейки местной памяти, содержащей 1 (бит 0 установлен),
;5538    ;         в качестве данных.
```

```
;5539    ;       7. Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка содержимого
;5540    ;          регистра виртуального адреса на 2(H).
;5541    ;       8. Запись в CSR1 для сброса L DIAG.CHK и установки L ECC DIS. Запись
;5542    ;          4(H) в WR1 (ожидаемые данные).
;5543    ;       9. Повторение шагов 6 и 7 для проверки содержимого регистра виртуального
;5544    ;          адреса на 4(H).
;5545    ;      10. Запись в CSR1 для установки L DIAG CHK и сброса L ECC DIS. Запись 6(H)
;5546    ;          в WR1 (ожидаемые данные).
;5547    ;      11. Повторение шагов 6 и 7 для проверки содержимого регистра виртуального
;5548    ;          адреса на 6(H).
;5549    ;      12. Запись в CSR1 для сброса L DIAG CHK и L ECC DIS. Запись 8(H) в WR1
;5550    ;          (ожидаемые данные).
;5551    ;      13. Повторение шагов 6 и 7 для проверки содержимого регистра виртуального
;5552    ;          адреса на 8(H).
;5553    ;
;5554    ;   ОШИБКИ:
;5555    ;
;5556    ;       ПРИМЕЧАНИЕ: ожидаемыми и полученными данными, печатаемыми в сообщении об
;5557    ;       ошибке, являются ожидаемое и полученное значения регистра виртуального
;5558    ;       адреса. Эти значения используются для определения, где микропрограмма
;5559    ;       памяти выполнила ветвление. см. описание наладки.
;5560    ;
;5561    ;       ошибка 1 - ветвление по BEN0=LVA00 работает неправильно при LVA00=0
;5562    ;       ошибка 2 - ветвление по BEN0=LVA00 или BEN0=L DIAG CHK и BEN1=L ECC DIS
;5563    ;                  работает неправильно при LVA00=1, L DIAG CHK=1 и L ECC DIS=1
;5564    ;       ошибка 3 - ветвление по BEN0=L DIAG CHK и BEN1=L ECC DIS работает непра-
;5565    ;                  вильно при L DIAG CHK=0 и L ECC DIS=1
;5566    ;       ошибка 4 - ветвление по BEN0=L DIAG CHK и BEN1=L ECC DIS работает непра-
;5567    ;                  вильно при L DIAG CHK=1 и L ECC DIS=0
;5568    ;       ошибка 5 - ветвление по BEN0=L DIAG CHK и BEN1=L ECC DIS работает непра-
;5569    ;                  вильно при L DIAG CHK=0 и L ECC DIS=0
;5570    ;
;5571    ;   НАЛАДКА:
;5572    ;
;5573    ;          ОШИБКА 1 - Эта ошибка указывает на неправильную работу мультиплексора
;5574    ;       BEN0 или его входов. Необходимо проверить низкий уровень сигнала LVA00 H
;5575    ;       на входе мультиплексора во время инструкции MEM.REQ. Источник этого сиг-
;5576    ;       нала уже был проверен, так что может быть обрыв цепи, на что указывает
;5577    ;       изменение уровня. Если этот сигнал правильный, необходимо проверить вхо-
;5578    ;       ды с 09, с 10 и с 11. Если модуль MCT может работать в пошаговом режиме,
;5579    ;       необходимо остановить его на первом микрослове после ветвления, где BEN0=
;5580    ;       LVA00. Сигналы выборки C 11 H, C 10 H и C 09 H должны быть равными 001(B)
;5581    ;       соответственно. Если эти входы правильные, повидимому, мультиплексор не-
;5582    ;       исправен. Если модуль MCT не может работать в пошаговом режиме, входы
;5583    ;       должны быть проверены при зацикливании по ошибке.
;5584    ;
;5585    ;          ОШИБКА 2 - Эта ошибка указывает на неправильную работу мультиплексоров
;5586    ;       BEN0, BEN1 или их входов. Если полученными данными (регистр виртуального
;5587    ;       адреса) является 0, тогда неправильно работает BEN0=LVA00. Остановите
;5588    ;       центральный процессор в пошаговом режиме на инструкции MEM.REQ, ко-
;5589    ;       торая посылает в регистр виртуального адреса данные=1, и проверьте вы-
;5590    ;       сокий уровень сигнала LVA00 H на входе мультиплексора BEN0. Если он
;5591    ;       правильный, следует проверить линии выборки. Если модуль MCT может
;5592    ;       работать в пошаговом режиме, остановите его на микрослове памяти
;5593    ;       после ветвления (BEN0=LVA00), и проверьте наличие 001(B) на входах C 11 H,
```

```
;5594    ;       С 10 Н и С 09 Н соответственно. Если входы правильные, повидимому, муль-
;5595    ;       типлексор неисправен. Если полученные данные больше 2, тогда неправильно
;5596    ;       работает BEN0=L DIAG CHK или BEN1=L ECC DIS. Если полученные данные=4(H),
;5597    ;       тогда неправильно работает BEN1=L ECC DIS. Если полученные данные=6(H),
;5598    ;       тогда неправильно работает BEN0=L ECC DIS. Если полученные данные=8(H),
;5599    ;       тогда неправильно работают обе цепи. Необходимо проверить входы соот-
;5600    ;       ветствующего мультиплексора, как описано выше. Если модуль MCT работает
;5601    ;       в пошаговом режиме, для проверки входов остановите его на втором микро-
;5602    ;       слове памяти после адреса ветвления. В противном случае необходи-
;5603    ;       мо зацикливание при ошибке. Входы выборки мультиплексора BEN0 должны
;5604    ;       быть 100(B) и сигнал L DIAG CHK H должен быть высоким. Входы выборки
;5605    ;       мультиплексора BEN1 должны быть 011(B) и сигнал L ECC DIS H должен быть
;5606    ;       высоким. Если входы правильные, подозревается мультиплексор.
;5607    ;
;5608    ; ПРИМЕЧАНИЕ:
;5609    ;       Сигналы L DIAG CHK H и L ECC DIS H могут быть проверены без останова MCT
;5610    ;       в пошаговом режиме, а останавливая в пошаговом режиме центральный процес-
;5611    ;       сор на иструкции MEM.REQ, которая вызывает эту проверку. Эти сигналы долж-
;5612    ;       ны быть истинными после записи в CSR1.
;5613    ;
;5614    ;          ОШИБКА 3 - Если полученные данные=2(H), тогда по-видимому, сигнал
;5615    ;       L DIAG CHK H не становится низким на входе мультиплексора BEN0. Любые
;5616    ;       другие значения полученных данных указывают неправильные входы выборки
;5617    ;       одного из двух мультиплексоров или неисправность самих мультиплексоров.
;5618    ;       Если модуль MCT может работать в пошаговом режиме, необходимо остановить
;5619    ;       микропрограмму памяти на втором микрослове после адреса ветвления (BEN0=
;5620    ;       L DIAG CHK, BEN1=L ECC DIS) и проверить входы мультиплексоров следующим
;5621    ;       образом: выборка мультиплексора BEN0 должна быть 100(B) и сигнал L DIAG
;5622    ;       CHK H должен быть низким, выборка мультиплексора BEN1 должна быть 011(B)
;5623    ;       и сигнал L ECC DIS H должен быть высоким. Если модуль MCT не может ра-
;5624    ;       ботать в пошаговом режиме, тогда для проверки входов выборки должно быть
;5625    ;       использовано зацикливание при ошибке.
;5626    ;
;5627    ; ПРИМЕЧАЕИЕ:
;5628    ;          Сигналы L DIAG CHK H и L ECC DIS H могут быть проверены без оста-
;5629    ;       нова MCT в пошаговом режиме, а останавливая центральный процессор на
;5630    ;       инструкции MEM.REQ, которая вызывает эту проверку. Эти сигналы будут
;5631    ;       истинными после записи в CSR1.
;5632    ;
;5633    ;          ОШИБКА 4 - если полученные данные=2(H), тогда, по-видимому, сигнал
;5634    ;       L ECC DIS H не становится низким на входе мультиплексора BEN1. Любые
;5635    ;       другие значения полученных данных указывают неправильные входы выборки
;5636    ;       одного из двух мультиплексоров или неисправность самих мультиплексоров.
;5637    ;       Если модуль MCT может работать в пошаговом режиме, необходимо остано-
;5638    ;       вить микропрограмму памяти на втором микрослове после адреса ветвления
;5639    ;       (BEN0=L DIAG CHK, BEN1=L ECC DIS) и проверить входы мультиплексоров сле-
;5640    ;       дующим образом: выборка мультиплексора BEN0 должна быть 100(B) и сигнал
;5641    ;       L DIAG CHK H должен быть высоким, выборка мультиплексора BEN1 должна быть
;5642    ;       011(B) и сигнал L ECC DIS H должен быть низким. Если модуль MCT не может
;5643    ;       работать в пошаговом режиме, тогда для проверки входов выборки должно
;5644    ;       использоваться зацикливание при ошибке.
;5645    ;
;5646    ; ПРИМЕЧАНИЕ:
;5647    ;          Сигналы L DIAG CHK H и L ECC DIS H могут быть проверены без
;5648    ;       останова MCT в пошаговом режиме, а останавливая центральный процессор
```

```
                ;5649   ;       на инструкции MEM.REQ, которая вызывает эту проверку. Эти сигналы будут
                ;5650   ;       истинными после записи в CSR1.
                ;5651   ;
                ;5652   ;          ОШИБКА 5 - Если это первая ошибка, скорее всего, неисправен мультип-
                ;5653   ;       лексор. Если полученные данные=4(H), тогда подозревается мультиплексор BEN1.
                ;5654   ;       если полученные данные=6(H), тогда подозревается мультиплексор BEN0.
                ;5655   ;       Если полученные данные=2(H), тогда подозреваются оба мультиплексора (это
                ;5656   ;       маловероятно).
                ;5657   ;
                ;5658    T.4:
U 07AF, B65E,15 ;5659            MOV LS[BEGIN.TEST] TO WR[0]                    ; установка бита 15 в WR[0] для слова управления и
                ;5660                                                           ; состояния
U 07B0, 3E80,15 ;5661            MOV WR[0] TO LS[CONTROL.STATUS]                ; установка бита 15 в слове управления и состояния. Бит
                ;5662                                                           ; 15 указывает начало теста для консольного процессора
U 07B1, 10F0,15 ;5663            MISC [SET.CP.ATTN]                             ; выдача сигнала CPU ATTN для консольного процессора
                ;5664    WAIT.T4.0:
U 07B2, 087B,24 ;5665            JMP [WAIT.T4.0]                                ; зацикливание для ожидания ответа консольного
                ;5666                                                           ; процессора
U 07B3, 0A1A,AC ;5667            JSR [SETUP.1]                                  ; установка масок, кода модуля и др.
U 07B4, B62C,15 ;5668            MOV LS[#FFFFFF00] TO WR[0]                     ; установка данных для маски ошибки
U 07B5, 3E8A,15 ;5669            MOV WR[0] TO LS[ERROR.MASK]                    ; установка маски ошибки для проверки младшего байта
                ;5670    LOOP.T4.1:
U 07B6, 2F82,95 ;5671            CLR WR[1]                                      ; ожидаемые данные
U 07B7, 1E9D,F5 ;5672            MEM.REQ[READ.MAINT.BR.CHECK] ADRS[ZERO] DT[LONG] ; выполнение проверки ветвления при LVA00=0
U 07B8, 3022,15 ;5673            MOV MEM.DATA TO WR[0]                          ; выборка результата (регистр виртуального адреса)
U 07B9, 0869,3C ;5674            JSR [CHECK.RESULT]                             ; проверка результата
U 07BA, 887B,64 ;5675            JMP [LOOP.T4.1]                                ; зацикливание при ошибке, если разрешено
U 07BB, FF82,15 ;5676            INC LS[ERROR.NUMBER]                           ; номер ошибки=2
U 07BC, 3672,15 ;5677            MOV LS[ECC.DIS] TO WR[0]                       ; подготовка установки бита запрета коррекции (ECC) в
                ;5678                                                           ; CSR1
U 07BD, 4774,15 ;5679            BIS LS[DIAG.CHK] TO WR[0]                      ; также установка бита диагностического контроля
U 07BE, 8A17,FC ;5680            JSR [WRITE.CSR1]                               ; запись данных из WR0 в CSR1
                ;5681    LOOP.T4.2:
U 07BF, B642,95 ;5682            MOV LS[#2] TO WR[1]                            ; ожидаемые данные
U 07C0, 9E41,F5 ;5683            MEM.REQ[READ.MAINT.BR.CHECK] ADRS[#1] DT[LONG] ; выполнение проверки ветвления при LVA00=1,
                ;5684                                                           ; L ECC DIS=1 и L DIAG CHK=1
U 07C1, 3022,15 ;5685            MOV MEM.DATA TO WR[0]                          ; выборка результата (регистр виртуального адреса)
U 07C2, 0869,3C ;5686            JSR [CHECK.RESULT]                             ; проверка результата
U 07C3, 8878,F4 ;5687            JMP [LOOP.T4.2]                                ; зацикливание при ошибке, если разрешено
U 07C4, FF82,15 ;5688            INC LS[ERROR.NUMBER]                           ; номер ошибки=3
U 07C5, 3672,15 ;5689            MOV LS[ECC.DIS] TO WR[0]                       ; подготовка установки бита запрета коррекции (ECC) в
                ;5690                                                           ; CSR1
U 07C6, 8A17,FC ;5691            JSR [WRITE.CSR1]                               ; запись данных из WR0 в CSR1
                ;5692    LOOP.T4.3:
U 07C7, B644,95 ;5693            MOV LS[#4] TO WR[1]                            ; ожидаемые данные
U 07C8, 9E41,F5 ;5694            MEM.REQ[READ.MAINT.BR.CHECK] ADRS[#1] DT[LONG] ; выполнение проверки ветвления при LVA00=1, L
                ;5695                                                           ; ECC DIS=1 и L DIAG CHK=0
U 07C9, 3022,15 ;5696            MOV MEM.DATA TO WR[0]                          ; выборка результата (регистр виртуального адреса)
U 07CA, 0869,3C ;5697            JSR [CHECK.RESULT]                             ; проверка результата
U 07CB, 887C,74 ;5698            JMP [LOOP.T4.3]                                ; зацикливание при ошибке, если разрешено
U 07CC, FF82,15 ;5699            INC LS[ERROR.NUMBER]                           ; номер ошибки=4
U 07CD, 3674,15 ;5700            MOV LS[DIAG.CHK] TO WR[0]                      ; подготовка установки бита диагностического контроля в
                ;5701                                                           ; CSR1
U 07CE, 8A17,FC ;5702            JSR [WRITE.CSR1]                               ; запись данных из WR0 в CSR1
                ;5703    LOOP.T4.4:
```

```
U 07CF,  B644,95 ;5704            MOV LS[#4] TO WR[1]                        ; ожидаемые данные=6. Установка бита 2
U 07D0,  C742,95 ;5705            BIS LS[BIT1] TO WR[1]                      ; сейчас ожидаемые данные=6
U 07D1,  9E41,F5 ;5706            MEM.REQ[READ.MAINT.BR.CHECK] ADRS[#1] DT[LONG] ; выполнение проверки ветвления при LVA00=1, L
                 ;5707                                                       ; ECC DIS=0 и L DIAG CHK=1
U 07D2,  3022,15 ;5708            MOV MEM.DATA TO WR[0]                      ; выборка результата (регистр виртуального адреса)
U 07D3,  0869,3C ;5709            JSR [CHECK.RESULT]                         ; проверка результата
U 07D4,  087C,F4 ;5710            JMP [LOOP.T4.4]                            ; зацикливание при ошибке, если разрешено
U 07D5,  FF82,15 ;5711            INC LS[ERROR.NUMBER]                       ; номер ошибки=5
U 07D6,  0A17,EC ;5712            JSR [CLEAR.CSR1]                           ; запись нуля в CSR1
                 ;5713    LOOP.T4.5:
U 07D7,  3646,95 ;5714            MOV LS[#8] TO WR[1]                        ; ожидаемые данные
U 07D8,  9E41,F5 ;5715            MEM.REQ[READ.MAINT.BR.CHECK] ADRS[#1] DT[LONG] ; выполнение проверки ветвления при LVA00=1, L
                 ;5716                                                       ; ECC DIS=0 и L DIAG CHK=0
U 07D9,  3022,15 ;5717            MOV MEM.DATA TO WR[0]                      ; выборка результата (регистр виртуального адреса)
U 07DA,  0869,3C ;5718            JSR [CHECK.RESULT]                         ; проверка результата
U 07DB,  087D,74 ;5719            JMP [LOOP.T4.5]                            ; зацикливание при ошибке, если разрешено
                 ;5720    END.T4:
```

```
;5721   .PAGE "ТЕСТЫ АДРЕСА, ДАННЫХ И СХЕМ УПРАВЛЕНИЯ ОБЩЕЙ ШИНЫ"
;5722   .TOC "ТЕСТ 5 - тест адреса общей шины (модуль МСТ)"
;5723   ;
;5724   ; ОПИСАНИЕ ТЕСТА:
;5725   ;
;5726   ;       Этот тест проверяет входы UBS AXX ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА,
;5727   ;   схемы управления этих ПМЛ, а также и сами ПМЛ. По завершении этого теста
;5728   ;   ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА будут полностью проверены. Путь данных
;5729   ;   следующий: центральный процессор выдает инструкцию MEM.REQ с полем
;5730   ;   MF = READ.MAINT.UBS (1F) (чтение с общей шины в режиме наладки). Начальное
;5731   ;   ветвление выполняется так, как описано в начале этого листинга. По адре-
;5732   ;   су начального ветвления устанавливается занятость памяти, IB BSY, разреша-
;5733   ;   ется физический адрес, адрес UB, данные UB, DATIP и выполняется ветвление по
;5734   ;   сигналам L DIAG CHK и L ECC DIS. До выполнения инструкции MEM.REQ в CSR1
;5735   ;   были сброшены биты диагностического контроля DIAG CHK и запрета коррекции
;5736   ;   ECC DIS. Поэтому ветвление в микропрограмме памяти в этой точке выполнит пе-
;5737   ;   реход к микрокоманде, которая проверяет схемы адреса общей шины. Первая мик-
;5738   ;   рокоманда памяти после ветвления поддерживает занятость памяти и возбужденное
;5739   ;   состояние IB BSY и сохраняет разрешение UB ADR. Сигнал UB ADR EN L помещает
;5740   ;   содержимое битов с 00 по 08 регистра виртуального адреса и битов с 09 по 17
;5741   ;   PA на линии UBS A0 по UBS A17, так что теперь они поступают на входы UBS AXX
;5742   ;   ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА (регистр виртуального адреса был загружен
;5743   ;   данными инструкцией центрального процессора MEM.REQ). Следующий микроцикл раз-
;5744   ;   решает загрузить внутренние буферы ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА со входов
;5745   ;   UBS AXX и поддерживает сигнал UB ADR EN, занятость памяти и разрешение IB BSY.
;5746   ;   Следующий цикл читает данные обратно в центральный процессор, помещая содержи-
;5747   ;   мое регистра виртуального адреса на шину MC BUS, как и при проверке функции
;5748   ;   READ.MAINT.ADR (выдача виртуального адреса на общую шину и чтение этих дан-
;5749   ;   ных). Используемыми тестовыми данными являются все 1, все 0 и бегущая 1. Би-
;5750   ;   ты регистра виртуального адреса 18 и 31 аппаратурой принудительно устанавли-
;5751   ;   ваются в низкий уровень, а биты 19-30 в высокий уровень. Эти биты также про-
;5752   ;   веряются (они возвращаются обратно как биты 18-29 высоким уровнем, а биты 30
;5753   ;   и 17 низким уровнем).
;5754   ;
;5755   ; ПРЕДПОЛОЖЕНИЯ:
;5756   ;
;5757   ;   Предполагается, что предыдущие тесты регистра виртуального адреса выполнены
;5758   ;   успешно.
;5759   ;
;5760   ; ШАГИ ТЕСТА:
;5761   ;
;5762   ;   1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;5763   ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;5764   ;      памяти.
;5765   ;   2. Выполнение записи в CSR1 для сброса битов ECC DIS и DIAG CHK. Эти данные
;5766   ;      используются для ветвления в микропрограмме памяти к правильной тестовой
;5767   ;      ветви.
;5768   ;   3. Загрузка тестовых данных из местной памяти в WR3.
;5769   ;   4. Сброс битов 31 и 18, установка битов 19-30 (эти биты всегда принудительно
;5770   ;      устанавливаются так аппаратурой до сдвига вправо). Выполнение ROR WR3 для
;5771   ;      получения ожидаемых данных и пересылка в WR1.
;5772   ;   5. Выдача инструкции с DT = LONGWORD (длинное слово) и MF = READ.MAINT.UBS
;5773   ;      (выдача виртуального адреса на общую шину и чтение его в качестве данных),
;5774   ;      используя данные из ячейки местной памяти, содержащей тестовые данные.
;5775   ;   6. Выполнение инструкции MOV MEM TO WR[3] для чтения и проверки результата.
```

```
                ;5776   ;       Биты 17-00 поступают с линий данных общей шины (они возвращаются сдви-
                ;5777   ;       нутыми вправо на 1 бит, так что результат будет в битах 16-00 и 31). Дру-
                ;5778   ;       гие биты проверяются на наличие 1 в битах 19-30 и 0 - в битах 31 и 18 (до
                ;5779   ;       сдвига) на регистре виртуального адреса.
                ;5780   ;    7. Повторение шагов с 3 до 5 с каждым набором данных.
                ;5781   ;
                ;5782   ; ОШИБКИ:
                ;5783   ;
                ;5784   ;    ошибка 1 - тест адреса общей шины выполняется неправильно с данными - все
                ;5785   ;               единицы.
                ;5786   ;    ошибка 2 - тест адреса общей шины выполняется неправильно с данными - все
                ;5787   ;               нули.
                ;5788   ;    ошибка 3 - тест адреса общей шины выполняется неправильно с данными - сдви-
                ;5789   ;               гаемая единица.
                ;5790   ;
                ;5791   ; НАЛАДКА:
                ;5792   ;
                ;5793   ;       ОШИБКА 1 - Эта ошибка указывает на неправильную работу ПМЛ РЕГИСТРА ВИРТУ-
                ;5794   ;    АЛЬНОГО АДРЕСА или схем адреса общей шины. Если МСТ работает в пошаговом режи-
                ;5795   ;    ме, необходимо остановить на микроцикле памяти, который разрешает загрузку
                ;5796   ;    регистра виртуального адреса с линий данных общей шины и проверить высокий
                ;5797   ;    уровень на входах VAR MUX SEL H и VAR LOAD H ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕ-
                ;5798   ;    СА. Также необходимо проверить высокие уровни на входах UBS AXX H. Если вхо-
                ;5799   ;    ды UBS AXX H неправильные, необходимо проверить низкий уровень сигнала UB
                ;5800   ;    ADR EN L на входе управления приемо-передатчиков общей шины и высокие уровни
                ;5801   ;    на входах LVAXX H. Также необходимо проверить входы UBS AXX L. Если входы
                ;5802   ;    ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА правильные, подозреваются сами ПМЛ. Если
                ;5803   ;    неправильные биты в зоне 20-31 (после сдвига), скорее всего неисправна сама
                ;5804   ;    ПМЛ, которая формирует LVA 19 H по LVA 30 H. Необходимо проверить заземление
                ;5805   ;    ножки 3 ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА, которая формирует LVA 31 H и ножки
                ;5806   ;    9 ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА, которая формирует LVA 18 H, если в ре-
                ;5807   ;    зультате биты 0 или 19 неправильные. Если пошаговый режим МСТ невозможен,
                ;5808   ;    тогда эти входы следует просматривать при зацикливании по ошибке.
                ;5809   ;
                ;5810   ;       ОШИБКА 2 - то же, что и для ошибки 1, за исключением того, что входы UBS
                ;5811   ;    AXX H ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА должны быть все нули. Другим возможным
                ;5812   ;    источником ошибки может быть пропадание сигнала ADDR PH L после его установ-
                ;5813   ;    ки. Это показывают неправильные полученные данные в битах с 10 по 18, которые
                ;5814   ;    преобретают высокие уровни вместо низких. Если модуль МСТ может работать в
                ;5815   ;    пошаговом режиме, необходимо остановиться на микроцикле памяти, который заг-
                ;5816   ;    ружает ПМЛ РЕГИСТРА ВИРТУАЛЬНОГО АДРЕСА с UBS AXX и проверить низкий уровень
                ;5817   ;    сигнала ADDR PH L. Если этот сигнал неправильный, необходимо проверить низ-
                ;5818   ;    кий уровень на входе CONT FUNC LAT L и высокий уровень на входе TB DATA EN L
                ;5819   ;    ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Если эти сигналы правильные, подозревается сама ПМЛ
                ;5820   ;    РАЗНЫЕ ФУНКЦИИ УПР.
                ;5821   ;
                ;5822   ;       ОШИБКА 3 - см.ошибку 2. Разница только в тестовых данных, которые представ-
                ;5823   ;    ляют собой сдвигаемую единицу. Вероятнее всего, ошибка появляется из-за закора-
                ;5824   ;    чивания связей.
                ;5825   ;
                ;5826    T.5:
U 07DC, B65E,15 ;5827           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;5828                                                   ; состояния
U 07DD, 3E80,15 ;5829           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;5830                                                   ; 15 указывает начало теста для консольного процессора
```

```
U 07DE, 10E0,15 ;5831            MISC [SET.CP.ATTN]                          ; выдача сигнала CPU ATTN для консольного процессора
                ;5832    WAIT.T5.0:
U 07DF, 887D,F4 ;5833            JMP [WAIT.T5.0]                             ; зацикливание для ожидания ответа консольного
                ;5834                                                        ; процессора
U 07E0, 0A1A,9C ;5835            JSR [SETUP]                                 ; установка масок, кода модуля и др.
U 07E1, 5F7F,95 ;5836            MCOM LS[BIT31] TO WR[3]                     ; сброс бита 31, установка других битов WR3
U 07E2, 4565,95 ;5837            BIC LS[BIT18] TO WR[3]                      ; сброс бита 18 (аппаратура модуля МСТ сбрасывает этим
                ;5838                                                        ; биты, так что и ожидаемые данные должны быть
                ;5839                                                        ; сброшены)
U 07E3, 2341,95 ;5840            ROR WR[3]                                   ; корректировка для аппаратного сдвига
                ;5841    LOOP.T5.1:
U 07E4, 2006,95 ;5842            MOV WR[3] TO WR[1]                          ; установка ожидаемых данных
U 07E5, 1F9F,F5 ;5843            MEM.REQ[READ.MAINT.UBS] ADRS[ONES] DT[LONG] ; пересылка содержимого регистра виртуального
                ;5844                                                        ; адреса на адресные линии общей шины (данные - все
                ;5845                                                        ; единицы)
U 07E6, 3022,15 ;5846            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 07E7, 0869,3C ;5847            JSR [CHECK.RESULT]                          ; проверка результата
U 07E8, 087E,44 ;5848            JMP [LOOP.T5.1]                             ; зацикливание при ошибке, если разрешено
U 07E9, FF82,15 ;5849            INC LS[ERROR.NUMBER]                        ; номер ошибки = 2
U 07EA, B635,95 ;5850            MOV LS[UBS.SET] TO WR[3]                    ; установка битов в WR3, которые устанавливаются
                ;5851                                                        ; аппаратурой модуля МСТ
U 07EB, 2341,95 ;5852            ROR WR[3]                                   ; коорректировка для аппаратного сдвига
                ;5853    LOOP.T5.2:
U 07EC, 2006,95 ;5854            MOV WR[3] TO WR[1]                          ; установка ожидаемых данных
U 07ED, 9F9D,F5 ;5855            MEM.REQ[READ.MAINT.UBS] ADRS[ZERO] DT[LONG] ; пересылка содержимого регистра виртуального
                ;5856                                                        ; адреса на адресные линии общей шины (данные - все
                ;5857                                                        ; нули)
U 07EE, 3022,15 ;5858            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 07EF, 0869,3C ;5859            JSR [CHECK.RESULT]                          ; проверка результата
U 07F0, 887E,C4 ;5860            JMP [LOOP.T5.2]                             ; зацикливание при ошибке, если разрешено
U 07F1, FF82,15 ;5861            INC LS[ERROR.NUMBER]                        ; номер ошибки = 3
U 07F2, E5F8,15 ;5862            CLR LS[OS]                                  ; очистка индекса
                ;5863    REPEAT.T5.3:
U 07F3, 36F7,95 ;5864            MOV LS[SHIFT.OS(4-0)] TO WR[3]              ; загрузка тестовых данных в WR3
U 07F4, C735,95 ;5865            BIS LS[UBS.SET] TO WR[3]                    ; установка битов, которые устанавливаются аппаратурой
                ;5866                                                        ; модуля МСТ
U 07F5, C57F,95 ;5867            BIC LS[BIT31] TO WR[3]                      ; сброс битов, которые сбрасываются аппаратурой модуля
                ;5868                                                        ; МСТ
U 07F6, 4565,95 ;5869            BIC LS[BIT18] TO WR[3]                      ; сброс битов, которые сбрасываются аппаратурой модуля
                ;5870                                                        ; МСТ
U 07F7, 2341,95 ;5871            ROR WR[3]                                   ; корректировка аппаратного  сдвига
                ;5872    LOOP.T5.3:
U 07F8, 2006,95 ;5873            MOV WR[3] TO WR[1]                          ; установка ожидаемых данных
U 07F9, 9FF7,F5 ;5874            MEM.REQ[READ.MAINT.UBS] ADRS[SHIFT.OS(4-0)] DT[LONG] ; пересылка содержимого регистра
                ;5875                                                        ; виртуального адреса на адресные шины (данные
                ;5876                                                        ; сдвигаемая единица)
U 07FA, 3022,15 ;5877            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 07FB, 0869,3C ;5878            JSR [CHECK.RESULT]                          ; проверка результата
U 07FC, 887F,B4 ;5879            JMP [LOOP.T5.3]                             ; зацикливание при ошибке, если разрешено
U 07FD, 7FF8,15 ;5880            INC LS[OS]                                  ; увеличечние индекса для следующего набора
U 07FE, 36F9,15 ;5881            MOV LS[OS] TO WR[2]                         ; подготовка для проверки, все ли наборы использованы
                ;5882              BIT LS[BIT5] WITH WR[2],                  ; проверка, что содержимое регистра OS
U 07FF, D94B,35 ;5883            DT(LONG)&SET.ALU.CC                         ; увеличено до 20(H) и установка кодов условий
U 0800, 887F,39 ;5884            JMP.IF[BITS.CLR] TO [REPEAT.T5.3]           ; повторение со следующим набором, если не все
                ;5885                                                        ; использованы
```

```
                ;5886    T5.END:
```

```
;5887    .PAGE "ТЕСТ 6 - тест данных общей шины (модули MCT и WCS)"
;5888    ;
;5889    ; ОПИСАНИЕ ТЕСТА:
;5890    ;
;5891    ;       Этот тест проверяет схемы данных общей шины, используя транзитные схемы
;5892    ;    модуля WCS. Все сигналы управления и тестовые наборы данных поступают из
;5893    ;    модуля MCT. Путь данных следующий: выдается инструкция центрального процес-
;5894    ;    сора MEM REQ с DT=LONGWORD и MF=READ.MAINT.UBS (1F(H)). Начальное ветвление
;5895    ;    выполняется так, как описано в начале этого листинга. Микроцикл памяти по
;5896    ;    адресу начального ветвления устанавливает MEM BSY и IB BSY и разрешает фи-
;5897    ;    зический адрес, DATIP, адрес UB и данные UB. Тогда ветвление зависит от би-
;5898    ;    тов DIAG CHK и ECC DIS, которые были установлены до инструкции MEM.REQ. Для
;5899    ;    этого теста бит DIAG CHK сброшен и бит ECC DIS установлен. По месту ветвле-
;5900    ;    ления буферы UB DIR на модуле WCS открыты, приемо-передатчики UB DATA на
;5901    ;    модуле WCS разрешены для приема тестового набора с шины MC BUS, на шину
;5902    ;    MC BUS пропускаются данные из регистра виртуального адреса и поддерживаются
;5903    ;    возбужденными сигналы MEM BSY и IB BSY. Следующий цикл повторяет предыдущие
;5904        команды, позволяющие распространение и установку данных. Следующий микроцикл
;5905        памяти закрывает буферы UB и оставляет разрешенными данные UB, MEM BSY и
;5906        IB BSY. Следующий микроцикл памяти открывает буферы UB обратно на шину MC BUS
;5907    ;   и поддерживает возбужденными MEM BUSY и IB BSY. Следующий микроцикл памяти
;5908    ;   зацикливается в ожидании снятия сигнала CPU GRANT и поддерживает разрешение
;5909    ;   буфера данных UB на шину MC BUS. Сигнал CPU GRANT снимается, когда централь-
;5910    ;   ный процессор выполнит инструкцию MOV MEM.DATA TO WR[0]. Заключительный мик-
;5911    ;   роцикл памяти проверяет следующий запрос центрального процессора и переходит
;5912    ;   на холостой цикл. Используемыми наборами данных являются все единицы, все
;5913    ;   нули и сдвигаемая единица. Будет проверено только 16 битов (биты полученных
;5914    ;   данных 14-00 и 31).
;5915    ;
;5916    ; ПРЕДПОЛОЖЕНИЯ:
;5917    ;
;5918    ;       Предполагается, что предыдущие тесты выполнены успешно. Необходимо заме-
;5919    ;    тить, что большинство схем, которые проверяет этот тест, находятся на модуле
;5920    ;    WCS, но все сигналы управления и тестовые данные порождаются на модуле MCT.
;5921    ;
;5922    ; ШАГИ ТЕСТА:
;5923    ;
;5924    ;       1.Установка номера ошибки и номера модуля в местной памяти (для распечатки
;5925    ;         ошибок) и сброс предыдущего номера ошибки в местной памяти.
;5926    ;       2.Установка маски ошибки для проверки 16 битов, запись в CSR1 для установ-
;5927    ;         ки бита ECC DIS и сброса бита DIAG CHK (эти биты будут использованы для
;5928    ;         ветвления к правильной тестовой ветви).
;5929    ;       3.Загрузка тестовых данных из местной памяти в WR1.
;5930    ;       4.Сдвиг WR[1] вправо на один бит.
;5931    ;       5.Выдача инструкции MEM.REQ с DT=LONGWORD и MF=READ.MAINT.UBS, используя
;5932    ;         ячейку местной памяти, содержащую тестовые данные, в качестве источника
;5933    ;         данных.
;5934    ;       6.Выполнение инструкции MOV MEM.DATA TO WR[0] для чтения результата и про-
;5935    ;         верки.
;5936    ;       7.Повторение шагов с 3 по 6 с другими наборами данных.
;5937    ;
;5938    ; ОШИБКИ:
;5939    ;
;5940    ;       ошибка 1 - схемы данных общей шины работают неправильно с данными - все единицы
;5941    ;       ошибка 2 - схемы данных общей шины работают неправильно с данными - все нули
```

```
                ;5942   ;       ошибка 3 - схемы данных общей шины работают неправильно с данными - сдвигае-
                ;5943   ;                  мая единица
                ;5944   ;
                ;5945   ; НАЛАДКА:
                ;5946   ;
                ;5947   ;       ОШИБКА 1 - Эта ошибка указывает на неправильную работу схем данных общей
                ;5948   ;    шины или управляющих сигналов. Большинство схем, проверяемых этим тестом, на-
                ;5949   ;    ходятся на модуле WCS. Если модуль MCT работает в пошаговом режиме, необходи-
                ;5950   ;    мо остановить на первом микроцикле после адреса начального ветвления (на пер-
                ;5951   ;    вом из двух циклов, которые выполняют те же действия). На модуле WCS необхо-
                ;5952   ;    димо проверить, что сигнал UB DATA EN L - низкий, а сигнал UB DIR LATCH L -
                ;5953   ;    высокий. Если какой-либо из этих сигналов неправильный, необходимо проследить
                ;5954   ;    их в обратном направлении до ПЗУ управляющей памяти на модуле MCT. Если эти
                ;5955   ;    сигналы правильные, необходимо проверить высокие уровни на всех входах D вось-
                ;5956   ;    миразрядных буферов. Если входы неправильные, необходимо проверить их в обрат-
                ;5957   ;    ном направлении до шины MC BUS. Если эти входы правильные, по-видимому, неисп-
                ;5958   ;    равен восьмиразрядный буфер или соединение шины MC BUS с контактами модуля.
                ;5959   ;    Выходы восьмиразрядного буфера можно проверить в автоматической работе модуля
                ;5960   ;    MCT (без останова) и с остановом центрального процессора на инструкции MEM.REQ.
                ;5961   ;    В этом случае сигнал UB DIR EN L должен быть низкий и сигнал UB DIR LATCH L
                ;5962   ;    - также низкий. Оба эти сигнала могут быть проверены в обратном направлении
                ;5963   ;    до ПЗУ управляющей памяти MCT.
                ;5964   ;
                ;5965   ;       ОШИБКА 2 - Можно выполнить такие же процедуры как и в случае ошибки 1. Вхо-
                ;5966   ;    ды восьмиразрядного буфера должны быть низкими. Если это первая ошибка, ско-
                ;5967   ;    рее всего, сигнал UB DIR EN L H не становится низким. Останов центрального
                ;5968   ;    процессора на инструкции MEM.REQ позволяет это проверить. Если этот сигнал
                ;5969   ;    правильный, тогда необходимо проверять также, как при ошибке 1.
                ;5970   ;       Другой причиной ошибки может быть неисправное устройство на общей шине,
                ;5971   ;    поддерживающее низкий уровень (возбужденный) на линии данных, так что на
                ;5972   ;    выходе будет наблюдаться 1.
                ;5973   ;
                ;5974   ;       ОШИБКА 3 - Подозреваются короткие замыкания между битами или неисправ-
                ;5975   ;    ность восьмиразрядного буфера на модуле MCT.
                ;5976   ;
                ;5977    T.6:
U 0801, B65E,15 ;5978           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;5979                                                   ; состояния
U 0802, 3E80,15 ;5980           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;5981                                                   ; 15 указывает начало теста для консольного процессора
U 0803, 10E0,15 ;5982           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN для консольного процессора
                ;5983    WAIT.T6.0:
U 0804, 8880,44 ;5984           JMP [WAIT.T6.0]                         ; зацикливание для ожидания ответа консольного
                ;5985                                                   ; процессора
U 0805, 0A1A,AC ;5986           JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0806, C740,15 ;5987           BIS LS[WCS] TO WR[0]                    ; добавление кода модуля WCS для распечатки
U 0807, 3E8C,15 ;5988           MOV WR[0] TO LS[MODULE.NUM]             ; запоминание кода модуля в местной памяти для указания
                ;5989                                                   ; модулей MCT и WCS
U 0808, 3636,15 ;5990           MOV LS[UBS.DATA] TO WR[0]               ; установка данных для маски ошибки
U 0809, 3E8A,15 ;5991           MOV WR[0] TO LS[ERROR.MASK]             ; запоминание маски ошибки для проверки битов 14-00 и
                ;5992                                                   ; бита 31
U 080A, 3672,15 ;5993           MOV LS[ECC.DIS] TO WR[0]                ; подготовка установки бита ECC DIS в CSR
U 080B, 8A17,FC ;5994           JSR [WRITE.CSR1]                        ; запись данных в CSR1 из WR0
                ;5995    LOOP.T6.1:
U 080C, 369E,95 ;5996           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
```

```
U 080D,  1F9F,F5 ;5997            MEM.REQ[READ.MAINT.UBS] ADRS[ONES] DT[LONG] ; выдача единиц на линии данных общей шины
U 080E,  3022,15 ;5998            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 080F,  0869,3C ;5999            JSR [CHECK.RESULT]                          ; проверка результата
U 0810,  0880,C4 ;6000            JMP [LOOP.T6.1]                             ; зацикливание при ошибке, если разрешено
U 0811,  FF82,15 ;6001            INC LS[ERROR.NUMBER]                        ; номер ошибки 2
                 ;6002    LOOP.T6.2:
U 0812,  2F82,95 ;6003            CLR WR[1]                                   ; ожидаемые данные
U 0813,  9F9D,F5 ;6004            MEM.REQ[READ.MAINT.UBS] ADRS[ZERO] DT[LONG] ; выдача нулей на линии данных общей шины
U 0814,  3022,15 ;6005            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 0815,  0869,3C ;6006            JSR [CHECK.RESULT]                          ; проверка результата
U 0816,  0881,24 ;6007            JMP [LOOP.T6.2]                             ; зацикливание при ошибке, если разрешено
U 0817,  FF82,15 ;6008            INC LS[ERROR.NUMBER]                        ; ошибка 3
U 0818,  E5F8,15 ;6009            CLR LS[OS]                                  ; очистка индекса
                 ;6010    REPEAT.T6.3:
U 0819,  36F7,95 ;6011            MOV LS[SHIFT.OS(4-0)] TO WR[3]              ; выборка тестовых данных
U 081A,  2341,95 ;6012            ROR WR[3]                                   ; корректировка для аппаратного сдвига
                 ;6013    LOOP.T6.3:
U 0818,  2006,95 ;6014            MOV WR[3] TO WR[1]                          ; ожидаемые данные
U 081C,  9FF7,F5 ;6015            MEM.REQ[READ.MAINT.UBS] ADRS[SHIFT.OS(4-0)] DT[LONG] ; выдача сдвигаемой единицы на линию
                 ;6016                                                        ; данных общей шины
U 081D,  3022,15 ;6017            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 081E,  0869,3C ;6018            JSR [CHECK.RESULT]                          ; проверка результата
U 081F,  0881,B4 ;6019            JMP [LOOP.T6.3]                             ; зацикливание при ошибке, если разрешено
U 0820,  7FF8,15 ;6020            INC LS[OS]                                  ; увеличение индекса для следующего набора
U 0821,  36F9,15 ;6021            MOV LS[OS] TO WR[2]                         ; подготовка для проверки, все ли наборы использованы
                 ;6022              BIT LS[BIT4] WITH WR[2],                  ; проверка, что содержимое регистра OS увеличено до
                 ;6023                                                        ; 10(H) и установка кодов условий
U 0822,  5949,35 ;6024            DT(LONG)&SET.ALU.CC                         ;
U 0823,  0881,99 ;6025            JMP.IF[BITS.CLR] TO [REPEAT.T6.3]           ; повторение со следующим набором, если не все
                 ;6026                                                        ; использованы
                 ;6027    END.T6:
```

```
                ;6028   .PAGE "ТЕСТ 7 - проверка схем инструкции MOV MEM.DATA TO LS (модуль DAP)"
                ;6029   ;
                ;6030   ; ОПИСАНИЕ ТЕСТА:
                ;6031   ;
                ;6032   ;     Этот тест проверяет схемы выполнения инструкции MOV MEM.DATA TO LS на моду-
                ;6033   ;   ле DAP. Следующий тест должен использовать эту функцию, но она не могла быть
                ;6034   ;   проверена, пока не была проверена базовая часть модуля MCT. Тест просто очи-
                ;6035   ;   щает ячейку местной памяти (LS8) и записывает единицы в регистр виртуального
                ;6036   ;   адреса, используя функцию памяти READ.MAINT.VAR. После этого считывается ре-
                ;6037   ;   зультат инструкцией MOV MEM.DATA TO LS[T8], LS8 пересылается в WR0 и проверя-
                ;6038   ;   ется результат. Тест повторяется с использованием нулевых данных.
                ;6039   ;
                ;6040   ; ПРЕДПОЛОЖЕНИЯ:
                ;6041   ;
                ;6042   ;   Предполагается, что все предыдущие тесты выполнены успешно. Этот тест прове-
                ;6043   ;   ряет схемы модуля DAP, которые будут использоваться в следующих тестах.
                ;6044   ;
                ;6045   ; ШАГИ ТЕСТА:
                ;6046   ;
                ;6047   ;   1) Установка маски ошибки, номера ошибки и номера модуля в местной памяти
                ;6048   ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
                ;6049   ;      мяти.
                ;6050   ;   2) Очистка LS8 и выдача инструкции MEM.REQ с MF=READ.MAINT.VAR и DT=LONGWORD.
                ;6051   ;      использование ячейки LS, содержащей все единицы в качестве данных вирту-
                ;6052   ;      ального адреса.
                ;6053   ;   3) Выполнение инструкции MOV MEM.DATA TO LS[T8], потом пересылка LS8 в WR0.
                ;6054   ;      проверка результата на все единицы.
                ;6055   ;   4) Повторение шагов 2 и 3 с нулевыми данными.
                ;6056   ;
                ;6057   ; ОШИБКИ:
                ;6058   ;
                ;6059   ;   ошибка 1 - инструкция MOV MEM.DATA TO LS работает неправильно с данными - все
                ;6060   ;              единицы.
                ;6061   ;   ошибка 2 - инструкция MOV MEM.DATA TO LS работает неправильно с данными - все
                ;6062   ;              нули.
                ;6063   ;
                ;6064   ; НАЛАДКА:
                ;6065   ;
                ;6066   ;
                ;6067   ;     ОШИБКА 1 - Эта ошибка указывает на неправильную работу ПЗУ УПР.ФУНКЦИЙ
                ;6068   ;   АЛУ или ПМЛ УПР.ИСТОЧНИКОМ-ПРИЕМНИКОМ модуля DAP. Также может подозреваться
                ;6069   ;   ПМЛ УПР.МЕСТНОЙ ПАМЯТИ. Необходимо остановить центральный процессор на инст-
                ;6070   ;   рукции MOV MEM.DATA TO LS и проверить наличие 3(В) на выходах ALU CTL 2 H по
                ;6071   ;   ALU CTL 0 H и высокого уровня на выходе DEST CTL 0 H ПЗУ УПР.ФУНКЦИЙ АЛУ.
                ;6072   ;   Также необходимо проверить низкий уровень на выходе DEST CTL 1 H и высокие
                ;6073   ;   уровни на выходах SRC CTL 2 H по SRC CTL 0 H ПМЛ УПР.ИСТОЧНИКОМ-ПРИЕМНИКОМ.
                ;6074   ;   Если любой из этих сигналов неправильный, подозревается микросхема, формиру-
                ;6075   ;   ющая этот сигнал. Если выходы управления АЛУ правильные, необходимо проверить
                ;6076   ;   низкий уровень на выходных ножках 12, 13 и 14 ПМЛ УПР.МЕСТНОЙ ПАМЯТИ. Если
                ;6077   ;   любой из этих выходов неправильный, подозревается ПМЛ УПР.МЕСТНОЙ ПАМЯТИ.
                ;6078   ;
                ;6079   ;     ОШИБКА 2 - то же, что и для ошибки 1.
                ;6080   ;
                ;6081   T.7:
U 0824, B65E,15 ;6082           MOV LS[BEGIN.TEST] TO WR[0]          ; установка бита 15 в WR[0] для слова управления и
```

```
                    ;6083                                                   ; состояния
U 0825,  3E80,15    ;6084           MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 15 в слове управления и состояния. Бит 15
                    ;6085                                                   ; указывает начало теста для конс.процессора
U 0826,  10E0,15    ;6086           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN для конс.процессора
                    ;6087       WAIT.T7.0:
U 0827,  0882,74    ;6088           JMP [WAIT.T7.0]                         ; зацикливание для ожидания ответа конс.процессора
U 0828,  0A1A,AC    ;6089           JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0829,  3642,15    ;6090           MOV LS[CPU] TO WR[0]                     ; установка кода модуля, бита 1 в WR[0]
U 082A,  3E8C,15    ;6091           MOV WR[0] TO LS[MODULE.NUM]              ; запоминание кода модуля в местной памяти для указания
                    ;6092                                                   ; модуля DAP
                    ;6093       LOOP.T7.1:
U 082B,  E510,15    ;6094           CLR LS[T8]                              ; дополнительный код ожидаемых данных
U 082C,  369E,95    ;6095           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 082D,  9D9E,75    ;6096           MEM.REQ[READ.MAINT.ADRS] ADRS[ONES] DT[LONG] ; запись единиц в регистр виртуального адреса
U 082E,  3810,15    ;6097           MOV MEM.DATA TO LS[T8]                   ; запоминание результата в ячейке 8 местной памяти
U 082F,  B610,15    ;6098           MOV LS[T8] TO WR[0]                      ; пересылка результата в WR0
U 0830,  0869,3C    ;6099           JSR [CHECK.RESULT]                      ; проверка результата
U 0831,  0882,B4    ;6100           JMP [LOOP.T7.1]                         ; зацикливание при ошибке, если разрешено
U 0832,  FF82,15    ;6101           INC LS[ERROR.NUMBER]                    ; ошибка 2
                    ;6102       LOOP.T7.2:
U 0833,  B69C,95    ;6103           MOV LS[ZERO] TO WR[1]                   ; ожидаемые данные
U 0834,  1D9C,75    ;6104           MEM.REQ[READ.MAINT.ADRS] ADRS[ZERO] DT[LONG] ; запись нулей в регистр виртуального адреса
U 0835,  3810,15    ;6105           MOV MEM.DATA TO LS[T8]                   ; запоминание результата в ячейке 8 местной памяти
U 0836,  B610,15    ;6106           MOV LS[T8] TO WR[0]                      ; пересылка результата в WR0
U 0837,  0869,3C    ;6107           JSR [CHECK.RESULT]                      ; проверка результата
U 0838,  0883,34    ;6108           JMP [LOOP.T7.2]                         ; зацикливание при ошибке, если разрешено
                    ;6109       END.T7:
```

```
;6110    .PAGE "ТЕСТ 8 - тест ветвления по UBC1, ICC0, UB ACTIVITY (модуль MCT)"
;6111    ;
;6112    ;
;6113    ; ОПИСАНИЕ ТЕСТА:
;6114    ;
;6115    ;       Этот тест проверяет схемы ветвления по сигналам L UBC1 H, IC0 H и D UB
;6116    ;    ACTIVITY L. Сами сигналы также будут проверяться. Эти сигналы управляются мик-
;6117    ;    ропрограммой памяти и для их проверки выполняется ветвление. Если ветвление
;6118    ;    выполняется правильно, тогда регистр виртуального адреса будет увеличен на 4.
;6119    ;    Чтение регистра виртуального адреса в конце теста проверяет, что все переходы
;6120    ;    выполнены, или указывает ту часть, в которой появится первая неисправность.
;6121    ;    Любая неисправность вызывает немедленное чтение регистра виртуального адреса.
;6122    ;    В начале теста в CSR1 были записаны биты DIAG CHK и ECC DIS. Эти биты исполь-
;6123    ;    зуются для ветвления к правильному тесту. Путь данных следующий: центральный
;6124    ;    процессор выдает микроинструкцию MEM.REQ с полем MF=READ.MAINT.UBS (1F). На-
;6125    ;    чальное ветвление в памяти выполняется, как описано в начале листинга. Микро-
;6126    ;    цикл по адресу начального ветвления разрешает данные UB, адрес UB, и устанав-
;6127    ;    ливает физический адрес, DATIP, IB и MEMORY BUSY. DATIP вызывает установку
;6128    ;    IC0 H и сброс IC1 H в начале следующего микроцикла. Этот цикл выполняет пе-
;6129    ;    реход по битам DIAG CHK и ECC DIS (установленным в CSR1 до инструкции MEM.REQ)
;6130    ;    к следующему циклу в этой последовательности. Этот следующий микроцикл уста-
;6131    ;    навливает DATO, поддерживает установленными IB BSY и MEMORY BUSY, поддерживает
;6132    ;    разрешение адреса UB и открывает буфер управления функцией. При установке DATO
;6133    ;    в начале следующего цикла сбрасывается IC0 H и устанавливается IC1 H. Оба эти
;6134    ;    сигнала поступают на приемопередатчики общей шины и формируют сигналы UB C0 H
;6135    ;    и UB C1, которые поступают на восьмиразрядный буфер, поддерживаемый открытым
;6136    ;    высоким уровнем сигнала CONT FUNC LAT L. На выходе формируются сигналы L UBC1
;6137         H и L UBC0 H. В этом цикле ветвлением по сигналу IC0 H (который еще высокий из
;6138         предыдущего цикла) формируется следующий микроцикл. Этот следующий микроцикл
;6139    ;    выполняет те же функции, как и последний цикл, за исключением того, что DATO
;6140    ;    не устанавливается и увеличивается регистр виртуального адреса. Это ветвление
;6141    ;    проверяет наличие низкого уровня сигнала IC0 H, который формируется в резуль-
;6142    ;    тате предыдущего DATO. Следующий цикл этого ветвления выполняет те же функции,
;6143    ;    за исключением того, что устанавливается DATI и ветвление выполняется по сигна-
;6144    ;    лу L UBC1. Сигнал L UBC1 H еще установлен от предыдущего установленного DATO.
;6145    ;    Установка DATI в этом цикле вызовет переход сигнала L UBC1 H в низкий уровень
;6146    ;    в начале следующего цикла. Следующий цикл является циклом задержки для уста-
;6147    ;    новки сигнала L UBC1 H до выполнения ветвления по этому сигналу. Этот следую-
;6148    ;    щий цикл поддерживает открытым CONT FUNC LAT, поддерживает MEMORY BUSY, IB
;6149    ;    BSY и разрешенный UB ADRS EN, увеличивает регистр виртуального адреса и вы-
;6150    ;    полняет ветвление по сигналу L UBC1 H. Сигнал L UBC1 H должен быть низким,
;6151    ;    так как сигнал DATI был на два цикла раньше. Следующий цикл только поддержи-
;6152    ;    вает установленным MEMORY BUSY и увеличивает регистр виртуального адреса.
;6153    ;    Этот цикл выполняет ветвление по сигналу D UB ACTIVITY. Сигнал D UB ACTIVITY
;6154    ;    L становится активным на втором цикле после снятия IB BSY. Следующие два цик-
;6155    ;    ла являются циклами задержки, которые только поддерживают возбужденным MEMORY
;6156    ;    BUSY. Эта задержка необходима для снятия сигнала L BBSY после распространения
;6157    ;    через несколько уровней схемы. Следующий цикл поддерживает возбужденным
;6158    ;    MEMORY BUSY, увеличивает регистр виртуального адреса и выполняет ветвление по
;6159    ;    сигналу D UB ACTIVITY L, который должен быть снят (высокий). Следующий цикл
;6160    ;    увеличивает регистр виртуального адреса, поддерживает возбужденным MEM BSY и
;6161    ;    переходит в микропрограмму READ.MAINT.ADR для пересылки текущего значения ре-
;6162    ;    гистра виртуального адреса в центральный процессор для проверки. Если в какой-
;6163    ;    либо момент встречается неправильное ветвление, будет выполняться микропрограм-
;6164    ;    ма READ.MAINT.ADR и значение регистра виртуального адреса этого момента будет
```

```
;6165    ;   указывать участок неисправности.
;6166    ;     Этот тест также проверяет вход L DT0 в ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. для уверен-
;6167    ;   ности, что он оказывает воздействие на выход IC0 H.
;6168    ;
;6169    ; ПРЕДПОЛОЖЕНИЯ:
;6170    ;
;6171    ;   Предполагается, что предыдущие тесты выполнены успешно.
;6172    ;
;6173    ; ШАГИ ТЕСТА:
;6174    ;
;6175    ;     1.Установка номера ошибки и номера модуля в местной памяти (для распечатки
;6176    ;       ошибок) и сброс предыдущего номера ошибки в местной памяти.
;6177    ;     2.Установка маски для проверки младшего байта и запись в CSR1 данных для ус-
;6178    ;       тановки битов DIAG CHK и ECC DIS для ветвления к правильному тесту.
;6179    ;     3.Загрузка C(H) в WR1 в качестве ожидаемых данных.
;6180    ;     4.Выдача инструкции MEM.REQ с MF=READ.MAINT.UBS (1F(H)). Использование ячей-
;6181    ;       ки местной памяти, содержащей все нули. Этим инициализируется в качестве
;6182    ;       адреса регистр виртуального адреса на 0 перед его инкрементированием. Поле
;6183    ;       DT инструкции MEM.REQ=LONGWORD (длинное слово).
;6184    ;     5.Выполнение инструкций MOV MEM.DATA TO LS[TB], MOV LS[TB] TO WR[0] и про-
;6185    ;       верка результата на C(H).
;6186    ;     6.Повторение шагов 4 и 5 с типом данных BYTE (байт) в инструкции MEM.REQ и
;6187    ;       ожидаемыми данными, равными 2.
;6188    ;
;6189    ; ОШИБКИ:
;6190    ;
;6191    ;     ошибка 1 - неправильное ветвление по сигналам IC0, L UBC1 или D UB ACTIVITY.
;6192    ;                Полученные данные указывают место появления ошибки (см.наладка).
;6193    ;     ошибка 2 - неправильно возбуждается IC0 H, когда тип данных=BYTE.
;6194    ;
;6195    ; НАЛАДКА:
;6196    ;
;6197    ;     ОШИБКА 1 - Полученные данные в распечатке ошибки указывают неправильное
;6198    ;   ветвление. Ими является значение регистра виртуального адреса в момент
;6199    ;   неисправности. Если регистр виртуального адреса=: 0 - неправильно выпол-
;6200    ;   нено ветвление по возбужденному сигналу IC0 H; 2(H) - неправильно выпол-
;6201    ;   нено ветвление по невозбужденному сигналу IC0 H; 4(H)- неправильно выпол-
;6202    ;   нено ветвление по возбужденному сигналу L UBC1 H; 6(H)- неправильно выпол-
;6203    ;   нено ветвление по невозбужденному сигналу L UBC1 H; 8(H)- неправильно выпол-
;6204    ;   нено ветвление по возбужденному сигналу D UB ACTIVITY L; A(H)- неправильно
;6205    ;   выполнилось ветвление по невозбужденному сигналу D UB ACTIVITY L.
;6206    ;     В следующих процедурах, если пошаговый режим MCT невозможен, сигналы дол-
;6207    ;   жны быть проверены при зацикливании по ошибке.
;6208    ;     Если регистр виртуального адреса=0, тогда необходимо проверить, что ус-
;6209    ;   тановка DATIP возбуждает IC0 H. Если MCT может работат в пошаговом режи-
;6210    ;   ме, остановите центральный процессор на инструкции MEM.REQ и остановите MCT
;6211    ;   на первом цикле по адресу начального ветвления (который устанавливает DATIP).
;6212    ;   входы ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. SPF0 H и SPF2 H должны быть высокими. Выход не
;6213    ;   должен измениться до следующего цикла. Остановите MCT на следующем цикле
;6214    ;   (BEN2 = LUB.C0) и проверьте высокий уровень сигнала IC0 H на выходе ПМЛ РАЗ-
;6215    ;   НЫЕ ФУНКЦИИ УПР. Если он неправильный, ПМЛ неисправна. Если на выходе ПМЛ
;6216    ;   правильный сигнал, необходимо проверить высокий уровень сигнала IC0 H на
;6217    ;   входе мультиплексора BEN2. Также необходимо проверить наличие 100(B) на
;6218    ;   входах выб[6:0]и A2,A1 и A0 соответственно. Если здесь правильно, по-видимому,
;6219    ;   неисправен мультиплексор.
```

;6220 ; Если регистр виртуального адреса=2, тогда необходимо проверить, что ус-
;6221 ; тановка DATO снимает IC0 H. Если MCT может работать в пошаговом режиме, ос-
;6222 ; тановите центральный процессор на инструкции MEM.REQ и остановите MCT на пер-
;6223 ; вом адресе после адреса начального ветвления (установка DATO). Необходимо
;6224 ; проверить наличие высокого уровня на входах SPF2 H, SPF1 H, низкого уровня на
;6225 ; входе SPF0 и высокого уровня на L DT0 H ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. СИГНАЛ L DT0 H
;6226 ; поступает из ПМЛ УПР.КОДАМИ УСЛОВИЙ модуля DAP. Этот сигнал был проверен внут-
;6227 ; ри модуля DAP предыдущими тестами модуля DAP, но он может не поступать на мо-
;6228 ; дуль MCT. Если эти входы правильные, необходимо остановить MCT на следующем
;6229 ; цикле (другое ветвление с BEN2=L UB.C0) и проверить низкий уровень на выходе
;6230 ; IC0 H. Если правильно, необходимо проверить низкий уровень сигнала IC0 H на
;6231 ; входе мультиплексора BEN2 и наличие 100(B) на линиях выборки A2,A1 и A0 соот-
;6232 ; ветственно. Если все эти входы правильные, по-видимому, мультиплексор неис-
;6233 ; правен.
;6234 ; Если регистр виртуального адреса=4, тогда необходимо проверить, что уста-
;6235 ; новка DATO возбуждает L UBC1 H. Если MCT может работать в пошаговом режиме,
;6236 ; остановите центральный процессор на инструкции MEM.REQ и остановите MCT на
;6237 ; первом адресе после адреса начального ветвления (установка DATO). Необходимо
;6238 ; проверить наличие высокого уровня на входах SPF2 H, SPF1 H и низкий уровень
;6239 ; на входе SPF0 H ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Если эти сигналы правильные, необхо-
;6240 ; димо перейти в пошаговом режиме на следующей цикл MCT (другое ветвление по
;6241 ; BEN2=L UB.C0) и проверить высокий уровень на выходе IC1 H. Если здесь пра-
;6242 ; вильно, необходимо проверить этот сигнал на входе приемо-передатчика общей
;6243 ; шины, который выдает UBC1 H. Также необходимо проверить высокий уровень сиг-
;6244 ; нала UBC1 H и низкий уровень сигнала UB ADR EN L. Сигнал UBC1 H поступает на
;6245 ; буфер управления функцией. Вход должен быть высоким и выход L UBC1 H должен
;6246 ; быть высоким. Также необходимо проверить высокий уровень сигнала
;6247 ; CONT FUNC LAT L. Если все эти сигналы правильные, необходимо проверить низ-
;6248 ; кий уровень сигнала SPF2 H на входе ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. (это предотв-
;6249 ; ращает изменение IC1 H в следующем цикле) и, остановив контроллер ОЗУ
;6250 ; на следующем цикле (BEN1=L UBC1), проверить, что сигнал L UBC1 H на входе
;6251 ; мультиплексора BEN1 высокий. Если нет, необходимо проследить этот сигнал в
;6252 ; обратном направлении. Если он изменился на выходе ПМЛ РАЗНЫЕ ФУНКЦИИ УПР., то
;6253 ; ПМЛ неисправна. Если этот сигнал высокий, необходимо проверить наличие 100(B)
;6254 ; на входах выборки A2, A1 и A0 на входе мультиплексора BEN1. Если правильно,
;6255 ; подозревается сам мультиплексор BEN1.
;6256 ; Если регистр виртуального адреса=6(H), тогда необходимо проверить, что
;6257 ; установка DATI снимает L UBC1 H. Если MCT может работать в пошаговом режи-
;6258 ; ме, остановите центральный процессор на инструкции MEM.REQ и остановите MCT
;6259 ; на третьем адресе после адреса начального ветвления (установка DATI). Необ-
;6260 ; ходимо проверить наличие высокого уровня на входе SPF2 H и низкого - на вхо-
;6261 ; де SPF1 H ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Если эти сигналы правильные, необходимо
;6262 ; остановить MCT на следующем цикле (цикл задержки без ветвления) и проверить
;6263 ; низкий уровень на выходе IC1 H. Если этот выход правильный, необходимо прос-
;6264 ; ледчть низкий уровень сигналов C1 по всей цепи до выхода сигнала L UBC1 H
;6265 ; из буфера управления функцией, как описано выше при ошибке, когда регистр
;6266 ; виртуального адреса=4. Если сигнал L UBC1 низкий, необходимо проверить низ-
;6267 ; кий уровень сигнала SPF2 H (это предотвращает от изменения IC1 H при следу-
;6268 ; ющем синхросигнале) на входе ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. и тогда перейти в пош-
;6269 ; говом режиме на следующий цикл (BEN1=L UBC1). Необходимо проверить низкий
;6270 ; уровень на входе L UBC1 H и наличие 100(B) на входах выборки A2, A1 и A0
;6271 ; соответственно мультиплексора BEN1. Если L UBC1 H высокий, необходимо прос-
;6272 ; ледить его в обратном направлении до ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Если IC1 H вы-
;6273 ; сокий, ПМЛ неисправна. Если входы мультиплексора BEN1 правильные, подозрева-
;6274 ; ется сам мультиплексор.

```
                    ;6275   ;     Если регистр виртуального адреса =8(H), тогда необходимо проверить, что про-
                    ;6276   ;   падание сигнала IB BSY возбуждает UB ACTIVITY L. Если МСТ может работать в по-
                    ;6277   ;   шаговом режиме, остановите центральный процессор на инструкции MEM.REQ и оста-
                    ;6278   ;   новите микропрограмму памяти на первом микроцикле, который выполняет ветвление
                    ;6279   ;   по сигналу D UB ACTIVITY L (BEN2=UNIBUS.ACTIVITY L). Необходимо проверить низ-
                    ;6280   ;   кий уровень сигнала D UB ACTIVITY L на входе мультиплексора BEN2. На входах
                    ;6281   ;   выборки A2,A1 и A0 должно быть 110 (B) соответственно. Если эти входы правиль-
                    ;6282   ;   ные, повидимому, мультиплексор неисправен. Если сигнал D UB ACTIVITY L высокий,
                    ;6283   ;   тогда необходимо проверить этот сигнал на выходе буфера, входом которого явля-
                    ;6284   ;   ется UB ACTIVITY L. Если сигнал UB ACTIVITY L высокий, необходимо проверить ПМЛ
                    ;6285   ;   АРБИТР. Необходимо проверить низкий уровень на входах I BBSY H и ARB C0 H и
                    ;6286   ;   высокий - на входе L BBSY H (ножка 4). Если эти входы правильные, но сигнал UB
                    ;6287   ;   ACTIVITY L высокий, неисправна ПМЛ АПБИТР.
                    ;6288   ;     Если L BBSY H низкий, возможно, что сигнал I BBSY H никогда не становится
                    ;6289   ;   высоким на ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. СИГНАЛ I BBSY A H должен был стать высоким
                    ;6290   ;   уже на два цикла раньше при выполнении этой тестовой ветви. Для проверки это-
                    ;6291   ;   го сигнала может быть удобно возвратиться на 2 цикла (новым запуском). Если
                    ;6292   ;   сигнал I BBSY A H становится высоким, а сигнал I BBSY H - нет, ПМЛ РАЗНЫЕ
                    ;6293   ;   ФУНКЦИИ УПР. неисправна. Если I BBSY H высокий, нужно проверить низкий уро-
                    ;6294   ;   вень IC0 H I BBSY A H на входах ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Если эти входы
                    ;6295   ;   правильные, но I BBSY H высокий, неисправна ПМЛ РАЗНЫЕ ФУНКЦИИ УПР.
                    ;6296   ;     Если регистр виртуального адреса = A(H), тогда необходимо проверить, что
                    ;6297   ;   сигнал D UB ACTIVITY L становится высоким через 5 циклов после снятия сиг-
                    ;6298   ;   нала I BBSY H. Если МСТ может работать в пошаговом режиме, остановите цент-
                    ;6299   ;   ральный процессор на инструкции MEM.REQ, а микропрограмму памяти останови-
                    ;6300   ;   те на втором цикле, который выполняет ветвление по сигналу UB ACTIVITY
                    ;6301   ;   (BEN2=UNIBUS.ACTIVITY.L). Это десятый цикл после начального ветвления, если
                    ;6302   ;   все предыдущие ветвления выполнялись правильно. Необходимо проверить высокий
                    ;6303   ;   уровень сигнала D UB ACTIVITY L на входе мультиплексора BEN2 и наличие
                    ;6304   ;   110(B) на линиях выборки A2, A1 и A0 соответственно. Если эти входы правиль-
                    ;6305   ;   ные, мультиплексор неисправен. Если D UB ACTIVITY L низкий, необходимо про-
                    ;6306   ;   верить его на буфере, входом которого является UB ACTIVITY L. Этот вход так-
                    ;6307   ;   же должен быть высоким и поступает из ПМЛ АРБИТР. Необходимо проверить низ-
                    ;6308   ;   кие уровни на входах L NPR H, L SACK H и L BBSY H и высокий уровень на входе
                    ;6309   ;   NPG L этой ПМЛ. Высокий уровень сигналов L SACK H и L BBSY H может поддержи-
                    ;6310   ;   ваться неисправным устройством на общей шине или неисправностью самой общей
                    ;6311   ;   шины. Кроме того, необходимо проверить буферы и сигналы T0 CLOCK, которые
                    ;6312   ;   поддерживают эти входы. Если все входы правильные, подозревается ПМЛ АРБИТР.
                    ;6313   ;     ОШИБКА 2 - Эта ошибка указывает, что сигнал IC0 H не стал высоким, когда вы-
                    ;6314   ;   полнялась инструкция MEM.REQ с типом данных = байт. Причиной ошибки скорее
                    ;6315   ;   всего является ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. или вход L DT0 H. Вход L DT0 H должен
                    ;6316   ;   быть низким после выполнения инструкции MEM.REQ с типом данных = байт. Если
                    ;6317   ;   этот вход правильный, тогда, повидимому, ПМЛ неисправна.
                    ;6318    T.8:
U 0839, B65E,15     ;6319           MOV LS[BEGIN.TEST] TO WR[0]          ; установка бита 15 в WR[0] для слова управления и
                    ;6320                                                ; состояния
U 083A, 3E80,15     ;6321           MOV WR[0] TO LS[CONTROL.STATUS]       ; установка бита 15 в слове управления и состояния. Бит
                    ;6322                                                ; 15 указывает начало теста для консольного процессора
U 083B, 10E0,15     ;6323           MISC [SET.CP.ATTN]                   ; выдача сигнала CPU ATTN консольному процессору
                    ;6324    WAIT.T8.0:
U 083C, 0883,C4     ;6325           JMP [WAIT.T8.0]                      ; зацикливание для ожидания ответа консольного
                    ;6326                                                ; процессора
U 083D, 0A1A,AC     ;6327           JSR [SETUP.1]                        ; установка масок, кода модуля и др.
U 083E, B62C,15     ;6328           MOV LS[#FFFFFF00] TO WR[0]            ; установка маски ошибки
U 083F, 3EBA,15     ;6329           MOV WR[0] TO LS[ERROR.MASK]           ; сохранение маски ошибки для проверки младшего байта
```

```
U 0840,  3672,15 ;6330            MOV LS[ECC.DIS] TO WR[0]                ; установка бита запрета коррекции для CSR
U 0841,  4774,15 ;6331            BIS LS[DIAG.CHK] TO WR[0]               ; установка бита DIAG CHK
U 0842,  8A17,FC ;6332            JSR [WRITE.CSR1]                        ; установка битов ECC DIS и DIAG CHK в CSR
U 0843,  B647,95 ;6333            MOV LS[#8] TO WR[3]                     ; подготовка C(H) для ожидаемых данных
U 0844,  4745,95 ;6334            BIS LS[#4] TO WR[3]                     ; WR3=C(H)
                 ;6335    LOOP.T8.1:
U 0845,  2006,95 ;6336            MOV WR[3] TO WR[1]                      ; ожидаемые данные
U 0846,  9F9D,F5 ;6337            MEM.REQ[READ.MAINT.UBS] ADRS[ZERO] DT[LONG] ; начало проверки с регистром виртуального
                 ;6338                                                    ; адреса=0
U 0847,  3810,15 ;6339            MOV MEM.DATA TO LS[T8]                  ; чтение значения регистра виртуального адреса в ячейку
                 ;6340                                                    ; местной памяти
U 0848,  B610,15 ;6341            MOV LS[T8] TO WR[0]                     ; пересылка полученных данных в WR0
U 0849,  0869,3C ;6342            JSR [CHECK.RESULT]                      ; проверка результата
U 084A,  8884,54 ;6343            JMP [LOOP.T8.1]                         ; зацикливание при ошибке, если разрешено
U 084B,  FF82,15 ;6344            INC LS[ERROR.NUMBER]                    ; ошибка 2
U 084C,  65FC,15 ;6345            CLR LS[SIZE]                            ; очистка регистра для указания типа данных=байт
                 ;6346    LOOP.T8.2:
U 084D,  B642,95 ;6347            MOV LS[#2] TO WR[1]                     ; ожидаемые данные
U 084E,  9F9D,95 ;6348            MEM.REQ[READ.MAINT.UBS] ADRS[ZERO] DT[BYTE] ; начало проверки с регистром виртуального адреса
                 ;6349                                                    ; =0 и типом данных=байт
                 ;6350              MOV MEM.DATA TO LS[T8],               ; чтение значения регистра виртуального адреса в ячейку
                 ;6351                                                    ; местной памяти
U 084F,  B810,55 ;6352            DT(SIZE)&SET.ALU.CC                     ; сохранение разрешения типа данных=байт
U 0850,  B610,15 ;6353            MOV LS[T8] TO WR[0]                     ; пересылка полученного значения в WR0
U 0851,  0869,3C ;6354            JSR [CHECK.RESULT]                      ; проверка результата
U 0852,  0884,D4 ;6355            JMP [LOOP.T8.2]                         ; зацикливание при ошибке, если разрешено
                 ;6356    END.T8:
```

```
;6357    .PAGE "ТЕСТ 9 - тест ветвления по MSYN и SSYN (модуль MCT)"
;6358    ;
;6359    ; ОПИСАНИЕ ТЕСТА:
;6360    ;
;6361    ;       Этот тест проверяет схемы ветвления по сигналам L MSYN H и L SSYN H. Также
;6362    ;   проверяются и сами сигналы. Поскольку во время этого теста на общей шине будет
;6363    ;   возбужден MSYN, вместе с MSYN будет передаваться адрес FFFFFE(H) для предот-
;6364    ;   вращения ответа с какого-либо устройства на общей шине. Сама общая шина не
;6365    ;   будет проверяться. Ветвление по невозбужденному MSYN в этом тесте не будет про-
;6366    ;   веряться, так как, если оно не работает, начальное ветвление с холостого цикла
;6367    ;   памяти будет неправильным уже в самой первой проверке.
;6368    ;       Путь данных следующий: выполняется запись в CSR1 для установки бита
;6369    ;   DIAG CHK и сброса бита ECC DIS. Эти биты используются для перехода в соотве-
;6370    ;   тствующую тестовую ветвь. Выдается инструкция MEM.REQ С полем MF=READ.MAINT.UBS.
;6371    ;   начальное ветвление выполняется как описано в начале этого листинга. По адре-
;6372    ;   су начального ветвления физический адрес, адрес UB и данные UB разрешены,
;6373    ;   IB BSY и BUS BUSY возбуждены и DATIP установлен. По ECC DIS и DIAG CHK выпол-
;6374    ;   няется переход в соответствующую тестовую ветвь. Следующий микроцикл выдает
;6375    ;   MSYN и SSYN, поддерживает разрешенным адрес UB и поддерживает возбужденными
;6376    ;   IB BSY и MEMORY BUSY. Следующий цикл выполняет те же функции, как и предыдущий.
;6377    ;   Следующий цикл поддерживает разрешенным адрес UB, возбужденными IB BSY и
;6378    ;   MEMORY BUSY и выполняет ветвление по L MSYN H для проверки, возбужден ли MSYN.
;6379    ;   Сигнал SSYN к этому времени уже распространяется через буферы. Сигналы MSYN и
;6380    ;   SSYN оба сбрасываются. Сброшенный MSYN сбрасывает SSYN двумя циклами позже. сле-
;6381    ;   дующий цикл поддерживает разрешенным адрес UB и возбужденными IB BSY и MEMORY
;6382    ;   BUSY. Кроме того, в этом цикле выполняется ветвление по L SSYN H для проверки,
;6383    ;   возбужден ли SSYN. Следующий цикл увеличивает содержимое регистра виртуального
;6384    ;   адреса и поддерживает разрешенным адрес UB, IB BSY и MEMORY BUSY возбужденными.
;6385    ;   Следующий цикл увеличивает содержимое регистра виртуального адреса, поддержива-
;6386    ;   ет возбужденным MEMORY BUSY и выполняет ветвление по L SSYN H для проверки, снят
;6387    ;   ли SSYN (SSYN должен сниматься в начале этого цикла, так как MSYN был снят тре-
;6388    ;   емя циклами раньше). Следующий цикл увеличивает содержимое регистра виртуального
;6389    ;   адреса и переходит на микропрограмму READ.MAINT.ADR для обратного чтения регист-
;6390    ;   РА виртуального адреса в центральный процессор. Если все ветвления выполнены,
;6391    ;   регистр виртуального адреса должен содержать B(H) (читаться как 5 после сдвига),
;6392    ;   когда тест завершен. Если ветвление выполняется неправильно, микропрограмма па-
;6393    ;   мяти будут переходить в микропрограмму READ.MAINT.ADR для обратного чтения ре-
;6394    ;   гистра виртуального адреса в центральный процессор.
;6395    ;
;6396    ; ПРЕДПОЛОЖЕНИЯ:
;6397    ;
;6398    ;   Предполагается, что все предыдущие тесты выполнены успешно. Также предпола-
;6399    ;   гается, что ветвление по L MSYN H выполняется правильно, когда MSYN низкий,
;6400    ;   так как это ветвление используется в начальном ветвлении из холостого цикла.
;6401    ;
;6402    ; ШАГИ ТЕСТА:
;6403    ;
;6404    ;   1. Установка номера ошибки и номера модуля в местной памяти (для распечатки
;6405    ;      ошибок) и очистка предыдущего номера ошибки в местной памяти.
;6406    ;   2. Установка маски ошибки для проверки только младшего байта и запись CSR1
;6407    ;      для установки бита DIAG CHK и сброса бита ECC DIS.
;6408    ;   3. Загрузка 5(H) в WR[1] в качестве ожидаемых данных.
;6409    ;   4. Выдача инструкции MEM.REQ с полем MF = READ.MAINT.UBS, используя в качес-
;6410    ;      ве адреса ячейку местной памяти, содержащую FFFFFE (таким образом никакое
;6411    ;      устройство общей шины не может ответить).
```

```
;6412   ;   5. Выполнение инструкции MOV MEM.DATA TO WR[13 и проверка результата на 5(H).
;6413   ;
;6414   ; ОШИБКИ:
;6415   ;
;6416   ;   ошибка 1 - неправильно выполняется ветвление по L MSYN или L SSYN. Получен-
;6417   ;              ные данные показывают, где появилась ошибка (см. наладку).
;6418   ;
;6419   ; НАЛАДКА:
;6420   ;
;6421   ;      ОШИБКА 1 - Полученные данные в сообщении об ошибке указывают неправильное
;6422   ;   ветвление. Этими данными является младший байт регистра виртуального адреса
;6423   ;   к моменту неисправности (после сдвига вправо). Ниже приводится сводка значе-
;6424   ;   ний регистра виртуального адреса и описание соответствующей неисправности:
;6425   ;   регистр виртуального адреса = FFFFFE - неправильно выполняется ветвление при
;6426   ;   возбужденном L MSYN; регистр виртуального адреса = 1(H) - неправильно выпол-
;6427   ;   няется ветвление при возбужденном L SSYN; регистр виртуального адреса
;6428   ;   = 3(H) - неправильно выполняется ветвление при невозбужденном L SSYN.
;6429   ;      В следующих процедурах наладки, если МСТ не может работать в пошаговом ре-
;6430   ;   жиме, сигналы должны быть проверены при зацикливании по ошибке.
;6431   ;      Если регистр виртуального адреса = FF, тогда необходимо проверить, что
;6432   ;   I MSYN H возбуждает L MSYN H. Если МСТ может работать в пошаговом режиме,
;6433   ;   тогда следует остановить центральный процессор на инструкции MEM.REQ, а мик-
;6434   ;   ропрограмму контроллера ОЗУ остановить на третьем микроцикле после адре-
;6435   ;   РА начального ветвления (ветвление по L MSYN, BEN0 = L MSYN). Необходимо
;6436   ;   проверить высокий уровень сигнала L MSYN H на входе мультиплексора BEN0 и
;6437   ;   наличие 111(B) на входах выборки A2,A1 и A0. Если эти входы правильные, по-
;6438   ;   видимому, неисправен мультиплексор BEN0. Если сигнал L MSYN H низкий, необхо-
;6439   ;   димо проследить правильность прохождения данного сигнала через буферы. Синх-
;6440   ;   росигнал буферов уже был проверен в предыдущем тесте. Также необходимо прос-
;6441   ;   ледить назад сигнал I MSYN, который поступает из ПЗУ управляющей памяти.
;6442   ;      Если регистр виртуального адреса = 1(H), тогда необходимо проверить, что
;6443   ;   сигнал I SSYN H возбуждает L SSYN H. Если МСТ может работать в пошаговом ре-
;6444   ;   жиме, следует остановить центральный процессор на инструкции MEM.REQ, а мик-
;6445   ;   ропрограмму памяти остановить на третьем микроцикле после адреса начального
;6446   ;   ветвления (ветвление по L MSYN, BEN0 = L MSYN). Сигнал L SSYN H уже должен
;6447   ;   быть высоким на входе мультиплексора BEN1, но микропрограмма не будет выпол-
;6448   ;   нять ветвления по этому сигналу до следующего микроцикла. Если L SSYN H низкий
;6449   ;   на входе мультиплексора BEN1, необходимо проследить назад правильность про-
;6450   ;   хождения данного сигнала через буферы до выхода L ISSYN H из ПМЛ СИГН.КОНТР.ПИ-
;6451   ;   ТАНИЯ И ИНИЦИАЦИИ. Если этот выход низкий, необходимо проверить высокий уровень
;6452   ;   на входе в MSYN H и низкий - на входе ENABLE ERROR H. Если эти входы правильные,
;6453   ;   тогда, по-видимому, ПМЛ неисправна. Для проверки необходимо повторить запуск
;6454   ;   и остановить МСТ на втором микроцикле после адреса начального ветвления (в цик-
;6455   ;   ле, непосредственно предшествующем BEN0 = L MSYN) и проверить высокий уровень
;6456   ;   на входах B MSYN H и I SSYN H и низкий уровень на входе ENABLE ERROR H ПМЛ СИГН.
;6457   ;   КОНР.ПИТАНИЯ И ИНИЦИАЦИИ. На выходе L ISSYN H этой ПМЛ должен быть высокий
;6458   ;   уровень. Если входы правильные, неисправна ПМЛ. Если сигнал L SSYN H высокий
;6459   ;   на входе мультиплексора BEN1, необходимо остановить микропрограмму на следующем
;6460   ;   микроцикле (BEN1 = L SSYN) и проверить наличие 101(B) на входах выборки A2,A1 и
;6461   ;   A0 соответственно. Если эти входы правильные и L SSYN H все еще высокий, тогда
;6462   ;   мультиплексор неисправен.
;6463   ;      Если регистр виртуального адреса = 3(H), тогда необходимо проверить, что
;6464   ;   снятие I MSYN H сбрасывает I SSYN H. Если МСТ может работать в пошаговом ре-
;6465   ;   жиме, центральный процессор необходимо остановить на инструкции MEM.REQ, а
;6466   ;   микропрограмму памяти остановить на четвертом микроцикле после адреса началь-
```

```
                    ;6467   ;    ного ветвления (BEN1 = L SSYN, регистр виртуального адреса не увеличивается).
                    ;6468   ;    Необходимо проверить низкий уровень на выходе L ISSYN H ПМЛ СИГН.КОНТР.ПИТА-
                    ;6469   ;    НИЯ И ИНИЦИАЦИИ. Если этот сигнал высокий, необходимо проверить низкий уро-
                    ;6470   ;    вень на входах в MSYN H и L INTR H. Если эти входы правильные, неисправна ПМЛ.
                    ;6471   ;    Если L ISSYN H низкий, необходимо остановить микропрограмму на следующем цик-
                    ;6472   ;    ле и проверить низкий уровень на выходе первого буфера, входом которого явля-
                    ;6473   ;    ется сигнал B SSYN H. Если здесь правильно, необходимо выполнить еще один шаг
                    ;6474   ;    и проверить второй буфер, который выдает низкий сигнал L SSYN H. Если здесь
                    ;6475   ;    правильно, необходимо проверить низкий уровень на входе L SSYN H мультиплек-
                    ;6476   ;    сора BEN1 и наличие 101(B) на входах выборки A2,A1 и A0 соответственно. Если
                    ;6477   ;    эти входы правильные, тогда мультиплексор неисправен.
                    ;6478   ;
                    ;6479    T.9:
U 0853, B65E,15     ;6480            MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR[0] для слова управления и
                    ;6481                                                       ; состояния
U 0854, 3E80,15     ;6482            MOV WR[0] TO LS[CONTROL.STATUS]            ; установка бита 15 в слове управления и состояния. Бит
                    ;6483                                                       ; 15 указывает начало теста для консольного процессора
U 0855, 10E0,15     ;6484            MISC [SET.CP.ATTN]                         ; выдача сигнала CPU ATTN консольному процессору
                    ;6485    WAIT.T9.0:
U 0856, 0885,64     ;6486            JMP [WAIT.T9.0]                            ; зацикливание для ожидания ответа консольного
                    ;6487                                                       ; процессора
U 0857, 0A1A,AC     ;6488            JSR [SETUP.1]                              ; установка масок, кода модуля и др.
U 0858, B62C,15     ;6489            MOV LS[#FFFFFF00] TO WR[0]                 ; установка маски ошибки
U 0859, 3EBA,15     ;6490            MOV WR[0] TO LS[ERROR.MASK]                ; запоминание маски ошибки для проверки младшего байта
U 085A, 3674,15     ;6491            MOV LS[DIAG.CHK] TO WR[0]                  ; установка бита DIAG CHK для записи в CSR
U 085B, 8A17,FC     ;6492            JSR [WRITE.CSR1]                           ; запись в CSR1 данных из WR[0] (используемых для
                    ;6493                                                       ; ветвления в тесте)
U 085C, 3645,95     ;6494            MOV  LS[#4] TO WR[3]                       ; подготовка 4 в WR3
U 085D, 2047,95     ;6495            INC WR[3]                                  ; WR3 содержит 5 (ожидаемые данные)
U 085E, DF2A,15     ;6496            MCOM LS[#FF000000] TO WR[0]                ; пересылка FFFFFF(H) в WR[0]
U 085F, 2100,15     ;6497            DEC WR[0]                                  ; WR0 = FFFFFE(H)
U 0860, BE12,15     ;6498            MOV WR[0] TO LS[T9]                        ; адрес в местную память
                    ;6499    LOOP.T9.1:
U 0861, 2006,95     ;6500            MOV WR[3] TO WR[1]                         ; установка ожидаемых данных
U 0862, 1F13,B5     ;6501            MEM.REQ[READ.MAINT.UBS] ADRS[T9] DT[WORD] ;   выполнение проверки с FFFFFE(H) в качестве
                    ;6502                                                       ; адреса
U 0863, 3022,15     ;6503            MOV MEM.DATA TO WR[0]                      ; выборка результата из регистра виртуального адреса
U 0864, 0869,3C     ;6504            JSR [CHECK.RESULT]                         ; проверка результата
U 0865, 8886,14     ;6505            JMP [LOOP.T9.1]                            ; зацикливание при ошибке, если разрешено
                    ;6506    END.T9:
```

```
;6507   .PAGE "ТЕСТЫ СДВИГАТЕЛЕЙ ДАННЫХ ECC"
;6508   .TOC "ТЕСТ А - запись/чтение через ПМЛ сдвигателей данных (модуль МСТ)"
;6509   ;
;6510   ;  ОПИСАНИЕ ТЕСТА:
;6511   ;
;6512   ;       Этот тест проверяет принципиальную исправность ПМЛ СДВИГАТЕЛЕЙ/РЕГ.ДАН-
;6513   ;       НЫХ путем пересылки данных в сдвигатели и обратно без выполнения сдвига.
;6514   ;       Будут проверены основные функции ПМЛ сдвигателей без использования ARRAY
;6515   ;       BUS или буферов ECC для запоминания данных. Путь данных следующий: выдает-
;6516   ;       ся инструкция MEM.REQ с полем MF=MAINT.ECC.DATA. Используемый адрес мест-
;6517   ;       ной памяти содержит данные 0(M) для дальнейшего ветвления в тесте. Эти дан-
;6518   ;       ные загружаются в регистр виртуального адреса и выполняется начальное вет-
;6519   ;       вление в соответствующую программу (см. описание начального ветвления в на-
;6520   ;       чале этого листинга). По адресу начального ветвления возбуждается занятость
;6521   ;       памяти и запрет сдвига, разрешаются биты физического адреса и передатчики
;6522   ;       на модуле DAP, чтобы данные были приняты DAP. В следующем цикле сдвигатели
;6523   ;       данных стробируются для запрета сдвига, повторяются все функции предыдущего
;6524   ;       цикла (за исключением того, что сигнал SET ADDR PH не нужен). Следующий
;6525   ;       цикл открывает буферы ECC, разрешает выход сдвигателей данных на ARRAY
;6526   ;       BUS, сбрасывает любые ошибки в CSR, поддерживает установленным MEMORY BUSY
;6527   ;       и выполняет ветвление по L CPU DR. Сигнал CPU DR (запрос данных) поступит,
;6528   ;       когда центральный процессор выполнит инструкцию MOV. В этом случае цент-
;6529   ;       ральный процессор посылает тестовый набор данных для ПМЛ сдвигателей. Ес-
;6530   ;       ли сигнал L CPU DR еще не возбужден, микропрограмма памяти зацикливается
;6531   ;       в ожидании его. Как только сигнал L CPU DR будет возбужден, следующий цикл
;6532   ;       возбуждает сигнал 2ND MEM CYC, открывает буферы ECC, разрешает выходы ПМЛ
;6533   ;       сдвигателей на ARRAY BUS и поддерживает разрешенными передатчики централь-
;6534   ;       ного процессора. Установленный сигнал 2ND MEM CYC поддерживает данные,, пе-
;6535   ;       реданные через шину MC BUS из центрального процессора, буферизированными
;6536   ;       внутри ПМЛ сдвигателей (данные были загружены в последнем микроцикле). Сле-
;6537   ;       дующий цикл поддерживает буферы ECC открытыми, поддерживает выход ПМЛ сдви-
;6538   ;       гателей на ARRAY BUS и ожидает возбуждения сигнала D DATA REC. Сигнал
;6539   ;       DATA RCVD возбуждается центральным процессором в конце (P2) инструкции MOV.
;6540   ;       Зацикливание здесь в ожидании возбуждения этого сигнала до возбуждения сиг-
;6541   ;       нала MEMORY BUSY гарантирует, что центральный процессор перейдет на следу-
;6542   ;       ющую микроинструкцию. Когда сигнал D DATA REC будет возбужден, микропрог-
;6543   ;       рамма памяти выполнит ветвление по LVA00 (который был сброшен в начале
;6544   ;       инструкции центрального процессора MEM.REQ) в соответствующий тест. Этот
;6545   ;       микроцикл возбуждает MEMORY BUSY, открывает буферы ECC, разрешает выходы
;6546   ;       ПМЛ сдвигателей на ARRAY BUS и разрешает передать контрольные биты из CSR1
;6547   ;       в контрольные биты ARRAY BUS. Этот цикл затем выполнит ветвление в соот-
;6548   ;       ветствующую тестовую ветвь в зависимости от состояния битов ECC DIS и
;6549   ;       DIAG CHK в CSR1. CSR1 был загружен до выдачи инструкции MEM.REQ. Для этого
;6550   ;       теста оба бита DIAG CHK и ECC DIS сброшены. Следующий цикл возбуждает
;6551   ;       MDR DAT OUT EN, который выдает тестовые данные, буферизованные в сдвигате-
;6552   ;       лях, на шину MC BUS и поддерживает возбужденным MEMORY BUSY. Когда тесто-
;6553   ;       вые данные появляются на шине MC BUS, они могут быть прочитаны инструкцией
;6554   ;       центрального процессора MOV MEM.DATA TO WR[0] и проверены. Следующий мик-
;6555   ;       роцикл сбрасывает MEMORY BUSY, поддерживает MDR DAT OUT EN возбужденным,
;6556   ;       пропускает выходы буферов ECC на шину ARRAY BUS (которая не используется
;6557   ;       в этом тесте) и проверяет сигнал L CPU DR (запрос данных). Если сигнал
;6558   ;       L CPU DR еще не безбужден, микропрограмма зацикливается для его ожидания.
;6559   ;       Цикл перед холостым циклом ожидает возбуждения сигнала D DAT REC, который
;6560   ;       означает, что центральный процессор имеет данные. Последний цикл возвраща-
;6561   ;       ет в холостой цикл. Используемые тестовые данные в этом тесте - все едини-
```

```
;6562    ;       цы, все нули и сдвигаемая единица.
;6563    ;
;6564    ;  ПРЕДПОЛОЖЕНИЯ:
;6565    ;
;6566    ;       Предполагается, что все предыдущие тесты выполнены успешно.
;6567    ;
;6568    ;  ШАГИ ТЕСТА:
;6569    ;
;6570    ;       1.Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;6571    ;         (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;6572    ;         памяти.
;6573    ;       2.Запись в CSR1 данных, обеспечивающих сброс битов DIAG CHK и ECC DIS.
;6574    ;       3.Загрузка в WR1 тестовых данных из местной памяти.
;6575    ;       4.Выдача инструкции MEM.REQ с DT=LONGWORD (11) и MF=MAINT.ECC.DATA,
;6576    ;         используя качестве адреса ячейку местной памяти, содержащую 0(H) (эти
;6577    ;         данные загружаются в регистр виртуального адреса и позже используются
;6578    ;         для ветвления).
;6579    ;       5.Выполнение инструкции WRITE.MEM LS с текущими тестовыми данными.
;6580    ;       6.Выполнение 5 инструкций NOP для установления шины MC BUS (этим гаранти-
;6581    ;         руется, что ПМЛ сдвигателей на самом деле возбуждают шину MC BUS).
;6582    ;       7.Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата.
;6583    ;       8.Повторение шагов с 2 по 6 с другими тестовыми данными.
;6584    ;
;6585    ;  ОШИБКИ:
;6586    ;
;6587    ;       ошибка 1 - данные искажаются при прохождении через ПМЛ СДВИГАТЕЛИ/РЕГ.
;6588    ;                  ДАННЫХ. Данные=все нули.
;6589    ;       ошибка 2 - данные искажаются при прохождении через ПМЛ СДВИГАТЕЛИ/РЕГ.
;6590    ;                  ДАННЫХ. Данные=все единицы.
;6591    ;       ошибка 3 - данные искажаются при прохождении через ПМЛ СДВИГАТЕЛИ/РЕГ.
;6592    ;                  ДАННЫХ. Данные=сдвигаемая единица.
;6593    ;
;6594    ;  НАЛАДКА:
;6595    ;
;6596    ;          Этот тест использует новый сигнал управления ветвлением, который может
;6597    ;       вызвать зависание микропрограммы памяти в ожидании сигнала D DATA REC H.
;6598    ;       Если тест сообщает о таймауте после инструкции MEM.REQ, необходимо прове-
;6599    ;       рить высокий уровень сигнала D DATA REC H на входе мультиплексора BEN3.
;6600    ;       На входах выборки A2, A1 и A0 должно быть 110(B) соответственно. Если
;6601    ;       эти входы правильные, по-видимому, мультиплексор неисправен. Если сигнал
;6602    ;       D DATA REC H низкий, необходимо проверить его на буфере, входом которого
;6603    ;       является сигнал DATA RCVD H. Сигнал DATA RCVD H был проверен на входе
;6604    ;       ПМЛ АРБИТР, но не как вход этого буфера.
;6605    ;
;6606    ;          ОШИБКА 1 - Эта ошибка может быть вызвана неисправной ПМЛ сдвигателей
;6607    ;       или множеством управляющих сигналов. Следует остановить центральный про-
;6608    ;       цессор на инструкции MEM.REQ и проверить сигналы на сдвигателях следующим
;6609    ;       образом (если не все биты искажены, выбирается ПМЛ, выходы которой непра-
;6610    ;       вильные): сигнал 2ND MEM CYC H должен быть низкий, сигнал MDR DAT OUT EN
;6611    ;       L должен быть высокий, сигналы A0 L и A1 L должны быть высокими и T0
;6612    ;       CLOCK (2) H должен переключаться (этот микроцикл зацикливается). Если
;6613    ;       сигнал 2ND MEM CYC H высокий, необходимо проверить его на выходе ПМЛ РАЗ-
;6614    ;       НЫЕ ФУНКЦИИ УПР. Если здесь неправильно, необходимо проверить наличие
;6615    ;       низкого уровня на входе SPF1. Выход первоначально установлен низким в хо-
;6616    ;       лостом цикле. Если выход высокий, необходимо вернуться в холостой цикл и
```

```
;6617    ;       проверить высокий уровень сигнала CONT FUNC LAT L. Если здесь правильно,
;6618    ;       выход должен быть низкий, иначе ПМЛ неисправна. Сигнал SPF1 должен оста-
;6619    ;       ваться низким до установки сигнала 2ND MEM CYC и выход не должен стать вы-
;6620    ;       соким до этого. Если A1 или A0 низкие, необходимо проверить ПМЛ УПР.СДВИ-
;6621    ;       ГАТЕЛЕМ ДАННЫХ. Сигнал ROT CLOCK H должен быть низким. Если этот вход пра-
;6622    ;       вильный, но выходы A1 и A0 низкие, должны быть проверены входы во время
;6623    ;       сигнала ROT CLOCK. Если пошаговый режим МСТ возможен, микропрограмму памя-
;6624    ;       ти следует остановить по адресу после начального ветвления (где имеется
;6625    ;       ROT CLOCK) и проверить высокий уровень на входах ROT C1 H, ROT C0 H,
;6626    ;       CPH/UBL H и ROT CLOCK H. Если все эти сигналы правильные, по-видимому, ПМЛ
;6627    ;       неисправна. Сигнал CPH/UBL приходит из ПМЛ УПР.ПРЕДВЫБОРКОЙ. Если сигнал
;6628    ;       CPH/UBL низкий, небхвдимо проверить низкий уровень сигнала CPU GRANT L и
;6629    ;       высокий - CONT FUNC LAT L, пока микропрограмма памяти циклится на цикле,
;6630    ;       который проверяет сигналы L CPU DR H (центральный процессор нужно остано-
;6631    ;       вить на инструкции MEM.REQ, а память автоматически циклится на этом мик-
;6632    ;       роцикле).
;6633    ;          Если входы сдвигателей данных правильные, центральный процессор нужно
;6634    ;       остановить на инструкции WRITE.MEM. Микропрограмма памяти будет циклиться
;6635    ;       в ожидании возбуждения сигнала центрального процессора DATA RCVD. Выходы
;6636    ;       теперь можно проверять на шине MC BUS и они должны содержать ожидаемые
;6637    ;       данные. Также нужно проверить высокий уровень на входе 2ND MEM CYC H.
;6638    ;       Сигнал 2ND MEM CYC H установен в предыдущем цикле возбуждением сигнала
;6639    ;       SPF1 H. Если сигнал 2ND MEM CYC H на ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. низкий и по-
;6640    ;       шаговый режим МСТ возможен, тогда необходимо снова запустить центральный
;6641    ;       процессор от инструкции MEM.REQ, разрешить пошаговый режим МСТ, остановить
;6642    ;       центральный процессор на инструкции WRITE.MEM, а микропрограмму памяти ос-
;6643    ;       тановить на следующем микроцикле (возбуждается 2ND MEM.CYC H). Необходимо
;6644    ;       проверить высокий уровень на входе SPF1 H ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. выход
;6645    ;       должен быть высоким. Если нет, ПМЛ неисправна. Затем нужно выполнить еще
;6646    ;       один шаг МСТ и проверить низкий уровень на входах CONT FUNC LAT L, SPF0 H,
;6647    ;       SPF1 H и SPF2 H. Если сигнал 2ND MEM CYC H сейчас становится низким, ПМЛ
;6648    ;       неисправна.
;6649    ;          Если сигнал 2ND MEM CYC был правильный на ПМЛ сдвигателей, но выходы
;6650    ;       были неправильные, необходимо проверить входы, остановив центральный про-
;6651    ;       цессор на инструкции MEM.REQ. Микропрограмма памяти должна циклиться в ожи-
;6652    ;       дании сигнала DATA REQ центрального процессора. Входы на шине MC BUS можно
;6653    ;       проверить во время этого цикла.
;6654    ;          Остановите центральный процессор на инструкции MOV MEM.DATA TO WR, и
;6655    ;       пусть микрокод памяти циклится на предпоследнем микрослове в тесте, ожидая
;6656    ;       возбуждения сигнала DATA RCVD центрального процессора. Необходимо проверить
;6657    ;       низкий уровень входа MDR DATA OUT EN L и высокий уровень - 2ND MEM CYC H
;6658    ;       ПМЛ сдвигателей данных. Выход на шине MC BUS должен сдержать ожидаемые дан-
;6659    ;       ные. Если входы правильные, по-видимому, ПМЛ сдвигателя неисправна (имеется
;6660    ;       небольшая вероятность, что управляющая память искажает при выдаче биты кор-
;6661    ;       рекции в некотором предыдущем цикле).
;6662    ;
;6663    ;          ОШИБКА 2 - Так же, как и при ошибке 1. Отличаются только тестовые данные.
;6664    ;
;6665    ;          ОШИБКА 3 - См. ошибку 1, но скорее всего подозревается сама ПМЛ сдвигате-
;6666    ;                     ля.
;6667    ;
;6668    ;
;6669    ;
;6670
;6671     T.A:
```

```
U 0866,  B65E,15 ;6672            MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR[0] для слова управления и
                 ;6673                                                       ; состояния
U 0867,  3E80,15 ;6674            MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                 ;6675                                                       ; 15 указывает начало теста консольному процессору
U 0868,  10E0,15 ;6676            MISC [SET.CP.ATTN]                         ; выдача сигнала CPU ATTN консольному процессору
                 ;6677    WAIT.TA.0:
U 0869,  0886,94 ;6678            JMP [WAIT.TA.0]                            ; зацикливание для ожидания ответа консольного
                 ;6679                                                       ; процессора
U 086A,  0A1A,9C ;6680            JSR [SETUP]                                ; установка масок, кода модуля и др. и очистка CSR1
                 ;6681    LOOP.TA.1:
U 086B,  B69C,95 ;6682            MOV LS[ZERO] TO WR[1]                      ; ожидаемые данные
U 086C,  9D9C,F5 ;6683            MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение проверки сдвигателей данных
U 086D,  B29C,15 ;6684            WRITE.MEM LS[ZERO]                         ; запись нулей в сдвигатели
U 086E,  0A1B,4C ;6685            JSR [NOP.DELAY]                            ; выполнение 5 циклов задержки для установления шины MC
                 ;6686                                                       ; BUS
U 086F,  3022,15 ;6687            MOV MEM.DATA TO WR[0]                      ; выборка результата из сдвигателей данных
U 0870,  0869,3C ;6688            JSR [CHECK.RESULT]                         ; проверка результата
U 0871,  8886,B4 ;6689            JMP [LOOP.TA.1]                            ; зацикливание при ошибке
U 0872,  FF82,15 ;6690            INC LS[ERROR.NUMBER]                       ; ошибка 2
                 ;6691    LOOP.TA.2:
U 0873,  369E,95 ;6692            MOV LS[ONES] TO WR[1]                      ; ожидаемые данные
U 0874,  9D9C,F5 ;6693            MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение проверки сдвигателей данных
U 0875,  329E,15 ;6694            WRITE.MEM LS[ONES]                         ; запись единиц в сдвигатели
U 0876,  0A1B,4C ;6695            JSR [NOP.DELAY]                            ; выполнение 5 циклов задержки для установления шины MC
                 ;6696                                                       ; BUS
U 0877,  3022,15 ;6697            MOV MEM.DATA TO WR[0]                      ; выборка результата из сдвигателей данных
U 0878,  0869,3C ;6698            JSR [CHECK.RESULT]                         ; проверка результата
U 0879,  8887,34 ;6699            JMP [LOOP.TA.2]                            ; зацикливание при ошибке, если разрешено
U 087A,  FF82,15 ;6700            INC LS[ERROR.NUMBER]                       ; ошибка 3
U 087B,  E5F8,15 ;6701            CLR LS[OS]                                 ; очистка индекса
                 ;6702    LOOP.TA.3:
U 087C,  B6F6,95 ;6703            MOV LS[SHIFT.OS(4-0)] TO WR[1]             ; выборка следующего набора тестовых данных
                 ;6704                                                       ; (ожидаемые данные)
U 087D,  9D9C,F5 ;6705            MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение проверки сдвигателей данных
U 087E,  B2F6,15 ;6706            WRITE.MEM LS[SHIFT.OS(4-0)]                ; запись сдвигаемой единицы в сдвигатели
U 087F,  0A1B,4C ;6707            JSR [NOP.DELAY]                            ; выполнение 5 циклов задержки для установления шины MC
                 ;6708                                                       ; BUS
U 0880,  3022,15 ;6709            MOV MEM.DATA TO WR[0]                      ; выборка результата из сдвигателей данных
U 0881,  0869,3C ;6710            JSR [CHECK.RESULT]                         ; проверка результата
U 0882,  8887,C4 ;6711            JMP [LOOP.TA.3]                            ; зацикливание при ошибке, если разрешено
U 0883,  7FF8,15 ;6712            INC LS[OS]                                 ; увеличение индекса для следующего набора
U 0884,  36F9,15 ;6713            MOV LS[OS] TO WR[2]                        ; проверка, что OS увеличен до 20 (H) и
                 ;6714              BIT LS[BIT5] WITH WR[2],                 ; установка кодов условий
U 0885,  D94B,35 ;6715            DT(LONG)&SET.ALU.CC                        ;
U 0886,  0887,C9 ;6716            JMP.IF[BITS.CLR] TO [LOOP.TA.3]            ; повторение со следующим набором, если не все
                 ;6717                                                       ; использованы
                 ;6718    END.TA:
```

```
;6719    .PAGE "ТЕСТ В - тест шины BUS ARRAY, регистров схемы ECC и CSR CKBT (модуль MCT)"
;6720    ;
;6721    ;  ОПИСАНИЕ ТЕСТА:
;6722    ;
;6723    ;        Этот тест проверяет регистры схемы ECC (кодов коррекции ошибок) и шину BUS
;6724    ;       ARRAY, выдавая данные из сдвигателей данных на шину BUS ARRAY и буферизируя
;6725    ;       данные в микросхемах ECC. Тогда данные таким же путем читаются обратно и
;6726    ;       проверяются. Также проверяются CSR1, CSR0 и регистр CKBT схемы ECC. В CSR1
;6727    ;       записываются контрольные биты и тест записывает эти биты в регистр CKBT
;6728    ;       схемы ECC. Тогда контрольные биты из регистра CKBT пересылаются на BUS
;6729    ;       ARRAY CB[7:1] и записываются в CSR0, где они могут быть проверены.
;6730    ;          Путь данных следующий: выдается инструкция MEM.REQ с полем MF=MAINT.
;6731    ;       ECC.DATA. Используемый адрес местной памяти содержит данные=0(H) для
;6732    ;       последующего ветвлений в тесте. Эти данные загружаются в регистр виртуаль-
;6733    ;       ного адреса и начальное ветвление передает управление в соответствующую
;6734    ;       программу (см. описание начального ветвления в начале этого листинга).
;6735    ;       По адресу начального ветвления устанавливаются MEMORY BUSY и запрет
;6736    ;       сдвига, разрешаются биты PA и передатчики на модуле DAP разрешаются,
;6737    ;       чтобы были приняты данные из центрального процессора. В следующем цикле
;6738    ;       сдвигатели данных стробируются с запретом сдвига и повторяются все функ-
;6739    ;       ции предыдущего цикла (за исключением того что установка сигнала ADDR PH
;6740    ;       не нужна). Следующий цикл открывает регистры схемы ECC, разрешает выходы
;6741    ;       сдвигателей данных на BUS ARRAY, очищает любые ошибки в CSR, поддерживает
;6742    ;       установленным MEMORY BUSY и выполняет ветвление по сигналу L CPU DR.
;6743    ;          Сигнал CPU DR (запрос данных) поступает, когда центральный процессор вы-
;6744    ;       полняет инструкцию MOV. В этом случае центральный процессор посылает тес-
;6745    ;       товые данные для ПМЛ сдвигателей данных. Если сигнал L CPU DR еще не воз-
;6746    ;       бужден, микропрограмма памяти зацикливается в ожидании его. Как только
;6747    ;       сигнал L CPU DR возбуждается, следующий микроцикл устанавливает
;6748    ;       2ND MEM CYC, открывает регистры схемы ECC, разрешает выходы ПМЛ сдвига-
;6749    ;       телей на BUS ARRAY и поддерживает разрешенными передатчики центрального
;6750    ;       процессора. Установка 2ND MEM CYC поддерживает данные, посылаемые через
;6751    ;       шину MC BUS из центрального процессора и буферизируемые внутри ПМЛ
;6752    ;       сдвигателей (данные были загружены в последнем микроцикле).
;6753    ;       Следующий цикл поддерживает открытыми регистры схемы ECC, поддерживает
;6754    ;       выход ПМЛ сдвигателей на шину BUS ARRAY и ожидает возбуждения сигнала
;6755    ;       D DATA REC. Сигнал DATA RCVD возбуждается центральным процессором в кон-
;6756    ;       це (P2) инструкции MOV. Зацикливание здесь, пока этот сигнал не будет
;6757    ;       возбужден, до возбуждения MEMORY BUSY гарантирует, что центральный про-
;6758    ;       цессор перейдет на следующую микроинструкцию.
;6759    ;          Когда сигнал D DATA REC возбуждается, микропрограмма памяти выполняет
;6760    ;       ветвление по сигналу LVA00 (который был сброшен к началу инструкции цент-
;6761    ;       рального процессора MEM.REQ) на соответствующую тестовую ветвь. Этот мик-
;6762    ;       роцикл возбуждает MEMORY BUSY, открывает регистры схемы ECC, разрешает вы-
;6763    ;       ходы ПМЛ сдвигателей на шину BUS ARRAY и разрешает выход контольных битов
;6764    ;       из CSR1 на шину контрольных битов BUS ARRAY CB. Микропрограмма тогда выпол-
;6765    ;       нит переход на соответствующую тестовую ветвь в зависимости от состояния би-
;6766    ;       тов ECC DIS и DIAG CHK в CSR1. CSR1 был загружен до выдачи инструкции
;6767    ;       MEM.REQ. Для этого теста биты DIAG CHK и ECC DIS установлены и CKBT (конт-
;6768    ;       рольные биты) записаны в 7 младших битах CSR1. Следующий микроцикл закрыва-
;6769    ;       ет регистры входных, выходных и контрольных битов схемы ECC. Сигнал 2ND MEM
;6770    ;       CYC сбрасывается, так что сдвигатели данных могут снять данные с BUS ARRAY
;6771    ;       позже. MEMORY BUSY остается возбужденным. Следующий микроцикл возбуждает би-
;6772    ;       ты синдрома на шине BUS ARRAY CB (обеспечивает, что буферы CKTB закрыты) и
;6773    ;       поддерживает возбужденным MEMORY BUSY. Следующий цикл выдает буферизирован-
```

```
;6774    ;       ные CKBT на шину BUS ARRAY CB [T:1], поддерживает возбужденным MEMORY BUSY
;6775    ;       и выполняет ветвление по ошибке ECC. Если ошибка была обнаружена (CKBT не
;6776    ;       соответствуют тем, которые сгенерированы для данных), тогда микропрограмма
;6777    ;       должна выполнить переход к инструкции, которая возбуждает сигнал CSR ERR
;6778    ;       ADDR CLK для ПМЛ УПР. CSR. В CSR1 будут возбуждены биты RDS или CRD, в за-
;6779    ;       висимости от того, была обнаружена одиночная ошибка или неисправимая ошиб-
;6780    ;       ка. Кроме того, будет возбужден сигнал DATA ERR, который возбуждает сиг-
;6781    ;       нал ERR SUM для центрального процессора. Буферизованные CKBT из микросхе-
;6782    ;       мы ECC поддерживаются разрешенными на шине BUS ARRAY CB. Если ошибок не
;6783    ;       обнаружено, сигнал CSR ERROR ADDR CLK не возбуждается и не будет сообще-
;6784    ;       ния о суммарной ошибке, RDS или CRD.
;6785    ;          Следующий цикл разрешает выход данных ECC на шину BUS ARRAY и разрешает
;6786    ;       ПМЛ сдвигателей данных для передачи этих данных на шину MC BUS. Также стро-
;6787    ;       бирует в CSR0 контрольные биты, которые были буферизированы в микросхеме ECC.
;6788    ;       Выдача данных ECC на шину BUS ARRAY и разрешение сдвигателям данных передать
;6789    ;       данные на шину MC BUS повторяется и в следующем цикле. Сейчас тестовые данные
;6790    ;       находятся на шине MC BUS и могут быть считаны центральным процессором инст-
;6791    ;       рукцией MOV.MEM.DATA TO WR[0] и проверены.
;6792    ;          Следующий цикл сбрасывает MEMORY BUSY, поддерживает возбужденным
;6793    ;       MDR DAT OUT EN, обеспечивает выход ECC на шину BUS ARRAY и проверяет
;6794    ;       сигнал L CPU DR (запрос данных). Если сигнал L CPU DR еще не возбужден,
;6795    ;       микропрограмма циклится в ожидании его. Цикл до холостого цикла циклится
;6796    ;       в ожидании возбуждения сигнала D DAT REC, означающего, что центральный
;6797    ;       процессор имеет данные. Последний цикл выполняет возврат в холостой цикл.
;6798    ;          Тестовые данные, используемые в этом тесте, следующие: для данных ECC-
;6799    ;       все единицы, все нули и сдвигаемая единица. Для контрольных битов ECC
;6800    ;       (CKBT) - все нули, все единицы и сдвигаемая единица.
;6801    ;
;6802    ;  ПРЕДПОЛОЖЕНИЯ:
;6803    ;
;6804    ;       Предполагается, что все предыдущие тесты выполнены успешно.
;6805    ;
;6806    ;  ШАГИ ТЕСТА:
;6807    ;
;6808    ;       1. Установка маски ошибки, номера ошибки и номера модуля в местной памя-
;6809    ;          ти (для распечатки ошибок) и очистка предыдущего номера ошибки в мест-
;6810    ;          ной памяти.
;6811    ;       2. Запись в CSR1 данных для установки битов DIAG CHK и ECC DIS. Также
;6812    ;          загрузка части CKBT (биты 0-6) тестовыми данными для проверки CKBT.
;6813    ;       3. Загрузка WR1 тестовыми данными из местной памяти (ожидаемый результат
;6814    ;          данных ECC).
;6815    ;       4. Выдача инструкции MEM.REQ с DT=LONGWORD (11) и MF=MAINT.ECC.DATA,
;6816    ;          используя ячейку местной памяти, которая содержит 0(H) в качестве ад-
;6817    ;          реса (загружается в регистр виртуального адреса и используется для бо-
;6818    ;          лее позднего ветвления).
;6819    ;       5. Выполнение инструкции WRITE.MEM LS с текущими тестовыми данными.
;6820    ;       6. Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата
;6821    ;          (данные ECC).
;6822    ;       7. Изменение маски ошибки для проверки битов 0-6 и чтение CSR0 для
;6823    ;          проверки данных скат.
;6824    ;       8. Восстановление маски ошибки для 32 битов и повторение шагов с 2 по 7
;6825    ;          со следующими тестовыми данными.
;6826    ;
;6827    ;  ОШИБКИ:
;6828    ;
```

```
;6829    ;       ошибка 1 - неправильно работает регистр схемы ECC, BUS ARRAY или сдвигатели с
;6830    ;                  данными - все единицы.
;6831    ;       ошибка 2 - неправильно работает регистр CKBT схемы ECC, CKBT BUS, CSR1 или
;6832    ;                  CSR0 с данными - все нули.
;6833    ;       ошибка 3 - неправильно работает регистр схемы ECC, BUS ARRAY или сдвигатели с
;6834    ;                  данными - все нули.
;6835    ;       ошибка 4 - непправильно работает регистр CKBT схемы ECC, CKBT BUS, CSR1 или CSR0 с
;6836    ;                  данными - все единицы.
;6837    ;       ошибка 5 - неправильно работает регистр схемы ECC, BUS ARRAY или сдвигатели с
;6838    ;                  данными - сдвигаемая единица.
;6839    ;       ошибка 6 - непправильно работает регистр CKBT схемы ECC, CKBT BUS, CSR1 или CSR0 с
;6840    ;                  данными - сдвигаемая единица.
;6841    ;
;6842    ;  НАЛАДКА:
;6843    ;
;6844    ;
;6845    ;          ОШИБКА 1 - Этот тест является первым, который использует шину BUS ARRAY
;6846    ;       или схемы ECC. Он также использует новый путь через ПМЛ сдвигателей данных.
;6847    ;       Прежде всего необходимо проверить выходы сдвигателей данных и управляю-
;6848    ;       щие сигналы для схемы ECC, остановив центральный процессор на инструкции
;6849    ;       WRITE.MEM. Микропрограмма памяти должна циклиться в ожидании сигнала
;6850    ;       DATA RCVD центрального процессора. Необходимо проверить низкие уровни
;6851    ;       выходов сдвигателей данных на BUS ARRAY DXX L. Если неправильные, необхо-
;6852    ;       димо проверить низкий уровень сигнала DIS WR BYT EN L на 11 ножке. Этот
;6853    ;       сигнал поступает прямо из ПЗУ управляющей памяти. Если сигнал разрешения
;6854    ;       низкий, по-видимому, ПМЛ неисправна (другие выходы использовались в пре-
;6855    ;       дыдущих тестах). Если выходы на шине BUS ARRAY DXX L правильные, необхо-
;6856    ;       димо проверить входы схемы ECC с шины BUS ARRAY. Также необходимо про-
;6857    ;       верить низкий уровень сигнала LAT DAT IN H (ножка 1 обеих микросхем ECC)
;6858    ;       и высокий уровень сигнала LAT OUTPUT BYT X L (ножки 3 и 45 обеих микро-
;6859    ;       схем ECC). Сигнал LAT OUTPUT BYT X L поступает из схем совпадения по
;6860    ;       низким уровням сигнала LAT OUTPUT BYT L из ПЗУ управляющей памяти и вы-
;6861    ;       ходов ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ. ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ
;6862    ;       должна выдать низкие уровни на ножках 12, 13, 14 и 15, которые поступают
;6863    ;       на одну половину схем совпадения по низким уровням. Если один из этих
;6864    ;       выходов высокий, необходимо проверить низкий уровень на входе ROT C0 H.
;6865    ;       Только этот сигнал необходим для формирования низких уровней на вышеупо-
;6866    ;       мянутых выходах. Если сигнал ROT C0 H правильный, ПМЛ неисправна.
;6867    ;          Если в этой точке все правильно, следует остановить центральный процес-
;6868    ;       сор на инструкции MOV MEM.DATA TO WR[0]. Память должна зациклиться на пред-
;6869    ;       последнем микроцикле в ожидании сигнала DATA RCVD центрального процессора.
;6870    ;       Необходимо проверить сигналы шины BUS ARRAY DXX L на входах ПМЛ сдвигателей
;6871    ;       данных. Они все должны быть низкими, и формируются микросхемой ECC. Если
;6872    ;       входы BUS ARRAY неправильные, необходимо проверить следующие входы микро-
;6873    ;       схемы ECC: входы LAT OUTPUT BYT X L должны быть низкими, входы LAT DAT
;6874    ;       IN H и OUTPUT BYT (0-3) H должны быть высокими. Сигналы OUTPUT BYTE и
;6875    ;       LAT DAT IN поступают прямо из ПЗУ управляющей памяти, в то время как
;6876    ;       сигналы LAT OUTPUT BYT X L поступают из схем совпадения по низким уров-
;6877    ;       ням. Если этот сигнал неправильный, необходимо проверить высокий уровень
;6878    ;       сигнала LAT OUTPUT BYT L (C63) на входе схемы совпадения. Если эти входы
;6879    ;       правильные, но выход неправильный, подозревается микросхема ECC.
;6880    ;          Если входы сдвигателей данных на шине BUS ARRAY правильные, необходимо
;6881    ;       проверить другие входы ПМЛ сдвигателей: входы 2ND MEM CYC H и MDR DAT OUT
;6882    ;       EN L должны быть низкими, а входы A0 L и A1 L должны быть высокими. Если
;6883    ;       эти входы правильные, выходы на шине BUS MC DXX H должны быть высокими.
```

```
                ;6884   ;       Если нет, неисправна ПМЛ сдвигателей.
                ;6885   ;          Другим возможным источником ошибки является то, что одна из плат матриц
                ;6886   ;       памяти блокирует линии BUS ARRAY DXX или закорачивает 2 линии между со-
                ;6887   ;       бой. Для проверки этого предположения необходимо удалить одну из плат
                ;6888   ;       матрицы памяти и запустить этот тест снова.
                ;6889   ;
                ;6890   ;          ОШИБКА 2 - Эта ошибка указывает на неправильную работу битов 0-6 регист-
                ;6891   ;       ров CSR1 или CSR0, или шины BUS ARRAY CB, регистра CKBT схемы ECC или линий
                ;6892   ;       управления. Следует остановить центральный процессор на инструкции WRITE.MEM.
                ;6893   ;       Микропрограмма памяти должна циклиться в ожидании сигнала DATA RCVD цент-
                ;6894   ;       рального процессора. Необходимо проверить низкий уровень на всех выходах
                ;6895   ;       буферов CSR1 (BUS ARRAY CB X L). Если неправильные, необходимо проверить
                ;6896   ;       низкий уровень сигнала CSR CB EN L (C43). Если он правильный, необходимо
                ;6897   ;       проверить изменение уровней других входов (отсутствие связи). Если уровни
                ;6898   ;       не меняются, необходимо запустить микропрограмму WRITE CSR для проверки
                ;6899   ;       синхросигналов и входов шины BUS MC. Если MCT не работает в пошаговом ре-
                ;6900   ;       жиме, должно использоваться зацикливание теста.
                ;6901   ;          Если выходы на шине BUS ARRAY CB X L правильные, необходимо проверить
                ;6902   ;       высокий уровень сигнала LAT CB REG L на микросхеме ECC и низкие уровни на
                ;6903   ;       входах BUS ARRAY CB X L микросхемы ECC.
                ;6904   ;          Если до этого места все правильно, и MCT может работать в пошаговом ре-
                ;6905   ;       жиме, следует подготовить MCT для пошагового режима, а центральный процессор
                ;6906   ;       остоновить на инструкции READ.MEM. Контролер ОЗУ необходимо довести до
                ;6907   ;       микроцикла, который выдает информацию на BUS ARRAY CB X L и стробирует
                ;6908   ;       CSR0 (этот цикл следует за ветвлением по ошибке BEN0=ERROR L, если нет ошиб-
                ;6909   ;       ки). Необходимо проверить низкие уровни на линиях BUS ARRAY CB X L, поступа-
                ;6910   ;       ющих в буфер CSR0. Также необходимо проверить высокий уровень сигнала
                ;6911   ;       CSR CB/SYN CLK H на буфере CSR0. Если эти сигналы правильные, тогда или бу-
                ;6912   ;       фер CSR0 неисправен, сигнал CSR CB/SYN CLK не становится низким, или сигнал
                ;6913   ;       RD CSR 0 L не становится низким для разрешения буфера, когда выполняется
                ;6914   ;       функция READ CSR. Если сигналы BUS ARRAY CB X L неправильные, то необходимо
                ;6915   ;       проверить, что сигнал OUTPUT CB BUS H (C26) на микросхеме ECC высокий, а
                ;6916   ;       сигнал LAT CB REG L низкий. Если эти сигналы правильные, подозревается мик-
                ;6917   ;       росхема ECC.
                ;6918   ;
                ;6919   ;          ОШИБКА 3 - Так же, как и при ошибке 1, за исключением того, что данные-
                ;6920   ;                  все нули.
                ;6921   ;
                ;6922   ;          ОШИБКА 4 - Так же, как и при ошибке 2, за исключением того, что данные-
                ;6923   ;                  все единицы.
                ;6924   ;
                ;6925   ;          ОШИБКА 5 - Во-первых, подозреваются короткие замыкания между двумя линия-
                ;6926   ;                  ми BUS ARRAY или внутренние закорачивания в микросхемах ECC или
                ;6927   ;                  ПМЛ сдвигателей данных.
                ;6928   ;
                ;6929   ;          ОШИБКА 6 - Во-первых, подозреваются короткие замыкания между двумя лини-
                ;6930   ;                  ями BUS ARRAY CBX или внутренние закорачивания в микросхемах ECC,
                ;6931   ;                  буфере CSR1 или буфере CSR0.
                ;6932   ;
                ;6933   ;
                ;6934   ;
                ;6935    T.B:
U 0887, B65E,15 ;6936            MOV LS[BEGIN.TEST] TO WR[0]            ; установка бита 15 в WR[0] для слова управления и
                ;6937                                                   ; состояния
U 0888, 3E80,15 ;6938            MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
```

```
                    ;6939                                   ; 15 указывает начало теста для консольного процессора
U 0889,  10E0,15    ;6940           MISC [SET.CP.ATTN]      ; выдача сигнала CPU ATTN консольному процессору
                    ;6941   WAIT.TB.0:
U 088A,  8888,A4    ;6942           JMP [WAIT.TB.0]         ; зацикливание для ожидания ответа консольного
                    ;6943                                   ; процессора
U 088B,  0A1A,AC    ;6944           JSR [SETUP.1]           ; установка масок, кода модуля и др.
U 088C,  8713,95    ;6945           MOV LS[^XFFFFFFB0] TO WR[3] ; установка маски ошибки для 7 битов (6-0) для ошибок с
                    ;6946                                   ; четными номерами
U 088D,  3641,15    ;6947           MOV LS[#1] TO WR[2]      ; установка 1 в WR2 (используется для установки номера
                    ;6948                                   ; ошибки)
U 088E,  3672,15    ;6949           MOV LS[ECC.DIS] TO WR[0] ; установка бита ECC DIS в WR[0]
U 088F,  4774,15    ;6950           BIS LS[DIAG.CHK] TO WR[0] ; также установка бита DIAG CHK
U 0890,  8A17,FC    ;6951           JSR [WRITE.CSR1]        ; установка битов ECC DIS и DIAG CHK в CSR1, очистка
                    ;6952                                   ; CKBT в битах 6-0 (тестовые данные)
                    ;6953   LOOP.TB.1:
U 0891,  E58A,15    ;6954           CLR LS[ERROR.MASK]      ; проверка всех битов
U 0892,  3EB3,15    ;6955           MOV WR[2] TO LS[ERROR.NUMBER] ; установка номера ошибки в местной памяти (номер
                    ;6956                                   ; ошибки=1)
U 0893,  369E,95    ;6957           MOV LS[ONES] TO WR[1]    ; ожидаемые данные для битов данных
U 0894,  9D9C,F5    ;6958           MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение теста схемы ECC
U 0895,  329E,15    ;6959           WRITE.MEM LS[ONES]      ; запись единичных данных в схему ECC. CKBT (контрольные
                    ;6960                                   ; биты) получают нулевые данные из CSR1
U 0896,  3022,15    ;6961           MOV MEM.DATA TO WR[0]    ; выборка битов данных результата (должны быть единицы)
U 0897,  0869,3C    ;6962           JSR [CHECK.RESULT]      ; проверка результата
U 0898,  8889,14    ;6963           JMP [LOOP.TB.1]         ; зацикливание при ошибке, если разрешено
U 0899,  3EB8,95    ;6964           MOV WR[3] TO LS[ERROR.MASK] ; запоминание маски в местной памяти
U 089A,  FFB2,15    ;6965           INC LS[ERROR.NUMBER]    ; ошибка 2
U 089B,  B69C,95    ;6966           MOV LS[ZERO] TO WR[1]    ; ожидаемые данные
U 089C,  9D9D,75    ;6967           MEM.REQ[READ.CSR] ADRS[#0] DT[LONG] ; чтение CKBT CSR0
U 089D,  3022,15    ;6968           MOV MEM.DATA TO WR[0]    ; выборка результата для CKBT
U 089E,  0869,3C    ;6969           JSR [CHECK.RESULT]      ; проверка результата
U 089F,  8889,14    ;6970           JMP [LOOP.TB.1]         ; зацикливание при ошибке, если разрешено
U 08A0,  4043,15    ;6971           ADD LS[#2] TO WR[2]      ; ошибка 3 в WR2
U 08A1,  B626,15    ;6972           MOV LS[#FF] TO WR[0]     ; установка в рабочем регистре битов 7-0
U 08A2,  4772,15    ;6973           BIS LS[ECC.DIS] TO WR[0] ; установка в рабочем регистре бита ECC DIS
U 08A3,  4774,15    ;6974           BIS LS[DIAG.CHK] TO WR[0] ; установка битов ECC DIS и DIAG CHK в CSR1
U 08A4,  8A17,FC    ;6975           JSR [WRITE.CSR1]        ; установка единичных CKBT в битах 6-0 (тестовые данные)
                    ;6976                                   ; (бит 7 в CSR1 также установлен, но не используется)
                    ;6977   LOOP.TB.3:
U 08A5,  E58A,15    ;6978           CLR LS[ERROR.MASK]      ; проверка всех битов
U 08A6,  3EB3,15    ;6979           MOV WR[2] TO LS[ERROR.NUMBER] ; установка номера ошибки в местной памяти (ошибка 3)
U 08A7,  B69C,95    ;6980           MOV LS[ZERO] TO WR[1]    ; ожидаемые данные для битов данных
U 08A8,  9D9C,F5    ;6981           MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение теста данных схемы ECC
U 08A9,  B29C,15    ;6982           WRITE.MEM LS[ZERO]      ; запись нулевых данных в схемы ECC. Биты CKBT получают
                    ;6983                                   ; единичные данные из CSR1
U 08AA,  3022,15    ;6984           MOV MEM.DATA TO WR[0]    ; выборка битов данных результата (должны быть
                    ;6985                                   ; нулевыми)
U 08AB,  0869,3C    ;6986           JSR [CHECK.RESULT]      ; проверка результата
U 08AC,  88BA,54    ;6987           JMP [LOOP.TB.3]         ; зацикливание при ошибке, если разрешено
U 08AD,  3EBB,95    ;6988           MOV WR[3] TO LS[ERROR.MASK] ; запоминание маски в местной памяти
U 08AE,  FFB2,15    ;6989           INC LS[ERROR.NUMBER]    ; ошибка 4
U 08AF,  369E,95    ;6990           MOV LS[ONES] TO WR[1]    ; ожидаемые данные
U 08B0,  9D9D,75    ;6991           MEM.REQ[READ.CSR] ADRS[#0] DT[LONG] ; чтение CSR0 для получения битов CKBT
U 08B1,  3022,15    ;6992           MOV MEM.DATA TO WR[0]    ; выборка результата для CKBT
U 08B2,  0869,3C    ;6993           JSR [CHECK.RESULT]      ; проверка результата
```

```
U 08B3, 088A,54 ;6994            JMP [LOOP.TB.3]                          ; зацикливание при ошибке, если разрешено
U 0884, 4043,15 ;6995            ADD LS[#2] TO WR[2]                      ; ошибка 5 в WR2
U 08B5, E5F8,15 ;6996            CLR LS[OS]                               ; очистка индекса
                ;6997    REPEAT.TB.5:
U 08B6, 36F6,15 ;6998            MOV LS[SHIFT.OS(4-0)] TO WR[0]           ; установка в рабочем регистре битов 7-0
U 08B7, 452C,15 ;6999            BIC LS[#FFFFFF00] TO WR[0]               ; очистка остальных битов рабочего регистра
U 08B8, 4772,15 ;7000            BIS LS[ECC.DIS] TO WR[0]                 ; установка бита ECC DIS в рабочем регистре
U 08B9, 4774,15 ;7001            BIS LS[DIAG.CHK] TO WR[0]                ; также установка бита DIAG CHK
U 08BA, 8A17,FC ;7002            JSR [WRITE.CSR1]                         ; установка битов ECC DIS и DIAG CHK в CSR1. Установка
                ;7003                                                     ; тестовых CKBT в битах 6-0.(бит 7 в CSR1 также
                ;7004                                                     ; установлен, но не используется)
                ;7005    LOOP.TB.5:
U 08BB, E58A,15 ;7006            CLR LS[ERROR.MASK]                       ; проверка всех битов
U 08BC, 3E83,15 ;7007            MOV WR[2] TO LS[ERROR.NUMBER]            ; установка номера ошибки в местной памяти (ошибка 5)
U 08BD, B6F6,95 ;7008            MOV LS[SHIFT.OS(4-0)] TO WR[1]           ; ожидаемые данные для битов данных
U 08BE, 9D9C,F5 ;7009            MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение теста данных схемы ECC
U 08BF, B2F6,15 ;7010            WRITE.MEM LS[SHIFT.OS(4-0)]              ; запись данных со сдвигаемой единицей в схемы ECC. Биты
                ;7011                                                     ; CKBT получают сдвигаемую единицу из CSR1
U 08C0, 3022,15 ;7012            MOV MEM.DATA TO WR[0]                    ; выборка битов данных результата
U 08C1, 0869,3C ;7013            JSR [CHECK.RESULT]                       ; проверка результата
U 08C2, 088B,B4 ;7014            JMP [LOOP.TB.5]                          ; зацикливание при ошибке, если разрешено
U 08C3, 3E8B,95 ;7015            MOV WR[3] TO LS[ERROR.MASK]              ; запоминание маски в местной памяти
U 08C4, FFB2,15 ;7016            INC LS[ERROR.NUMBER]                     ; ошибка 6
U 08C5, B6F6,95 ;7017            MOV LS[SHIFT.OS(4-0)] TO WR[1]           ; ожидаемые данные
U 08C6, 9D9D,75 ;7018            MEM.REQ[READ.CSR] ADRS[#0] DT[LONG]      ; чтение CSR0 для получения CKBT
U 08C7, 3022,15 ;7019            MOV MEM.DATA TO WR[0]                    ; выборка результата (CKBT)
U 08C8, 0869,3C ;7020            JSR [CHECK.RESULT]                       ; проверка результата
U 08C9, 088B,B4 ;7021            JMP [LOOP.TB.5]                          ; зацикливание при ошибке, если разрешено
U 08CA, 7FF8,15 ;7022            INC LS[OS]                               ; увеличение индекса для следующего набора тестовых данных
U 08CB, B6F8,15 ;7023            MOV LS[OS] TO WR[0]                      ; подготовка проверки, все ли наборы использованы
                ;7024              BIT LS[BIT5] WITH WR[0],               ; проверка, увеличен ли OS до 20(H)
U 08CC, 594A,35 ;7025            DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 08CD, 088B,69 ;7026            JMP.IF[BITS.CLR] TO [REPEAT.TB.5]        ; повторение со следующим набором, если не все
                ;7027                                                     ; использованы
                ;7028    END.TB:
```

```
;7029    .PAGE "ТЕСТ С - тест генерации контрольных битов схемой ECC (модуль МСТ)"
;7030    ;
;7031    ; ОПИСАНИЕ ТЕСТА:
;7032    ;
;7033    ;       Этот тест проверяет схемы генерации контрольных битов в схемах ECC модуля
;7034    ;    МСТ. Используются тестовые данные, которые должны создать известные контроль-
;7035    ;    ные биты, которые записываются в CSR0 для проверки. Используемые наборы долж-
;7036    ;    ны обнаружить любое зависание или закорачивание сигналов на линиях PART SYND
;7037    ;    XX L и BUS ARRAY CB X L. Тест также проверет основные схемы генерации. Путь
;7038    ;    данных следующий: выдается инструкция MEM.REQ с полем MF = MAINT.ECC.DATA.,
;7039    ;    используемый адрес местной памяти содержит 0(H) для более позднего ветвления в
;7040    ;    тесте. Эти данные загружаются в регистр виртуального адреса и выполняется на-
;7041    ;    чальное ветвление в соответствующую программу (см. описание начального ветвле-
;7042    ;    ния в начале этого листинга). По адресу ветвления возбуждаются MEMORY BUSY и
;7043    ;    запрет сдвига, разрешаются биты PA и передатчики модуля центрального процес-
;7044    ;    сора для приема данных из центрального процессора. В следующем цикле строби-
;7045    ;    руются сдвигатели данных для запрета свига и функции предыдущего цикла повто-
;7046    ;    ряются (за исключением того, что сигнал SET ADDR PH не нужен). Следующий цикл
;7047    ;    открывает регистры схемы ECC, разрешает выход сдвигателей данных на шину BUS
;7048    ;    ARRAY, очищает любые ошибки в CSR, поддерживает возбужденным MEMORY BUSY и
;7049    ;    выполняет ветвление по сигналу L CPU DR. Сигнал CPU DR (запрос данных) посту-
;7050    ;    пает, когда центральный процессор выполняет инструкцию MOV. В этом случае
;7051    ;    центральный процессор посылает тестовый набор данных для ПМЛ сдвигателей. Ес-
;7052    ;    ли сигнал L CPU DR еще не выставлен, микропрограмма памяти циклится в ожида-
;7053    ;    нии его. Как только L CPU DR появится, следующий микроцикл возбуждает 2ND MEM
;7054    ;    CYC, открывает регистры схемы ECC, разрешает выходы ПМЛ сдвигателей на BUS
;7055    ;    ARRAY и поддерживает разрешенными передатчики центрального процессора. Уста-
;7056    ;    новленный 2ND MEM CYC поддерживает данные, передаваемые через шину BUS MC из
;7057    ;    центрального процессора, буферизованными внутри ПМЛ сдвигателей (данные были
;7058    ;    загруженны в последнем микроцикле). Следующий цикл поддерживает открытыми ре-
;7059    ;    гистры схемы ECC, держит выходы ПМЛ сдвигателей на BUS ARRAY и ожидает воз-
;7060    ;    буждения сигнала D DATA REC. Сигнал DATA RCVD выставляется центральным про-
;7061    ;    цессором в конце (P2) инструкции MOV. Зацикливание здесь, пока сигнал будет
;7062    ;    возбужден прежде, чем возбуждается MEMORY BUSY, обеспечивает, что центральный
;7063    ;    процессор перейдет на следующую микроинструкцию. Когда сигнал D DATA REC бу-
;7064    ;    дет выставлен, микропрограмма памяти выполнит ветвление по сигналу LVA00 (ко-
;7065    ;    торый был сброшен в начале инструкции центрального процессора MEM.REQ) на со-
;7066    ;    ответствующую тестовую ветвь. Этот микроцикл выставит MEMORY BUSY, откроет
;7067    ;    регистры схемы ECC, разрешит выходы ПМЛ сдвигателей на BUS ARRAY и разрешит
;7068    ;    передачу контрольных битов из CSR1 на линии контрольных битов BUS ARRAY. За-
;7069    ;    тем произойдет ветвление на соответствующую проверку в зависимости от состо-
;7070    ;    яния битов ECC DIS и DIAG CHK в CSR1. CSR1 был загружен до выдачи инструкции
;7071    ;    MEM.REQ. Для этой проверки бит DIAG CHK сброшен, а ECC DIS установлен. Биты
;7072    ;    6-0 содержат дополнение ожидаемых CKBT, которые будут буферизированы в схемах
;7073    ;    ECC до генерации новых CKBT. По месту ветвления возбуждается сигнал GENERATE,
;7074    ;    2ND MEM.CYC сбрасывается, а MEMORY BUSY остается возбужденным. Следующий цикл
;7075    ;    повторяет все функции предыдушего, за исключением того, что сигнал 2ND MEM CYC
;7076    ;    уже сброшен. Следующий цикл повторяет предыдущий цикл и разрешает выход сгене-
;7077    ;    рированных CKBT на BUS ARRAY. Следующий цикл повторяет последний цикл, а так-
;7078    ;    же стробирует данные с шины BUS ARRAY CB в CSR0. Следующие циклы разрешают вы-
;7079    ;    дачу данных ECC на шину BUS MC через сдвигатели данных, выполняют зацикливание
;7080    ;    в ожидании сигнала центрального процессора DATA REQ, а потом сигнала централь-
;7081    ;    ного процессора DATA RCVD, и возврат в холостой цикл. Данные из схемы ECC про-
;7082    ;    веряются, чтобы убедиться, что генерация не изменила их, а CSR0 читается для
;7083    ;    проверки, что были сгенерированы правильные CKBT. Используются следующие тесто-
```

```
;7084    ;    вые данные (CKBT начинаются от CT и уменьшаются до C1): все нули, которые по-
;7085    ;    рождают CKBT=0111100(B), все единицы, которые порождают такие же CKBT, как и все
;7086    ;    нули, данные 80000000(H), которые порождают CKBT=1000011(B), данные 1(H), кото-
;7087    ;    рые порождают CKBT=1100100(B), 2(H), которые порождают CKBT=0100101(B), 4(H),
;7088    ;    которые порождают CKBT=0100110(B), 10(H), которые порождают CKBT=0100000 и дан-
;7089    ;    ные 100(H), которые порождают CKBT=1010100(B). Необходимо заметить, что CKBT
;7090    ;    читаются обратно из схем ECC и инвертируются до записи в CSR0. Фактически полу-
;7091    ;    ченные данные являются дополнением значения CKBT, показанного выше.
;7092    ;
;7093    ; ПРЕДПОЛОЖЕНИЯ:
;7094    ;
;7095    ;    Предполагается, что все предыдущие тесты выполнены успешно.
;7096    ;
;7097    ; ШАГИ ТЕСТА:
;7098    ;
;7099    ;    1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;7100    ;       (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;7101    ;       мяти.
;7102    ;    2. Запись в CSR1 для сброса бита DIAG CHK и установки бита ECC DIS и запись
;7103    ;       дополнительного кода ожидаемых CKBT в биты CSR1 6-0.
;7104    ;    3. Загрузка тестовых данных в WR1 из местной памяти (ожидаемые данные ECC).
;7105    ;    4. Выдача инструкции MEM.REQ с DT = LONGWORD(11) и MF = MAINT.ECC.DATA,
;7106    ;       используя ячейку местной памяти, которая содержит 0(H) в качестве адреса
;7107    ;       (он загружается в регистр виртуального адреса и используется для более
;7108    ;       позднего ветвления).
;7109    ;    5. Выполнение инструкции WRITE.MEM LS с текущими тестовыми данными ECC.
;7110    ;    6. Выполнение инструкции MOVE MEM.DATA TO WR[0] и проверка, что данные не
;7111    ;       изменены.
;7112    ;    7. Изменение маски ошибки для проверки битов 6-0, загрузка в WR1 ожидаемого
;7113    ;       результата для CKBT, чтение CSR0 и проверка.
;7114    ;    8. Повторение шагов с 2 по 7 со следующими тестовыми данными (шаг 6 выполняет-
;7115    ;       ся только с тестовыми данными - все единицы и все нули).
;7116    ;
;7117    ; ОШИБКИ:
;7118    ;
;7119    ;    ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными являются данные ECC, когда маска
;7120    ;    ошибки - все нули. Ожидаемыми и полученными данными являются контрольные биты,
;7121    ;    сгенерированные схемами ECC, когда маска ошибки - FFFFFF80.
;7122    ;
;7123    ;    ошибка 1 - функция генерации CKBT изменяет данные, буферизованные в схемах
;7124    ;               ECC. Данные - все нули.
;7125    ;    ошибка 2 - неправильно работает генерация CKBT в схемах ECC или управление с данными
;7126    ;               ECC - все нули.
;7127    ;    ошибка 3 - функция генерации CKBT изменяет данные, буферизованные в схемах
;7128    ;               ECC. Данные - все единицы.
;7129    ;    ошибка 4 - неправильно работает генерация CKBT в схемах ECC или управление с данными
;7130    ;               ECC - все единицы.
;7131    ;    ошибка 5 - неправильно работает генерация CKBT в схемах ECC или управление с данными
;7132    ;               ECC - 80000000(H).
;7133    ;    ошибка 6 - неправильно работает генерация CKBT в схемах ECC с данными ECC - 1(H).
;7134    ;    ошибка 7 - неправильно работает генерация CKBT в схемах ECC с данными ECC - 2(H).
;7135    ;    ошибка 8 - неправильно работает генерация CKBT в схемах ECC с данными ECC - 4(H).
;7136    ;    ошибка 9 - неправильно работает генерация CKBT в схемах ECC с данными ECC - 10(H).
;7137    ;    ошибка A - неправильно работает генерация CKBT в схемах ECC с данными ECC - 100(H).
;7138    ;
```

```
;7139    ; НАЛАДКА:
;7140    ;
;7141    ;       ОШИБКА 1 - Эта ошибка указывает на возможную неисправность микросхемы ECC,
;7142    ;    в которой обнаружен ошибочный бит, или на неправильный сигнал CORR DIS L(C29).
;7143    ;    Во время зацикливания по ошибке необходимо просмотреть сигнал CORR DIS L(C29),
;7144    ;    поступающий на микросхему ECC, связанную с ошибочным битом данных. Этот сиг-
;7145    ;    нал не должен становиться низким во время зацикливания в этом тесте. Если сиг-
;7146    ;    нал остается высоким, по-видимому, микросхема ECC неисправна.
;7147    ;
;7148    ;       ОШИБКА 2 - Эта ошибка указывает неправильную генерацию битов CKBT в ECC или
;7149    ;    неправильное управление схем ECC. Если пошаговый режим МСТ невозможен, для прос-
;7150    ;    мотра интересующих сигналов должно использоваться зацикливание при ошибке. Если
;7151    ;    пошаговый режим МСТ возможен, во-первых, подведите центральный процессор на инст-
;7152    ;    рукцию WRITE.MEM, подготовьте пошаговый режим МСТ, остановите центральный процес-
;7153    ;    сор на инструкции MOV MEM.DATA TO WRI0] и остановите микропрограмму памяти на
;7154    ;    четвертом микроцикле микропрограммы GENERATE (4 цикла после ветвления по сигна-
;7155    ;    лам ECC.DIS и DIAG.CHK). Необходимо проверить высокий уровень на входе GENERATE
;7156    ;    H(C27) микросхемы ECC для младшего слова. Этот сигнал поступает прямо с ПЗУ уп-
;7157    ;    равляющей памяти. Если он правильный, необходимо проверить высокий уровень сиг-
;7158    ;    нала на входе OUTPUT CB/SYN H(C28) микросхемы ECC для старшего слова. Также не-
;7159    ;    обходимо проверить ножку 26 (OUT CB BUS), 23 (HI WORD) и 33 (GEN), чтобы убе-
;7160    ;    диться, что они заземлены, а ножка 24 (LAT CB) подсоединена к высокому уровню.
;7161    ;    Также необходимо проверить подсоединение к высокому уровню ножки 34 (OUT SYND)
;7162    ;    микросхемы ECC для младшего слова. Наконец, необходимо проверить высокий уровень
;7163    ;    сигнала CSR CB EN L(C43), который разрешает выход CSR1 на шину BUS ARRAY CB. Ес-
;7164    ;    ли все эти входы правильные, необходимо проверить сигналы PART SYND XX L на вхо-
;7165    ;    де ECC старшего слова. Все они должны быть единицы. Если нет, микросхема ECC
;7166    ;    младшего слова неисправна, или неисправна сама линия связи. Если здесь правиль-
;7167    ;    но, микросхема ECC старшего слова должна выдавать правильные CKBT на шину BUS
;7168    ;    ARRAY CB. Для этой ошибки правильными CKBT являются 1000011(B) от CB1 L соот-
;7169    ;    ветственно. Иначе микросхема ECC старшего слова неисправна.
;7170    ;
;7171    ;       ОШИБКА 3 - См. ошибку 1
;7172    ;
;7173    ;       ОШИБКА 4 - Так же, как и при ошибке 2. Входные данные разные, но все входы
;7174    ;                  управления и выходы CKBT идентичны.
;7175    ;
;7176    ;       ОШИБКА 5 - Этот набор должен дополнить исходные CKBT. Линии PART SYND X L
;7177    ;                  из младшего слова еще содержат все единицы, но CKBT из старшего
;7178    ;                  слова на шине BUS ARRAY CB должны быть 0111100. Все сигналы уп-
;7179    ;                  равления такие же, как при ошибке 2.
;7180    ;
;7181    ;       ОШИБКА 6 - Эта ошибка указывает на возможную неисправность микросхем ECC
;7182    ;                  или линий PART SYND X L между двумя микросхемами ECC. Используя
;7183    ;                  пошаговый режим, как и при ошибке 2, проверьте наличие 010111(B)
;7184    ;                  на линиях PART SYND X L OT T L до 1 L соответственно. Выходы
;7185    ;                  старшего слова на шине BUS ARRAY CB должны содержать 0011011(B)
;7186    ;                  от CBT до CB1 соответственно.
;7187    ;
;7188    ;       ОШИБКА 7 - Эта ошибка указывает на возможную неисправность микросхем ECC
;7189    ;                  или линий PART SYND X L между двумя микросхемами ECC. Исполь-
;7190    ;                  зуя пошаговый режим, как и при ошибке 2, проверьте наличие
;7191    ;                  1100110(B) на PART SYND H L от T L до 1 L соответственно. Выхо-
;7192    ;                  ды старшего слова на шине BUS ARRAY CB должны содержать
;7193    ;                  1011010(B) от CBT до CB1 соответственно.
```

```
                    ;7194   ;
                    ;7195   ;       ОШИБКА 8 - Эта ошибка указывает на возможную неисправность микросхем ECC
                    ;7196   ;                  или PART SYND X L между двумя микросхемами ECC. Используя поша-
                    ;7197   ;                  говый режим, как и при ошибке 2, проверяется наличие 1100101(B)
                    ;7198   ;                  на линиях PART SYND X L от T L до 1 L соответственно. Выходы
                    ;7199   ;                  старшего слова на шине BUS ARRAY CB должны содержать 1011001(B)
                    ;7200   ;                  от CBT до CB1 соответственно.
                    ;7201   ;
                    ;7202   ;       ОШИБКА 9 - Эта ошибка указывает на возможную неисправность микросхем ECC
                    ;7203   ;                  или линий PART SYND X L между двумя микросхемами ECC. Используя
                    ;7204   ;                  пошаговый режим, как и при ошибке 2, проверьте наличие 1100011(B)
                    ;7205   ;                  на линиях PART SYND X L от T L до 1 L соответственно. Выходы
                    ;7206   ;                  старшего слова на шине BUS ARRAY CB должны содержать 1011111(B)
                    ;7207   ;                  от CBT до CB1 соответственно.
                    ;7208   ;
                    ;7209   ;       ОШИБКА A - Эта ошибка указывает на возможную неисправность микросхем ECC.
                    ;7210   ;                  Для определения, которая из них неисправна, испльзуется пошаговый
                    ;7211   ;                  режим, как и при ошибке 2, и проверяется наличие 0010111(B) на
                    ;7212   ;                  линиях PART SYND X L от T L до 1 L соответственно. Выходы старшего
                    ;7213   ;                  слова на шине BUS ARRAY CB должны содержать 1011001(B) от CBT до
                    ;7214   ;                  CB1 соответственно.
                    ;7215    T.C:
U 08CE, B65E,15     ;7216            MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR[0] для слова управления и
                    ;7217                                                       ; состояния
U 08CF, 3E80,15     ;7218            MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                    ;7219                                                       ; 15 указывает начало теста консольному процессору
U 08D0, 10E0,15     ;7220            MISC [SET.CP.ATTN]                         ; выдача сигнала CPU ATTN консольному процессору
                    ;7221    WAIT.TC.0:
U 08D1, 088D,14     ;7222            JMP [WAIT.TC.0]                            ; зацикливание при ожидании ответа консольного
                    ;7223                                                       ; процессора
U 08D2, 0A1A,AC     ;7224            JSR [SETUP.1]                              ; установка масок, кода модуля и др.
U 08D3, B700,15     ;7225            MOV LS[CKBTS.01111100] TO WR[0]             ; формирование дополнения ожидаемых CKBT для записи в
                    ;7226                                                       ; CSR1
U 08D4, 4772,15     ;7227            BIS LS[ECC.DIS] TO WR[0]                   ; установка бита ECC DIS B CSR1 для ветвления
U 08D5, 8A17,FC     ;7228            JSR [WRITE.CSR1]                           ; запись CSR1 с данными из WR0
U 08D6, B701,95     ;7229            MOV LS[CKBTS.01111100] TO WR[3]             ; запоминание дополнения ожидаемых CKBT B WR3
U 08D7, A0C7,95     ;7230            COM WR[3]                                  ; WR3 содержит CKBT (CKBT из схемы ECC инвертируются,
                    ;7231                                                       ; когда читаются)
U 08D8, 3713,15     ;7232            MOV LS[#FFFFFF80] TO WR[2]                 ; установка маски ошибки для 7 битов
                    ;7233    LOOP.TC.1:
U 08D9, B640,15     ;7234            MOV LS[#1] TO WR[0]                        ; 1 в WR0
U 08DA, BE82,15     ;7235            MOV WR[0] TO LS[ERROR.NUMBER]              ; установка номера ошибки в местной памяти
U 08DB, B69C,95     ;7236            MOV LS[ZERO] TO WR[1]                      ; ожидаемые данные
U 08DC, E58A,15     ;7237            CLR LS[ERROR.MASK]                         ; установка маски ошибки для проверки всех битов
U 08DD, 9D9C,F5     ;7238            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG]  ; подготовка записи данных ECC
U 08DE, B29C,15     ;7239            WRITE.MEM LS[#0]                           ; запись нулевых данных ECC в схемы ECC
U 08DF, 3022,15     ;7240            MOV MEM.DATA TO WR[0]                      ; проверка, что данные ECC не изменены
U 08E0, 0869,3C     ;7241            JSR [CHECK.RESULT]                         ; проверка результата
U 08E1, 888D,94     ;7242            JMP [LOOP.TC.1]                            ; зацикливание при ошибке, если разрешено
U 08E2, FFB2,15     ;7243            INC LS[ERROR.NUMBER]                       ; ошибка 2
U 08E3, 2006,95     ;7244            MOV WR[3] TO WR[1]                         ; ожидаемые данные для CKBT
U 08E4, BE88,15     ;7245            MOV WR[2] TO LS[ERROR.MASK]                ; изменение маски ошибки для проверки 7 битов(6-0)
U 08E5, 9D9D,75     ;7246            MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]      ; чтение сгенерированных CKBT из CSR0
U 08E6, 3022,15     ;7247            MOV MEM.DATA TO WR[0]                      ; выборка данных
U 08E7, 0869,3C     ;7248            JSR [CHECK.RESULT]                         ; проверка результата
```

```
U 08E8, 888D,94 ;7249            JMP [LOOP.TC.1]                          ; зацикливание при ошибке, если разрешено
                ;7250    LOOP.TC.3:
U 08E9, 36C0,15 ;7251            MOV LS[#3(H)] TO WR[0]                   ; загрузка 3 в WR0
U 08EA, BE82,15 ;7252            MOV WR[0] TO LS[ERROR.NUMBER]            ; установка номера ошибки в местной памяти
U 08EB, 369E,95 ;7253            MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 08EC, E58A,15 ;7254            CLR LS[ERROR.MASK]                       ; установка маски ошибки для проверки всех битов
U 08ED, 9D9C,F5 ;7255            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи данных ECC
U 08EE, 329E,15 ;7256            WRITE.MEM LS[ONES]                       ; запись единичных данных ECC в схему ECC
U 08EF, 3022,15 ;7257            MOV MEM.DATA TO WR[0]                    ; проверка, что данные ECC не изменены
U 08F0, 0869,3C ;7258            JSR [CHECK.RESULT]                       ; проверка результата
U 08F1, 888E,94 ;7259            JMP [LOOP.TC.3]                          ; зацикливание при ошибке, если разрешено
U 08F2, FF82,15 ;7260            INC LS[ERROR.NUMBER]                     ; ошибка 4
U 08F3, 2006,95 ;7261            MOV WR[3] TO WR[1]                       ; ожидаемые данные для CKBT
U 08F4, BE8B,15 ;7262            MOV WR[2] TO LS[ERROR.MASK]              ; изменение маски ошибки для проверки 7 битов (6-0)
U 08F5, 9D9D,75 ;7263            MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]    ; чтение сгенерированных CKBT из CSR0
U 08F6, 3022,15 ;7264            MOV MEM.DATA TO WR[0]                    ; выборка данных
U 08F7, 0869,3C ;7265            JSR [CHECK.RESULT]                       ; проверка результата
U 08F8, 888E,94 ;7266            JMP [LOOP.TC.3]                          ; зацикливание при ошибке, если разрешено
U 08F9, B67F,15 ;7267            MOV LS[BIT31] TO WR[2]                   ; данные B0000000 для записи в матрицу памяти
U 08FA, 3703,95 ;7268            MOV LS[CKBTS.1000011] TO WR[3]           ; дополнение ожидаемых CKBT
U 08FB, 0890,CC ;7269            JSR [CHECK.TC.CKBT]                      ; проверка генерации CKBT при ошибке 5
U 08FC, 3641,15 ;7270            MOV LS[#1] TO WR[2]                      ; данные 1(H) для записи в матрицу памяти
U 08FD, 3705,95 ;7271            MOV LS[CKBTS.1100100] TO WR[3]           ; дополнение ожидаемых CKBT
U 08FE, 0890,CC ;7272            JSR [CHECK.TC.CKBT]                      ; проверка генерации CKBT при ошибке 6
U 08FF, B643,15 ;7273            MOV LS[#2] TO WR[2]                      ; данные 2(H) для записи в матрицу памяти
U 0900, B707,95 ;7274            MOV LS[CKBTS.0100101] TO WR[3]           ; дополнение ожидаемых CKBT
U 0901, 0890,CC ;7275            JSR [CHECK.TC.CKBT]                      ; проверка генерации при ошибке 7
U 0902, B645,15 ;7276            MOV LS[#4] TO WR[2]                      ; данные 4(H) для записи в матрицу памяти
U 0903, 3709,95 ;7277            MOV LS[CKBTS.0100110] TO WR[3]           ; дополнение ожидаемых CKBT
U 0904, 0890,CC ;7278            JSR [CHECK.TC.CKBT]                      ; проверка генерации CKBT при ошибке 8
U 0905, B649,15 ;7279            MOV LS[#10] TO WR[2]                     ; данные 10(H) для записи в матрицу памяти
U 0906, B70B,95 ;7280            MOV LS[CKBTS.0100000] TO WR[3]           ; дополнение ожидаемых CKBT
U 0907, 0890,CC ;7281            JSR [CHECK.TC.CKBT]                      ; проверка генерации CKBT при ошибке 9
U 0908, B651,15 ;7282            MOV LS[#100] TO WR[2]                    ; данные 100(H) для записи в матрицу памяти
U 0909, B70D,95 ;7283            MOV LS[CKBTS.1010100] TO WR[3]           ; дополнение ожидаемых CKBT
U 090A, 0890,CC ;7284            JSR [CHECK.TC.CKBT]                      ; проверка генерации CKBT при ошибке A
U 090B, 8891,84 ;7285            JMP [END.TC]                             ; тест выполнен
                ;7286     ;
                ;7287     ; Подпрограммы теста
                ;7288     ;
                ;7289    CHECK.TC.CKBT:
U 090C, FF82,15 ;7290            INC LS[ERROR.NUMBER]                     ; увеличение номера ошибки
U 090D, A0C7,95 ;7291            COM WR[3]                                ; WR3 содержит ожидаемые CKBT
U 090E, BE11,15 ;7292            MOV WR[2] TO LS[T8]                      ; запоминание данных для записи в матрицу памяти
                ;7293    LOOP.TC.5.A:
U 090F, 2006,95 ;7294            MOV WR[3] TO WR[1]                       ; ожидаемые данные для CKBT
U 0910, 9D9C,F5 ;7295            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка для записи данных ECC
U 0911, 3210,15 ;7296            WRITE.MEM LS[T8]                         ; запись данных ECC в схемы ECC
U 0912, 3022,15 ;7297            MOV MEM.DATA TO WR[0]                    ; завершение микроцикла, но проверка результата не
                ;7298                                                     ; выполняется
U 0913, 9D9D,75 ;7299            MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]    ; чтение сгенерированных CKBT из CSR0
U 0914, 3022,15 ;7300            MOV MEM.DATA TO WR[0]                    ; выборка данных
U 0915, 0869,3C ;7301            JSR [CHECK.RESULT]                       ; проверка результата
U 0916, 8890,F4 ;7302            JMP [LOOP.TC.5.A]                        ; зацикливание при ошибках с 5 по A, если разрешено
U 0917, 5B00,14 ;7303            RETURN                                   ; выполнено
```

```
                ;7304    END.TC:
```

```
;7305    .PAGE "ТЕСТ D - тест ошибки ECC (без коррекции) (модули MCT и DAP)"
;7306    ;
;7307    ; ОПИСАНИЕ ТЕСТА:
;7308    ;
;7309    ;       Этот тест проверяет, что схемы ECC правильно обнаруживают ошибку одного или
;7310    ;    двух битов и что сигналы RDS, CRD и ERR SUM вырабатывается правильно. Также
;7311    ;    проверяется схема пропуска микроинструкции в модуле центрального процессора при
;7312    ;    наличии ошибки памяти. Бит INH REP CRD (запрета сообщения о корректируемых ошиб-
;7313    ;    ках в данных) в CSR1 переключается при ошибке CRD для гарантии, что он запреща-
;7314    ;    ет установку бита CRD в CSR1. Путь данных такой же, как и в тесте шины BUS ARRAY,
;7315    ;    регистров схемы ECC, включая тест генерации CKBT (см. описание пути данных этих
;7316    ;    тестов). Используемые тестовые данные - все нули в качестве данных ECC со сле-
;7317    ;    дующими наборами CKBT (от CB T до CB 1 соответственно): 0111100(B), который не
;7318    ;    создает ошибки, 1100100 (B), который должен создать ошибку одиночного бита (CRD)
;7319    ;    и 1111101 (B), который должен создать ошибку двух битов (RDS). Заметим, что CKBT
;7320    ;    записываются и читаются как дополнение данных, что обусловлено аппаратурой.
;7321    ;
;7322    ; ПРЕДПОЛОЖЕНИЯ:
;7323    ;
;7324    ;    Предполагается, что все предыдущие тесты выполнены успешно. Здесь проверяет-
;7325    ;    ся схема пропуска микроинструкции центрального процессора при наличии ошибки
;7326    ;    памяти (ERR SUM), так что модуль центр.процессора может быть причиной неис-
;7327    ;    правности.
;7328    ;
;7329    ; ШАГИ ТЕСТА:
;7330    ;
;7331    ;    1) Установка маски ошибки, номера ошибки и номера молуля в местной памяти
;7332    ;       (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;7333    ;       мяти.
;7334    ;    2) Запись данных в CSR1 для установки битов DIAG CHK и ECC DIS. Также загруз-
;7335    ;       ка CKBT (битов 0-6), которые не создают ошибки (0111100(B) от CBT до CB1
;7336    ;       соответственно). Данные CKBT должны быть инвертированны до записи в CSR.
;7337    ;    3) Очистка WR[1]. Это ожидаемые данные со схемы ECC.
;7338    ;    4) Выдача инструкции MEM.REQ с DT=LONGWORD (11) и MF=MAINT.ECC.DATA, исполь-
;7339    ;       зуя ячейку местной памяти, которая содержит 0(H) в качестве адреса (он
;7340    ;       загружается в регистр виртуального адреса и используется для более поздне-
;7341    ;       го ветвления).
;7342    ;    5) Выполнение инструкции WRITE.MEM LS с данными - все нули.
;7343    ;    6) Выполнение инструкции MOV MEM DATA TO WR[0]. Проверка результата на все
;7344    ;       нули (без изменения).
;7345    ;    7) Изменение маски ошибки для проверки битов 14-23 и 25-31 регистра CSR1.
;7346    ;       Загрузка WR1 ожидаемыми данными в регистре CSR1.
;7347    ;    8) Чтение CSR1 для проверки установленных битов DIAG CHK и ECC DIS и очищен-
;7348    ;       ных других битов.
;7349    ;    9) Изменение кода модуля для указания при распечатке модуля центрального
;7350    ;       процессора. Выполнение функции SKIP (пропуска) при отсутствии ошибки па-
;7351    ;       мяти (ERR SUM отсуствует) и проверка пропуска.
;7352    ;   10) Запись CSR1 данными CKBT, которые формируют ошибку одиночного бита с
;7353    ;       данными - все нули (CKBT=1100100(B) от CBT до CB1 соответственно). CKBT
;7354    ;       должны быть инвертированны до записи в CSR).
;7355    ;   11) Восстановление маски ошибки для данных из 32 битов и восстановление номера
;7356    ;       модуля для распечатки только модуля MCT.
;7357    ;   12) Очистка WR[1]. Это ожидаемые данные из схемы ECC. Выдача инструкции
;7358    ;       MEM.REQ как описано в шаге 4.
;7359    ;   13) Выполнение инструкции WRITE.MEM LS с тестовыми данными - нули.
```

```
;7360    ;  14) Выполнение инструкции MOV MEM.DATA TO WR[0]. Проверка результата на все
;7361    ;      нули (без изменения).
;7362    ;  15) Изменение маски ошибки для проверки битов 14-23 и 25-31 CSR1. Загрузка
;7363    ;      WR1 ожидаемыми данными регистра CSR1.
;7364    ;  16) Чтение CSR1 и проверка установки битов CRD (бит 30), DIAG CHK и ECC DIS
;7365    ;      и очистки других битов.
;7366    ;  17) Изменение кода модуля, чтобы иметь модуль DAP для распечатки. Выполнение
;7367    ;      функции SKIP (пропуска) при отсутствии ошибки памяти (ERR SUM отсутствует)
;7368    ;      и проверка, что пропуска не было. Очистка CSR2 и проверка, что бит CRD в
;7369    ;      CSR1 сброшен.
;7370    ;  18) Повторение шагов с 10 по 17 с CKTB, которые формируют двойную ошибку.
;7371    ;      проверка установки битов RDS (бит 31), DIAG CHK, ECC DIS и очистки других
;7372    ;      битов. CKTB=1111101(B) от CB 7 до CB 1 соответственно. CKBT должны быть
;7373    ;      инвертированы до записи в CSR.
;7374    ;  19) Повторение шагов с 10 по 17 с CKTB для одиночной ошибки, при установленном
;7375    ;      INH REP CRD в CSR1. Проверка, что пропуска не было и что бит CRD в CSR1
;7376    ;      не установлен.
;7377    ;  20) Проверка, что сигнал ERR SUM доходит до центрального процессора достаточ-
;7378    ;      но быстро.
;7379    ;
;7380    ; ОШИБКИ:
;7381    ;
;7382    ;   ошибка 1 - данные ECC были изменены при использовании нулевых данных и CKBT
;7383    ;              которые не создают ошибки. Коррекция также была запрещена.
;7384    ;   ошибка 2 - биты ошибок CSR1 не были сброшены, хотя данные и CKBT не должны
;7385    ;              формировать никакой ошибки.
;7386    ;   ошибка 3 - неправильно работает функция центрального процессора - пропуск
;7387    ;              микроинструкции при отсутствии ошибки памяти.
;7388    ;   ошибка 4 - данные ECC были изменены при использовании нулевых данных и CKBT
;7389    ;              для ошибки в одиночном бите, даже когда коррекция была запрещена.
;7390    ;   ошибка 5 - биты ошибок CSR1 не установлены правильно при ошибке в одиночном
;7391    ;              бите (CRD).
;7392    ;   ошибка 6 - функция центрального процессора - пропуск микроинструкции при от-
;7393    ;              сутствии ошибок памяти выполняет пропуск при ошибке одиночного бита.
;7394    ;   ошибка 7 - запись в CSR2 не сбрасывает бита CRD в CSR1.
;7395    ;   ошибка 8 - данные ECC были изменены при использовании нулевых данных и CKBT
;7396    ;              для формирования ошибки двух битов, даже когда коррекция была
;7397    ;              запрещена.
;7398    ;   ошибка 9 - биты ошибок CSR1 не установлены правильно при ошибке двух битов (RDS).
;7399    ;   ошибка A - функция центрального процессора - пропуск при отсутствии ошибки па-
;7400    ;              мяти выполняет пропуск при ошибке двух битов.
;7401    ;   ошибка B - запись в CSR2 не сбрасывает бита RDS в CSR1.
;7402    ;   ошибка C - не все биты ошибок CSR1 сбрасываются при получении ошибки одиноч-
;7403    ;              ного бита (CRD) с установленным INH REP CRD.
;7404    ;   ошибка D - функция центрального процессора - пропуск при отсутствии ошибки па-
;7405    ;              мяти выполняет пропуск при ошибке одиночного бита с установленным
;7406    ;              INH REP CRD (бит ECC DIS был возбужден, так что ошибка памяти так-
;7407    ;              же должна быть возбуждена).
;7408    ;   ошибка E - пропуск при отсутствии ошибки памяти выполняет пропуск даже когда
;7409    ;              сигнал ERR SUM был возбужден на MCT.
;7410    ;
;7411    ; НАЛАДКА:
;7412    ;
;7413    ;      ОШИБКА 1 - Эта ошибка указывает на возможную неисправность схем ECC. По-
;7414    ;   дозревается схема ECC, связанная с неправильным битом в сообщении об ошибке.
```

```
;7415   ;
;7416   ;       ОШИБКА 2 - Эта ошибка указывает, что или биты ошибок CSR не сбрасываются,
;7417   ;    или схемы ECC формируют неправильный сигнал ошибок, или неправильно работает
;7418   ;    схема CSR1. С остановом по ошибке сигналы ошибки из схемы ECC можно проверить
;7419   ;    после останова теста по ошибке. Необходимо проверить сигналы ERROR L и SINGLE
;7420   ;    ERROR L, сформированные схемой ECC старшего слова. Оба сигнала должны быть
;7421   ;    высокими. Если нет, повидимому, неисправна одна из микросхем ECC. Если эти
;7422   ;    выходы правильные, необходимо запустить снова от холостого цикла памяти, раз-
;7423   ;    решить пошаговый режим MCT (если возможно), остановить центральный процессор
;7424   ;    на инструкции MEM.REQ и остановить микропрограмму памяти на втором цикле
;7425   ;    после начального ветвления (цикл CLR ERR). Необходимо проверить низкий уро-
;7426   ;    вень сигнала CLR ERR L (C40) на ПМЛ CSR1 и высокий уровень сигнала CPU ERR
;7427   ;    SUM CLK H (C48). Далее перейти на следующий цикл и проверить, что сигнал CPU
;7428   ;    ERR SUM CLK H становится низким. Если это выполняется, ПМЛ должна сбрасываться.
;7429   ;    Если любой из этих выходов неправильный (CSR 14-23), когда тест выполняется
;7430   ;    неуспешно, по-видимому, ПМЛ неисправна, если входы правильные (выходы разре-
;7431   ;    шаются только когда выполняется чтение CSR, поэтому они не могут быть прос-
;7432   ;    мотрены непосредственно здесь). Если пошаговый режим MCT невозможен, тогда
;7433   ;    необходимо зациклить по ошибке и проверить. Если биты 30 или 31 неправильные,
;7434   ;    необходимо проверить низкий уровень сигналов L RDS H и L CRD H на выходе
;7435   ;    ПМЛ УПР.ОШИБКАМИ ДАННЫХ после микроцикла памяти CLR ERR. Во время этого цик-
;7436   ;    ла должны быть высокие уровни на входах CPH/UBL и CSR ERR SUM CLK H (C42).
;7437   ;    Сигнал CPH/UBL поступает из ПМЛ УПР.ПРЕДВЫБОРКОЙ и высокий уровень уже
;7438   ;    был проверен в предыдущем тесте. Если эти сигналы правильные, тогда ПМЛ
;7439   ;    неисправна, если выход не содержит низкого уровня. Если бит 30 или 31
;7440   ;    неправильный, возможно, что буфер или соединения с микросхемой неисправны,
;7441   ;    когда выполняется обратное чтение CRD и RDS на шину BUS MC. Это можно прове-
;7442   ;    рить после останова по ошибке.
;7443   ;
;7444   ;       ОШИБКА 3 - Эта ошибка указывает на возможную неисправность схем формирования
;7445   ;    сигнала ERR SUM на модуле MCT или неправильный пропуск при отсутствии ошибки
;7446   ;    памяти (ERR SUM отсутствует) на модуле центрального процессора. После останова
;7447   ;    по ошибке необходимо проверить сигнал ERR SUM H на модуле MCT. Если этот сигнал
;7448   ;    низкий, по-видимому, неисправность есть в модуле центр.процессора или в ген-
;7449   ;    монтажном соединении. В модуле DAP необходимо проверить низкий уровень в
;7450   ;    соединении ERR SUM H с ПМЛ УПР.МИАСС ЕКВЕНСЕРОМ. Если здесь правильно, то по-
;7451   ;    видимому, неисправна ПМЛ, но должно быть проверено наличие 1D(H) на входах
;7452   ;    с CSR 04 H по CSR 00 H и высокий уровень на входе JUMP INST L. Если сигнал
;7453   ;    ERR SUM H на модуле MCT высокий, тогда необходимо проследить в обратном нап-
;7454   ;    равлении до ПМЛ. Если причиной неисправности является CSR1A или CSR1B, тогда
;7455   ;    наверное, эта ПМЛ неисправна. Если сигнал DATA ERR L низкий, тогда необходимо
;7456   ;    проверить выход ПМЛ УПР.ОШИБКАМИ ДАННЫХ. Если выход низкий, то по-видимому,
;7457   ;    ПМЛ УПР.ОШИБКАМИ ДАННЫХ неисправна.
;7458   ;
;7459   ;       ОШИБКА 4 - Скорее всего подозревается сама схема ECC. Необходимо проверить
;7460   ;    низкий уровень сигнала COR DIS L, поступающего на обе микросхемы.
;7461   ;
;7462   ;       ОШИБКА 5 - Эта ошибка может быть вызвана схемами ECC, неисправностями в схе-
;7463   ;    мах ветвления или в схемах CSR1. Маловероятно, что неправильными будут биты
;7464   ;    14-23. Это первая проверка. Если неправильные биты 30 или 31, прежде всего необ-
;7465   ;    ходимо проверить сигналы ERROR L и SINGLE ERR L, сформированные микросхемой ECC
;7466   ;    старшего слова. Оба эти сигналы должны быть низкими. Если нет, по-видимому, не-
;7467   ;    исправна одна из микросхем ECC. Если эти сигналы правильные, необходимо прове-
;7468   ;    рить их на входе ПМЛ УПР.ОШИБКАМИ ДАННЫХ. Если здесь правильно и пошаговый ре-
;7469   ;    жим MCT возможен, центральный процессор следует остановить на инструкции WRITE
```

```
;7470    ;   MEM.LS. Микропрограмма памяти должна циклиться, ожидая DATA RCVD из центрального
;7471    ;   процессора. Необходимо разрешить пошаговый режим MCT, тогда еще раз остановить
;7472    ;   центральный процессор на инструкции MOV MEM.DATA TO WR[0]. Теперь следует оста-
;7473    ;   новить микропрограмму памяти через 4 цикла на цикле ветвления по ошибке
;7474    ;   (BEN0=ERROR L). Необходимо проверить низкий уровень сигнала ERROR L на входе
;7475    ;   мультиплексора BEN0. Также необходимо проверить наличие 011(B) на линиях выбор-
;7476    ;   ки A2,A1 и A0 соответственно. Если здесь правильно, необходимо выполнить еще
;7477    ;   один цикл MCT и микропрограмма должна оказаться на микроцикле, который возбуж-
;7478    ;   дает сигнал CSR ERR ADDR CLK A L (C45). Если ветвление выполняется неправильно,
;7479    ;   неисправен мультиплексор BEN0. Иначе необходимо проверить входы ПМЛ УПР.ОШИБКА-
;7480    ;   МИ данных следующим образом: сигналы CSR ERR ADDR CLK A L, CSR ERR SUM CLK H
;7481    ;   SINGLE ERR L и ERROR L все должны быть низкими, сигнал CPH/UBL должен быть
;7482    ;   высоким, INH REP CRD H должен быть низким и WR CSR L должен быть высоким.
;7483    ;   Если эти входы правильные, тогда L RDS H должен быть низким и L CRD H должен
;7484    ;   быть высоким. ПМЛ неисправна, если эти выходы неправильные. Если здесь пра-
;7485    ;   вильно, необходимо проверить, что эти сигналы поступают на шинный формирователь.
;7486    ;   Шинный формирователь неисправен, если до его входов все правильно. Еще одной
;7487    ;   возможностью является то, что сигналы L CRD H и L RDS H изменяются позже в мик-
;7488    ;   ропрограмме. Если ПМЛ УПР.ОШИБКАМИ ДАННЫХ исправна, это не должно произойти,
;7489    ;   пока не возбуждены сигналы WR CSR или CSR ERR SUM CLK. Если пошаговый режим MCT
;7490    ;   не возможен, необходимо зациклить по ошибке и, используя для синхронизации сиг-
;7491    ;   нал CSR ERR ADDR CLK A L, проверить входы ПМЛ УПР.ОШИБКАМИ ДАННЫХ. Если CSR
;7492    ;   ERR ADDR CLK A L не возбуждается, схема ветвления может работать неправильно.
;7493    ;
;7494    ;       ОШИБКА 6 - Эта ошибка указывает на возможную неисправность схем ERR SUM на
;7495    ;   модуле MCT или схем ветвления по ERR SUM на модуле DAP. После останова по ошиб-
;7496    ;   ке необходимо проверить высокий уровень сигнала ERR SUM на модуле MCT. Если он
;7497    ;   правильный, неисправность имеется на модуле DAP или в генмонтажном соединении.
;7498    ;   Необходимо проверить высокий уровень сигнала ERR SUM H на входе ПМЛ УПР.МИКРО-
;7499    ;   СЕКВЕНСЕРОМ. Если здесь правильно, по-видимому, ПМЛ УПР.МИАСС ЕКВЕНСЕРОМ не-
;7500    ;   исправна, но для гарантии входы можно проверить, остановив центральный процес-
;7501    ;   сор на инструкции пропуска при отсутствии ошибки памяти (SKIP IF MEM.REF.OK).
;7502    ;   входы от CSR 04 до CSR 00 должны содержать 1D(H) и вход JUMP INST L должен
;7503    ;   быть высоким.
;7504    ;       Если сигнал ERR SUM H на модуле MCT низкий, необходимо проверить сигнал DATA
;7505    ;   ERR L на одном инвертирующем входе вентиля "ИЛИ". Он должен быть низким здесь и
;7506    ;   на выходе ПМЛ УПР.ОШИБКАМИ ДАННЫХ. Если выход ПМЛ неправильный, необходимо про-
;7507    ;   верить входы ПМЛ УПР.ОШИБКАМИ ДАННЫХ, как описано выше при ошибке 5.
;7508    ;
;7509    ;       ОШИБКА 7 - Эта ошибка указывает возможную неисправность ПМЛ УПР.ОШИБКАМИ
;7510    ;   ДАННЫХ или входов WR CSR L (C46) и CSR ERR SUM CLK H. Необходимо проверить
;7511    ;   низкий уровень входа WR CSR L и высокий уровень входа CSR ERR SUM CLK во вре-
;7512    ;   мя инструкции JSR [CLEAR.CSR2]. Если эти входы правильные, по-видимому, неисп-
;7513    ;   равна ПМЛ.
;7514    ;
;7515    ;       ОШИБКА 8 - То же, что и для ошибки 4.
;7516    ;
;7517    ;       ОШИБКА 9 - Эта ошибка указывает на возможную неисправность схем ECC или
;7518    ;   схем ПМЛ УПР.ОШИБКАМИ ДАННЫХ. После останова по ошибке необходимо проверить
;7519    ;   низкий уровень на выходе ERROR L и высокий уровень на выходе SINGLE ERR L
;7520    ;   схемы ECC старшего слова. Если они неправильные, по-видимому, неисправна од-
;7521    ;   на из схем ECC. Иначе необходимо проверить входы ПМЛ УПР.ОШИБКАМИ ДАННЫХ.
;7522    ;   Также можно проверить выходы на возможную неисправность схемы шинного форми-
;7523    ;   рователя. Если выходы неправильные, необходимо проверить другие входы ПМЛ
;7524    ;   УПР.ОШИБКАМИ ДАННЫХ в пошаговом режиме или при зацикливании, как описано выше
```

```
                    ;7525   ;   при ошибке 5. Отличие содержится только в том, что сигнал SINGLE ERROR L дол-
                    ;7526   ;   жен быть высоким и формировать сигнал L CRD H низкого уровня, а сигнал L RDS
                    ;7527   ;   H высокого уровня.
                    ;7528   ;
                    ;7529   ;       ОШИБКА A - Эта ошибка указывает на возможную неисправность самой ПМЛ УПР.
                    ;7530   ;   ОШИБКАМИ ДАННЫХ.
                    ;7531   ;
                    ;7532   ;       ОШИБКА B - То же, что и для ошибки 7.
                    ;7533   ;
                    ;7534   ;       ОШИБКА C - Эта ошибка указывает на возможную неисправность ПМЛ УПР.ОШИБКА-
                    ;7535   ;   МИ ДАННЫХ или входа INH REP CRD H. Необходимо проверить высокий уровень этого
                    ;7536   ;   входа после останова по ошибке. Если правильный, по-видимому, неисправна ПМЛ
                    ;7537   ;   УПР.ОШИБКАМИ ДАННЫХ.
                    ;7538   ;
                    ;7539   ;       ОШИБКА D - Эта ошибка указывает, что неисправна ПМЛ УПР.ОШИБКАМИ ДАННЫХ.
                    ;7540   ;   Для гарантии необходимо проверить сигнал L ECC DIS H. После останова по ошиб-
                    ;7541   ;   ке сигнал L ECC DIS H должен быть высоким. Другие входы были проверены при
                    ;7542   ;   других ошибках.
                    ;7543   ;
                    ;7544   ;       ОШИБКА E - Эта ошибка указывает, что сигнал ERR SUM L не доходит до центр.
                    ;7545   ;   процессора достаточно быстро. Приостанов до проверки ERR SUM длится 2 цикла
                    ;7546   ;   памяти (до снятия MEM BUSY). Сигнал ERR SUM должен успевать распространиться
                    ;7547   ;   через необходимые схемы в течении этого времени. Если эта ошибка появляется,
                    ;7548   ;   подозревается недостаточное быстродействие микросхем.
                    ;7549   ;
                    ;7550
                    ;7551    T.D:
U 0918, B65E,15     ;7552            MOV LS[BEGIN.TEST] TO WR[0]            ; установка бита 15 в WR[0] для слова управления и
                    ;7553                                                   ; состояния
U 0919, 3E80,15     ;7554            MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                    ;7555                                                   ; 15 указывает начало теста для конс.процессора
U 091A, 10E0,15     ;7556            MISC [SET.CP.ATTN]                     ; выдача сигнала CPU ATTN конс.процессору
                    ;7557    WAIT.TD.0:
U 091B, 8891,B4     ;7558            JMP [WAIT.TD.0]                        ; зацикливание для ожидания ответа консольного
                    ;7559                                                   ; процессора
U 091C, 0A1A,AC     ;7560            JSR [SETUP.1]                          ; установка масок, кода модуля и др.
U 091D, 3673,95     ;7561            MOV LS[ECC.DIS] TO WR[3]               ; установка бита ECC DIS в WR3
U 091E, 4775,95     ;7562            BIS LS[DIAG.CHK] TO WR[3]              ; также установка бита DIAG CHK в WR3. Сейчас WR3
                    ;7563                                                   ; содержит ожидаемые данные для ошибки 2
U 091F, B700,15     ;7564            MOV LS[CKBTS.0111100] TO WR[0]         ; пересылка в WR0 битов CKBT для формирования
                    ;7565                                                   ; отсутствия ошибки
U 0920, A0C0,15     ;7566            COM WR[0]                              ; дополнение CKBT - корректировка для аппаратуры
U 0921, 452C,15     ;7567            BIC LS[#FFFFFF00] TO WR[0]             ; очистка всех битов, за исключением младшего байта
U 0922, 2EC6,15     ;7568            BIS WR[3] TO WR[0]                     ; установка битов DIAG CHK и ECC DIS в WR0
U 0923, 8A17,FC     ;7569            JSR [WRITE.CSR1]                       ; запись данных из WR[0] в CSR1
U 0924, B629,15     ;7570            MOV LS[#FFFF] TO WR[2]                 ; маска ошибки для ошибки 2. Маскируются 16 младших
                    ;7571                                                   ; битов (не проверяются)
U 0925, 455D,15     ;7572            BIC LS[VALID.ERR] TO WR[2]             ; проверка ошибки истинности (бит 14)
U 0926, C55F,15     ;7573            BIC LS[TB.PAR.ERR] TO WR[2]            ; и проверка ошибки паритета буфера трансляции (бит 15)
U 0927, 4771,15     ;7574            BIS LS[BIT24] TO WR[2]                 ; не проверяется - неиспользуемый бит (бит 24)
                    ;7575    LOOP.TD.1:
U 0928, B640,15     ;7576            MOV LS[#1] TO WR[0]                    ; 1 в WR0
U 0929, BE82,15     ;7577            MOV WR[0] TO LS[ERROR.NUMBER]          ; установка номера ошибки в местной памяти
U 092A, E58A,15     ;7578            CLR LS[ERROR.MASK]                     ; проверка всех битов для ошибки 1
U 092B, 2F82,95     ;7579            CLR WR[1]                              ; ожидаемые данные ECC=0(H)
```

```
U 092C, 9D9C,F5 ;7580            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка для записи CKBT из CSR1 и данных из
                ;7581                                                      ; местной памяти
U 092D, B29C,15 ;7582            WRITE.MEM LS[ZERO]                        ; запись данных ECC=0(H)
U 092E, 3022,15 ;7583            MOV MEM.DATA TO WR[0]                     ; выборка результата - данные ECC (должен быть 0)
U 092F, 0869,3C ;7584            JSR [CHECK.RESULT]                        ; проверка, что данные ECC не изменены
U 0930, 8892,84 ;7585            JMP [LOOP.TD.1]                           ; зацикливание по ошибке, если разрешено
U 0931, FF82,15 ;7586            INC LS[ERROR.NUMBER]                      ; ошибка 2
U 0932, BE8B,15 ;7587            MOV WR[2] TO LS[ERROR.MASK]               ; изменение маски ошибки для проверки битов 14-23 и
                ;7588                                                      ; 25-31
U 0933, 2006,95 ;7589            MOV WR[3] TO WR[1]                        ; ожидаемые данные (установлены биты ECC DIS и DIAG
                ;7590                                                      ; CHK)
U 0934, 9D45,75 ;7591            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]     ; подготовка чтения CSR1
U 0935, 3022,15 ;7592            MOV MEM.DATA TO WR[0]                     ; выборка содержимого CSR1
U 0936, 0869,3C ;7593            JSR [CHECK.RESULT]                        ; проверка результата
U 0937, 8892,84 ;7594            JMP [LOOP.TD.1]                           ; зацикливание по ошибке, если разрешено
U 0938, FF82,15 ;7595            INC LS[ERROR.NUMBER]                      ; ошибка 3
                ;7596    LOOP.TD.3:
U 0939, 5B00,1D ;7597            SKIP.IF[MEM.REF.OK]                       ; проверка схем пропуска. Эта инструкция должна
                ;7598                                                      ; выполнить пропуск инструкции JSR, которая в последующем
                ;7599                                                      ; выдает сообщение о неисправности пропуска
                ;7600                                                      ; при ошибке памяти
U 093A, 889A,2C ;7601            JSR [ERROR.TD.3]                          ;
U 093B, 3673,95 ;7602            MOV LS[ECC.DIS] TO WR[3]                  ; установка ECC DIS в WR3
U 093C, 4775,95 ;7603            BIS LS[DIAG.CHK] TO WR[3]                 ; также установка DIAG CHK в WR3
U 093D, 3704,15 ;7604            MOV LS[CKBTS.1100100] TO WR[0]            ; пересылка CKBT для ошибки одиночного бита в WR0
U 093E, A0C0,15 ;7605            COM WR[0]                                 ; дополнение CKBT - корректировка для аппаратуры
U 093F, 452C,15 ;7606            BIC LS[#FFFFFF00] TO WR[0]                ; очистка всех битов, за исключением младшего байта
U 0940, 2EC6,15 ;7607            BIS WR[3] TO WR[0]                        ; установка битов DIAG CHK и ECC DIS в WR0
U 0941, 8A17,FC ;7608            JSR [WRITE.CSR1]                          ; запись данных из WR[0] в CSR1
U 0942, C77D,95 ;7609            BIS LS[CRD] TO WR[3]                      ; WR3 содержит ожидаемые данные для ошибки 5
                ;7610    LOOP.TD.4:
U 0943, 3644,15 ;7611            MOV LS[#4] TO WR[0]                       ; пересылка 4 в WR[0]
U 0944, 8E82,15 ;7612            MOV WR[0] TO LS[ERROR.NUMBER]             ; установка номера ошибки в местной памяти
U 0945, E58A,15 ;7613            CLR LS[ERROR.MASK]                        ; проверка всех битов для ошибки 4
U 0946, 2F82,95 ;7614            CLR WR[1]                                 ; ожидаемые данные ECC=0(H)
U 0947, 9D9C,F5 ;7615            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1 и данных из местной
                ;7616                                                      ; памяти
U 0948, B29C,15 ;7617            WRITE.MEM LS[ZERO]                        ; запись данных ECC=0(H)
U 0949, 3022,15 ;7618            MOV MEM.DATA TO WR[0]                     ; выборка результата - данные ECC (должен быть 0)
U 094A, 0869,3C ;7619            JSR [CHECK.RESULT]                        ; проверка, что данные ECC не изменены
U 094B, 0894,34 ;7620            JMP [LOOP.TD.4]                           ; зацикливание по ошибке, если разрешено
U 094C, FF82,15 ;7621            INC LS[ERROR.NUMBER]                      ; ошибка 5
U 094D, BE8B,15 ;7622            MOV WR[2] TO LS[ERROR.MASK]               ; изменение маски ошибки для проверки битов 14-23 и
                ;7623                                                      ; 25-31
U 094E, 2006,95 ;7624            MOV WR[3] TO WR[1]                        ; ожидаемые данные (установлены CRD, ECC DIS и DIAG
                ;7625                                                      ; CHK)
U 094F, 9D45,75 ;7626            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]     ; подготовка чтения CSR1
U 0950, 3022,15 ;7627            MOV MEM.DATA TO WR[0]                     ; выборка содержимого CSR1
U 0951, 0869,3C ;7628            JSR [CHECK.RESULT]                        ; проверка результата
U 0952, 0894,34 ;7629            JMP [LOOP.TD.4]                           ; зацикливание по ошибке, если разрешено
U 0953, FF82,15 ;7630            INC LS[ERROR.NUMBER]                      ; ошибка 6
                ;7631    LOOP.TD.6:
U 0954, 5B00,1D ;7632            SKIP.IF[MEM.REF.OK]                       ; проверка схемы пропуска. Эта инструкция не должна
                ;7633                                                      ; выполнять пропуска
U 0955, 0895,74 ;7634            JMP [TEST.TD.7]                           ; если здесь, ошибки нет. Продолжение проверки
```

```
U 0956, 089A,5C ;7635            JSR [ERROR.TD.6]                        ; если здесь, сообщение о неисправности пропуска при
                ;7636                                                    ; ошибке памяти
                ;7637    TEST.TD.7:
U 0957, FF82,15 ;7638            INC LS[ERROR.NUMBER]                    ; ошибка 7
U 0958, 457D,95 ;7639            BIC LS[CRD] TO WR[3]                    ; ожидаемые данные содержат сброшенный бит CRD
                ;7640    LOOP.TD.7:
U 0959, 8A1E,3C ;7641            JSR [CLEAR.CSR2]                        ; выполнение записи в CSR2
U 095A, 2006,95 ;7642            MOV WR[3] TO WR[1]                      ; ожидаемые данные (установлены ECC DIS и DIAG CHK)
U 095B, 9D45,75 ;7643            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 095C, 3022,15 ;7644            MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1. Бит CRD должен быть сброшен
                ;7645                                                    ; записью в CSR2
U 095D, 0869,3C ;7646            JSR [CHECK.RESULT]                      ; проверка результата
U 095E, 8895,94 ;7647            JMP [LOOP.TD.7]                         ; зацикливание по ошибке, если разрешено
                ;7648    TEST.TD.8:
U 095F, FF82,15 ;7649            INC LS[ERROR.NUMBER]                    ; ошибка 8
U 0960, 3682,15 ;7650            MOV LS[ERROR.NUMBER] TO WR[0]           ; выборка подготовленного номера ошибки
U 0961, 3E10,15 ;7651            MOV WR[0] TO LS[T8]                     ; и запоминание во временной ячейке местной памяти
U 0962, 3673,95 ;7652            MOV LS[ECC.DIS] TO WR[3]                ; установка бита ECC DIS в WR3
U 0963, 4775,95 ;7653            BIS LS[DIAG.CHK] TO WR[3]               ; также установка бита DIAG CHK в WR3
U 0964, 370E,15 ;7654            MOV LS[CKBTS.1111101] TO WR[0]          ; пересылка CKBT для некорректируемой ошибки в WR0
U 0965, A0C0,15 ;7655            COM WR[0]                               ; дополнение CKBT - корректировка для аппаратуры
U 0966, 452C,15 ;7656            BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех битов, за исключением младшего байта
U 0967, 2EC6,15 ;7657            BIS WR[3] TO WR[0]                      ; установка битов DIAG CHK и ECC DIS в WR0
U 0968, 8A17,FC ;7658            JSR [WRITE.CSR1]                        ; запись данных из WR[0] в CSR1
U 0969, 477F,95 ;7659            BIS LS[RDS] TO WR[3]                    ; сейчас WR3 содержит ожидаемые данные для ошибки 8
                ;7660    LOOP.TD.8:
U 096A, B610,15 ;7661            MOV LS[T8] TO WR[0]                     ; пересылка 8 в WR[0]
U 096B, BE82,15 ;7662            MOV WR[0] TO LS[ERROR.NUMBER]           ; установка номера ошибки в местной памяти
U 096C, E58A,15 ;7663            CLR LS[ERROR.MASK]                      ; проверка всех битов для ошибки 8
U 096D, 2F82,95 ;7664            CLR WR[1]                               ; ожидаемые данные ECC=0(H)
U 096E, 9D9C,F5 ;7665            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1 и данных из местной
                ;7666                                                    ; памяти
U 096F, B29C,15 ;7667            WRITE.MEM LS[ZERO]                      ; запись данных ECC=0(H)
U 0970, 3022,15 ;7668            MOV MEM.DATA TO WR[0]                   ; выборка результата - данных ECC (должен быть 0)
U 0971, 0869,3C ;7669            JSR [CHECK.RESULT]                      ; проверка, что данные ECC не изменены
U 0972, 8896,A4 ;7670            JMP [LOOP.TD.8]                         ; зацикливание по ошибке, если разрешено
U 0973, FF82,15 ;7671            INC LS[ERROR.NUMBER]                    ; ошибка 9
U 0974, BE8B,15 ;7672            MOV WR[2] TO LS[ERROR.MASK]             ; изменение маски ошибки для проверки битов 14-23 и
                ;7673                                                    ; 25-31
U 0975, 2006,95 ;7674            MOV WR[3] TO WR[1]                      ; ожидаемые данные (RDS, ECC DIS и DIAG CHK
                ;7675                                                    ; установлены)
U 0976, 9D45,75 ;7676            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0977, 3022,15 ;7677            MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0978, 0869,3C ;7678            JSR [CHECK.RESULT]                      ; проверка результата
U 0979, 8896,A4 ;7679            JMP [LOOP.TD.8]                         ; зацикливание по ошибке, если разрешено
U 097A, FF82,15 ;7680            INC LS[ERROR.NUMBER]                    ; ошибка A
                ;7681    LOOP.TD.A:
U 097B, 5B00,1D ;7682            SKIP.IF[MEM.REF.OK]                     ; проверка схем пропуска. Эта инструкция не должна
                ;7683                                                    ; выполнять пропуска
U 097C, 8897,E4 ;7684            JMP [TEST.TD.B]                         ; если здесь, ошибки нет. Продолжение проверки
U 097D, 889A,8C ;7685            JSR [ERROR.TD.A]                        ; если здесь, сообщение о неисправности пропуска при
                ;7686                                                    ; ошибке памяти
                ;7687    TEST.TD.B:
U 097E, FF82,15 ;7688            INC LS[ERROR.NUMBER]                    ; ошибка B
U 097F, C57F,95 ;7689            BIC LS[RDS] TO WR[3]                    ; ожидаемые данные содержат сброшенный бит RDS
```

```
                        ;  ТЕСТ D - тест ошибки ECC (без коррекции) (модули MCT и DAP)

                        ;7690    LOOP.TD.B:
U 0980, BA1E,3C         ;7691            JSR [CLEAR.CSR2]                        ; выполнение записи в CSR2
U 0981, 2006,95         ;7692            MOV WR[3] TO WR[1]                      ; ожидаемые данные (установлены биты ECC DIS и DIAG
                        ;7693                                                    ; CHK)
U 0982, 9D45,75         ;7694            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0983, 3022,15         ;7695            MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1. Бит RDS должен быть сброшен
                        ;7696                                                    ; при записи в CSR2
U 0984, 0869,3C         ;7697            JSR [CHECK.RESULT]                      ; проверка результата
U 0985, 0898,04         ;7698            JMP [LOOP.TD.B]                         ; зацикливание по ошибке, если разрешено
                        ;7699    TEST.TD.C:
U 0986, FF82,15         ;7700            INC LS[ERROR.NUMBER]                    ; ошибка C
U 0987, 3673,95         ;7701            MOV LS[ECC.DIS] TO WR[3]                ; установка бита ECC DIS в WR3
U 0988, 4775,95         ;7702            BIS LS[DIAG.CHK] TO WR[3]               ; также установка бита DIAG CHK в WR3
U 0989, 4779,95         ;7703            BIS LS[INH.CRD] TO WR[3]                ; сейчас WR3 содержит ожидаемые данные для ошибки B
U 098A, 3704,15         ;7704            MOV LS[CKBTS.1100100] TO WR[0]          ; пересылка в WR0 CKBT для ошибки одиночного бита в WR0
U 098B, A0C0,15         ;7705            COM WR[0]                               ; дополнение CKBT - корректировка для аппаратуры
U 098C, 452C,15         ;7706            BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех битов, кроме младшего байта
U 098D, 2EC6,15         ;7707            BIS WR[3] TO WR[0]                      ; установка битов INH REP CRD, DIAG CHK и ECC DIS в WR0
U 098E, 8A17,FC         ;7708            JSR [WRITE.CSR1]                        ;
                        ;7709    LOOP.TD.C:
U 098F, 9D9C,F5         ;7710            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1 и данных из местной
                        ;7711                                                    ; памяти
U 0990, B29C,15         ;7712            WRITE.MEM LS[ZERO]                      ; запись данных ECC=0(H)
U 0991, 3022,15         ;7713            MOV MEM.DATA TO WR[0]                   ; завершение цикла, но проверка не выполняется
U 0992, 2006,95         ;7714            MOV WR[3] TO WR[1]                      ; ожидаемые данные (INH CRD, ECC DIS и DIAG CHK
                        ;7715                                                    ; установлены)
U 0993, 9D45,75         ;7716            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0994, 3022,15         ;7717            MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0995, 0869,3C         ;7718            JSR [CHECK.RESULT]                      ; проверка результата
U 0996, 0898,F4         ;7719            JMP [LOOP.TD.C]                         ; зацикливание по ошибке, если разрешено
U 0997, FF82,15         ;7720            INC LS[ERROR.NUMBER]                    ; ошибка D
                        ;7721    LOOP.TD.D:
U 0998, 5800,1D         ;7722            SKIP.IF[MEM.REF.OK]                     ; проверка схем пропуска. Эта инструкция не должна
                        ;7723                                                    ; выполнять пропуска
U 0999, 0899,B4         ;7724            JMP [TEST.TD.E]                         ; если здесь, нет ошибки. Продолжение проверки
U 099A, 889A,BC         ;7725            JSR [ERROR.TD.D]                        ; если здесь, сообщение о неисправности пропуска при
                        ;7726                                                    ; ошибке памяти
                        ;7727    TEST.TD.E:
U 099B, FF82,15         ;7728            INC LS[ERROR.NUMBER]                    ; ошибка E
                        ;7729    LOOP.TD.E:
U 099C, 9D9C,F5         ;7730            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1 и данных из местной
                        ;7731                                                    ; памяти
U 099D, B29C,15         ;7732            WRITE.MEM LS[ZERO]                      ; запись данных ECC=0(H)
                        ;7733              MOV MEM.DATA TO WR[0],                ; завершение цикла. Просмотр ошибки памяти
U 099E, B022,1D         ;7734            SKIP.IF[MEM.REF.OK]                     ; проверка схем пропуска. Эта инструкция не должна
                        ;7735                                                    ; выполнять пропуска
U 099F, 889B,D4         ;7736            JMP [END.TD]                            ; если здеь, нет ошибки. Конец теста
U 09A0, 889A,EC         ;7737            JSR [ERROR.TD.E]                        ; если здесь, сообщение о неисправности пропуска при
                        ;7738                                                    ; ошибке памяти
U 09A1, 889B,D4         ;7739            JMP [END.TD]                            ; ошибка, но зацикливания по ошибке нет. Выполнено
                        ;7740    ERROR.TD.3:
U 09A2, 089B,1C         ;7741            JSR [REPORT.TD.SKIP.ERROR]              ; вывод сообщения об ошибке при ошибке пропуска
U 09A3, 8893,94         ;7742            JMP [LOOP.TD.3]                         ; если возврат сюда, зацикливание при ошибке
U 09A4, 5800,14         ;7743            RETURN                                  ; возврат без зацикливания при ошибке
                        ;7744    ERROR.TD.6:
```

```
U 09A5, 089B,1C ;7745            JSR [REPORT.TD.SKIP.ERROR]              ; вывод сообщения об ошибке при ошибке пропуска
U 09A6, 0895,44 ;7746            JMP [LOOP.TD.6]                         ; если возврат сюда, зацикливание при ошибке
U 09A7, 5B00,14 ;7747            RETURN                                  ; нет зацикливания при ошибке. Продолжение проверки
                ;7748    ERROR.TD.A:
U 09A8, 889B,3C ;7749            JSR [REPORT.TD.SKIP.ERROR.NO.CPU]       ; вывод сообщения об ошибке при ошибке пропуска
U 09A9, 8897,B4 ;7750            JMP [LOOP.TD.A]                         ; если возврат сюда, зацикливание при ошибке
U 09AA, 5B00,14 ;7751            RETURN                                  ; нет зацикливания при ошибке. Продолжение проверки
                ;7752    ERROR.TD.D:
U 09AB, 889B,3C ;7753            JSR [REPORT.TD.SKIP.ERROR.NO.CPU]       ; вывод сообщения об ошибке при ошибке пропуска
U 09AC, 0899,B4 ;7754            JMP [LOOP.TD.D]                         ; если возврат сюда, зацикливание при ошибке
U 09AD, 5B00,14 ;7755            RETURN                                  ; нет зацикливания при ошибке. Продолжение проверки
                ;7756    ERROR.TD.E:
U 09AE, 889B,3C ;7757            JSR [REPORT.TD.SKIP.ERROR.NO.CPU]       ; вывод сообщения об ошибке при ошибке пропуска
U 09AF, 8899,C4 ;7758            JMP [LOOP.TD.E]                         ; если возврат сюда, зацикливание при ошибке
U 09B0, 5B00,14 ;7759            RETURN                                  ; нет зацикливания при ошибке. Продолжение проверки
                ;7760    REPORT.TD.SKIP.ERROR:
U 09B1, 3642,15 ;7761            MOV LS[CPU] TO WR[0]                    ; адрес модуля DAP для распечатки кода
                ;7762                                                    ; модуля
U 09B2, 6D8C,15 ;7763            BIS WR[0] TO LS[MODULE.NUM]             ; восстановление кода модуля в местной памяти
                ;7764    RE 'ORT.TD.SKIP.ERROR.NO.CPU:
U 09B3, B66E,15 ;7765            MOV LS[NA.EXP.REC] TO WR[0]             ; установка бита 23 в WR0 для указания не печатать
                ;7766                                                    ; ожидаемых или полученных данных, если появляется
                ;7767                                                    ; ошибка
U 09B4, 3E80,15 ;7768            MOV WR[0] TO LS[CONTROL.STATUS]         ; восстановление слова управления и состояния
U 09B5, 2F80,15 ;7769            CLR WR[0]                               ; очистка WR0
U 09B6, 369E,95 ;7770            MOV LS[ONES] TO WR[1]                   ; установка WR1. Сейчас подпрограмма ошибки должна
                ;7771                                                    ; сообщить об ошибке (но EXP и REC не будут печататься)
U 09B7, 0869,3C ;7772            JSR [CHECK.RESULT]                      ; сообщение об ошибке
U 09B8, 5B00,14 ;7773            RETURN                                  ; возврат для зацикливания при ошибке
U 09B9, 3644,15 ;7774            MOV LS[MCT] TO WR[0]                    ; установка кода модуля. Установка бита 2 в WR0
U 09BA, 3E8C,15 ;7775            MOV WR[0] TO LS[MODULE.NUM]             ; запоминание кода модуля в местной памяти для указания
                ;7776                                                    ; только модуля MCT
U 09BB, E580,15 ;7777            CLR LS[CONTROL.STATUS]                  ; снова разрешение печати EXP и REC
U 09BC, DB00,16 ;7778            RETURN+1                                ; возврат без зацикливания при ошибке
                ;7779    END.TD:
```

```
;7780    .PAGE "ТЕСТ Е - тест коррекции ECC (модуль МСТ)"
;7781    ;
;7782    ; ОПИСАНИЕ ТЕСТА:
;7783    ;
;7784    ;       Этот тест проверяет схемы коррекции ECC, чтобы убедиться, что любой бит мо-
;7785    ;    жет быть скорректирован из "1" в "0" и из "0" в 1. Тест также проверяет схемы
;7786    ;    МСТ ветвления при ошибке и ветвления при ошибке одиночного бита (BEN0 = ERROR.L
;7787    ;    и BEN2 = SINGLE.ERR.L). CSR должен быть загружен контрольными битами CKBT, ко-
;7788    ;    торые соотвествуют данным ECC - все нули или все единицы. Затем в буферы данных
;7789    ;    ECC вводятся данные со сдвигаемой единицей и со сдвигаемым нулем и в полученных
;7790    ;    данных проверяется коррекция на все нули или все единицы соответственно.
;7791    ;    Ошибки двух битов также будут проверяться, чтобы убедиться, что они обнаружи-
;7792    ;    ваются, но не корректируются. Путь данных следующий: выдается инструкция MEM.
;7793    ;    REQ с полем MF = MAINT.ECC.DATA. Используемый адрес местной памяти содержит
;7794    ;    данные 0(H) для более позднего ветвления в тесте. Эти данные загружаются в ре-
;7795    ;    гистр виртуального адреса и выполняется начальное ветвление на соответствующую
;7796    ;    программу (см. описание начального ветвления в начале этого листинга). По ад-
;7797    ;    ресу начального ветвления устанавливаются MEMORY BUSY и запрет сдвига, разреша-
;7798    ;    ются биты PA и приемо-передатчики на модуле центрального процессора для обеспе-
;7799    ;    чения передачи данных из центрального процессора. В следующем цикле стробируют-
;7800    ;    ся сдвигатели данных для запрета сдвига и повторяются функции предыдущего цикла
;7801    ;    (за исключением того, что не нужен SET ADDR PH). Следующий цикл открывает буфе-
;7802    ;    ры ECC, разрешает выходы сдвигателей данных на шину BUS ARRAY, очищает любые
;7803    ;    ошибки в CSR, поддерживает установленным MEMORY BUSY и выполняет по сигналу L
;7804    ;    CPU DR. Сигнал CPU DR (запрос данных) поступает, когда центральный процессор
;7805    ;    выполняет инструкцию MOV. В этом случае центральный процессор пересылает набор
;7806    ;    тестовых данных для ПМЛ сдвигателей. Если сигнал L CPU DR еще не возбужден, мик-
;7807    ;    ропрограмма памяти будет циклиться в ожидании его. Как только сигнал L CPU DR
;7808    ;    возбуждается, следующий микроцикл возбуждает 2ND MEM.CYC, открывает буферы ECC,
;7809    ;    разрешает ПМЛ сдвигателей на шину BUS ARRAY и поддерживает разрешенными пере-
;7810    ;    датчики центрального процессора. Установленный 2ND MEM CYC поддерживает данные,
;7811    ;    пересылаемые через шину BUS MC из центрального процессора, буферизованными
;7812    ;    внутри ПМЛ сдвигателей (данные были загружены в последнем микроцикле). Сле-
;7813    ;    дующий цикл поддерживает открытыми буферы ECC, держит выходы ПМЛ сдвигате-
;7814    ;    лей на шине BUS ARRAY и ожидает возбуждения сигнала D DATA REC. Сигнал DATA
;7815    ;    RCVD возбуждается центральным процессорам в конце (P2) инструкции централь-
;7816    ;    ного процессора MOV. Зацикливание здесь, пока будет возбужден этот сигнал,
;7817    ;    до возбуждения MEMORY BUSY обеспечивает, что центральный процессор перейдет
;7818    ;    на следующую микроинструкцию.
;7819    ;       Когда сигнал D DATA REC появится, микропрограмма памяти выполнит ветвление
;7820    ;    по сигналу LVA00 (который был сброшен в начале инструкции центрального процес-
;7821    ;    сора MEM.REQ) на соответствующую тестовую ветвь. Этот микроцикл должен возбудить
;7822    ;    MEMORY BUSY, открыть буферы ECC, разрешить выходы ПМЛ сдвигателей на шину BUS
;7823    ;    ARRAY. Затем этот цикл должен выполнить ветвление на соответствующую проверку
;7824    ;    в зависимости от состояния битов ECC DIS и DIAG CHK в CSR1. CSR1 был загружен
;7825    ;    до выдачи инструкции MEM.REQ. Для этого теста бит DIAG CHK установлен и ECC
;7826    ;    DIS сброшен. После ветвления сигнал 2ND MEMORY CYCLE сбрасывается, а MEMORY
;7827    ;    BUSY остается возбужденным. Ветвление выполняется также по сигналу ERROR L.
;7828    ;    Если ошибки нет, данные ECC помещаются на шину BUS MC и память зацикливается
;7829    ;    в ожидании сигнала CPU DATA REQ. Затем МСТ ожидает сигнала CPU DATA RCVD, пос-
;7830    ;    ле которого переходит в холостой цикл. Если ошибка имеется, следующий цикл
;7831    ;    стробирует RDS и CRD В CSR1, разрешает биты синдрома из схемы ECC старшего
;7832    ;    слова на шину BUS ARRAY СВ и выполняет ветвление по одиночной ошибке. Если об-
;7833    ;    наружена двойная ошибка, синдромы буферизируются в CSR0 и данные из схемы ECC
;7834    ;    читаются центральным процессором, как описано выше. Если появляется ошибка оди-
```

```
;7835    ;   ночного бита, следующий цикл разрешает коррекцию, выдает биты синдрома на шину
;7836    ;   CB BUS и открывает выходы буфера данных ECC. Следующий цикл все это повторяет,
;7837    ;   а также стробирует биты синдрома в CSR0. Следующий цикл закрывает выход буфера
;7838    ;   данных ECC, поддерживает разрешение коррекции. Затем результат читается цент-
;7839    ;   ральным процессором, как описано раньше и проверяется. Используемые тестовые
;7840    ;   данные - все единицы и все нули с установленными CKBT для создания отсутствия
;7841    ;   ошибки. После этого следует сдвигаемая единца с установленными CKBT для коррек-
;7842    ;   ции сдвигаемого бита.
;7843    ;
;7844    ; ПРЕДПОЛОЖЕНИЯ:
;7845    ;
;7846    ;   Предполагается, что все предыдущие тесты выполнены успешно.
;7847    ;
;7848    ; ШАГИ ТЕСТА:
;7849    ;
;7850    ;   1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;7851    ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;7852    ;      памяти.
;7853    ;   2. Запись в CSR0 единиц выполнением теста CKBT. Затем запись в CSR1 для
;7854    ;      установки бита DIAG CHK, сброса бита ECC DIS и запись дополнения CKBT
;7855    ;      для данных - все единицы или все нули (дополнение значения 0111100(B)
;7856    ;      от CBT до CB1 соответственно).
;7857    ;   3. Загрузка в WR1 тестовых данных из местной памяти (ожидаемые данные ECC).
;7858    ;   4. Выдача инструкции MEM.REQ C DT = LONGWORD(11) и MF = MAINT.ECC.DATA, ис-
;7859    ;      пользуя ячейку местной памяти, которая содержит 0(H), качестве адреса (он
;7860    ;      загружается в регистр виртуального адреса и используется для более поздне-
;7861    ;      го ветвления).
;7862    ;   5. Выполнение инструкции WRITE.MEM LS с текущими тестовыми данными ECC.
;7863    ;   6. Выполнение инструкции MOVE MEM.DATA TO WR[0] и проверка, что данные не
;7864    ;      изменены.
;7865    ;   7. Изменение маски ошибки для проверки битов 6-0, загрузка в WR1 ожидаемого
;7866    ;      результата CSR0 (единицы), чтение CSR0 и проверка. Это гарантирует, что
;7867    ;      выполнено правильное ветвление по сигналу ERROR.L.
;7868    ;   8. Изменение маски ошибки для проверки битов 31 и 30 CSR1 (RDS и CRD). Чтение
;7869    ;      CSR1 и проверка, что биты RDS и CRD сброшены. Выполнение пропуска при от-
;7870    ;      сутствии ошибки памяти (ERROR SUM отсутствует) в центральном процессоре,
;7871    ;      для проверки пропуска.
;7872    ;   9. Восстановление маски ошибки на 32 бита и повторение шагов с 2 по 6 со сле-
;7873    ;      дующими тестовыми данными (нули).
;7874    ;  10. Повторение шагов с 2 до 5 с тестовыми данными - сдвигаемая единица.
;7875    ;  11. Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка на все нули (от-
;7876    ;      корректированные данные). Повторение шагов 8 и 10, пока единица не будет
;7877    ;      продвинута через все биты. Проверка установки CRD и пропуска при ошибке
;7878    ;      памяти на шаге 8.
;7879    ;  12. Повторение шага 10 с данными - сдвигаемый нуль. Выполнение инструкции MOV
;7880    ;      MEM.DATA TO WR[0] и проверка на все единицы (откорректированные данные).
;7881    ;  13. Повторение шагов с 2 до 5 с тестовыми данными для двойной ошибки в качест-
;7882    ;      ве данных ECC. Тестовыми данными для двойной ошибки являются 3(H) и 30000(H)
;7883    ;      (двойная ошибка в младшем слове и в старшем слове соответственно).
;7884    ;  14. Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка, что данные не из-
;7885    ;      менены (двойные ошибки не корректируются).
;7886    ;  15. Повторение шага 8 для проверки установки RDS, сброса CRD и отсутствия
;7887    ;      пропуска при отсутствии ошибки памяти.
;7888    ;  16. Установка INH REP CRD В CSR1 и проверка пропуска и неустановленного CRD
;7889    ;      при ошибке одиночного бита.
```

```
;7890    ;  17. Установка INH REP CRD В CSR1 и проверка отсутствия пропуска и установки
;7891    ;      RDS при ошибке двух битов.
;7892    ;
;7893    ; ОШИБКИ:
;7894    ;
;7895    ;   ПРИМЕЧАНИЕ: Ожидаемыми и плолученными данными являются: данные ECC, когда
;7896    ;   маска ошибки - все нули, данные CSR0, когда маска ошибки - FFFFFF80 и дан-
;7897    ;   ные CSR1, когда маска ошибки - 3FFFFFFF.
;7898    ;
;7899    ;   ошибка 1 - данные ECC изменены при СКВТ, установленных для отсутствия ошибки
;7900    ;              (единичные данные).
;7901    ;
;7902    ;   ошибка 2 - СКВТ в схемах ECC изменены при условии отсутствия ошибки или
;7903    ;              неправильно работает ветвление по ошибке.
;7904    ;
;7905    ;   ошибка 3 - бит RDS или CRD установлен в CSR1 при условии отсутствия ошибки.
;7906    ;
;7907    ;   ошибка 4 - в центральном процессоре не произошел пропуск по отсутствию ошибки
;7908    ;              памяти при отсутствии ошибки памяти (из-за неисправности в модуле
;7909    ;              МСТ).
;7910    ;
;7911    ;   ошибка 5 - данные ECC изменены при СКВТ, установленных для отсутствия ошибки
;7912    ;              (данные - нули).
;7913    ;
;7914    ;   ошибка 6 -  схема ECC работает неправильно при коррекции ошибки одиночного бита
;7915    ;               из 1 в 0. Данные в поле OTHER являются эталоном данных, откорректи-
;7916    ;               рованных на все нули.
;7917    ;
;7918    ;   ошибка 7 - установленный бит RDS и/или сброшенный CRD после ошибки одиноч-,
;7919    ;              ного бита. Данные в поле OTHER являются эталоном данных, откоррек-
;7920    ;              тированных на все нули.
;7921    ;
;7922    ;   ошибка 8 - функция центрального процессора - пропуск по отсутствию ошибки па-
;7923    ;              мяти выполняет пропуск после ошибки одиночного бита.
;7924    ;
;7925    ;   ошибка 9 - схема ECC работает неправильно при коррекции одиночного бита из 0
;7926    ;              в 1. Данные в поле OTHER являются эталоном данных, откорректирован-
;7927    ;              ных на все единицы.
;7928    ;   ошибка A - схема ECC изменила данные при ошибке двух битов или неправильно
;7929    ;              выполняется ветвление при ошибке одиночного бита с данными = 3(H).
;7930    ;
;7931    ;   ошибка B - бит RDS сброшен и/или CRD установлен при ошибке двух битов с
;7932    ;              данными = 3(H).
;7933    ;
;7934    ;   ошибка C - функция центрального процессора - пропуск при отсутствии ошибки па-
;7935    ;              мяти выполняет пропуск после ошибки двух битов.
;7936    ;
;7937    ;   ошибка D - схема ECC изменила данные при ошибке двух битов или неправильно
;7938    ;              выполняется ветвление при ошибке одиночного бита с данными =
;7939    ;              30000(H).
;7940    ;
;7941    ;   ошибка E - сбрасывается RDS и/или устанавливается CRD при ошибке двух битов
;7942    ;              с данными = 30000(H).
;7943    ;
;7944    ;   ошибка F - в CSR1 устанавливается CRD или RDS при ошибке одиночного бита с
```

```
;7945    ;              установленным INH REP CRD (запрет сообщения о корректируемых
;7946    ;              ошибках данных).
;7947    ;    ошибка 10 - в центральном процессоре не произошел пропуск по отсутствию ошиб-
;7948    ;                ки памяти при ошибке одиночного бита с установленным INH REP CRD.
;7949    ;
;7950    ;    ошибка 11 - сбрасывается RDS или устанавливается CRD после ошибки двух би-
;7951    ;                тов с установленным INH REP CRD.
;7952    ;
;7953    ;    ошибка 12 - функция центрального процессора - пропуск по отсутствию ошибки па-
;7954    ;                мяти выполняет пропуск при ошибке двух битов с установленным
;7955    ;                INH REP CRD.
;7956    ;
;7957    ; НАЛАДКА:
;7958    ;
;7959    ;   ПРИМЕЧАНИЕ: Данные, записанные в части данных микросхем ECC, печатаются под
;7960    ;   OTHER в сообщении об ошибке при тестовых данных со сдвигаемой единицей и сдви-
;7961    ;   гаемым нулем.
;7962    ;
;7963    ;      ОШИБКА 1 - Эта ошибка указывет на возможную неисправность самих микросхем
;7964    ;   ECC. Трудно определить, которая микросхема ECC работает неправильно, но все
;7965    ;   входные линии, за исключением CORR DIS L при высоком уровне, были проверены.
;7966    ;   Если CORR DIS L имеет высокий уровень на обеих микросхемах, подозревается од-
;7967    ;   на из этих микросхем.
;7968    ;
;7969    ;      ОШИБКА 2 - Эта ошибка скорее всего появляется в результате неправильного
;7970    ;   ветвления по сигналу ERROR L в направлении отсутствия ошибки. Если ветвление
;7971    ;   при ошибке выбирает путь ошибки, синдромы из схем ECC записываются в CSR0.
;7972    ;   Они будут отличными от CKBT, первоначально записанных В CSR0. Если пошаговый
;7973    ;   режим MCT возможен, центральный процессор необходимо остановить на инструкции
;7974    ;   WRITE.MEM LS, разрешить пошаговый режим MCT, остановить центральный процессор
;7975    ;   на инструкции MOV MEM.DATA TO WR[0] и остановить MCT на ветвлении по сигналу
;7976    ;   ERROR L (BEN0=ERROR L). Необходимо проверить высокий уровень сигнала ERROR L
;7977    ;   на входе мультиплексора BEN0 и наличие 011(B) на входах выборки A2, A1,и A0
;7978    ;   соответственно. Если здесь правильно, выполняется шаг MCT и проверяется от-
;7979    ;   сутствие ветвления. Если ветвление произошло, неисправен мультиплексор BEN0.
;7980    ;
;7981    ;      ОШИБКА 3 - Эта ошибка скорее всего указывает на неправильное ветвление или
;7982    ;   возможную неисправность схем ECC. Если схемы ECC исправны, условие отсутст-
;7983    ;   вия ошибки должно вызвать обход сигнала CSR ERR ADR CLK, необходимый для ус-
;7984    ;   тановки битов в CSR. Если схемы ECC неисправны, биты CRD и RDS могут принять
;7985    ;   любые значения.
;7986    ;
;7987    ;      ОШИБКА 4 - Если это первая ошибка, подозревается ПМЛ УПР.ОШИБКАМИ ДАННЫХ.
;7988    ;
;7989    ;      ОШИБКА 5 - То же, что и для ошибки 1.
;7990    ;
;7991    ;      ОШИБКА 6 - Эта ошибка указывает на возможную неисправность одной из микро-
;7992    ;   схем ECC или схем ветвления по сигналу SINGLE ERR L. Необходимо проверить низ-
;7993    ;   кий уровень сигналов SINGLE ERR L и ERROR L на выходе микросхемы ECC старшего
;7994    ;   слова. Сигналы можно проверить после останова по ошибке под управлением кон-
;7995    ;   сольного процессора. Если один из этих сигналов неправильный, тогда неисправ-
;7996    ;   на одна из схем ECC. Если оба сигнала низкие, возможно, что неправильно вы-
;7997    ;   полняется ветвление по сигналу SINGLE ERR. Если ветвление неправильное, сиг-
;7998    ;   нал CORR DIS L не станет высоким и коррекции не будет. Прежде всего, необходи-
;7999    ;   мо проверить низкий уровень сигнала SINGLE ERR L на входе мультиплексора BEN2.
```

```
;8000    ;   Если здесь правильно и пошаговый режим MCT возможен, центральный процессор
;8001    ;   следует остановить на инструкции WRITE.MEM LS, разрешить пошаговый режим MCT,
;8002    ;   здесь центральный процессор остановить на инструкции MOV MEM.DATA TO WR[0] и
;8003    ;   MCT остановить на цикле ветвления по сигналу SINGLE ERR (BEN2 = SINGLE.ERR.L).
;8004    ;   Необходимо проверить сигнал SINGLE ERR L на входе мультиплексора BEN2 и нали-
;8005    ;   чие 011(B) на линиях выборки A2, A1, A0 соответственно. Если здесь правильно,
;8006    ;   необходимо выполнить еще один шаг MCT. Должен быть переход на цикл, в котором
;8007    ;   сигнал CORR DIS L становится высоким. Если ветвление неправильное, неисправен
;8008    ;   мультиплексор BEN2. Иначе необходимо проверить, что сигнал CORR DIS L высокий
;8009    ;   на обеих схемах ECC. Если правильно, по-видимому, неисправна одна из микросхем
;8010    ;   ECC. Если пошаговый режим MCT невозможен, необходимо проверить сигнал CORR DIS
;8011    ;   L во время зацикливания по ошибке.
;8012    ;
;8013    ;      ОШИБКА 7 - Если это первая ошибка, возможно, что не возбуждается сигнал
;8014    ;   SINGLE ERR L микросхемой ECC старшего слова, но возможно, что неправильно
;8015    ;   выполняется ветвление по сигналу SINGLE ERR L (BEN2=SINGLE.ERR.L). Другие
;8016    ;   схемы, используемые для управления ПМЛ УПР.ОШИБКАМИ ДАННЫХ, уже проверялись.
;8017    ;   Если пошаговой режим MCT возможен, необходимо проверить входы к ПМЛ УПР.
;8018    ;   ОШИБКАМИ ДАННЫХ во время цикла, который возбуждает CSR ERR ADDR CLK A L. Дол-
;8019    ;   жен быть низкий уровень на входах CSR ERR ADDR CLK A L, SINGLE ERR L, ERROR
;8020    ;   L, CSR ERR SUM CLK H и INH REP CRD H. Необходимо проверить высокий уровень на
;8021    ;   входах WR CSR L и CPU/UBL. Дефектный бит управляющей памяти может также быть
;8022    ;   причиной ошибки.
;8023    ;
;8024    ;      ОШИБКА 8 - Если эта ошибка появляется, скорее всего подозревается сама ПМЛ
;8025    ;   УПР.ОШИБКАМИ ДАННЫХ.
;8026    ;
;8027    ;      ОШИБКА 9 - Если это первая ошибка, скорее всего подозревается одна из мик-
;8028    ;   росхем ECC.
;8029    ;
;8030    ;      ОШИБКА A - Эта ошибка указывает на возможную неисправность одной из микрос-
;8031    ;   схем ECC или схем ветвления по сигналу SINGLE ERR. Необходимо проверить низкий
;8032    ;   уровень сигнала ERROR L и высокий уровень сигнала SINGLE ERR L на выходе мик-
;8033    ;   росхемы ECC старшего слова. Это можно выполнить после того, как консольный
;8034    ;   процессор выполнит останов по ошибке. Если хотя бы один из этих сигналов непра-
;8035    ;   вильный, тогда неисправна одна из микросхем ECC. Если они правильные, необхо-
;8036    ;   димо проверить высокий уровень сигнала SINGLE ERR L на мультиплексоре BEN2.
;8037    ;   Если здесь правильно и пошаговый режим MCT возможен, необходимо проследить от-
;8038    ;   сутствие ветвления по методике, описанной для ошибки 6. Если микропрограмма за-
;8039    ;   канчивается установкой высокого уровня сигнала DIS CORR L, тогда, по-видимому,
;8040    ;   неисправен мультиплексор BEN2. Иначе подозревается одна из микросхем ECC.
;8041    ;
;8042    ;      ОШИБКА B - Скорее всего подозревается одна из микросхем ECC. Ветвление по
;8043    ;   сигналу ERROR L могло пройти по пути отсутствия ошибки, из-за того, что схема
;8044    ;   ECC не возбудила сигнала ERROR L. Это могло оставить данные неизмененными, но
;8045    ;   не произошло стробирование CSR1 для установки RDS.
;8046    ;
;8047    ;      ОШИБКА C - если эта ошибка появляется, скорее всего подозревается сама ПМЛ
;8048    ;   УПР.ОШИБКАМИ ДАННЫХ.
;8049    ;
;8050    ;      ОШИБКА D - подозревается одна из микросхем ECC.
;8051    ;
;8052    ;      ОШИБКА E - то же, что и для ошибки B.
;8053    ;
;8054    ;      ОШИБКИ ОТ F ДО 12 - Эти ошибки указывают на возможную неисправность ПМЛ УПР.
```

```
                ;8055   ;   ОШИБКАМИ ДАННЫХ.
                ;8056   ;
                ;8057    T.E:
U 09BD, B65E,15 ;8058            MOV LS[BEGIN.TEST] TO WR[0]            ; установка бита 15 в WR[0] для слова управления и
                ;8059                                                   ; состояния
U 09BE, 3E80,15 ;8060            MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и  состояния. Бит
                ;8061                                                   ; 15 указывает начало теста  консольному процессору
U 09BF, 10E0,15 ;8062            MISC [SET.CP.ATTN]                     ; выдача сигнала CPU ATTN консольному процессору
                ;8063    WAIT.TE.0:
U 09C0, 889C,04 ;8064            JMP [WAIT.TE.0]                        ; зацикливание для ожидания ответа консольного
                ;8065                                                   ; процессора
U 09C1, 0A1A,AC ;8066            JSR [SETUP.1]                          ; установка масок, кода модуля и др.
                ;8067    WRITE.CSR0:
U 09C2, 3672,15 ;8068            MOV LS[ECC.DIS] TO WR[0]               ; установка бита ECC DIS В WR[0]
U 09C3, 4774,15 ;8069            BIS LS[DIAG.CHK] TO WR[0]              ; также установка бита DIAG CHK
U 09C4, C726,15 ;8070            BIS LS[#FF] TO WR[0]                   ; также установка CKBT
U 09C5, 8A17,FC ;8071            JSR [WRITE.CSR1]                       ; установка битов ECC DIS и DIAG CHK В CSR1. Установка
                ;8072                                                   ; тестовых CKBT в битах 6-0
U 09C6, 9D9C,F5 ;8073            MEM.REQ[MAINT.ECC.DATA] ADRS[ZERO] DT[LONG] ; выполнение записи единиц в CKBT (сейчас CSR0
                ;8074                                                   ; содержит единицы)
U 09C7, 329E,15 ;8075            WRITE.MEM LS[ONES]                     ; запись единичных данных в схему ECC. Биты CKBT получают
                ;8076                                                   ; из CSR1 единичные данные
U 09C8, 3022,15 ;8077            MOV MEM.DATA TO WR[0]                  ; чтение результата для окончания микроцикла
U 09C9, B700,15 ;8078            MOV LS[CKBTS.01111100] TO WR[0]        ; подготовка CKBT для данных - все нули или все единицы
U 09CA, A0C0,15 ;8079            COM WR[0]                              ; дополнение CKBT - корректировка для аппаратуры
U 09CB, 452C,15 ;8080            BIC LS[#FFFFFF00] TO WR[0]             ; очистка всех битов, кроме младшего байта
U 09CC, 4774,15 ;8081            BIS LS[DIAG.CHK] TO WR[0]              ; установка бита DIAG CHK В WR0
U 09CD, 8A17,FC ;8082            JSR [WRITE.CSR1]                       ; запись данных из WR0 В CSR1
U 09CE, 3713,15 ;8083            MOV LS[#FFFFFF80] TO WR[2]             ; подготовка маски ошибки в WR2 для проверки битов 6-0
                ;8084    LOOP.TE.1:
U 09CF, B640,15 ;8085            MOV LS[#1] TO WR[0]                    ; 1 в WR0
U 09D0, BE82,15 ;8086            MOV WR[0] TO LS[ERROR.NUMBER]          ; подготовка номера ошибки в местной памяти
U 09D1, E58A,15 ;8087            CLR LS[ERROR.MASK]                     ; проверка всех битов
U 09D2, 369E,95 ;8088            MOV LS[ONES] TO WR[1]                  ; ожидаемые данные
U 09D3, 9D9C,F5 ;8089            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данные из местной
                ;8090                                                   ; памяти
U 09D4, 329E,15 ;8091            WRITE.MEM LS[ONES]                     ; запись единичных данных в схему ECC
U 09D5, 3022,15 ;8092            MOV MEM.DATA TO WR[0]                  ; чтение данных обратно (должны быть не изменены)
U 09D6, 0869,3C ;8093            JSR [CHECK.RESULT]                     ; проверка результата
U 09D7, 889C,F4 ;8094            JMP [LOOP.TE.1]                        ; зацикливание при ошибке, если разрешено
                ;8095    TEST.TE.2:
U 09D8, FF82,15 ;8096            INC LS[ERROR.NUMBER]                   ; ошибка 2
U 09D9, BE88,15 ;8097            MOV WR[2] TO LS[ERROR.MASK]            ; маска ошибки для проверки битов 6-0
U 09DA, 369E,95 ;8098            MOV LS[ONES] TO WR[1]                  ; ожидаемые данные (CSR0 не изменен)
U 09DB, 9D9D,75 ;8099            MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]  ; подготовка чтения CSR0
U 09DC, 3022,15 ;8100            MOV MEM.DATA TO WR[0]                  ; выборка CKBT из CSR0. Они должны быть не изменены,
                ;8101                                                   ; после записи в начале этого теста
U 09DD, 0869,3C ;8102            JSR [CHECK.RESULT]                     ; проверка результата
U 09DE, 889C,F4 ;8103            JMP [LOOP.TE.1]                        ; зацикливание при ошибке, если разрешено
                ;8104    TEST.TE.3:
U 09DF, FF82,15 ;8105            INC LS[ERROR.NUMBER]                   ; ошибка 3
U 09E0, 5F7E,15 ;8106            MCOM LS[RDS] TO WR[0]                  ; сброс бита 31 в WR0. Установка других
U 09E1, 457C,15 ;8107            BIC LS[CRD] TO WR[0]                   ; сброс бита 30 в WR0
U 09E2, 3E8A,15 ;8108            MOV WR[0] TO LS[ERROR.MASK]            ; подготовка проверки битов 30 и 31
U 09E3, 2F82,95 ;8109            CLR WR[1]                              ; ожидаемые данные (оба бита сброшены)
```

```
U 09E4, 9D45,75 ;8110            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; подготовка чтения CSR1
U 09E5, 3022,15 ;8111            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 09E6, 0869,3C ;8112            JSR [CHECK.RESULT]                     ; проверка результата
U 09E7, 889C,F4 ;8113            JMP [LOOP.TE.1]                        ; зацикливание при ошибке, если разрешено
                ;8114    TEST.TE.4:
U 09E8, FF82,15 ;8115            INC LS[ERROR.NUMBER]                   ; ошибка 4
U 09E9, 5800,1D ;8116            SKIP.IF[MEM.REF.OK]                    ; пропуск, если нет ошибки памяти (должен быть пропуск)
U 09EA, 08A7,4C ;8117            JSR [ERROR.TE.4]                       ; ошибка, переход сюда. Сообщение об ошибке
                ;8118    TEST.TE.5:
U 09EB, FF82,15 ;8119            INC LS[ERROR.NUMBER]                   ; ошибка 5
U 09EC, E58A,15 ;8120            CLR LS[ERROR.MASK]                     ; проверка всех битов
                ;8121    LOOP.TE.5:
U 09ED, B69C,95 ;8122            MOV LS[ZERO] TO WR[1]                  ; ожидаемые данные
U 09EE, 9D9C,F5 ;8123            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8124                                                   ; памяти
U 09EF, B29C,15 ;8125            WRITE.MEM LS[ZERO]                     ; запись нулевых данных в схему ECC
U 09F0, 3022,15 ;8126            MOV MEM.DATA TO WR[0]                  ; чтение данных обратно (должны быть не изменены)
U 09F1, 0869,3C ;8127            JSR [CHECK.RESULT]                     ; проверка результата
U 09F2, 889E,D4 ;8128            JMP [LOOP.TE.5]                        ; зацикливание при ошибке, если разрешено
                ;8129    TEST.TE.6:
U 09F3, FF82,15 ;8130            INC LS[ERROR.NUMBER]                   ; ошибка 6
U 09F4, 3683,95 ;8131            MOV LS[ERROR.NUMBER] TO WR[3]          ; запоминание номера ошибки
U 09F5, DF7F,15 ;8132            MCOM LS[RDS] TO WR[2]                  ; сброс бита 31 в WR2, установка других
U 09F6, C57D,15 ;8133            BIC LS[CRD] TO WR[2]                   ; сброс бита 30 в WR2. Используется в качестве маски
                ;8134                                                   ; ошибки для ошибки 7
U 09F7, E5F8,15 ;8135            CLR LS[OS]                             ; очистка индекса
                ;8136    REPEAT.TE.6:
U 09F8, 36F6,15 ;8137            MOV LS[SHIFT.OS(4-0)] TO WR[0]         ; выборка тестовых данных ECC
U 09F9, BE88,15 ;8138            MOV WR[0] TO LS[ADDRESS.DATA]          ; пересылка данных в ячейку OTHER для распечатки
U 09FA, B670,15 ;8139            MOV LS[OTHER.DATA] TO WR[0]            ; установка бита 24
U 09FB, 3E80,15 ;8140            MOV WR[0] TO LS[CONTROL.STATUS]        ; сообщение консольному процессору для печати тестовых
                ;8141                                                   ; данных ECC под "OTHER" в сообщении об ошибке
                ;8142    LOOP.TE.6:
U 09FC, BE83,95 ;8143            MOV WR[3] TO LS[ERROR.NUMBER]          ; подготовка номера ошибки в местной памяти
U 09FD, E58A,15 ;8144            CLR LS[ERROR.MASK]                     ; проверка всех битов
U 09FE, B69C,95 ;8145            MOV LS[ZERO] TO WR[1]                  ; ожидаемые данные
U 09FF, 9D9C,F5 ;8146            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8147                                                   ; памяти
U 0A00, B2F6,15 ;8148            WRITE.MEM LS[SHIFT.OS(4-0)]            ; запись данных сдвигаемой единицы в схему ECC
U 0A01, 3022,15 ;8149            MOV MEM.DATA TO WR[0]                  ; чтение данных обратно (должны быть откорректированы на
                ;8150                                                   ; все нули)
U 0A02, 0869,3C ;8151            JSR [CHECK.RESULT]                     ; проверка результата
U 0A03, 889F,C4 ;8152            JMP [LOOP.TE.6]                        ; зацикливание при ошибке, если разрешено
                ;8153    TEST.TE.7:
U 0A04, FF82,15 ;8154            INC LS[ERROR.NUMBER]                   ; ошибка 7
U 0A05, BE88,15 ;8155            MOV WR[2] TO LS[ERROR.MASK]            ; подготовка проверки битов 30 и 31
U 0A06, 367C,95 ;8156            MOV LS[CRD] TO WR[1]                   ; ожидаемые данные (CRD установлен, RDS сброшен)
U 0A07, 9D45,75 ;8157            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; подготовка чтения CSR1
U 0A08, 3022,15 ;8158            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 0A09, 0869,3C ;8159            JSR [CHECK.RESULT]                     ; проверка результата
U 0A0A, 889F,C4 ;8160            JMP [LOOP.TE.6]                        ; зацикливание при ошибке, если разрешено
U 0A0B, 7FF8,15 ;8161            INC LS[OS]                             ; увеличение индекса для следующего набора
U 0A0C, B6F8,15 ;8162            MOV LS[OS] TO WR[0]                    ; подготовка проверки, все ли наборы использованы
                ;8163              BIT LS[BIT5] WITH WR[0],             ; проверка, увеличен ли OS до 20(H) и
U 0A0D, 594A,35 ;8164            DT(LONG)&SET.ALU.CC                    ; и установка кодов условий
```

```
U 0A0E, B89F,89 ;8165            JMP.IF[BITS.CLR] TO [REPEAT.TE.6]      ; повторение со следующим набором, если не все
                ;8166                                                   ; использованы
                ;8167    TEST.TE.8:
U 0A0F, FF82,15 ;8168            INC LS[ERROR.NUMBER]                   ; ошибка 8
                ;8169    LOOP.TE.8:
U 0A10, 5B00,1D ;8170            SKIP.IF[MEM.REF.OK]                    ; пропуск, если нет ошибки памяти (не должно быть
                ;8171                                                   ; пропуска)
U 0A11, 08A1,34 ;8172            JMP [TEST.TE.9]                        ;
U 0A12, 08A7,7C ;8173            JSR [ERROR.TE.8]                       ; ошибка, если переход сюда. сообщение об ошибке
                ;8174    TEST.TE.9:
U 0A13, FF82,15 ;8175            INC LS[ERROR.NUMBER]                   ; ошибка 9
U 0A14, E58A,15 ;8176            CLR LS[ERROR.MASK]                     ; проверка всех битов
U 0A15, E5F8,15 ;8177            CLR LS[OS]                             ; очистка индекса
                ;8178    REPEAT.TE.9:
U 0A16, 5FF6,15 ;8179            MCOM LS[SHIFT.OS(4-0)] TO WR[0]        ; выборка тестовых данных, дополнение и пересылка в
                ;8180                                                   ; WR[0]
U 0A17, 3E10,15 ;8181            MOV WR[0] TO LS[T8]                    ; запоминание в ячейке временного хранения местной
                ;8182                                                   ; памяти
U 0A18, BE88,15 ;8183            MOV WR[0] TO LS[ADDRESS.DATA]          ; пересылка в ячейку других данных для распечатки
U 0A19, B670,15 ;8184            MOV LS[OTHER.DATA] TO WR[0]            ; установка бита 24
U 0A1A, 3E80,15 ;8185            MOV WR[0] TO LS[CONTROL.STATUS]        ; сообщение консольному процессору для печати тестовых
                ;8186                                                   ; данных ECC под "OTHER" в сообщении об ошибке
                ;8187    LOOP.TE.9:
U 0A1B, 369E,95 ;8188            MOV LS[ONES] TO WR[1]                  ; ожидаемые данные
U 0A1C, 9D9C,F5 ;8189            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8190                                                   ; памяти
U 0A1D, 3210,15 ;8191            WRITE.MEM LS[T8]                       ; запись данных сдвигаемого нуля в схему ECC
U 0A1E, 3022,15 ;8192            MOV MEM.DATA TO WR[0]                  ; чтение данных обратно (должны быть откорректированы на
                ;8193                                                   ; все единицы)
U 0A1F, 0869,3C ;8194            JSR [CHECK.RESULT]                     ; проверка результата
U 0A20, 88A1,B4 ;8195            JMP [LOOP.TE.9]                        ; зацикливание при ошибке, если разрешено
U 0A21, 7FF8,15 ;8196            INC LS[OS]                             ; увеличение индекса для следующего набора
U 0A22, B6F8,15 ;8197            MOV LS[OS] TO WR[0]                    ; подготовка проверки, все ли наборы использованы
                ;8198              BIT LS[BIT5] WITH WR[0],             ; проверка, увеличен ли OS до 20(H)
U 0A23, 594A,35 ;8199            DT(LONG)&SET.ALU.CC                    ; и установка кодов условий
U 0A24, 88A1,69 ;8200            JMP.IF[BITS.CLR] TO [REPEAT.TE.9]      ; повторение со следующим набором, если не все
                ;8201                                                   ; использованы
                ;8202    TEST.TE.A:
U 0A25, E580,15 ;8203            CLR LS[CONTROL.STATUS]                 ; очистка слова управления и состояния
U 0A26, FF82,15 ;8204            INC LS[ERROR.NUMBER]                   ; ошибка A
U 0A27, 3683,95 ;8205            MOV LS[ERROR.NUMBER] TO WR[3]          ; запоминание номера ошибки
U 0A28, DF7F,15 ;8206            MCOM LS[RDS] TO WR[2]                  ; очистка бита 31 в WR2, установка других
U 0A29, C57D,15 ;8207            BIC LS[CRD] TO WR[2]                   ; очистка  бита 30 в WR2. Используется в качестве маски
                ;8208                                                   ; ошибки 8
                ;8209    LOOP.TE.A:
U 0A2A, BE83,95 ;8210            MOV WR[3] TO LS[ERROR.NUMBER]          ; установка номера ошибки в местной памяти
U 0A2B, E58A,15 ;8211            CLR LS[ERROR.MASK]                     ; проверка всех битов
U 0A2C, B6C0,95 ;8212            MOV LS[#3(H)] TO WR[1]                 ; ожидаемые данные
U 0A2D, 9D9C,F5 ;8213            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8214                                                   ; памяти
U 0A2E, B2C0,15 ;8215            WRITE.MEM LS[#3(H)]                    ; запись данных 3(H) в схему ECC (ошибка двух битов)
U 0A2F, 3022,15 ;8216            MOV MEM.DATA TO WR[0]                  ; чтение данных обратно (не должны быть модифицированы)
U 0A30, 0869,3C ;8217            JSR [CHECK.RESULT]                     ; проверка результата
U 0A31, 08A2,A4 ;8218            JMP [LOOP.TE.A]                        ; зацикливание при ошибке, если разрешено
                ;8219    TEST.TE.B:
```

```
U 0A32, FF82,15 ;8220            INC LS[ERROR.NUMBER]                        ; ошибка B
U 0A33, BE8B,15 ;8221            MOV WR[2] TO LS[ERROR.MASK]                 ; подготовка для проверки битов 30 и 31
U 0A34, B67E,95 ;8222            MOV LS[RDS] TO WR[1]                        ; ожидаемые данные (RDS  установлен, CRD сброшен)
U 0A35, 9D45,75 ;8223            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; подготовка чтения CSR1
U 0A36, 3022,15 ;8224            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 0A37, 0869,3C ;8225            JSR [CHECK.RESULT]                          ; проверка результата
U 0A38, 08A2,A4 ;8226            JMP [LOOP.TE.A]                             ; зацикливание при ошибке, если разрешено
                ;8227    TEST.TE.C:
U 0A39, FF82,15 ;8228            INC LS[ERROR.NUMBER]                        ; ошибка C
U 0A3A, 5800,1D ;8229            SKIP.IF[MEM.REF.OK]                         ; пропуск, если нет ошибки памяти (не должно быть
                ;8230                                                        ; пропуска)
U 0A3B, 08A3,D4 ;8231            JMP [TEST.TE.D]                             ;
U 0A3C, 88A7,AC ;8232            JSR [ERROR.TE.C]                            ; ошибка, если переход сюда. Сообщение об ошибке
                ;8233    TEST.TE.D:
U 0A3D, FF82,15 ;8234            INC LS[ERROR.NUMBER]                        ; ошибка D
U 0A3E, 3683,95 ;8235            MOV LS[ERROR.NUMBER] TO WR[3]               ; запоминание номера ошибки
U 0A3F, DF7F,15 ;8236            MCOM LS[RDS] TO WR[2]                       ; сброс бита 31 в WR2, установка других
U 0A40, C57D,15 ;8237            BIC LS[CRD] TO WR[2]                        ; сброс бита 30 в WR2. Используется в качестве маски
                ;8238                                                        ; ошибки для ошибки E
                ;8239    LOOP.TE.D:
U 0A41, BE83,95 ;8240            MOV WR[3] TO LS[ERROR.NUMBER]               ; установка номера ошибки в местной памяти
U 0A42, E58A,15 ;8241            CLR LS[ERROR.MASK]                          ; проверка всех битов
U 0A43, B734,95 ;8242            MOV LS[#30000] TO WR[1]                     ; ожидаемые данные
U 0A44, 9D9C,F5 ;8243            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8244                                                        ; памяти
U 0A45, B334,15 ;8245            WRITE.MEM LS[#30000]                        ; запись данных 30000(H) в схему ECC (ошибка двух
                ;8246                                                        ; битов)
U 0A46, 3022,15 ;8247            MOV MEM.DATA TO WR[0]                       ; чтение данных обратно (не должны быть модифицированы)
U 0A47, 0869,3C ;8248            JSR [CHECK.RESULT]                          ; проверка результата
U 0A48, 88A4,14 ;8249            JMP [LOOP.TE.D]                             ; зацикливание при ошибке, если разрешено
                ;8250    TEST.TE.E:
U 0A49, FF82,15 ;8251            INC LS[ERROR.NUMBER]                        ; ошибка E
U 0A4A, BE8B,15 ;8252            MOV WR[2] TO LS[ERROR.MASK]                 ; подготовка для проверки битов 30 и 31
U 0A4B, B67E,95 ;8253            MOV LS[RDS] TO WR[1]                        ; ожидаемые данные (RDS установлен, CRD сброшен)
U 0A4C, 9D45,75 ;8254            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; подготовка чтения CSR1
U 0A4D, 3022,15 ;8255            MOV MEM.DATA TO WR[0]                       ; выборка результата
U 0A4E, 0869,3C ;8256            JSR [CHECK.RESULT]                          ; проверка результата
U 0A4F, 88A4,14 ;8257            JMP [LOOP.TE.D]                             ; зацикливание при ошибке, если разрешено
                ;8258    TEST.TE.F:
U 0A50, B700,15 ;8259            MOV LS[CKBTS.0111100] TO WR[0]              ; подготовка CKBT для данных - все нули или все единицы
U 0A51, A0C0,15 ;8260            COM WR[0]                                   ; дополнение CKBT - коррекция для аппаратуры
U 0A52, 452C,15 ;8261            BIC LS[#FFFFFF00] TO WR[0]                  ; очистка всех битов, кроме младшего байта
U 0A53, 4774,15 ;8262            BIS LS[DIAG.CHK] TO WR[0]                   ; установка бита DIAG CHK В WR0
U 0A54, 4778,15 ;8263            BIS LS[INH.CRD] TO WR[0]                    ; установка бита запрета сообщения о корректируемых
                ;8264                                                        ; ошибках данных
U 0A55, 8A17,FC ;8265            JSR [WRITE.CSR1]                            ; запись в CSR1 данных из WR0
U 0A56, FF82,15 ;8266            INC LS[ERROR.NUMBER]                        ; ошибка F
U 0A57, 3683,95 ;8267            MOV LS[ERROR.NUMBER] TO WR[3]               ; сохранение номера ошибки в WR3
                ;8268    LOOP.TE.F:
U 0A58, BE83,95 ;8269            MOV WR[3] TO LS[ERROR.NUMBER]               ; установка номера ошибки
U 0A59, 2F82,95 ;8270            CLR WR[1]                                   ; ожидаемые данные
U 0A5A, 9D9C,F5 ;8271            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи CKBT из CSR1, данных из местной
                ;8272                                                        ; памяти
U 0A5B, 3240,15 ;8273            WRITE.MEM LS[#1]                            ; запись данных = 1 в схему ECC (ошибка одиночного
                ;8274                                                        ; бита)
```

```
U 0A5C, 3022,15 ;8275            MOV MEM.DATA TO WR[0]                          ; чтение данных обратно для завершения цикла (без
                ;8276                                                           ; проверки)
U 0A5D, 9D45,75 ;8277            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]          ; подготовка чтения CSR1
U 0A5E, 3022,15 ;8278            MOV MEM.DATA TO WR[0]                          ; выборка результата
U 0A5F, 0869,3C ;8279            JSR [CHECK.RESULT]                             ; проверка результата
U 0A60, 08A5,84 ;8280            JMP [LOOP.TE.F]                                ; зацикливание при ошибке, если разрешено
                ;8281    TEST.TE.10:
U 0A61, FF82,15 ;8282            INC LS[ERROR.NUMBER]                           ; ошибка 10
                ;8283    LOOP.TE.10:
U 0A62, 5B00,1D ;8284            SKIP.IF[MEM.REF.OK]                            ; пропуск, если нет ошибки памяти (должен быть пропуск)
U 0A63, 08A7,DC ;8285            JSR [ERROR.TE.10]                              ; ошибка, если переход сюда. Сообщение об ошибке
                ;8286    TEST.TE.11:
U 0A64, FF82,15 ;8287            INC LS[ERROR.NUMBER]                           ; ошибка 11
U 0A65, 3683,95 ;8288            MOV LS[ERROR.NUMBER] TO WR[3]                  ; сохранение номера ошибки в WR3
                ;8289    LOOP.TE.11:
U 0A66, BE83,95 ;8290            MOV WR[3] TO LS[ERROR.NUMBER]                  ; установка номера ошибки
U 0A67, B67E,95 ;8291            MOV LS[RDS] TO WR[1]                           ; ожидаемые данные
U 0A68, 9D9C,F5 ;8292            MEM.REQ[MAINT.ECC.DATA] ADRS[#0] DT[LONG] ; подготовка записи СКВТ из CSR1, данных из местной
                ;8293                                                           ; памяти
U 0A69, B2C0,15 ;8294            WRITE.MEM LS[#3(H)]                            ; запись данных 3(H) в схему ECC (ошибка двух битов)
U 0A6A, 3022,15 ;8295            MOV MEM.DATA TO WR[0]                          ; чтение данных обратно для завершения цикла (без
                ;8296                                                           ; проверки)
U 0A6B, 9D45,75 ;8297            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]          ; подготовка чтения CSR1
U 0A6C, 3022,15 ;8298            MOV MEM.DATA TO WR[0]                          ; выборка результата
U 0A6D, 0869,3C ;8299            JSR [CHECK.RESULT]                             ; проверка результата
U 0A6E, 88A6,64 ;8300            JMP [LOOP.TE.11]                               ; зацикливание при ошибке, если разрешено
                ;8301    TEST.TE.12:
U 0A6F, FF82,15 ;8302            INC LS[ERROR.NUMBER]                           ; ошибка 12
U 0A70, 5B00,1D ;8303            SKIP.IF[MEM.REF.OK]                            ; пропуск, если нет ошибки памяти (не должно быть
                ;8304                                                           ; пропуска)
U 0A71, 88A8,D4 ;8305            JMP [END.TE]                                   ; нет ошибки. Тест выполнен
U 0A72, 88A8,0C ;8306            JSR [ERROR.TE.12]                              ; ошибка, если переход сюда. Сообщение об ошибке
U 0A73, 88A8,D4 ;8307            JMP [END.TE]                                   ; тест выполнен
                ;8308    ERROR.TE.4:
U 0A74, 88A8,3C ;8309            JSR [REPORT.TE.SKIP.ERROR]                     ; вывод сообщения об ошибке с неприменимыми ожидаемыми
                ;8310                                                           ; и полученными данными
U 0A75, 889C,F4 ;8311            JMP [LOOP.TE.1]                                ; зацикливание при ошибке, если разрешено
U 0A76, 5B00,14 ;8312            RETURN                                         ; возврат без зацикливания при ошибке
                ;8313    ERROR.TE.8:
U 0A77, 88A8,3C ;8314            JSR [REPORT.TE.SKIP.ERROR]                     ; вывод сообщения об ошибке с неприменимыми ожидаемыми
                ;8315                                                           ; и полученными данными
U 0A78, 08A1,04 ;8316            JMP [LOOP.TE.8]                                ; зацикливание при ошибке, если разрешено
U 0A79, 5B00,14 ;8317            RETURN                                         ; возврат без зацикливания при ошибке
                ;8318    ERROR.TE.C:
U 0A7A, 88A8,3C ;8319            JSR [REPORT.TE.SKIP.ERROR]                     ; вывод сообщения об ошибке с неприменимыми ожидаемыми
                ;8320                                                           ; и полученными данными
U 0A7B, 08A2,A4 ;8321            JMP [LOOP.TE.A]                                ; зацикливание при ошибке, если разрешено
U 0A7C, 5B00,14 ;8322            RETURN                                         ; возврат без зацикливания при ошибке
                ;8323    ERROR.TE.10:
U 0A7D, 88A8,3C ;8324            JSR [REPORT.TE.SKIP.ERROR]                     ; вывод сообщения об ошибке с неприменимыми ожидаемыми
                ;8325                                                           ; и полученными данными
U 0A7E, 08A5,84 ;8326            JMP [LOOP.TE.F]                                ; зацикливание при ошибке, если разрешено
U 0A7F, 5B00,14 ;8327            RETURN                                         ; возврат без зацикливания при ошибке
                ;8328    ERROR.TE.12:
U 0A80, 88A8,3C ;8329            JSR [REPORT.TE.SKIP.ERROR]                     ; вывод сообщения об ошибке с неприменимыми ожидаемыми
```

```
                 ;8330                                                  ; и полученными данными
U 0AB1, 88A6,64  ;8331          JMP [LOOP.TE.11]                       ; зацикливание при ошибке, если разрешено
U 0AB2, 5B00,14  ;8332          RETURN                                 ; возврат без зацикливания при ошибке
                 ;8333  REPORT.TE.SKIP.ERROR:
U 0AB3, B66E,15  ;8334          MOV LSI[NA.EXP.REC] TO WR[0]           ; установка бита 23 в WR0 для указания - не печатать
                 ;8335                                                  ; ожидаемых и полученных данных, если появляется ошибка
U 0AB4, 3E80,15  ;8336          MOV WR[0] TO LSI[CONTROL.STATUS]       ; восстановление слова управления и состояния
U 0AB5, 2F80,15  ;8337          CLR WR[0]                              ; очистка WR0
U 0AB6, 369E,95  ;8338          MOV LSI[ONES] TO WR[1]                 ; установка единиц в WR1. Сейчас подпрограмма обработки
                 ;8339                                                  ; ошибки будет сообщать об ошибке (но ожидаемые и
                 ;8340                                                  ; полученные данные не будут печататься)
U 0AB7, 0869,3C  ;8341          JSR [CHECK.RESULT]                     ; сообщение об ошибке
U 0AB8, 88AB,B4  ;8342          JMP [ERROR.LOOP.TE]                    ; зацикливание при ошибке, если переход сюда
U 0AB9, E580,15  ;8343          CLR LSI[CONTROL.STATUS]                ; снова разрешается распечатка ожидаемых и полученных
                 ;8344                                                  ; данных
U 0ABA, D800,16  ;8345          RETURN+1                               ; возврат без зацикливания при ошибке
                 ;8346  ERROR.LOOP.TE:
U 0ABB, E580,15  ;8347          CLR LSI[CONTROL.STATUS]                ; снова разрешается распечатка ожидаемых и полученных
                 ;8348                                                  ; данных
U 0ABC, 5B00,14  ;8349          RETURN                                 ; возврат в зацикливание при ошибке
                 ;8350  END.TE:
```

```
;8351    .PAGE "ТЕСТЫ ЦИКЛИЧЕСКИХ СДВИГОВ ДАННЫХ"
;8352    .TOC "ТЕСТ F - циклический сдвиг на 9 битов вправо (модуль MCT)"
;8353    ;
;8354    ; ОПИСАНИЕ ТЕСТА:
;8355    ;
;8356    ;       Этот тест проверяет ПМЛ сдвигателей данных на циклический сдвиг байта
;8357    ;   вправо. Выдается инструкция MEM.REQ с полем MF=ROTATE.BYTE.RIGHT (6) (цикли-
;8358    ;   ческий сдвиг байта вправо). Путь данных следующий: выдается инструкция MEM.
;8359    ;   REQ с тестовыми данными, поступающими из местной памяти, и полем MF=ROTATE.
;8360    ;   BYTE.RIGHT. Тестовые данные загружаются в регистр виртуального адреса и вы-
;8361    ;   полняется начальное ветвление на микропрограмму памяти для циклического сдви-
;8362    ;   га вправо (см.описание начального ветвления в начале этого листинга). По ад-
;8363    ;   ресу начального ветвления устанавливается MEMORY BUSY, разрешаются биты РА
;8364    ;   фического адреса и регистр обхода виртуального адреса (VAR BYPASS). В следу-
;8365    ;   ющем цикле сдвигатели данных запрешаются, VAR BYPASS, ADDR PH и MEMORY BUSY
;8366    ;   отстаются возбужденными и разрешаются передатчики MCT. Затем тестовые данные,
;8367    ;   переданные центральным процессором, снова помешаются на шину BUS MC со сдви-
;8368    ;   гом на 1 бит вправо (благодаря передатчикам). В следующем цикле биты шины BUS
;8369    ;   MC передаются на BUS ARRAY через ПМЛ сдвигателей и открываются входные регист-
;8370    ;   ры схем ECC, ТАК что данные могут быть сохранены. Выходные регистры схем ECC
;8371    ;   также открываются. VAR BYPASS, ADDR PH и MEMORY BUSY остаются возбужденными.
;8372    ;   В следующем цикле данные, сохраняемые в схеме ECC, выдаются на шину BUS ARRAY
;8373    ;   и ПМЛ сдвигателей разрешаются для циклического сдвига этих данных на 8 битов
;8374    ;   вправо, а результат пересылается на шину BUS MC. VAR BYPASS и MEMORY BUSY ос-
;8375    ;   таются возбужденными (VAR BYPASS не обязательно оставить возбужденным, но он
;8376    ;   ни на что не влияет). Следующий цикл циклится при сброшенном MEMORY BUSY в ожи-
;8377    ;   дании сигнала DATA RCVD центрального процесора. Инструкция MOV в микропрограмме
;8378    ;   центрального процессора вызывает возбуждение сигнала DATA RCVD и принимает
;8379    ;   сдвинутые данные с шины BUS MC для проверки. Когда сигнал центрального процес-
;8380    ;   сора DATA RCVD возбуждается, микропрограмма MCT переходит в ячейку, которая вы-
;8381    ;   полняет переход в холостой цикл. Используемые тестовые данные - сдигаемая
;8382    ;   единица.
;8383    ;
;8384    ; ПРЕДПОЛОЖЕНИЯ:
;8385    ;
;8386    ;   Предполагается, что предыдущие тесты выполены успешно.
;8387    ;
;8388    ; ШАГИ ТЕСТА:
;8389    ;
;8390    ;   1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;8391    ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;8392    ;      памяти.
;8393    ;   2. Загрузка WR2 тестовыми данными из местной памяти.
;8394    ;   3. Пересылка 1 в бит 23 WR2 для получения ожидаемых данных в правом разря-
;8395    ;      де. Запоминание этих данных в WR1 в качестве ожидаемых данных.
;8396    ;   4. Выдача инструкции MEM.REQ С DT=LONGWORD (11) и MF=ROTATE.BYTE.RIGHT (6),
;8397    ;      с использованием в качестве источника данных ячейки местной памяти, ин-
;8398    ;      дексируемой регистром OS.
;8399    ;   5. Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата.
;8400    ;   6. Сдвиг WR2 влево и повторение шагов с 4 по 5, пока все тестовые данные не
;8401    ;      будут проверены.
;8402    ;
;8403    ; ОШИБКИ:
;8404    ;
;8405    ;   ошибка 1 - циклический сдвиг данных выполняется неправильно с данными со
```

```
                ;8406   ;               сдвигаемой единицей.
                ;8407   ;
                ;8408   ; НАЛАДКА:
                ;8409   ;
                ;8410   ;
                ;8411   ;       ОШИБКА 1 - Эта ошибка указывает на возможную неисправность ПМЛ сдвигателей
                ;8412   ;    данных или ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Входы A0 L и A1 L ПМЛ сдвигателей дан-
                ;8413   ;    ных могут быть проверены после останова центрального процессора на инструкции
                ;8414   ;    MOV MEM.DATA TO WR[0]. Память будет циклиться в ожидании сигнала DATA RCVD
                ;8415   ;    центрального процессора. Необходимо проверить, что A0 L низкий и A1 L высокий.
                ;8416   ;    Если они правильные, необходимо проверить низкий уровень сигналов 2ND MEM CYC
                ;8417   ;    H и MDR DAT OUT EN L. Если здесь правильно, подозреваются сами ПМЛ сдвигателей
                ;8418   ;    данных. Если входы A0 L или A1 L неправильные, необходимо их проверить на вы-
                ;8419   ;    ходе ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если там они неправильные и пошаговый режим
                ;8420   ;    МСТ возможен, необходимо установить пошаговый режим МСТ, остановить центральный
                ;8421   ;    процессор на инструкции MEM.REQ, а память остановить на цикле, который выпол-
                ;8422   ;    няет SET.ROT.BYTE и ROT.CLOCK. Необходимо проверить высокие уровни на входах
                ;8423   ;    ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ CPH/UBL и ROT C0 H и низкий уровень на входе ROT C1
                ;8424   ;    H. Сигнал ROT CLOCK H также должен быть высоким. Если они правильные, повидимо-
                ;8425   ;    му ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ неисправна. Если пошаговый режим МСТ невозможен,
                ;8426   ;    эти входы необходимо проверить при зацикливании по ошибке.
                ;8427   ;       Другим источником ошибки может быть ПМЛ УПР.РЕГИСТРАМ ДАННЫХ ПАМЯТИ. Необ-
                ;8428   ;    ходимо проверить низкий уровень на выходах (ножки 12-15), которые разрешают
                ;8429   ;    сигналу LAT OUTPUT BYT L стать высоким. Если здесь неправильно, необходимо
                ;8430   ;    проверить низкий уровень на входе 2ND MEM CYC H и высокий уровень на входах
                ;8431   ;    CPH/UBL, ROT C0 H, A1 L и A0 L (в то время, когда входы регистров схемы ECC
                ;8432   ;    открыты). Если здесь правильно, а ножки 12-15 не имеют низкого уровня, ПМЛ
                ;8433   ;    неисправна.
                ;8434    T.F:
U 0A8D, B65E,15 ;8435            MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;8436                                                    ; состояния
U 0A8E, 3E80,15 ;8437            MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;8438                                                    ; 15 указывает начало теста консольному процессору
U 0A8F, 10E0,15 ;8439            MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN консольному процессору
                ;8440    WAIT.TF.0:
U 0A90, 88A9,04 ;8441            JMP [WAIT.TF.0]                         ; зацикливание для ожидания ответа консольного
                ;8442                                                    ; процессора
U 0A91, 0A1A,AC ;8443            JSR [SETUP.1]                           ; установка масок,когда модуля и др.
U 0A92, E5F8,15 ;8444            CLR LS[OS]                              ; очистка индекса
U 0A93, 366F,15 ;8445            MOV LS[BIT23] TO WR[2]                  ; первый эталон ожидаемых данных
                ;8446    LOOP.TF.1:
U 0A94, A004,95 ;8447            MOV WR[2] TO WR[1]                      ; ожидаемые данные
U 0A95, 19F7,75 ;8448            MEM.REQ[ROTATE.BYTE.RIGHT] ADRS[SHIFT.OS(4-0)] DT[LONG] ; выполнение сдвига данных вправо
U 0A96, 3022,15 ;8449            MOV MEM.DATA TO WR[0]                   ; выробка результата
U 0A97, 0869,3C ;8450            JSR [CHECK.RESULT]                      ; проверка результата
U 0A98, 08A9,44 ;8451            JMP [LOOP.TF.1]                         ; зацикливание при ошибке, если разрешено
U 0A99, 23C1,15 ;8452            ROL WR[2]                               ; следующие ожидаемые данные
U 0A9A, 7FF8,15 ;8453            INC LS[OS]                              ; увеличение индекса для следующего набора
U 0A9B, B6F8,15 ;8454            MOV LS[OS] TO WR[0]                     ; подготовка проверки, все ли наборы использованы
                ;8455              BIT LS[BIT5] WITH WR[0],              ; проверка,увеличен ли OS до 20(H)
U 0A9C, 594A,35 ;8456            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0A9D, 88A9,49 ;8457            JMP.IF[BITS.CLR] TO [LOOP.TF.1]         ; повторение со следующим набором, если не все
                ;8458                                                    ; использованы
                ;8459    END.TF:
```

```
;8460    .PAGE "ТЕСТ 10 - циклический сдвиг на 15 битов влево (модуль МСТ)"
;8461    ;
;8462    ; ОПИСАНИЕ ТЕСТА:
;8463    ;
;8464    ;       Этот тест проверяет ПМЛ сдвигателей данных при циклическом сдвиге влево
;8465    ;    на слово. Выдается инструкция MEM.REQ с полем MF=WORD.SWAP(7). Путь данных
;8466    ;    следующий: выдается инструкция MEM.REQ с используемыми тестовыми данными, пос-
;8467    ;    тупающими из местной памяти и полем MF=WORD.SWAP. Тестовые данные загружают-
;8468    ;    ся в регистр виртуального адреса и выполняется начальное ветвление микропрог-
;8469    ;    раммы памяти на циклический сдвиг влево (см. описание начального ветвления в
;8470    ;    начале этого листинга). По адресу начального ветвления устанавливается MEMORY
;8471    ;    BUSY, разрешаются биты PA и VAR BYPASS. В следующем цикле запрещаются сдвига-
;8472    ;    тели данных, остаются возбужденными VAR BYPASS, ADDR PH и MEMORY BUSY и раз-
;8473    ;    решаются передатчики МСТ. Этим тестовые данные передаются обратно на шину BUS
;8474    ;    MC помимо центрального процессора, сдвинутые на 1 бит вправо (благодаря пере-
;8475    ;    датчикам). В следующем цикле биты шины BUS MC передаются на BUS ARRAY через
;8476    ;    ПМЛ сдвигателей и открываются входные регистры схемы ECC, так что данные могут
;8477    ;    быть сохранены. Выходные регистры схемы ECC так же разрешаются. VAR BYPASS,
;8478    ;    ADDR PH и MEMORY BUSY остаются возбужденными. В следующем цикле разрешается
;8479    ;    выдача на шину BUS ARRAY данных, сохраняемых в схеме ECC, разрешаются ПМЛ сдви-
;8480    ;    гателей для циклического сдвига этих данных на 16 битов влево и результат пе-
;8481    ;    ресылается на шину BUS MC. VAR BYPASS и MEMORY BUSY остаются возбужденными
;8482    ;    (VAR BYPASS не обязательно оставить возбужденным, но он ни на что не влияет).
;8483    ;    Следующий цикл циклится в ожидании сигнала DATA RCVD центрального процессора
;8484    ;    со сброшенным MEMORY BUSY. Инструкция MOV в микропрограмме центрального про-
;8485    ;    цессора вызовет возбуждение сигнала DATA RCVD и примет сдвинутые данные с шины
;8486    ;    BUS MC для проверки. Когда сигнал центрального процессора DATA RCVD возбужда-
;8487    ;    дается, микропрограмма МСТ переходит в ячейку, которая выполняет переход в хо-
;8488    ;    лостой цикл. Используемые тестовые данные - сдвигаемая единица.
;8489    ;
;8490    ; ПРЕДПОЛОЖЕНИЯ:
;8491    ;
;8492    ;   Предполагается, что предыдущие тесты выполнены успешно.
;8493    ;
;8494    ; ШАГИ ТЕСТА:
;8495    ;
;8496    ;       1.Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;8497    ;         (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;8498    ;         памяти.
;8499    ;       2.Загрузка в WR2 тестовых данных из местной памяти.
;8500    ;       3.Пересылка 1 в бит 15 WR2 для получения ожидаемых данных в правом раз-
;8501    ;         ряде. Запоминание этих данных в качестве ожидаемых данных.
;8502    ;       4.Выдача инструкции MEM.REQ с DT=LONGWORD (11) и MF=WORD.SVAP (7), исполь-
;8503    ;         зуя ячейку местной памяти, индексируемую регистром OS, в качестве источ-
;8504    ;         ника данных.
;8505    ;       5.Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата.
;8506    ;       6.Сдвиг WR2 влево и повторение шагов с 4 по 5, пока все тестовые данные не
;8507    ;         будут проверены.
;8508    ;
;8509    ; ОШИБКИ:
;8510    ;
;8511    ;   ошибка 1 - циклический сдвиг данных выполняется неправильно с данными со
;8512    ;              сдвигаемой единицей.
;8513    ;
;8514    ; НАЛАДКА:
```

```
                ;8515   ;
                ;8516   ;       ОШИБКА 1 - Эта ошибка указывает на возможную неисправность ПМЛ сдвигателей
                ;8517   ;    данных или ПМЛ УПР. СДВИГАТЕЛЕМ ДАННЫХ. Входы A0 L и A1 L ПМЛ сдвигателей дан-
                ;8518   ;    ных могут быть проверены после останова центрального процессора на инструк-
                ;8519   ;    ции MOV.MEM.DATA TO WR[0]. Память будет циклиться в ожидании сигнала CPU DATA
                ;8520   ;    RCVD. необходимо проверить, что A0 L высокий и A1 L низкий. Если правильно,
                ;8521   ;    необходимо проверить низкий уровень сигналов 2ND MEM.CYC H и MDR DAT OUT EN L.
                ;8522   ;    Если правильно, подозреваются сами ПМЛ сдвигателей данных. Если входы A0 L и
                ;8523   ;    A1 L неправильные, необходимо их проверить на выходе ПМЛ УПР.СДВИГАТЕЛЕМ ДАН-
                ;8524   ;    НЫХ. Если они неправильные, и пошаговый режим MCT возможен, следует разрешить
                ;8525   ;    пошаговый режим MCT, центральный процессор остановить на инструкции MEM REQ,
                ;8526   ;    а память останавливается на цикле, который выполняет SET.ROT.BYTE и ROT.CLOCK.
                ;8527   ;    необходимо проверить высокие уровни на входах CPH/UBL и ROT C1 L ПМЛ УПР.СДВИ-
                ;8528   ;    ГАТЕЛЕМ ДАННЫХ и низкий уровень на входе ROT C0 H. Сигнал ROT CLOCK H также
                ;8529   ;    должен быть высоким. Если правильно, повидимому, неисправна ПМЛ УПР.СДВИГАТЕ-
                ;8530   ;    ЛЕМ ДАННЫХ. Если пошаговый режим MCT невозможен, эти входы необходимо проверить
                ;8531   ;    при зацикливании по ошибке.
                ;8532   ;
                ;8533    T.10:
U 0A9E, B65E,15 ;8534            MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR[0] для слова управления и
                ;8535                                                       ; состояния
U 0A9F, 3E80,15 ;8536            MOV WR[0] TO LS[CONTROL.STATUS]            ; установка бита 15 в слове управления и состояния. Бит
                ;8537                                                       ; 15 указывает начало теста консольному процессору
U 0AA0, 10E0,15 ;8538            MISC [SET.CP.ATTN]                         ; выдача сигнала CPU ATTN консольному процессору
                ;8539    WAIT.T10.0:
U 0AA1, 08AA,14 ;8540            JMP [WAIT.T10.0]                           ; зацикливание для ожидания ответа консольного
                ;8541                                                       ; процессора
U 0AA2, 0A1A,AC ;8542            JSR [SETUP.1]                              ; установка масок, кода модуля и др.
U 0AA3, E5F8,15 ;8543            CLR LS[OS]                                 ; очистка индекса
U 0AA4, 365F,15 ;8544            MOV LS[BIT15] TO WR[2]                     ; первый набор ожидаемых данных
                ;8545    LOOP.T10.1:
U 0AA5, A004,95 ;8546            MOV WR[2] TO WR[1]                         ; ожидаемые данные
U 0AA6, 99F7,F5 ;8547            MEM.REQ[WORD.SWAP] ADRS[SHIFT.OS(4-0)] DT[LONG] ; выполнение сдвига данных влево
U 0AA7, 3022,15 ;8548            MOV  MEM.DATA TO WR[0]                     ; выборка результата
U 0AA8, 0869,3C ;8549            JSR [CHECK.RESULT]                         ; проверка результата
U 0AA9, 88AA,54 ;8550            JMP [LOOP.T10.1]                           ; зацикливание при ошибке, если разрешено
U 0AAA, 23C1,15 ;8551            ROL WR[2]                                  ; следующий набор ожидаемых данных
U 0AAB, 7FF8,15 ;8552            INC LS[OS]                                 ; подготовка проверки, все ли наборы использованы
U 0AAC, B6F8,15 ;8553            MOV LS[OS] TO WR[0]                        ; проверка, увеличен ли OS до 20(X) и установка кодов
                ;8554                                                       ; условий
                ;8555              BIT LS[BIT5] WITH WR[0],                 ;
U 0AAD, 594A,35 ;8556            DT(LONG)&SET.ALU.CC                        ;
U 0AAE, 08AA,59 ;8557            JMP.IF[BITS.CLR] TO [LOOP.T10.1]           ; повторение со следующим набором, если не все
                ;8558                                                       ; использованы
                ;8559    END.T10:
```

```
;8560    .PAGE  "ТЕСТЫ БУФЕРА ТРАНСЛЯЦИИ, ПАМЯТИ ОТОБРАЖЕНИЯ ОБЩЕЙ ШИНЫ И ПРИЗНАКОВ"
;8561    .TOC "ТЕСТ 11 - тест памяти буфера трансляции (модуль МСТ)"
;8562    ;
;8563    ;  ОПИСАНИЕ ТЕСТА:
;8564    ;
;8565    ;          Этот тест проверяет часть памяти TB (буфера трансляции) для адресов цент-
;8566    ;       рального процессора. Доступ к адресам с 0 по 7F(H) (128 десятичных адресов)
;8567    ;       возможен через микропрограмму контроллера ОЗУ WRITE.TB. Те же адреса можно
;8568    ;       считывать обратно для проверки, используя программу READ.TB. Другие адреса в
;8569    ;       зоне 1K X 4 микросхем памяти или не используются, или проверяются тестами па-
;8570    ;       мяти отображения адресов общей шины. Адрес обращения к памяти буфера трансля-
;8571    ;       ции определяется битами регистра виртуального адреса с 09 по 14 и битом 31.
;8572    ;       Биты с 7 по 9 адреса памяти аппаратно устанавливаются в 0. Биты с 0 по 8 и с
;8573    ;       15 по 30 регистра виртуального адреса не влияют на выборку адреса буфера
;8574    ;       трансляции. Адрес посылается из местной памяти во время инструкции централь-
;8575    ;       ного процессора MEM.REQ с полем MF=WRITE.TB (1H) или READ.TB (13H). Инструк-
;8576    ;       ция центрального процессора WRITE.MEM LS посылает данные, которые должны быть
;8577    ;       записаны в буфер трансляции следующим образом: биты местной памяти 0-15 пос-
;8578    ;       тупают на биты PA 09-24 соответственно, а биты 25-31 местной памяти поступа-
;8579    ;       ют на биты BYTE OFFSET, MODIFY, PROT A, PROT B, PROT C, PROT L и VALID соот-
;8580    ;       ветственно. Данные для записи в буфер трансляции сдвигаются в сдвигателях
;8581    ;       данных и запоминаются в регистрах схемы ECC. Затем они считываются и записы-
;8582    ;       ваются в буфер трансляции. Необходимо заметить, что бит 24 PA не используется
;8583    ;       и в буфер трансляции записываются только биты 9-23 PA.
;8584    ;          Путь данных при записи следующий: выдается инструкция MEM.REQ с MF=WRITE.
;8585    ;       TB и DT=LONGWORD. Начальное ветвление выполняется как описано в начале этого
;8586    ;       листинга. По адресу начального ветвления устанавливается DATI для сброса IC0,
;8587    ;       приемо-передатчики устанавливаются для приема данных из центрального процес-
;8588    ;       сора для записи в буфер трансляции и приемо-передатчики на МСТ устанавливают-
;8589    ;       ются для передачи данных из шины BUS MC в биты PA и в другие биты данных бу-
;8590    ;       фера трансляции. Устанавливаются MEMORY BUSY и ROTATE BYTE. Следующий цикл
;8591    ;       повторяет эту последовательность и дополняется сигналом ROT CLK для разреше-
;8592    ;       шения функции циклического сдвига на байт. Следующий цикл циклится в ожидании
;8593    ;       сигнала центрального процессора DATA REQ (который поступает при инструкции
;8594    ;       центрального процессора WRITE.MEM LS). Этот сигнал сбрасывает MEMORY BUSY для
;8595    ;       снятия приостанова центрального процессора, разрешает приемо-передатчики
;8596    ;       центрального процессора для передачи данных из центрального процессора и ус-
;8597    ;       танавливает МСТ для пропуска данных из MC BUS в буфер трансляции. Когда сигнал
;8598    ;       центрального процессора CPU DATA REQ становится активным, следующий цикл разре-
;8599    ;       шает выход данных сдвигателей на шину ARRAY BUS и открывает входные регистры
;8600    ;       схемы ECC для приема этих данных.
;8601    ;          Приемо-передатчики центрального процессора разрешаются для приема данных из
;8602    ;       центрального процессора, приемо-передатчики МСТ устанавливаются для пропуска
;8603    ;       данных в буфер трансляции и очищается CPUG (это гарантирует, что сигнал CPU
;8604    ;       GRANT снимется, даже если центральный процессор находится в пошаговом режиме).
;8605    ;       Следующий цикл только устанавливает запрет циклического сдвига и поддерживает
;8606    ;       установленными приемо-передатчики МСТ для пропуска данных в буфер трансляции.
;8607    ;       Этот цикл закрывает ECC, так что данные остаются буферизированными в схеме ECC.
;8608    ;       Следующий цикл выдает синхросигнал циклического сдвига для запрета сдвига и
;8609    ;       разрешает выход данных ECC на шину ARRAY BUS. Сдвигатели разрешаются для пере-
;8610    ;       сылки этих данных на шину MC BUS и приемо-передатчики МСТ разрешаются для пере-
;8611    ;       сылки данных шины MC BUS на входы данных буфера трансляции. Разрешение записи
;8612    ;       в буфер трансляции имеется, но запись не произойдет, пока разрешение записи не
;8613    ;       изменится (станет высоким). Следующий цикл повторяет эти действия, за исключе-
;8614    ;       нием синхросигнала циклического сдвига. Этот цикл выполняет ветвление по сигна-
```

лу IC0, который был сброшен при установке DATI в цикле начального ветвления. Этот цикл поддерживает разрешенными данные на входе буфера трансляции, но снимает разрешение записи буфера трансляции. Это вызывает запись в буфер трансляции. Затем микропрограмма переходит в холостой цикл.

Микропрограмма чтения запускается инструкцией MEM.REQ с полем MF=READ.TB (13H). Начальное ветвление выполняется, как описано в начале этого листинга. По адресу начального ветвления разрешается обход регистра виртуального адреса (VAR BYPASS), устанавливается MEMORY BUSY и разрешаются приемо-передатчики MCT, которые пропускают данные буфера трансляции, но не биты PA.

Следующий цикл поддерживает MEMORY BUSY, разрешает приемо-передатчики MCT, обход регистра виртуального адреса и разрешает другие приемо-передатчики MCT, которые пропускают биты PA. сейчас, когда данные буфера трансляции находятся на шине MC BUS, следующий цикл снимает MEMORY BUSY, поддерживая разрешенными приемо-перетдатчики MCT и циклится в ожидании снятия CPU GRANT (это произойдет, когда будет выполняться инструкция центрального процессора MOV MEM.DATA TO WR и возбудит сигнал центрального процессора DATA RCVD). Когда сигнал CPU GRANT снимается, следующий цикл подготавливает цикл общей шины и переходит в холостой цикл. Адрес буфера трансляции был обеспечен во время инструкции MEM.REQ, которая запустила микропрограмму чтения буфера трансляции. Данные возвращаются обратно без сдвига. Следовательно, биты с BYTE OFFSET по VALID возвращаются на биты с 01 по 07 соответственно, PA 09-23 возвращаются на биты 08-22 соответственно.

Вышеупомянутые циклы используют тестовые данные: все единицы, все нули, сдвигаемая единица, сдвигаемый нуль. Потом выполняется тест со сдвигаемым битом в адресе для проверки адресных линий. В конце выполняется тест "БЕГУЩЕЙ ИНВЕРСИИ" для проверки задержек схем адреса. Этот тест выполняется как тест "МАРШ", за исключением того, что он выполняется 7 раз, каждый раз переключая разные биты адреса с максимальной скоростью. "МАРШ" должен записать фон нулей, потом пройти буфер трансляции в порядке возрастания адресов с чтением нулей и записью единиц. Когда последний адрес будет проверен, "МАРШ" должен выполнить чтение единиц и запись нулей в буфер трансляции в порядке убывания адресов.

ПРЕДПОЛОЖЕНИЯ:

Предполагается, что все предыдущие тесты выполнены успешно.

ШАГИ ТЕСТА:

1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти (для распечатки ошибок) и очистка предыдущего номера ошибки в местной памяти.
2. Загрузка всех единиц в WR1 и выполнение инструкции MEM.REQ с MF=WRITE.TB и DT=LONGWORD. Ячейка местной памяти, содержащая нули, используется в качестве адреса.
3. Выполнение инструкции WRITE.MEM LS с ячейкой местной памяти, содержащей единицы.
4. Выполнение инструкции MEM.REQ с MF=READ.TB и DT=LONGWORD. Ячейка местной памяти, содержащая нули, используется в качестве адреса.
5. Выполнение инструкции MOV.MEM.DATA TO WR[0] и проверка результата на единицы.
6. Повторение шагов с 2 по 5 с данными - все нули в шагах 3 и 5.
7. Повторение шагов с 2 по 5 с данными - сдвигаемая 1 в шагах 3 и 5. Ожидаемыми данными в WR1 являются данные, переданные со сдвигом на 8 битов влево.

```
;8670    ;       8.Повторение шагов с 2 по 5 с данными - сдвигаемый нуль в шагах 3 и 5.
;8671    ;         ожидаемыми данными в WR1 являются данные, переданные со сдвигом на 8
;8672    ;         битов влево.
;8673    ;       9.Повторение шагов с 2 по 5 с тестовыми данными - нули, единицы, сдвигае-
;8674    ;         мая единица, сдвигаемый нуль при последовательном наращивании адресов
;8675    ;         с каждым набором данных. Этим проверяются все ячейки памяти, к которым
;8676    ;         может быть выполнено обращение. Ожидаемыми данными в WR1 являются дан-
;8677    ;         ные, переданные со сдвигом на 8 битов влево.
;8678    ;      10.Выполнение подтеста с наборами типа "МАРШ" при изменении адреса сдви-
;8679    ;         гом бита, содержащего 1, через адресное поле из нулей. Запись фона нулей,
;8680    ;         чтение нулей, запись единиц, чтение нулей. Это повторяется 7 раз, пока
;8681    ;         единица сдвигается через адресные линии. Этим проверяются адресные ли-
;8682    ;         нии и грубые отказы в микросхеме памяти.
;8683    ;      11.Выполнение теста "БЕГУЩЕЙ ИНВЕРСИИ" (описан выше).
;8684    ;
;8685    ;   ОШИБКИ:
;8686    ;
;8687    ;       Примечание: данными под OTHER является проверяемый адрес буфера трансля-
;8688    ;       ции.
;8689    ;
;8690    ;       ошибка 1 - неправильно выполняется запись или чтение буфера трансляции с
;8691    ;                  данными - все единицы, адрес 0.
;8692    ;       ошибка 2 - неправильно выполняется запись или чтение буфера трансляции с
;8693    ;                  данными - все нули, адрес 0.
;8694    ;       ошибка 3 - неправильно выполняется запись или чтение буфера трансляции с
;8695    ;                  данными - сдвигаемая единица, адрес 0.
;8696    ;       ошибка 4 - неправильно выполняется запись или чтение буфера трансляции с
;8697    ;                  данными - сдвигаемый нуль, адрес 0.
;8698    ;       ошибка 5 - неправильно работает буфер трансляции с данными - все единицы.
;8699    ;       ошибка 6 - неправильно работает буфер трансляции с данными - все нули.
;8700    ;       ошибка 7 - неправильно работает буфер трансляции с данными - сдвигаемая 1.
;8701    ;       ошибка 8 - неправильно работает буфер трансляции с данными - сдвигаемый 0.
;8702    ;       ошибка 9 - неправильно работает адрес буфера трансляции в тесте со сдвигае-
;8703    ;                  мым битом в адресе.
;8704    ;       ошибка A - неправильно работает буфер трансляции с набором "БЕГУЩЕЙ ИНВЕР-
;8705    ;                  СИИ" при наращивании адресов.
;8706    ;       ошибка B - неправильно работает буфер трансляции с набором "БЕГУЩЕЙ ИНВЕР-
;8707    ;                  СИИ" при уменьшении адресов.
;8708    ;
;8709    ;   НАЛАДКА:
;8710    ;
;8711    ;       Примечание: адрес буфера трансляции, который работает неправильно, в сооб-
;8712    ;       щении об ошибке печатается под OTHER (другие данные)(за исключением ошибок
;8713    ;       A и D). Биты располагаются так, чтобы получить действительный адрес памяти
;8714    ;       в шестнадцатеричном формате.
;8715    ;
;8716    ;          ОШИБКА 1 - Эта ошибка указывает на возможную неисправность схем буфера
;8717    ;       трансляции. Если пошаговый режим MCT возможен, центральный процессор сле-
;8718    ;       дует остановить на инструкции MEM.REQ и разрешить пошаговый режим MCT. Да-
;8719    ;       лее центральный процессор необходимо остановить на инструкции WRITE.MEM LS.
;8720    ;       Затем остановить MCT на цикле, в котором выполняется ветвление по сигналу
;8721    ;       IC0 (BEN2=LUB.C0). Необходимо проверить входы BUS MC D01- BUS MC D23, пос-
;8722    ;       тупающие на приемо-передатчики, выходами которых являются биты PA и биты
;8723    ;       от BYTE OFFSET до VALID. Все линии BUS MC DXX должны иметь высокий уровень.
;8724    ;       Если нет, одна из ПМЛ сдвигателя данных, по-видимому, неисправна (другие
```

;8725 ; цепи уже были проверены). Неисправной ПМЛ является та, которая формирует
;8726 ; неправильный сигнал BUS MC.
;8727 ; Если сигналы BUS MC правильные, необходимо проверить высокий уровень
;8728 ; сигнала TB DATA DIR RD L, низкие уровни сигналов TB DATA EN L и TB DATA
;8729 ; EN+MAINT L на всех трех буферах. Все они поступают непосредственно из уп-
;8730 ; равляющей памяти. Если здесь правильно, выходы буферов (выходы B) должны
;8731 ; иметь высокий уровень. Если нет, подозревается буфер, выдающий неправиль-
;8732 ; ный бит.
;8733 ; Если выходы буферов правильные, необходимо проверить высокие уровни на
;8734 ; всех входах памяти 1KX4. Ошибка здесь указывает на неисправное соедине-
;8735 ; ние. Необходимо проверить низкий уровень сигнала TB WE L на ножке 10 каж-
;8736 ; дой микросхемы памяти 1KX4. Этот сигнал поступает непосредственно из уп-
;8737 ; равляющей памяти. Если этот сигнал правильный, данные будут записаны в
;8738 ; память на следующем цикле, когда сигнал TB WE L станет высоким. Затем сле-
;8739 ; дует выполнить еще один цикл MCT и проверить, что сигнал TB WE L стано-
;8740 ; вится высоким, ADDR PH H низким и что выходы на ножках 11-14 микросхемы
;8741 ; памяти высокие. Если входы правильные, а выход низкий, по-видимому, мик-
;8742 ; росхема памяти неисправна.
;8743 ; Если все до этого места правильно, тогда неправильно работает программа
;8744 ; READ.TB. Центральный процессор следует остановить в пошаговом режиме на инст-
;8745 ; рукции MOV MEM.DATA TO WR[0]. MCT должен циклиться в ожидании снятия сигнала
;8746 ; CPU GRANT. Необходимо проверить низкие уровни сигналов TB DATA DIR RD L и TB
;8747 ; DATA EN L на приемо-передатчиках MCT. Также необходимо проверить, что входы
;8748 ; стороны B содержат высокие уровни. Если эти сигналы правильные, тогда необхо-
;8749 ; димо проверить высокие уровни на выходах стороны A. Неправильные входы указы-
;8750 ; вают на неисправность приемо-передатчика.
;8751 ;
;8752 ; ОШИБКА 2 - Так же, как и при ошибке 1, за исключением того, что входы
;8753 ; данных приемо-передатчиков и микросхем памяти содержат низкие уровни,
;8754 ; вместо высоких.
;8755 ;
;8756 ; ОШИБКА 3 - Эта ошибка указывает на возможные закорачивания между лини-
;8757 ; ями данных, поступающими на буфер трансляции, или в самих микросхемах па-
;8758 ; мяти.
;8759 ;
;8760 ; ОШИБКА 4 - Так же, как и при ошибке 3.
;8761 ;
;8762 ; ОШИБКИ с 5 по 8 - Эти ошибки скорее всего указывают неисправный адрес
;8763 ; микросхемы памяти, указанный под OTHER.
;8764 ;
;8765 ; ОШИБКА 9 - Эта ошибка указывает на возможную неисправность адресных ли-
;8766 ; ний или возможную неисправность адресных схем внутри микросхемы. Если все
;8767 ; биты неправильные, подозреваются адресные линии. Адресные линии можно
;8768 ; проверить в режиме SOMM (останов по адресу микрокоманды) на инструкции
;8769 ; MEM.REQ с MF=WRITE.TB. Каждый раз, когда центральный процессор останавли-
;8770 ; вается по совпадению адреса микрокоманды, можно проверить адресные линии
;8771 ; на модуле MCT. Единица должна сдвигаться через адресные линии от LVA 09 H
;8772 ; по LVA 14 H, А затем в TBA 6 H после каждой команды CONTINUE (продолже-
;8773 ; ние). TBA 7 по TBA 9 всегда должны быть низкими. Если TBA 6 H не стано-
;8774 ; вится высоким, необходимо проверить мультиплексор 4x2, который на выходе
;8775 ; формирует адрес TBA. Сигнал выборки UB TB SEL H должен быть низким. Этот
;8776 ; сигнал может быть проверен в обратном направлении до схемы "ИЛИ" по низ-
;8777 ; ким уровням, входами которой являются сигналы CPH/UBL H и UB TB SEL L. Оба
;8778 ; эти входа должны быть высокими. Если нет, подозреваются ПМЛ, которые форми-
;8779 ; руют эти сигналы.

```
                    ;8780   ;          Другой возможностью появления ошибки может быть неправильная работа схем
                    ;8781   ;       адреса в микропрограмме READ.TB. Эта микропрограмма в начале использует
                    ;8782   ;       регистр обхода регистра виртуального адреса. Останов по SOMM на инструкции
                    ;8783   ;       MEM.REQ с MF=READ.TB позволяет проверить адресные линии. После каждой ко-
                    ;8784   ;       манды CONTINUE следует проверить адресные линии на сдвиг единицы через по-
                    ;8785   ;       ле нулей.
                    ;8786   ;          Если все биты правильные, подозревается неисправная ячейка в самой микро-
                    ;8787   ;       схеме памяти. Если неправильной является группа из 4 битов, подозревается
                    ;8788   ;       обрыв связи адресной линии к этой микросхеме памяти.
                    ;8789   ;
                    ;8790   ;          ОШИБКА A и B - Эти ошибки скорее всего указывают на неисправность мик-
                    ;8791   ;       росхемы памяти - и/или отдельной ячейки, или схем переключения адреса.
                    ;8792   ;       Адрес можно определить из ячейки 9(H) местной памяти. Биты 9-14 соответ-
                    ;8793   ;       ствуют битам 0-5 на входе микросхемы памяти. Бит 31 соответствует биту
                    ;8794   ;       6 на входе микросхемы памяти.
                    ;8795   ;
                    ;8796   ;
                    ;8797   ;
                    ;8798    T.11:
U 0AAF, B65E,15     ;8799           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                    ;8800                                                   ; состояния
U 0AB0, 3E80,15     ;8801           MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 15 в слове управления и состояния. Бит
                    ;8802                                                   ; 15 указывает начало теста консольному процессору
U 0AB1, 10E0,15     ;8803           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN консольному процессору
                    ;8804    WAIT.T11.0:
U 0AB2, B8AB,24     ;8805           JMP [WAIT.T11.0]                        ; зацикливание для ожидания ответа консольного
                    ;8806                                                   ; процессора
U 0AB3, 0A1A,AC     ;8807           JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0AB4, B62A,15     ;8808           MOV LS[#FF000000] TO WR[0]               ; установка старшего байта маски ошибки
U 0AB5, C76E,15     ;8809           BIS LS[BIT23] TO WR[0]                   ; установка бита 23
U 0AB6, C740,15     ;8810           BIS LS[BIT0] TO WR[0]                    ; установка бита 0. Сейчас биты с 1 по 22 будут
                    ;8811                                                   ; проверяться на ошибки
U 0AB7, 3E8A,15     ;8812           MOV WR[0] TO LS[ERROR.MASK]              ; запоминание маски ошибки в местной памяти
U 0AB8, 6588,15     ;8813           CLR LS[ADDRESS.DATA]                     ; загрузка 0 для распечатки адреса под OTHER в сообщении
                    ;8814                                                   ; об ошибке
U 0AB9, B670,15     ;8815           MOV LS[OTHER.DATA] TO WR[0]              ; установка бита 24 в WR0
U 0ABA, 3E80,15     ;8816           MOV WR[0] TO LS[CONTROL.STATUS]          ; консольный процессор будет печатать под OTHER при
                    ;8817                                                   ; ошибке
                    ;8818    LOOP.T11.1:
U 0ABB, 369E,95     ;8819           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0ABC, 989C,F5     ;8820           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи в буфер трансляции по адресу 0
U 0ABD, 329E,15     ;8821           WRITE.MEM LS[ONES]                      ; запись всех единиц в буфер трансляции
U 0ABE, 9C9D,F5     ;8822           MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; подготовка чтения буфера трансляции
U 0ABF, 3022,15     ;8823           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0AC0, 0869,3C     ;8824           JSR [CHECK.RESULT]                      ; проверка результата
U 0AC1, B8AB,B4     ;8825           JMP [LOOP.T11.1]                        ; зацикливание при ошибке, если разрешено
U 0AC2, FFB2,15     ;8826           INC LS[ERROR.NUMBER]                    ; ошибка 2
                    ;8827    LOOP.T11.2:
U 0AC3, B69C,95     ;8828           MOV LS[ZERO] TO WR[1]                   ; ожидаемые данные
U 0AC4, 989C,F5     ;8829           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи в буфер трансляции по адресу 0
U 0AC5, B29C,15     ;8830           WRITE.MEM LS[ZERO]                      ; запись всех 0 в буфер трансляции
U 0AC6, 9C9D,F5     ;8831           MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; подготовка чтения буфера трансляции
U 0AC7, 3022,15     ;8832           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0AC8, 0869,3C     ;8833           JSR [CHECK.RESULT]                      ; проверка результата
U 0AC9, B8AC,34     ;8834           JMP [LOOP.T11.2]                        ; зацикливание при ошибке, если разрешено
```

```
U 0ACA, FF82,15 ;8835            INC LS[ERROR.NUMBER]                    ; ошибка 3
U 0ACB, E5F8,15 ;8836            CLR LS[OS]                              ; очистка индекса
U 0ACC, 3651,95 ;8837            MOV LS[BIT8] TO WR[3]                   ; первые ожидаемые данные (полученные данные сдвинуты на
                ;8838                                                    ; 8 разрядов влево)
                ;8839    LOOP.T11.3:
U 0ACD, 2006,95 ;8840            MOV WR[3] TO WR[1]                      ; ожидаемые данные
U 0ACE, 989C,F5 ;8841            MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи в буфер трансляции по адресу 0
U 0ACF, B2F6,15 ;8842            WRITE.MEM LS[SHIFT.OS(4-0)]             ; запись данных в буфер трансляции - сдвигаемая единица
U 0AD0, 9C9D,F5 ;8843            MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; подготовка чтения буфера трансляции по адресу 0
U 0AD1, 3022,15 ;8844            MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0AD2, 0869,3C ;8845            JSR [CHECK.RESULT]                      ; проверка результата
U 0AD3, 08AC,D4 ;8846            JMP [LOOP.T11.3]                        ; зацикливание при ошибке, если разрешено
U 0AD4, A3C1,95 ;8847            ROL WR[3]                               ; следующий набор ожидаемых данных
U 0AD5, 7FF8,15 ;8848            INC LS[OS]                              ; увеличение индекса для следующего набора
U 0AD6, B6F8,15 ;8849            MOV LS[OS] TO WR[0]                     ; подготовка проверки, все ли наборы использованы
                ;8850              BIT LS[BIT5] WITH WR[0],              ; проверка, увеличен ли OS до 20(H)
U 0AD7, 594A,35 ;8851            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0AD8, 88AC,D9 ;8852            JMP.IF[BITS.CLR] TO [LOOP.T11.3]        ; повторение со следующим набором, если не все
                ;8853                                                    ; использованы
U 0AD9, FF82,15 ;8854            INC LS[ERROR.NUMBER]                    ; ошибка 4
U 0ADA, E5F8,15 ;8855            CLR LS[OS]                              ; очистка индекса
U 0ADB, 5F51,95 ;8856            MCOM LS[BIT8] TO WR[3]                  ; первые ожидаемые данные (полученные данные сдвинуты на
                ;8857                                                    ; 8 разрядов влево)
                ;8858    REPEAT.T11.4:
U 0ADC, 5FF6,15 ;8859            MCOM LS[SHIFT.OS(4-0)] TO WR[0]         ; выборка текущего набора тестовых данных
U 0ADD, 3E10,15 ;8860            MOV WR[0] TO LS[T8]                     ; пересылка набора данных в ячейку временного хранения
                ;8861                                                    ; местной памяти
                ;8862    LOOP.T11.4:
U 0ADE, 2006,95 ;8863            MOV WR[3] TO WR[1]                      ; ожидаемые данные
U 0ADF, 989C,F5 ;8864            MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи в буфер трансляции по адресу 0
U 0AE0, 3210,15 ;8865            WRITE.MEM LS[T8]                        ; запись данных в буфер трансляции - сдвигаемый нуль
U 0AE1, 9C9D,F5 ;8866            MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; подготовка чтения буфера трансляции по адресу 0
U 0AE2, 3022,15 ;8867            MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0AE3, 0869,3C ;8868            JSR [CHECK.RESULT]                      ; проверка результата
U 0AE4, 88AD,E4 ;8869            JMP [LOOP.T11.4]                        ; зацикливание при ошибке, если разрешено
U 0AE5, A3C1,95 ;8870            ROL WR[3]                               ; следующий набор ожидаемых данных
U 0AE6, 7FF8,15 ;8871            INC LS[OS]                              ; увеличение индекса для следующего набора
U 0AE7, B6F8,15 ;8872            MOV LS[OS] TO WR[0]                     ; подготовка проверки, все ли наборы использованы
                ;8873              BIT LS[BIT5] WITH WR[0],              ; проверка, увеличен ли OS до 20(H)
U 0AE8, 594A,35 ;8874            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0AE9, 88AD,C9 ;8875            JMP.IF[BITS.CLR] TO [REPEAT.T11.4]      ; повторение со следующим набором, если не все
                ;8876                                                    ; использованы
U 0AEA, FF82,15 ;8877            INC LS[ERROR.NUMBER]                    ; ошибка 5
U 0AEB, 6512,15 ;8878            CLR LS[T9]                              ; очистка адреса буфера трансляции (адрес запоминается в
                ;8879                                                    ; LS9)
U 0AEC, B640,15 ;8880            MOV LS[#1] TO WR[0]                     ; пересылка 1 в WR0
U 0AED, BEFC,15 ;8881            MOV WR[0] TO LS[SIZE]                   ; загрузка регистра размера кодом типа данных - слово
                ;8882    LOOP.T11.5:
U 0AEE, 369E,95 ;8883            MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0AEF, 9812,F5 ;8884            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи в буфер трансляции
U 0AF0, 329E,15 ;8885            WRITE.MEM LS[ONES]                      ; запись всех единиц в буфер трансляции
U 0AF1, 9C13,F5 ;8886            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения буфера трансляции
U 0AF2, 3022,15 ;8887            MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0AF3, 0869,3C ;8888            JSR [CHECK.RESULT]                      ; проверка результата
U 0AF4, 88AE,E4 ;8889            JMP [LOOP.T11.5]                        ; зацикливание при ошибке, если разрешено
```

```
U 0AF5, 0A18,2C ;8890            JSR [NEXT.TB.ADDRESS]                  ; вычисление следующего адреса буфера трансляции
U 0AF6, 88AE,E4 ;8891            JMP [LOOP.T11.5]                       ; возврат, если не все адреса использованы
U 0AF7, FF82,15 ;8892            INC LS[ERROR.NUMBER]                   ; ошибка 6
U 0AF8, 6512,15 ;8893            CLR LS[T9]                             ; очистка адреса буфера трансляции (адрес запоминается в
                ;8894                                                   ; LS9)
U 0AF9, 6588,15 ;8895            CLR LS[ADDRESS.DATA]                   ; очистка адреса информации для распечатки при ошибке
                ;8896    LOOP.T11.6:
U 0AFA, B69C,95 ;8897            MOV LS[ZERO] TO WR[1]                  ; ожидаемые данные
U 0AFB, 9812,F5 ;8898            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]    ; подготовка записи в буфер трансляции
U 0AFC, B29C,15 ;8899            WRITE.MEM LS[ZERO]                     ; запись всех нулей в буфер трансляции
U 0AFD, 9C13,F5 ;8900            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]     ; подготовка чтения буфера трансляции
U 0AFE, 3022,15 ;8901            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 0AFF, 0869,3C ;8902            JSR [CHECK.RESULT]                     ; проверка результата
U 0B00, 88AF,A4 ;8903            JMP [LOOP.T11.6]                       ; зацикливание при ошибке, если разрешено
U 0B01, 0A18,2C ;8904            JSR [NEXT.TB.ADDRESS]                  ; вычисление следующего адреса буфера трансляции
U 0B02, 88AF,A4 ;8905            JMP [LOOP.T11.6]                       ; возврат, если не все адреса использованы
U 0B03, FF82,15 ;8906            INC LS[ERROR.NUMBER]                   ; ошибка 7
U 0B04, E5F8,15 ;8907            CLR LS[OS]                             ; очистка индекса
U 0B05, 6512,15 ;8908            CLR LS[T9]                             ; очистка адреса буфера трансляции (адрес запоминается в
                ;8909                                                   ; LS9)
U 0B06, 3651,95 ;8910            MOV LS[BIT8] TO WR[3]                  ; первые ожидаемые данные (полученные данные)
U 0B07, 6588,15 ;8911            CLR LS[ADDRESS.DATA]                   ; очистка адреса информации для распечатки при ошибке
                ;8912    LOOP.T11.7:
U 0B08, 2006,95 ;8913            MOV WR[3] TO WR[1]                     ; ожидаемые данные
U 0B09, 9812,F5 ;8914            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]    ; подготовка записи в буфер трансляции
U 0B0A, B2F6,15 ;8915            WRITE.MEM LS[SHIFT.OS(4-0)]            ; запись данных в буфер трансляции - сдвигаемая единица
U 0B0B, 9C13,F5 ;8916            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]     ; подготовка чтения буфера трансляции
U 0B0C, 3022,15 ;8917            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 0B0D, 0869,3C ;8918            JSR [CHECK.RESULT]                     ; проверка результата
U 0B0E, 88B0,84 ;8919            JMP [LOOP.T11.7]                       ; зацикливание при ошибке, если разрешено
U 0B0F, A3C1,95 ;8920            ROL WR[3]                              ; следующий набор ожидаемых данных
U 0B10, 7FF8,15 ;8921            INC LS[OS]                             ; увеличение индекса для следующего набора
U 0B11, B6F8,15 ;8922            MOV LS[OS] TO WR[0]                    ; подготовка проверки, все ли наборы использованы
                ;8923              BIT LS[BIT5] WITH WR[0],             ; проверка, увеличен ли OS до 20(H)
U 0B12, 594A,35 ;8924            DT(LONG)&SET.ALU.CC                    ; и установка кодов условий
U 0B13, 08B0,89 ;8925            JMP.IF[BITS.CLR] TO [LOOP.T11.7]       ; повторение со следующим набором, если не все
                ;8926                                                   ; использованы
U 0B14, E5F8,15 ;8927            CLR LS[OS]                             ; очистка индекса
U 0B15, 0A18,2C ;8928            JSR [NEXT.TB.ADDRESS]                  ; вычисление следующего адреса буфера трансляции
U 0B16, 88B0,84 ;8929            JMP [LOOP.T11.7]                       ; возврат, если не все адреса использованы
U 0B17, FF82,15 ;8930            INC LS[ERROR.NUMBER]                   ; ошибка 8
U 0B18, E5F8,15 ;8931            CLR LS[OS]                             ; очистка индекса
U 0B19, 6512,15 ;8932            CLR LS[T9]                             ; очистка адреса буфера трансляции (адрес запоминается в
                ;8933                                                   ; LS9)
U 0B1A, 6588,15 ;8934            CLR LS[ADDRESS.DATA]                   ; очистка адреса информации для распечатки при ошибке
U 0B1B, 5F51,95 ;8935            MCOM LS[BIT8] TO WR[3]                 ; первые ожидаемые данные (полученные данные сдвинуты на
                ;8936                                                   ; 8 разрядов влево)
                ;8937    REPEAT.T11.8:
U 0B1C, 5FF6,15 ;8938            MCOM LS[SHIFT.OS(4-0)] TO WR[0]        ; выборка текущего набора тестовых данных
U 0B1D, 3E10,15 ;8939            MOV WR[0] TO LS[T8]                    ; пересылка набора данных в ячейку временного хранения
                ;8940                                                   ; местной памяти
                ;8941    LOOP.T11.8:
U 0B1E, 2006,95 ;8942            MOV WR[3] TO WR[1]                     ; ожидаемые данные
U 0B1F, 9812,F5 ;8943            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]    ; подготовка записи в буфер трансляции
U 0B20, 3210,15 ;8944            WRITE.MEM LS[T8]                       ; запись данных в буфер трансляции - сдвигаемый нуль
```

```
U 0B21, 9C13,F5 ;8945            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения буфера трансляции
U 0B22, 3022,15 ;8946            MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0B23, 0B69,3C ;8947            JSR [CHECK.RESULT]                      ; проверка результата
U 0B24, 08B1,E4 ;8948            JMP [LOOP.T11.8]                        ; зацикливание при ошибке, если разрешено
U 0B25, A3C1,95 ;8949            ROL WR[3]                               ; следующий набор ожидаемых данных
U 0B26, 7FF8,15 ;8950            INC LS[OS]                              ; увеличение индекса для следующего набора
U 0B27, B6F8,15 ;8951            MOV LS[OS] TO WR[0]                     ; подготовка проверки все ли наборы использованы
                ;8952              BIT LS[BIT5] WITH WR[0],              ; проверка, увеличен ли OS до 20(H)
U 0B28, 594A,35 ;8953            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0B29, 08B1,C9 ;8954            JMP.IF[BITS.CLR] TO [REPEAT.T11.8]      ; повторение со следующим набором, если не все
                ;8955                                                    ; использованы
U 0B2A, E5F8,15 ;8956            CLR LS[OS]                              ; очистка индекса
U 0B2B, 0A18,2C ;8957            JSR [NEXT.TB.ADDRESS]                   ; вычисление следующего адреса буфера трансляции
U 0B2C, 88B1,C4 ;8958            JMP [REPEAT.T11.8]                      ; возврат, если не все адреса использованы
U 0B2D, FF82,15 ;8959            INC LS[ERROR.NUMBER]                    ; ошибка 9
U 0B2E, 6512,15 ;8960            CLR LS[T9]                              ; очистка ячейки местной памяти, содержащей адрес
                ;8961    BACK.T11.9:
U 0B2F, 9812,F5 ;8962            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи фона в буфер трансляции
U 0B30, B29C,15 ;8963            WRITE.MEM LS[ZERO]                      ; пересылка нулей в буфер трансляции
U 0B31, 0A18,2C ;8964            JSR [NEXT.TB.ADDRESS]                   ; вычисление следующего адреса буфера трансляции
U 0B32, 88B2,F4 ;8965            JMP [BACK.T11.9]                        ; возврат, если еще не все выполнено
U 0B33, 6588,15 ;8966            CLR LS[ADDRESS.DATA]                    ; очистка адреса информации, который выбирается для
                ;8967                                                    ; распечатки
U 0B34, 6512,15 ;8968            CLR LS[T9]                              ; очистка ячейки хранения адреса буфера трансляции
U 0B35, 3651,95 ;8969            MOV LS[BIT8] TO WR[3]                   ; установка бита 8 в WR3. Циклическим сдвигом WR3 будет
                ;8970                                                    ; генерироваться первый сдвинутый адрес (1(H))
U 0B36, B67F,15 ;8971            MOV LS[BIT31] TO WR[2]                  ; установка бита 31 в WR2. Циклическим сдвигом WR2 будет
                ;8972                                                    ; генерироваться первый сдвинутый адрес (1(H)) для
                ;8973                                                    ; распечатки при ошибке
                ;8974    LOOP.T11.9A:
U 0B37, 2F82,95 ;8975            CLR WR[1]                               ; ожидаемые данные
U 0B38, 9C13,F5 ;8976            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения буфера трансляции
U 0B39, 3022,15 ;8977            MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0B3A, 0B69,3C ;8978            JSR [CHECK.RESULT]                      ; проверка результата
U 0B3B, 88B3,74 ;8979            JMP [LOOP.T11.9A]                       ; зацикливание при ошибке, если разрешено
                ;8980    LOOP.T11.9B:
U 0B3C, 369E,95 ;8981            MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0B3D, 9812,F5 ;8982            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи в буфер трансляции
U 0B3E, 329E,15 ;8983            WRITE.MEM LS[ONES]                      ; запись единиц в буфер трансляции
U 0B3F, 9C13,F5 ;8984            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения единиц из буфера трансляции
U 0B40, 3022,15 ;8985            MOV MEM.DATA TO WR[0]                   ; чтение результата
U 0B41, 0B69,3C ;8986            JSR [CHECK.RESULT]                      ; проверка результата
U 0B42, 08B3,C4 ;8987            JMP [LOOP.T11.9B]                       ; зацикливание при ошибке, если разрешено
U 0B43, A3C1,95 ;8988            ROL WR[3]                               ; сдвиг адреса для пересылки 1 в следующий бит
                ;8989              BIT LS[BIT0] WITH WR[3],              ; проверка, установлен ли бит 0 (означает, что бит 31
                ;8990                                                    ; уже был проверен)
U 0B44, 5941,B5 ;8991            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0B45, 08B4,D1 ;8992            JMP.IF[BIT.SET] TO [TEST.T11.A]         ; переход на следующий тест, если выполнено
                ;8993              BIT LS[BIT15] WITH WR[3],             ; проверка, что биты 5-0 проверены (если последний тест
                ;8994                                                    ; был с установленным битом 14, после сдвига бит 15
                ;8995                                                    ; установлен)
U 0B46, 595F,B5 ;8996            DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0B47, 5B00,09 ;8997            SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит 15 не
                ;8998                                                    ; установлен
U 0B48, 367F,95 ;8999            MOV LS[BIT31] TO WR[3]                  ; иначе установка бита 31 для следующего адреса и сброс
```

```
                    ;9000                                                   ; бита 15
U 0B49,  BE13,95    ;9001           MOV WR[3] TO LS[T9]                     ; запоминание адреса
U 0B4A,  23C1,15    ;9002           ROL WR[2]                               ; адрес информации для распечатки при ошибке
U 0B4B,  3E89,15    ;9003           MOV WR[2] TO LS[ADDRESS.DATA]           ; запоминание в местной памяти
U 0B4C,  B8B3,74    ;9004           JMP [LOOP.T11.9A]                       ; повторение, пока все биты адреса не будут проверены
                    ;9005     TEST.T11.A:
U 0B4D,  E580,15    ;9006           CLR LS[CONTROL.STATUS]                  ; запрет распечатки под OTHER
U 0B4E,  FF82,15    ;9007           INC LS[ERROR.NUMBER]                    ; ошибка A
U 0B4F,  3682,15    ;9008           MOV LS[ERROR.NUMBER] TO WR[0]           ; сохранение номера ошибки
U 0B50,  BE14,15    ;9009           MOV WR[0] TO LS[T10]                    ; в ячейке временного хранения местной памяти
U 0B51,  3653,15    ;9010           MOV LS[BIT9] TO WR[2]                   ; подготовка увеличения (увеличивается бит 0 адреса для
                    ;9011                                                   ; микросхемы памяти)
U 0B52,  B653,95    ;9012           MOV LS[BIT9] TO WR[3]                   ; подготовка переключения бита с максимальной скоростью
                    ;9013                                                   ; первым переключаемым битом является бит 0 памяти
U 0B53,  6512,15    ;9014           CLR LS[T9]                              ; очистка ячейки местной памяти, содержащей адрес
                    ;9015     BACK.T11.A:
U 0B54,  9812,F5    ;9016           MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи фона в буфер трансляции
U 0B55,  B29C,15    ;9017           WRITE.MEM LS[ZERO]                      ; пересылка нулей в буфер трансляции
U 0B56,  0A18,2C    ;9018           JSR [NEXT.TB.ADDRESS]                   ; вычисление следующего адреса буфера трансляции
U 0B57,  B8B5,44    ;9019           JMP [BACK.T11.A]                        ; возврат, если еще не выполнено
                    ;9020     REPEAT.T11.A.1:
U 0B58,  6512,15    ;9021           CLR LS[T9]                              ; запуск от адреса 0 буфера трансляции
U 0B59,  3614,15    ;9022           MOV LS[T10] TO WR[0]                    ; выборка номера ошибки
U 0B5A,  BE82,15    ;9023           MOV WR[0] TO LS[ERROR.NUMBER]           ; пересылка в местную память для распечатки
                    ;9024     LOOP.T11.A.1:
U 0B5B,  2F82,95    ;9025           CLR WR[1]                               ; ожидаемые данные
U 0B5C,  9C13,F5    ;9026           MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения буфера трансляции
U 0B5D,  3022,15    ;9027           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0B5E,  0869,3C    ;9028           JSR [CHECK.RESULT]                      ; проверка результата
U 0B5F,  B8B5,B4    ;9029           JMP [LOOP.T11.A.1]                      ; зацикливание при ошибке, если разрешено
U 0B60,  9812,F5    ;9030           MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи дополнения в буфер трансляции
U 0B61,  329E,15    ;9031           WRITE.MEM LS[ONES]                      ; запись единиц в ту же ячейку
                    ;9032             ADD WR[3] TO LS[T9],                  ;
U 0B62,  EC13,D5    ;9033           DT(SIZE)&SET.ALU.CC                     ; и установка кодов условий
U 0B63,  08B5,B0    ;9034           JMP.IF[N.CLR] TO [LOOP.T11.A.1]         ; если бит 15 не установлен, повторение со следующим
                    ;9035                                                   ; адресом
U 0B64,  EC13,15    ;9036           ADD WR[2] TO LS[T9]                     ; иначе увеличение адреса ниже переключенного бита
U 0B65,  B65E,15    ;9037           MOV LS[BIT15] TO WR[0]                  ; подготовка сброса бита 15 в ячейке 9 местной памяти
U 0B66,  6F12,15    ;9038           XOR WR[0] TO LS[T9]                     ; сброс бита 15 в адресе
                    ;9039             BIT WR[3] WITH LS[T9],                ; если переключаемый бит сейчас установлен -
                    ;9040                                                   ; выполнено,
U 0B67,  D913,B5    ;9041           DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0B68,  08B5,B9    ;9042           JMP.IF[BITS.CLR] TO [LOOP.T11.A.1]      ; повторение теста от нового адреса, если не выполнено
U 0B69,  3612,15    ;9043           MOV LS[T9] TO WR[0]                     ; выборка адреса из местной памяти
U 0B6A,  C528,15    ;9044           BIC LS[#FFFF] TO WR[0]                  ; очистка битов с 9 по 14 (вместе с другими, которые уже
                    ;9045                                                   ; сброшены)
                    ;9046             XOR LS[BIT31] TO WR[0],               ; переключение бита 31
U 0B6B,  437E,35    ;9047           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0B6C,  BE12,15    ;9048           MOV WR[0] TO LS[T9]                     ; пересылка нового адреса в LS 9
U 0B6D,  B8B5,B8    ;9049           JMP.IF[N.SET] TO [LOOP.T11.A.1]         ; если бит 31 был сброшен, повторение, используя тот же
                    ;9050                                                   ; самый переключаемый бит, но при установленном бите
                    ;9051                                                   ; 31, иначе подготовка запуска уменьшением адреса
U 0B6E,  6512,15    ;9052           CLR LS[T9]                              ; запуск от адреса 0 буфера трансляции (будет дополнение
                    ;9053                                                   ; для запуска от верхнего адреса)
U 0B6F,  FF82,15    ;9054           INC LS[ERROR.NUMBER]                    ; ошибка B
```

```
                    ;9055    REPEAT.T11.B.1:
U 0B70,  7D12,15    ;9056            COM LS[T9]                          ; дополнение адреса для получения убывающих адресов
                    ;9057    LOOP.T11.B.1:
U 0B71,  369E,95    ;9058            MOV LS[ONES] TO WR[1]               ; ожидаемые данные
U 0B72,  9C13,F5    ;9059            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]  ; подготовка чтения буфера трансляции
U 0B73,  3022,15    ;9060            MOV MEM.DATA TO WR[0]               ; выборка результата
U 0B74,  0869,3C    ;9061            JSR [CHECK.RESULT]                  ; проверка результата
U 0B75,  08B7,14    ;9062            JMP [LOOP.T11.B.1]                  ; зацикливание при ошибке, если разрешено
U 0B76,  9812,F5    ;9063            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG] ; подготовка записи дополнения в буфер трансляции
U 0B77,  B29C,15    ;9064            WRITE.MEM LS[ZERO]                  ; запись нулей в ту же ячейку
U 0B78,  7D12,15    ;9065            COM LS[T9]                          ; восстановление дополнения для получения оригинала
                    ;9066              ADD WR[3] TO LS[T9],              ; переключение бита в адресе
U 0B79,  EC13,D5    ;9067            DT(SIZE)&SET.ALU.CC                 ; и установка кодов условий
U 0B7A,  08B7,00    ;9068            JMP.IF[N.CLR] TO [REPEAT.T11.B.1]   ; если бит 15 не установлен, повторение со следующим
                    ;9069                                                ; адресом
U 0B7B,  EC13,15    ;9070            ADD WR[2] TO LS[T9]                 ; иначе увеличение адреса ниже переключенного бита
U 0B7C,  B65E,15    ;9071            MOV LS[BIT15] TO WR[0]              ; подготовка сброса бита 15 в LS9
U 0B7D,  6F12,15    ;9072            XOR WR[0] TO LS[T9]                 ; сброс бита 15 в адресе
                    ;9073              BIT WR[3] WITH LS[T9],            ; если переключенный бит 15 сейчас установлен -
                    ;9074                                                ; выполнено,
U 0B7E,  D913,B5    ;9075            DT(LONG)&SET.ALU.CC                 ; установка кодов условий
U 0B7F,  08B7,09    ;9076            JMP.IF[BITS.CLR] TO [REPEAT.T11.B.1] ; повторение теста от нового адреса, если не выполнено
U 0B80,  3612,15    ;9077            MOV LS[T9] TO WR[0]                 ; выборка адреса из местной памяти
U 0B81,  C528,15    ;9078            BIC LS[#FFFF] TO WR[0]              ; очистка битов с 9 по 14 (вместе с другими, которые уже
                    ;9079                                                ; сброшены)
                    ;9080              XOR LS[BIT31] TO WR[0],           ; переключение бита 31 и
U 0B82,  437E,35    ;9081            DT(LONG)&SET.ALU.CC                 ; установка кодов условий
U 0B83,  BE12,15    ;9082            MOV WR[0] TO LS[T9]                 ; пересылка нового адреса в LS9
U 0B84,  88B7,08    ;9083            JMP.IF[N.SET] TO [REPEAT.T11.B.1]   ; если бит 31 был сброшен, повторение, с использованием
                    ;9084                                                ; того же самого переключаемого бита, но с установленным
                    ;9085                                                ; битом 31
U 0B85,  A3C1,95    ;9086            ROL WR[3]                           ; следующий бит для переключения с максимальной
                    ;9087                                                ; скоростью
                    ;9088              BIT LS[BIT15] WITH WR[3],         ; проверка, установлен ли бит 15 и
U 0B86,  595F,B5    ;9089            DT(LONG)&SET.ALU.CC                 ; установка кодов условий
U 0B87,  08B5,89    ;9090            JMP.IF[BITS.CLR] TO [REPEAT.T11.A.1] ; повторение теста с новым переключаемым битом, если бит
                    ;9091                                                ; 15 не установлен, иначе подготовка переключения
                    ;9092                                                ; следующего бита 31
U 0B88,  367F,95    ;9093            MOV LS[BIT31] TO WR[3]              ; подготовка переключения с максимальной скоростью
                    ;9094                                                ; сейчас будет переключен бит 6 памяти
U 0B89,  6512,15    ;9095            CLR LS[T9]                          ; запуск от адреса 0 буфера трансляции
U 0B8A,  3614,15    ;9096            MOV LS[T10] TO WR[0]                ; выборка номера ошибки
U 0B8B,  BE82,15    ;9097            MOV WR[0] TO LS[ERROR.NUMBER]       ; запоминание в местной памяти для распечатки ошибки
                    ;9098    LOOP.T11.A1.1:
U 0B8C,  2F82,95    ;9099            CLR WR[1]                           ; ожидаемые данные
U 0B8D,  9C13,F5    ;9100            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]  ; подготовка чтения буфера трансляции
U 0B8E,  3022,15    ;9101            MOV MEM.DATA TO WR[0]               ; выборка результата
U 0B8F,  0869,3C    ;9102            JSR [CHECK.RESULT]                  ; проверка результата
U 0B90,  8888,C4    ;9103            JMP [LOOP.T11.A1.1]                 ; зацикливание при ошибке, если разрешено
U 0B91,  9812,F5    ;9104            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG] ; подготовка записи дополнения в буфер трансляции
U 0B92,  329E,15    ;9105            WRITE.MEM LS[ONES]                  ; запись единиц в ту же ячейку
                    ;9106              XOR WR[3] TO LS[T9],              ; переключение бита в адресе
U 0B93,  EF13,B5    ;9107            DT(LONG)&SET.ALU.CC                 ; и установка кодов условий
U 0B94,  8888,C8    ;9108            JMP.IF[N.SET] TO [LOOP.T11.A1.1]    ; если бит 31 не сброшен, повторение со следующим
                    ;9109                                                ; адресом
```

```
                 ;9110              ADD WR[2] TO LS[T9],                    ; иначе увеличение адреса ниже переключаемого бита
U 0B95, 6C13,55  ;9111            DT(SIZE)&SET.ALU.CC                     ; и установка кодов условий
U 0B96, 08B8,C0  ;9112            JMP.IF[N.CLR] TO [LOOP.T11.A1.1]        ; повторение теста с новым адресом, если не выполнено
U 0B97, 6512,15  ;9113            CLR LS[T9]                              ; запуск от адреса 0 буфера трансляции (будет выполнено
                 ;9114                                                    ; дополнение для формирования верхнего адреса)
U 0B98, FF82,15  ;9115            INC LS[ERROR.NUMBER]                    ; ошибка B
                 ;9116    REPEAT.T11.B1.1:
U 0B99, 7D12,15  ;9117            COM LS[T9]                              ; дополнение адреса для уменьшения
                 ;9118    LOOP.T11.B1.1:
U 0B9A, 369E,95  ;9119            MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0B9B, 9C13,F5  ;9120            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка чтения буфера трансляции
U 0B9C, 3022,15  ;9121            MOV MEM.DATA TO WR[0]                   ; получение результата
U 0B9D, 0869,3C  ;9122            JSR [CHECK.RESULT]                      ; проверка результата
U 0B9E, 08B9,A4  ;9123            JMP [LOOP.T11.B1.1]                     ; зацикливание при ошибке, если разрешено
U 0B9F, 9812,F5  ;9124            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи дополнения в буфере трансляции
U 0BA0, B29C,15  ;9125            WRITE.MEM LS[ZERO]                      ; запись нулей в ту же ячейку
U 0BA1, 7D12,15  ;9126            COM LS[T9]                              ; восстановление адреса для получения оригинала
                 ;9127              XOR WR[3] TO LS[T9],                  ; переключение бита в адресе и
U 0BA2, EF13,B5  ;9128            DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0BA3, 08B9,98  ;9129            JMP.IF[N.SET] TO [REPEAT.T11.B1.1]      ; если бит 31 не сброшен, повторение со следующим
                 ;9130                                                    ; адресом
                 ;9131              ADD WR[2] TO LS[T9],                   ; иначе увеличение адреса ниже переключаемого бита и
U 0BA4, 6C13,55  ;9132            DT(SIZE)&SET.ALU.CC                     ; установка кодов условий
U 0BA5, 88B9,90  ;9133            JMP.IF[N.CLR] TO [REPEAT.T11.B1.1]      ; повторение теста с новым адресом, если не выполнено,
                 ;9134                                                    ; иначе выполнено
                 ;9135    END.T11:
```

```
;9136    .PAGE "ТЕСТ 12 - тест памяти отображения адресов общей шины (модуль МСТ)"
;9137    ;
;9138    ; ОПИСАНИЕ ТЕСТА:
;9139    ;
;9140    ;       Этот тест проверяет часть памяти 1KX4 буфера трансляции модуля МСТ, пред-
;9141    ;    назначеную для отображения адресов общей шины. Обращение к адресам 200(H) по
;9142    ;    3FF(H) (512 десятичных адресов) возможно через микропрограмму контроллера ОЗУ
;9143    ;    WRITE.UBS.MAP (запись в область отображения общей шины). Эти же адреса можно
;9144    ;    прочитать обратно для проверки программой READ.UBS.MAP (чтение области отоб-
;9145    ;    ражения общей шины).
;9146    ;       Другие адреса памяти 1KX4 либо не используются, либо проверены предыдущим
;9147    ;    тестом. Адреса обращения к памяти буфера трансляции определяются битами 09
;9148    ;    по 17 регистра виртуального адреса. Бит 9 адреса памяти устанавливается в
;9149    ;    1 аппаратно. Биты регистра виртуального адреса с 0 по 8 и с 18 по 31 не вли-
;9150    ;    яют на выборку адреса памяти отображения общей шины. Адрес передается из
;9151    ;    местной памяти во время инструкции MEM.REQ центрального процессора с полем
;9152    ;    MF=WRITE.UBS.MAP(0F) или READ.UBS MAP(12H). Инструкция WRITE.MEM LS централь-
;9153    ;    ного процессора передает данные, которые должны быть записаны в область ото-
;9154    ;    бражения общей шины следующим образом: биты местной памяти 00-15 поступают в
;9155    ;    биты PA 09-24 соответственно, биты 25-31 местной памяти поступают в биты
;9156    ;    BYTE OFFSET,MODIFY, PROT A, PROT B,PROT C, PROT D и VALID соответственно.
;9157    ;    Данные, которые записываются в область отображения общей шины, циклически
;9158    ;    сдвигаются в сдвигателях данных и запоминаются в регистрах схемы ECC. Затем
;9159    ;    данные считываются из этих регистров и записываются в память буфера трансля-
;9160    ;    ции. Заметим, что бит PA24 не используется, и в буфер трансляции записываются
;9161    ;    только биты PA 9-23.
;9162    ;       Путь данных при записи следующий: выдается инструкция MEM.REQ с полем
;9163    ;    MF=WRITE.UBS.MAP и DT=LONGWORD. Начальное ветвление выполняется как описано
;9164    ;    в начале этого листинга. По адресу начального ветвления приемо-передатчики
;9165    ;    подготавливаются для приема из центрального процессора данных для записи в
;9166    ;    буфер трансляции, а приемо-передатчики МСТ подготавливаются для передачи
;9167    ;    данных из шины BUS MC в биты PA и другие биты данных буфера трансляции. Ус-
;9168    ;    танавливаются MEMORY BUSY, ROTATE BYTE и устанавливается TB MUX UNIBUS для
;9169    ;    разрешения декодирования адресов общей шины. Остальной путь данных такой же,
;9170    ;    как в микропрограмме WRITE.TB предыдущего теста, начиная с адреса начального
;9171    ;    ветвления.
;9172    ;       Микропрограмма чтения инициализируется инструкцией MEM.REQ с полем MF=READ
;9173    ;    .UBS.MAP(12H). Начальное ветвление выполняется как описано в начале этого лис-
;9174    ;    тинга. По адресу начального ветвления разрешается обход регистра виртуально-
;9175    ;    го адреса, устанавливаются MEMORY BUSY и разрешаются приемо-передатчики МСТ,
;9176    ;    которые пропускают данные буфера трансляции, а не биты PA. TB MUX UNIBUS так-
;9177    ;    же установлен для подготовки адресов общей шины. Остальной путь данных такой
;9178    ;    же, как и в микропрограмме READ.TB предыдущего теста, начиная с адреса на-
;9179    ;    чального ветвления.
;9180    ;       Упомянутые выше циклы используют тестовые данные: все единицы, все нули,
;9181    ;    сдвигаемая единица, сдвигаемый нуль. Потом выполняется тест со сдвигаемым би-
;9182    ;    том в адресе для проверки адресных линий. В конце выполняется тест "БЕГУЩЕЙ
;9183    ;    ИНВЕРСИИ" для проверки задержек схем адреса. Этот тест выполняется как тест
;9184    ;    "МАРШ", за исключением того, что выполняется 9 раз, каждый раз переключая раз-
;9185    ;    ные биты адреса с максимальной скоростью. "МАРШ" должен записать фон нулей,
;9186    ;    потом при последовательном возрастании адресов пройти память отображения об-
;9187    ;    щей шины чтением нулей и записью единиц. Когда последний адрес будет проверен,
;9188    ;    "МАРШ" должен при последовательном уменьшении адресов выполнить чтение еди-
;9189    ;    ниц и запись нулей.
;9190    ;
```

```
;9191    ; ПРЕДПОЛОЖЕНИЯ:
;9192    ;
;9193    ;   Предполагается, что все предыдущие тесты выполнены успешно.
;9194    ;
;9195    ; ШАГИ ТЕСТА:
;9196    ;
;9197    ;     1.Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;9198    ;   (для распечатки ошибок) и очистка предыдущего номера ошибки в местной памяти.
;9199    ;   Маска ошибки разрешает проверку битов с 1 по 22.
;9200    ;     2.Загрузка всех единиц в WR1 и выполнение инструкции MEM.REQ с MF=WRITE.
;9201    ;   UBS.MAP и DT=LONGWORD. Ячейка местной памяти, содержащая нули, используется
;9202    ;   в качестве адреса (так как бит 9 адреса памяти устанавливается высоким прину-
;9203    ;   дительно, запись будет выполняться по адресу 200(H))
;9204    ;     3.Выполнение инструкции WRITE.MEM LS с ячейки местной памяти, содержащей
;9205    ;   все единицы.
;9206    ;     4.Выполнение инструкции MEM.REQ с MF=READ.UBS.MAP и DT=LONGWORD. Ячейка
;9207    ;   местной памяти, содержащая нули, используется в качестве адреса (читается
;9208    ;   адрес 200(H)
;9209    ;     5.Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата на
;9210    ;   единицы.
;9211    ;     6.Повторение шагов с 2 по 5 с данными - нули в шагах 3 и 5.
;9212    ;     7.Повторение шагов с 2 по 5 с данными - сдвигаемая единица в шагах 3 и 5.
;9213    ;   ожидаемыми данными в WR1 являются данные, переданные со сдвигом на 8 битов
;9214    ;   влево.
;9215    ;     8.Повторение шагов с 2 по 5 с данными - сдвигаемый нуль в шагах 3 и 5. Ожи-
;9216    ;   даемыми данными в WR1 являются данные, переданные со сдвигом на 8 битов вле-
;9217    ;   во.
;9218    ;     9.Повторение шагов с 2 по 5 с тестовыми данными: нули, единицы, сдвигаемая
;9219    ;   единица, сдвигаемый нуль для всех адресов в порядке возрастания с каждым набо-
;9220    ;   ром данных. Этим проверяются все ячейки памяти, к которым может быть выполне-
;9221    ;   но обращение. Ожидаемыми данными в WR1 являются данные, переданные со сдвигом
;9222    ;   на 8 битов влево.
;9223    ;     10.Выполнение подтеста с набором типа "МАРШ" при изменении адреса сдвигом
;9224    ;   бита содержащего 1, через адресное поле из нулей. Запись фона нулей, чтение ну-
;9225    ;   лей, запись единиц, чтение нулей. это повторяется 9 раз, пока единица сдвигает-
;9226    ;   ся через адресные линии. Этим проверяются адресные линии и грубые отказы в мик-
;9227    ;   росхеме памяти.
;9228    ;     11.Выполнение теста "БЕГУЩЕЙ ИНВЕРСИИ" (описан выше).
;9229    ;
;9230    ; ОШИБКИ:
;9231    ;
;9232    ;     ПРИМЕЧАНИЕ: Данными под OTHER является проверяемый адрес буфера трансляции.
;9233    ;
;9234    ;     ошибка 1 - неправильно выполняется запись или чтение памяти отображения об-
;9235    ;                общей шины с данными - все единицы. Адрес памяти 200(H).
;9236    ;     ошибка 2 - неправильно выполняется запись или чтение памяти отображения
;9237    ;                общей шины с данными - все нули. Адрес памяти 200(H).
;9238    ;     ошибка 3 - неправильно выполняется запись или чтение памяти отображения
;9239    ;                общей шины с данными - сдвигаемая единица. Адрес памяти 200(H).
;9240    ;     ошибка 4 - неправильно выполняется запись или чтение памяти отображения
;9241    ;                общей шины с данными - сдвигаемый нуль. Адрес памяти 200(H)
;9242    ;     ошибка 5 - неправильно работает память отображения общей шины с данными
;9243    ;                - все единицы.
;9244    ;     ошибка 6 - неправильно работает память отображения общей шины с данными
;9245    ;                - все нули.
```

```
                ;9246   ;      ошибка 7 - неправильно работает память отображения общей шины с данными
                ;9247   ;                 - сдвигаемая единица.
                ;9248   ;      ошибка 8 - неправильно работает память отображения общей шины с данными
                ;9249   ;                 - сдвигаемый нуль.
                ;9250   ;      ошибка 9 - неправильно работает память отображения общей шины в тесте со
                ;9251   ;                 сдвигаемым битом в адресе.
                ;9252   ;      ошибка А - неправильно работает память отображения общей шины с набором
                ;9253   ;                 "БЕГУЩЕЙ ИНВЕРСИИ" при наращивании адресов.
                ;9254   ;      ошибка В - неправильно работает память отображения общей шины с набором
                ;9255   ;                 "БЕГУЩЕЙ ИНВЕРСИИ" при уменьшении адресов.
                ;9256   ;
                ;9257   ; НАЛАДКА:
                ;9258   ;
                ;9259   ;      ПРИМЕЧАНИЕ: Адрес памяти отображения общей шины, который работает неправиль-
                ;9260   ;    но, в сообщении об ошибке печатается под OTHER (другие данные) (за исключением
                ;9261   ;    ошибок А и В). Биты располагаются так, чтобы получить действительный адрес па-
                ;9262   ;    мяти в шестнадцатеричном формате.
                ;9263   ;
                ;9264   ;      ОШИБКИ с 1 по 8 - Скорее всего, неправильно работает сама микросхема памяти,
                ;9265   ;    так как все входы данных и схемы управления были проверены в предыдущем тесте
                ;9266   ;    памяти буфера трансляции. Отличается только адрес и не влияет на тест данных.
                ;9267   ;
                ;9268   ;      ОШИБКА 9 - Эта ошибка указывает на возможную неисправность адресных линий
                ;9269   ;    или адресных схем внутри микросхемы. если неисправны все биты, подозреваются
                ;9270   ;    адресные линии. Адресные линии можно проверить в режиме SOMM (останов по сов-
                ;9271   ;    падению микроадреса) на инструкции MEM.REQ с полем MF=WRITE.UBS.MAP.
                ;9272   ;       Центральный процессор остановится при каждом совпадении микроадреса и мож-
                ;9273   ;    но проверить адресные линии на модуле МСТ. Единица должна сдвигаться через
                ;9274   ;    адресные линии от LVA 09 H по LVA 14 H, а затем на TBA 6 H,7 H и 8 H после-
                ;9275   ;    каждой команды CONTINUE (продолжение). TBA9 всегда должен быть высоким. Если
                ;9276   ;    TBA 6 H, 7 H или 8 H неправильно переходит в высокий уровень, необходимо про-
                ;9277   ;    верить мультиплексор 4X2, выходы которого являются адресом TBA. Сигнал выбор-
                ;9278   ;    ки UB TB SEL H должен быть высоким. Этот сигнал может быть проверен в обрат-
                ;9279   ;    ном направлении на инверторе "ИЛИ" со входами CPH/UBL H и UB TB SEL L. Сигнал
                ;9280   ;    UB TB SEL L должен быть низким. Если нет, подозревается ПМЛ, которая формирует
                ;9281   ;    этот сигнал.
                ;9282   ;       Если не все биты неправильные, подозревается неисправная ячейка в микросхеме
                ;9283   ;    памяти. Если неправильной является группа из четырех битов, подозревается обрыв
                ;9284   ;    адресной линии, поступающей на эту микросхему памяти (скорее всего адресные ли-
                ;9285   ;    нии 7,8 или 9).
                ;9286   ;       Другим источником неисправности может быть LVA 15 H, поступающий из регис-
                ;9287   ;    тра виртуального адреса в микропрограмме чтения буфера трансляции. Для про-
                ;9288   ;    верки, что этот бит переключается правильно, можно использовать зацикливание
                ;9289   ;    при ошибке.
                ;9290   ;
                ;9291   ;      ОШИБКА А и В - Эти ошибки скорее всего указывают на неисправность в микро-
                ;9292   ;    схеме памяти - или в отдельной ячейке или в схемах переключения адреса.
                ;9293   ;      Адрес может быть определен чтением ячейки 9(H) местной памяти. Биты 9-14
                ;9294   ;    соответствуют битам 0-5 на микросхеме памяти. Бит 31 соответствует биту 6 на
                ;9295   ;    микросхеме памяти.
                ;9296
                ;9297    T.12:
U 0BA6, B65E,15 ;9298           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;9299                                                   ; состояния
U 0BA7, 3E80,15 ;9300           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
```

```
                ;9301                                                   ; 15 указывает начало теста консольному процессору
U 0BA8, 10E0,15 ;9302           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN консольному процессору
                ;9303   WAIT.T12.0:
U 0BA9, 08BA,94 ;9304           JMP [WAIT.T12.0]                        ; зацикливание для ожидания ответа клнсольного
                ;9305                                                   ; процессора
U 0BAA, 0A1A,AC ;9306           JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0BAB, B640,15 ;9307           MOV LS[#1] TO WR[0]                     ; 1 в WR0
U 0BAC, BEFC,15 ;9308           MOV WR[0] TO LS[SIZE]                   ; подготовка регистра размера для типа данных = слово
U 0BAD, B62A,15 ;9309           MOV LS[#FF000000] TO WR[0]              ; установка старшего байта маски ошибки
U 0BAE, C76E,15 ;9310           BIS LS[BIT23] TO WR[0]                  ; установка бита 23
U 0BAF, C740,15 ;9311           BIS LS[BIT0] TO WR[0]                   ; и установка бита 0. Сейчас биты с 1 по 22 будут
                ;9312                                                   ; проверяться на наличие ошибки
U 0BB0, 3E8A,15 ;9313           MOV WR[0] TO LS[ERROR.MASK]             ; запоминание маски ошибки в местной памяти
U 0BB1, B652,15 ;9314           MOV LS[#200] TO WR[0]                   ; пересылка значения 200 в WR0
U 0BB2, BE88,15 ;9315           MOV WR[0] TO LS[ADDRESS.DATA]           ; загрузка 200 для распечатки адреса под OTHER в
                ;9316                                                   ; сообщении об ошибке
U 0BB3, B670,15 ;9317           MOV LS[OTHER.DATA] TO WR[0]             ; установка бита 24 в WR0
U 0BB4, 3E80,15 ;9318           MOV WR[0] TO LS[CONTROL.STATUS]         ; консольный процессор будет выполнять печать под
                ;9319                                                   ; OTHER при ошибке
                ;9320   LOOP.T12.1:
U 0BB5, 369E,95 ;9321           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0BB6, 1B9D,F5 ;9322           MEM.REQ[WRITE.UBS.MAP] ADRS[#0] DT[LONG] ; подготовка записи в память отображения общей шины по
                ;9323                                                   ; адресу 200
U 0BB7, 329E,15 ;9324           WRITE.MEM LS[ONES]                      ; запись всех единиц в память отображения общей шины
U 0BB8, 1C9D,75 ;9325           MEM.REQ[READ.UBS.MAP] ADRS[#0] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0BB9, 3022,15 ;9326           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0BBA, 0869,3C ;9327           JSR [CHECK.RESULT]                      ; проверка результата
U 0BBB, 88B8,54 ;9328           JMP [LOOP.T12.1]                        ; зацикливание при ошибке, если разрешено
U 0BBC, FF82,15 ;9329           INC LS[ERROR.NUMBER]                    ; ошибка 2
                ;9330   LOOP.T12.2:
U 0BBD, B69C,95 ;9331           MOV LS[ZERO] TO WR[1]                   ; ожидаемые данные
U 0BBE, 1B9D,F5 ;9332           MEM.REQ[WRITE.UBS.MAP] ADRS[#0] DT[LONG] ; подготовка записи в память отображения общей шины
                ;9333                                                   ; по адресу 200
U 0BBF, B29C,15 ;9334           WRITE.MEM LS[ZERO]                      ; запись всех нулей в память отображения общей шины
U 0BC0, 1C9D,75 ;9335           MEM.REQ[READ.UBS.MAP] ADRS[#0] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0BC1, 3022,15 ;9336           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0BC2, 0869,3C ;9337           JSR [CHECK.RESULT]                      ; проверка результата
U 0BC3, 08BB,D4 ;9338           JMP [LOOP.T12.2]                        ; зацикливание при ошибке, если разрешено
U 0BC4, FF82,15 ;9339           INC LS[ERROR.NUMBER]                    ; ошибка 3
U 0BC5, E5F8,15 ;9340           CLR LS[OS]                              ; очистка индекса
U 0BC6, 3651,95 ;9341           MOV LS[BIT8] TO WR[3]                   ; первые ожидаемые данные (полученные данные сдвинуты на
                ;9342                                                   ; 8 битов влево)
                ;9343   LOOP.T12.3:
U 0BC7, 2006,95 ;9344           MOV WR[3] TO WR[1]                      ; ожидаемые данные
U 0BC8, 1B9D,F5 ;9345           MEM.REQ[WRITE.UBS.MAP] ADRS[#0] DT[LONG] ; подготовка записи в память отображения общей шины по
                ;9346                                                   ; адресу 200
U 0BC9, B2F6,15 ;9347           WRITE.MEM LS[SHIFT.OS(4-0)]             ; запись сдвигаемой единицы в память отображения общей
                ;9348                                                   ; шины
U 0BCA, 1C9D,75 ;9349           MEM.REQ[READ.UBS.MAP] ADRS[#0] DT[LONG]; подготовка чтения памяти отображения общей шины по
                ;9350                                                   ; адресу 200
U 0BCB, 3022,15 ;9351           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0BCC, 0869,3C ;9352           JSR [CHECK.RESULT]                      ; проверка результата
U 0BCD, 88BC,74 ;9353           JMP [LOOP.T12.3]                        ; зацикливание при ошибке, если разрешено
U 0BCE, A3C1,95 ;9354           ROL WR[3]                               ; следующий набор ожидаемых данных
U 0BCF, 7FF8,15 ;9355           INC LS[OS]                              ; увеличение индекса для следующего набора
```

```
U 0BD0, B6F8,15 ;9356            MOV LS[OS] TO WR[0]                    ; подготовка проверки, всё ли наборы использованы
                ;9357              BIT LS[BIT5] WITH WR[0],             ; проверка, увеличен ли OS до 20(H) и установка кодов
                ;9358                                                   ; условий
U 0BD1, 594A,35 ;9359            DT(LONG)&SET.ALU.CC                    ;
U 0BD2, 08BC,79 ;9360            JMP.IF[BITS.CLR] TO [LOOP.T12.3]       ; повторение со следующим набором, если не все
                ;9361                                                   ; использованы
U 0BD3, FF82,15 ;9362            INC LS[ERROR.NUMBER]                   ; ошибка 4
U 0BD4, E5F8,15 ;9363            CLR LS[OS]                             ; очистка индекса
U 0BD5, 5F51,95 ;9364            MCOM LS[BIT8] TO WR[3]                 ; первые ожидаемые данные (полученные данные сдвинуты на
                ;9365                                                   ; 8 разрядов влево)
                ;9366    REPEAT.T12.4:
U 0BD6, 5FF6,15 ;9367            MCOM LS[SHIFT.OS(4-0)] TO WR[0]        ; выборка текущего набора тестовых данных
U 0BD7, 3E10,15 ;9368            MOV WR[0] TO LS[T8]                    ; пересылка тестовых данных в ячейку временного хранения
                ;9369                                                   ; местной памяти
                ;9370    LOOP.T12.4:
U 0BD8, 2006,95 ;9371            MOV WR[3] TO WR[1]                     ; ожидаемые данные
U 0BD9, 189D,F5 ;9372            MEM.REQ[WRITE.UBS.MAP] ADRS[#0] DT[LONG] ; подготовка записи в память отображения общей шины по
                ;9373                                                   ; адресу 200
U 0BDA, 3210,15 ;9374            WRITE.MEM LS[T8]                       ; запись сдвигаемого нуля в память отображения общей
                ;9375                                                   ; шины
U 0BDB, 1C9D,75 ;9376            MEM.REQ[READ.UBS.MAP] ADRS[#0] DT[LONG]; подготовка чтения памяти отображения общей шины по
                ;9377                                                   ; адресу 200
U 0BDC, 3022,15 ;9378            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 0BDD, 0869,3C ;9379            JSR [CHECK.RESULT]                     ; проверка результата
U 0BDE, 08BD,84 ;9380            JMP [LOOP.T12.4]                       ; зацикливание при ошибке, если разрешено
U 0BDF, A3C1,95 ;9381            ROL WR[3]                              ; следующий набор ожидаемых данных
U 0BE0, 7FF8,15 ;9382            INC LS[OS]                             ; увеличение индекса для следующего набора
U 0BE1, B6F8,15 ;9383            MOV LS[OS] TO WR[0]                    ; подготовка проверки, все ли наборы использованы
                ;9384              BIT LS[BIT5] WITH WR[0],             ; проверка, увеличен ли OS до 20(H) и установка кодов
                ;9385                                                   ; условий
U 0BE2, 594A,35 ;9386            DT(LONG)&SET.ALU.CC                    ;
U 0BE3, 08BD,69 ;9387            JMP.IF[BITS.CLR] TO [REPEAT.T12.4]     ; повторение со следующим набором, если не все
                ;9388                                                   ; использованы
U 0BE4, FF82,15 ;9389            INC LS[ERROR.NUMBER]                   ; ошибка 5
U 0BE5, 6512,15 ;9390            CLR LS[T9]                             ; очистка адреса памяти отображения общей шины (адрес
                ;9391                                                   ; хранится в ячейке 9 местной памяти)
                ;9392    LOOP.T12.5:
U 0BE6, 369E,95 ;9393            MOV LS[ONES] TO WR[1]                  ; ожидаемые данные
U 0BE7, 1B13,F5 ;9394            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи в память отображения общей шины
U 0BE8, 329E,15 ;9395            WRITE.MEM LS[ONES]                     ; запись всех единиц в память отображения общей шины
U 0BE9, 1C13,75 ;9396            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0BEA, 3022,15 ;9397            MOV MEM.DATA TO WR[0]                  ; выборка результата
U 0BEB, 0869,3C ;9398            JSR [CHECK.RESULT]                     ; проверка результата
U 0BEC, 88BE,64 ;9399            JMP [LOOP.T12.5]                       ; зацикливание при ошибке, если разрешено
U 0BED, 8A18,CC ;9400            JSR [NEXT.UB.ADDRESS]                  ; вычисление следующего адреса памяти отображения общей
                ;9401                                                   ; шины
U 0BEE, 88BE,64 ;9402            JMP [LOOP.T12.5]                       ; возврат сюда, если не все адреса использованы
U 0BEF, FF82,15 ;9403            INC LS[ERROR.NUMBER]                   ; ошибка 6
U 0BF0, 6512,15 ;9404            CLR LS[T9]                             ; очистка адреса памяти отображения общей шины (адрес
                ;9405                                                   ; хранится в ячейке 9 местной памяти)
U 0BF1, B652,15 ;9406            MOV LS[#200] TO WR[0]                  ; пересылка значения 200 в WR0
U 0BF2, BE88,15 ;9407            MOV WR[0] TO LS[ADDRESS.DATA]          ; загрузка 200 для распечатки под OTHER в качестве адреса
                ;9408                                                   ; в сообщении об ошибке
                ;9409    LOOP.T12.6:
U 0BF3, B69C,95 ;9410            MOV LS[ZERO] TO WR[1]                  ; ожидаемые данные
```

```
U 0BF4, 1B13,F5 ;9411           MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи в память отображения общей шины
U 0BF5, B29C,15 ;9412           WRITE.MEM LS[ZERO]                       ; запись всех нулей в память отображения общей шины
U 0BF6, 1C13,75 ;9413           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0BF7, 3022,15 ;9414           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 0BF8, 0869,3C ;9415           JSR [CHECK.RESULT]                       ; проверка результата
U 0BF9, 08BF,34 ;9416           JMP [LOOP.T12.6]                         ; зацикливание при ошибке, если разрешено
U 0BFA, 8A18,CC ;9417           JSR [NEXT.UB.ADDRESS]                    ; вычисление следующего адреса памяти отображения общей
                ;9418                                                    ; шины
U 0BFB, 08BF,34 ;9419           JMP [LOOP.T12.6]                         ; возврат сюда, если не все адреса использованы
U 0BFC, FF82,15 ;9420           INC LS[ERROR.NUMBER]                     ; ошибка 7
U 0BFD, E5F8,15 ;9421           CLR LS[OS]                               ; очистка индекса
U 0BFE, 6512,15 ;9422           CLR LS[T9]                               ; очистка адреса памяти отображения общей шины (адрес
                ;9423                                                    ; хранится в ячейке 9 местной памяти)
U 0BFF, 3651,95 ;9424           MOV LS[BIT8] TO WR[3]                    ; первые ожидаемые данные (полученные данные)
U 0C00, B652,15 ;9425           MOV LS[#200] TO WR[0]                    ; пересылка значения 200 в WR0
U 0C01, BE88,15 ;9426           MOV WR[0] TO LS[ADDRESS.DATA]            ; загрузка 200 для распечатки под OTHER в сообщении об
                ;9427                                                    ; ошибке в качестве адреса
                ;9428   LOOP.T12.7:
U 0C02, 2006,95 ;9429           MOV WR[3] TO WR[1]                       ; ожидаемые данные
U 0C03, 1B13,F5 ;9430           MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи в память отображения общей шины
U 0C04, B2F6,15 ;9431           WRITE.MEM LS[SHIFT.OS(4-0)]              ; запись сдвигаемой единицы в память отображения общей
                ;9432                                                    ; шины
U 0C05, 1C13,75 ;9433           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0C06, 3022,15 ;9434           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 0C07, 0869,3C ;9435           JSR [CHECK.RESULT]                       ; проверка результата
U 0C08, 08C0,24 ;9436           JMP [LOOP.T12.7]                         ; зацикливание при ошибке, если разрешено
U 0C09, A3C1,95 ;9437           ROL WR[3]                                ; следующий набор ожидаемых данных
U 0C0A, 7FF8,15 ;9438           INC LS[OS]                               ; увеличение индекса для следующего набора
U 0C0B, B6F8,15 ;9439           MOV LS[OS] TO WR[0]                      ; подготовка проверки, все ли наборы использованы
                ;9440             BIT LS[BIT5] WITH WR[0],               ; проверка, увеличен ли OS до 20(H) и установка кодов
                ;9441                                                    ; условий
U 0C0C, 594A,35 ;9442           DT(LONG)&SET.ALU.CC                      ;
U 0C0D, 88C0,29 ;9443           JMP.IF[BITS.CLR] TO [LOOP.T12.7]         ; повторение со следующим набором, если не все
                ;9444                                                    ; использованы
U 0C0E, E5F8,15 ;9445           CLR LS[OS]                               ; очистка индекса
U 0C0F, 8A18,CC ;9446           JSR [NEXT.UB.ADDRESS]                    ; вычисление следующего адреса памяти отображения общей
                ;9447                                                    ; шины
U 0C10, 08C0,24 ;9448           JMP [LOOP.T12.7]                         ; возврат сюда, если не все адреса использованы
U 0C11, FF82,15 ;9449           INC LS[ERROR.NUMBER]                     ; ошибка 8
U 0C12, E5F8,15 ;9450           CLR LS[OS]                               ; очистка индекса
U 0C13, 6512,15 ;9451           CLR LS[T9]                               ; очистка адреса памяти отображения общей шины (адрес
                ;9452                                                    ; хранится в ячейке 9 местной памяти)
U 0C14, B652,15 ;9453           MOV LS[#200] TO WR[0]                    ; пересылка значения 200 в WR0
U 0C15, BE88,15 ;9454           MOV WR[0] TO LS[ADDRESS.DATA]            ; пересылка 200 для распечатки под OTHER в сообщении об
                ;9455                                                    ; ошике в качестве адреса
U 0C16, 5F51,95 ;9456           MCOM LS[BIT8] TO WR[3]                   ; первые ожидаемые данные (полученны данные сдвинутые на
                ;9457                                                    ; 8 разрядов влево)
                ;9458   REPEAT.T12.8:
U 0C17, 5FF6,15 ;9459           MCOM LS[SHIFT.OS(4-0)] TO WR[0]          ; выборка текущего набора тестовых данных
U 0C18, 3E10,15 ;9460           MOV WR[0] TO LS[T8]                      ; пересылка набора данных в ячейку временного хранения
                ;9461                                                    ; местной памяти
                ;9462   LOOP.T12.8:
U 0C19, 2006,95 ;9463           MOV WR[3] TO WR[1]                       ; ожидаемые данные
U 0C1A, 1B13,F5 ;9464           MEM.REQ[WRITE.UBS MAP] ADRS[T9] DT[LONG] ; подготовка записи в память отображения общей шины
U 0C1B, 3210,15 ;9465           WRITE.MEM LS[T8]                         ; запись сдвигаемого нуля в память отображения общей
```

```
                    ;9466                                                   ; шины
U 0C1C, 1C13,75     ;9467           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0C1D, 3022,15     ;9468           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0C1E, 0869,3C     ;9469           JSR [CHECK.RESULT]                      ; проверка результата
U 0C1F, 08C1,94     ;9470           JMP [LOOP.T12.8]                        ; зацикливание при ошибке, если разрешено
U 0C20, A3C1,95     ;9471           ROL WR[3]                               ; следующий набор ожидаемых данных
U 0C21, 7FF8,15     ;9472           INC LS[OS]                              ; увеличение индекса для следующего набора
U 0C22, B6F8,15     ;9473           MOV LS[OS] TO WR[0]                     ; подготовка проверки, все ли наборы использованы
                    ;9474             BIT LS[BIT5] WITH WR[0],              ; проверка, увеличен ли OS до 20(H), установка кодов
                    ;9475                                                   ; условий
U 0C23, 594A,35     ;9476           DT(LONG)&SET.ALU.CC                     ;
U 0C24, 08C1,79     ;9477           JMP.IF[BITS.CLR] TO [REPEAT.T12.8]      ; повторение со следующим набором, если не все
                    ;9478                                                   ; использованы
U 0C25, E5F8,15     ;9479           CLR LS[OS]                              ; очистка индекса
U 0C26, 8A18,CC     ;9480           JSR [NEXT.UB.ADDRESS]                   ; вычисление следующего адреса памяти отображения общей
                    ;9481                                                   ; шины
U 0C27, 88C1,74     ;9482           JMP [REPEAT.T12.8]                      ; возврат, если не все адреса проверены
U 0C28, FF82,15     ;9483           INC LS[ERROR.NUMBER]                    ; ошибка 9
U 0C29, 6512,15     ;9484           CLR LS[T9]                              ; очистка ячейки местной памяти, содержащей адрес
                    ;9485       BACK.T12.9:
U 0C2A, 1B13,F5     ;9486           MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи фона в память отображения общей
                    ;9487                                                   ; шины
U 0C2B, B29C,15     ;9488           WRITE.MEM LS[ZERO]                      ; запись нулей в память отображения общей шины
U 0C2C, 8A18,CC     ;9489           JSR [NEXT.UB.ADDRESS]                   ; вычисление следующего адреса памяти отображения общей
                    ;9490                                                   ; шины
U 0C2D, 08C2,A4     ;9491           JMP [BACK.T12.9]                        ; возврат, если не все выполнено
U 0C2E, B652,15     ;9492           MOV LS[#200] TO WR[0]                   ; пересылка значения 200 в WR0
U 0C2F, BE88,15     ;9493           MOV WR[0] TO LS[ADDRESS.DATA]           ; загрузка 200 для распечатки в качестве адреса под
                    ;9494                                                   ; OTHER в сообщении об ошибке .
U 0C30, 6512,15     ;9495           CLR LS[T9]                              ; очистка ячейки хранения адреса памяти отображения об-
                    ;9496                                                   ; щей шины
U 0C31, 3651,95     ;9497           MOV LS[BIT8] TO WR[3]                   ; установка бита 8 в WR3. Сдвиг WR3 для формирования
                    ;9498                                                   ; первого сдвигаемого адреса (1(H))
U 0C32, B67F,15     ;9499           MOV LS[BIT31] TO WR[2]                  ; установка бита 31 в WR2. Сдвиг WR2 для формирования
                    ;9500                                                   ; первого сдвигаемого адреса (201(H)) для распечатки при
                    ;9501                                                   ; ошибке (после установки бита 9)
                    ;9502       LOOP.T12.9A:
U 0C33, 2F82,95     ;9503           CLR WR[1]                               ; ожидаемые данные
U 0C34, 1C13,75     ;9504           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0C35, 3022,15     ;9505           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0C36, 0869,3C     ;9506           JSR [CHECK.RESULT]                      ; проверка результата
U 0C37, 88C3,34     ;9507           JMP [LOOP.T12.9A]                       ; зацикливание при ошибке, если разрешено
                    ;9508       LOOP.T12.9B:
U 0C38, 369E,95     ;9509           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0C39, 1B13,F5     ;9510           MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи в память отображения общей шины
U 0C3A, 329E,15     ;9511           WRITE.MEM LS[ONES]                      ; запись единиц в память отображения общей шины
U 0C3B, 1C13,75     ;9512           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения единиц из памяти отображения общей
                    ;9513                                                   ; шины
U 0C3C, 3022,15     ;9514           MOV MEM.DATA TO WR[0]                   ; чтение результата
U 0C3D, 0869,3C     ;9515           JSR [CHECK.RESULT]                      ; проверка результата
U 0C3E, 08C3,84     ;9516           JMP [LOOP.T12.9B]                       ; зацикливание при ошибке, если разрешено
U 0C3F, A3C1,95     ;9517           ROL WR[3]                               ; сдвиг адреса для пересылки 1 в следующий бит
U 0C40, BE13,95     ;9518           MOV WR[3] TO LS[T9]                     ; запоминание адреса
U 0C41, 23C1,15     ;9519           ROL WR[2]                               ; адрес информации для распечатки при ошибке
U 0C42, 3E89,15     ;9520           MOV WR[2] TO LS[ADDRESS.DATA]           ; запоминание в местной памяти
```

```
U 0C43, B652,15 ;9521            MOV LS[#200] TO WR[0]                    ; подготовка установки бита 9 в печатаемом адресе
U 0C44, ED88,15 ;9522            BIS WR[0] TO LS[ADDRESS.DATA]            ; сейчас адрес памяти буфера трансляции правильный
                ;9523              BIT LS[BIT18] WITH WR[3],              ; проверка, проверены ли биты 8-0 (если последний тест
                ;9524                                                     ; был с установленным битом 17, то после сдвига бит 18
                ;9525                                                     ; будет установлен)
U 0C45, 5965,B5 ;9526            DT(LONG)&SET.ALU.CC                      ; установка кодов условий
U 0C46, 08C3,39 ;9527            JMP.IF[BITS.CLR] TO [LOOP.T12.9A]        ; повторение теста со следующим сдвинутым адресом, если
                ;9528                                                     ; бит 18 не установлен
                ;9529    TEST.T12.A:
U 0C47, E580,15 ;9530            CLR LS[CONTROL.STATUS]                   ; запрет печати под OTHER
U 0C48, FF82,15 ;9531            INC LS[ERROR.NUMBER]                     ; ошибка A
U 0C49, 3682,15 ;9532            MOV LS[ERROR.NUMBER] TO WR[0]            ; сохранение номера ошибки
U 0C4A, BE14,15 ;9533            MOV WR[0] TO LS[T10]                     ; в ячейке временного хранения местной памяти
U 0C4B, 3653,15 ;9534            MOV LS[BIT9] TO WR[2]                    ; подготовка увеличения (увеличивается бит 0 адреса
                ;9535                                                     ; микросхемы памяти)
U 0C4C, B653,95 ;9536            MOV LS[BIT9] TO WR[3]                    ; подготовка бита для переключения с максимальной
                ;9537                                                     ; скоростью. Первым переключаемым битом является бит 0
                ;9538                                                     ; памяти
U 0C4D, 6512,15 ;9539            CLR LS[T9]                               ; очистка ячейки местной памяти, содержащей адрес
                ;9540    BACK.T12.A:
U 0C4E, 1B13,F5 ;9541            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи фона в память отображения общей
                ;9542                                                     ; шины
U 0C4F, B29C,15 ;9543            WRITE.MEM LS[ZERO]                       ; пересылка нулей в память отображения общей шины
U 0C50, BA18,CC ;9544            JSR [NEXT.UB.ADDRESS]                    ; вычисление следующего адреса памяти отображения общей
                ;9545                                                     ; шины
U 0C51, B8C4,E4 ;9546            JMP [BACK.T12.A]                         ; возврат, если еще не все выполнено
                ;9547    REPEAT.T12.A.1:
U 0C52, 6512,15 ;9548            CLR LS[T9]                               ; запуск от адреса 200 памяти отображения общей шины
U 0C53, 3614,15 ;9549            MOV LS[T10] TO WR[0]                     ; выборка номера ошибки
U 0C54, BE82,15 ;9550            MOV WR[0] TO LS[ERROR.NUMBER]            ; пересылка в местную память для распечатки
                ;9551    LOOP.T12.A.1:
U 0C55, 2F82,95 ;9552            CLR WR[1]                                ; ожидаемые данные
U 0C56, 1C13,75 ;9553            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0C57, 3022,15 ;9554            MOV MEM.DATA TO WR[0]                    ; выборка результата
U 0C58, 0869,3C ;9555            JSR [CHECK.RESULT]                       ; проверка результата
U 0C59, B8C5,54 ;9556            JMP [LOOP.T12.A.1]                       ; зацикливание при ошибке, если разрешено
U 0C5A, 1B13,F5 ;9557            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи дополнения в память отображения
                ;9558                                                     ; общей шины
U 0C5B, 329E,15 ;9559            WRITE.MEM LS[ONES]                       ; запись единиц в эту ячейку
U 0C5C, 6C13,95 ;9560            ADD WR[3] TO LS[T9]                      ; переключение бита в адресе
U 0C5D, B664,15 ;9561            MOV LS[BIT18] TO WR[0]                   ; подготовка проверки, установлен ли бит 18
                ;9562              BIT WR[0] WITH LS[T9],                 ; проверка бита 18
U 0C5E, D912,35 ;9563            DT(LONG)&SET.ALU.CC                      ; установка кодов условий
U 0C5F, 08C5,59 ;9564            JMP.IF[BITS.CLR] TO [LOOP.T12.A.1]       ; если бит 18 не установлен, повторение со следующим
                ;9565                                                     ; адресом
U 0C60, EC13,15 ;9566            ADD WR[2] TO LS[T9]                      ; иначе увеличение адреса ниже переключаемого бита
U 0C61, B664,15 ;9567            MOV LS[BIT18] TO WR[0]                   ; подготовка очистки бита 18 в ячейке 9 местной памяти
U 0C62, 6F12,15 ;9568            XOR WR[0] TO LS[T9]                      ; сброс бита 18 в адресе
                ;9569              BIT WR[3] WITH LS[T9],                 ; если переключаемый бит сейчас установен, выполнено
U 0C63, D913,B5 ;9570            DT(LONG)&SET.ALU.CC                      ; установка кодов условий
U 0C64, 08C5,59 ;9571            JMP.IF[BITS.CLR] TO [LOOP.T12.A.1]       ; повторение теста с новым адресом, если не выполнено
                ;9572                                                     ; иначе подготовка запуска уменьшения адреса
U 0C65, 6512,15 ;9573            CLR LS[T9]                               ; запуск от адреса 200 памяти отображения общей шины
                ;9574                                                     ; (будет выполнено дополнение для запуска от верхнего
                ;9575                                                     ; адреса)
```

```
U 0C66, FF82,15 ;9576           INC LS[ERROR.NUMBER]                    ; ошибка B
                ;9577    REPEAT.T12.B.1:
U 0C67, 7D12,15 ;9578           COM LS[T9]                              ; дополнение адреса для получения уменьшающихся адресов
                ;9579    LOOP.T12.B.1:
U 0C68, 369E,95 ;9580           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0C69, 1C13,75 ;9581           MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения памяти отображения общей шины
U 0C6A, 3022,15 ;9582           MOV MEM.DATA TO WR[0]                   ; выборка результата
U 0C6B, 0869,3C ;9583           JSR [CHECK.RESULT]                      ; проверка результата
U 0C6C, 08C6,84 ;9584           JMP [LOOP.T12.B.1]                      ; зацикливание при ошибке, если разрешено
U 0C6D, 1B13,F5 ;9585           MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи дополнения в память отображения
                ;9586                                                   ; общей шины
U 0C6E, B29C,15 ;9587           WRITE.MEM LS[ZERO]                      ; запись нулей в эту ячейку
U 0C6F, 7D12,15 ;9588           COM LS[T9]                              ; восстановление адреса для получения оригинала
U 0C70, 6C13,95 ;9589           ADD WR[3] TO LS[T9]                     ; переключение бита в адресе
U 0C71, B664,15 ;9590           MOV LS[BIT18] TO WR[0]                  ; подготовка проверки, установлен ли бит 18
                ;9591             BIT WR[0] WITH LS[T9],                ; проверка, установлен ли бит 18 и установка кодов
U 0C72, D912,35 ;9592           DT(LONG)&SET.ALU.CC                     ; условий
U 0C73, 88C6,79 ;9593           JMP.IF[BITS.CLR] TO [REPEAT.T12.B.1]    ; если бит 18 не установлен, повторение со следующим
                ;9594                                                   ; адресом
U 0C74, EC13,15 ;9595           ADD WR[2] TO LS[T9]                     ; иначе увеличение адреса ниже переключаемого бита
U 0C75, B664,15 ;9596           MOV LS[BIT18] TO WR[0]                  ; подготовка сброса бита 18 в ячейке 9 местной памяти
U 0C76, 6F12,15 ;9597           XOR WR[0] TO LS[T9]                     ; сброс бита 18 в адресе
                ;9598             BIT WR[3] WITH LS[9],                 ; если переключаемый бит установлен, выполнено,
U 0C77, D913,B5 ;9599           DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0C78, 88C6,79 ;9600           JMP.IF[BITS.CLR] TO [REPEAT.T12.B.1]    ; повторение теста с новым адресом, если не выполнено
                ;9601                                                   ; иначе изменение переключаемого бита
U 0C79, A3C1,95 ;9602           ROL WR[3]                               ; следующий бит для переключения с максимальной
                ;9603                                                   ; скоростью
                ;9604             BIT LS[BIT18] WITH WR[3],             ; проверка, установлен ли бит 18 и установка кодов
U 0C7A, 5965,B5 ;9605           DT(LONG)&SET.ALU.CC                     ; условий
U 0C7B, 88C5,29 ;9606           JMP.IF[BITS.CLR] TO [REPEAT.T12.A.1]    ; повторение теста с новым переключенным битом, если бит
                ;9607                                                   ; 18 не установлен, иначе выполнено
                ;9608    END.T12:
```

```
;9609    .PAGE "ТЕСТ 13 - тест последовательной записи в две ячейки буфера трансляции (модуль MCT)"
;9610    ;
;9611    ; ОПИСАНИЕ ТЕСТА:
;9612    ;
;9613    ;       Этот тест проверяет микропрограмму последовательной записи в две ячейки
;9614    ;   буфера трансляции. Эта функция использует те же схемы, как и запись в буфер
;9615    ;   трансляции, но содержит незначительные различия в микропрограмме. Этот тест,
;9616    ;   в основном, проверяет ПЗУ управляющей памяти, так как никакие новые схемы не
;9617    ;   будут проверяться.
;9618    ;       Пути данных для этой функции буфера трансляции почти идентичны путям дан-
;9619    ;   ных микропрограммы записи в буфер трансляции. Адрес начального ветвления от-
;9620    ;   личается от микропрограммы записи буфера трансляции в том, что устанавливает
;9621    ;   DATIP вместо DATI. Этим устанавливается IC0, так что в конце микропрограммы
;9622    ;   выполняется переход. По этому переходу увеличивается регистр виртуального ад-
;9623    ;   реса и разрешение записи буфера трансляции снимается. Остальная часть управ-
;9624    ;   ляющего слова повторяет события цикла, который выполняет переход по IC0.
;9625    ;   Следующий цикл устанавливает DATI, так что IC0 сбрасывается, и повторяет пре-
;9626    ;   дыдущие события, за исключением увеличения регистра виртуального адреса. Мик-
;9627    ;   ропрограмма тогда возвращается в обычную микропрограмму записи в буфер транс-
;9628    ;   ляции в цикл, непосредственно предшествующий ветвлению по IC0. Таким образом,
;9629    ;   опять происходит запись в буфер трансляции по новому адресу.
;9630    ;
;9631    ; ПРЕДПОЛОЖЕНИЯ:
;9632    ;
;9633    ;   Предполагается, что все предыдущие тесты выполнены успешно.
;9634    ;
;9635    ; ШАГИ ТЕСТА:
;9636    ;
;9637    ;   1) Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;9638    ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной
;9639    ;      памяти. Маска ошибки разрешает проверку битов с 1 по 22.
;9640    ;   2) Загрузка ячейки местной памяти единицами в 9 младших битах и нулями во
;9641    ;      всех других битах.
;9642    ;   3) Загрузка всех единиц в WR1 и выполнение инструкции MEM.REQ с полем MF=
;9643    ;      WRITE.TB и DT=LONGWORD, используя ячейку местной памяти, загруженную в
;9644    ;      шаге 2.
;9645    ;   4) Выполнение инструкции WRITE.MEM LS с ячейкой местной памяти, содержащей
;9646    ;      все единицы.
;9647    ;   5) Увеличение ячейки местной памяти для записи в следующий адрес буфера
;9648    ;      трансляции и повторение шагов 3 и 4.
;9649    ;   6) Повторение шага 2 и выполнение инструкции MEM.REQ с MF=READ.TB и DT=
;9650    ;      LONGWORD, используя ячейку местной памяти, загруженную в шаге 2.
;9651    ;   7) Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата на все
;9652    ;      единицы. Повторение шагов 6 и 7 для проверки следующей ячейки.
;9653    ;   8) Загрузка нулей в WR1, повторение шага 2 и выполнение инструкции MEM.REQ
;9654    ;      с MF=WRITE.TB.STEP, DT=LONGWORD.
;9655    ;   9) Выполнение инструкции WRITE.MEM LS с ячейкой местной памяти, содержащей
;9656    ;      все нули.
;9657    ;  10) Повторение шагов 6 и 7 для уверенности, что была запись в 2 ячейки
;9658    ;      буфера трансляции.
;9659    ;
;9660    ; ОШИБКИ:
;9661    ;
;9662    ;   ошибка 1 - неправильно работает запись или чтение буфера трансляции (эта
;9663    ;              ошибка не должна появляться).
```

```
                ;9664   ;    ошибка 2 - неправильно работает последовательная запись в 2 ячейки буфера
                ;9665   ;               трансляции. Данными под OTHER является адрес буфера трансляции,
                ;9666   ;               запись по которому была неуспешной.
                ;9667   ;
                ;9668   ; НАЛАДКА:
                ;9669   ;
                ;9670   ;
                ;9671   ;       ОШИБКА 1 - Эта ошибка указывает на возможную неисправность при основной
                ;9672   ;   записи или чтении буфера трансляции, что было проверено предыдущим тестом.
                ;9673   ;   Необходимо пропустить тест памяти буфера трансляции заново.
                ;9674   ;
                ;9675   ;       ОШИБКА 2 - Эта ошибка указывает на возможную неисправность микрокодов в
                ;9676   ;   ПЗУ управляющей памяти. Все схемы, используемые в этом тесте, были проверены
                ;9677   ;   предыдущим тестом.
                ;9678   ;
                ;9679
                ;9680   T.13:
U 0C7C, B65E,15 ;9681           MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;9682                                                   ; состояния
U 0C7D, 3E80,15 ;9683           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;9684                                                   ; 15 указывает конс.процессору начало теста
U 0C7E, 10E0,15 ;9685           MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN для конс.процессора
                ;9686   WAIT.T13.0:
U 0C7F, 0BC7,F4 ;9687           JMP [WAIT.T13.0]                        ; зацикливание для ожидания ответа консольного
                ;9688                                                   ; процессора
U 0C80, 0A1A,AC ;9689           JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0C81, B62A,15 ;9690           MOV LS[#FF000000] TO WR[0]              ; установка старшего байта маски ошибки
U 0C82, C76E,15 ;9691           BIS LS[BIT23] TO WR[0]                  ; также установка бита 23
U 0C83, C740,15 ;9692           BIS LS[BIT0] TO WR[0]                   ; и бита 0
U 0C84, 3E8A,15 ;9693           MOV WR[0] TO LS[ERROR.MASK]             ; будут проверяться биты с 1 по 22
                ;9694   LOOP.T13.1:
U 0C85, 3627,15 ;9695           MOV LS[#FF] TO WR[2]                    ; установка в рабочем регистре битов 0-7
U 0C86, C751,15 ;9696           BIS LS[BIT8] TO WR[2]                   ; сейчас биты 0-8 установлены
U 0C87, 3E13,15 ;9697           MOV WR[2] TO LS[T9]                     ; запоминание этих данных в качестве адреса буфера
                ;9698                                                   ; трансляции в ячейке 9 местной памяти
U 0C88, 9812,F5 ;9699           MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи данных в буфер трансляции по адресу 0
U 0C89, 329E,15 ;9700           WRITE.MEM LS[ONES]                      ; пересылка единиц в буфер трансляции по адресу 0
U 0C8A, FF12,15 ;9701           INC LS[T9]                              ; сейчас адрес 1 буфера трансляции в ячейке 9 местной
                ;9702                                                   ; памяти
U 0C8B, 9812,F5 ;9703           MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи данных в буфер трансляции по адресу 1
U 0C8C, 329E,15 ;9704           WRITE.MEM LS[ONES]                      ; пересылка единиц в буфер трансляции по адресу 1
U 0C8D, 369E,95 ;9705           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0C8E, 3E13,15 ;9706           MOV WR[2] TO LS[T9]                     ; адрес 0 буфера трансляции
U 0C8F, 9C13,F5 ;9707           MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка проверки данных буфера трансляции
U 0C90, 3022,15 ;9708           MOV MEM.DATA TO WR[0]                   ; выборка данных буфера трансляции по адресу 0
U 0C91, 0869,3C ;9709           JSR [CHECK.RESULT]                      ; проверка на единицы
U 0C92, 08C8,54 ;9710           JMP [LOOP.T13.1]                        ; зацикливание при ошибке, если разрешено
U 0C93, 369E,95 ;9711           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 0C94, FF12,15 ;9712           INC LS[9]                               ; адрес 1 буфера трансляции
U 0C95, 9C13,F5 ;9713           MEM.REQ[READ.TB] ADRS[T9] DT[LONG]      ; подготовка проверки данных буфера трансляции
U 0C96, 3022,15 ;9714           MOV MEM.DATA TO WR[0]                   ; выборка данных буфера трансляции по адресу 1
U 0C97, 0869,3C ;9715           JSR [CHECK.RESULT]                      ; проверка на единицы
U 0C98, 08C8,54 ;9716           JMP [LOOP.T13.1]                        ; зацикливание при ошибке, если разрешено
U 0C99, FF82,15 ;9717           INC LS[ERROR.NUMBER]                    ; ошибка 2
U 0C9A, B670,15 ;9718           MOV LS[OTHER.DATA] TO WR[0]             ; подготовка адреса для распечатки под OTHER
```

```
U 0C9B, 3E80,15 ;9719            MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 24 в слове управления. Конс.процессор
                ;9720                                                     ; будет выполнять печать под OTHER при ошибке
                ;9721    LOOP.T13.2:
U 0C9C, 3E13,15 ;9722            MOV WR[2] TO LS[T9]                      ; подготовка адреса 0 буфера трансляции
U 0C9D, 6588,15 ;9723            CLR LS[ADDRESS.DATA]                     ; адрес для распечатки при ошибке
U 0C9E, 2F82,95 ;9724            CLR WR[1]                                ; ожидаемые данные
U 0C9F, 1C12,F5 ;9725            MEM.REQ[WRITE.TB.STEP] ADRS[T9] DT[LONG] ; подготовка записи в 2 последовательные ячейки буфера
                ;9726                                                     ; трансляции
U 0CA0, B29C,15 ;9727            WRITE.MEM LS[ZERO]                       ; запись нулей в буфер трансляции по адресам 0 и 1
U 0CA1, 9C13,F5 ;9728            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]       ; подготовка проверки буфера трансляции по адресу 0
U 0CA2, 3022,15 ;9729            MOV MEM.DATA TO WR[0]                    ; выборка данных
U 0CA3, 0869,3C ;9730            JSR [CHECK.RESULT]                       ; проверка на нули
U 0CA4, 88C9,C4 ;9731            JMP [LOOP.T13.2]                         ; зацикливание при ошибке, если разрешено
U 0CA5, FF12,15 ;9732            INC LS[T9]                               ; установка адреса 1
U 0CA6, FF88,15 ;9733            INC LS[ADDRESS.DATA]                     ; адрес для распечатки при ошибке (1)
U 0CA7, 2F82,95 ;9734            CLR WR[1]                                ; ожидаемые данные
U 0CA8, 9C13,F5 ;9735            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]       ; подготовка проверки буфера трансляции по адресу 1
U 0CA9, 3022,15 ;9736            MOV MEM.DATA TO WR[0]                    ; выборка данных
U 0CAA, 0869,3C ;9737            JSR [CHECK.RESULT]                       ; проверка на нули
U 0CAB, 88C9,C4 ;9738            JMP [LOOP.T13.2]                         ; зацикливание при ошибке, если разрешено
                ;9739    END.T13:
```

```
;9740    .PAGE "ТЕСТ 14 - тест *МАРШ* для всей памяти ТВ и адресной линии ТВА 09 (модуль МСТ)"
;9741    ;
;9742    ; ОПИСАНИЕ ТЕСТА:
;9743    ;
;9744    ;       Тест адресной линии ТВА 09 проверяет, что ТВА 09 имеет высокий уровень при
;9745    ;    записи в область отображения общей шины, низкий - при записи в область транс-
;9746    ;    ляции адресов центрального процессора.
;9747    ;       Предыдущие тесты проверили только, что этот бит не изменяется на протяже-
;9748    ;    нии всего теста. Если этот бит формируется неправильно, запись из общей шины
;9749    ;    может произойти в область, предназначенную для трансляции адресов центрального
;9750    ;    процессора или наоборот. Тест "МАРШ" будет выполняться для всех адресов памяти
;9751    ;    (т.е., для части памяти, выполняющей трансляцию адресов общей шины и для части
;9752    ;    трансляции адресов центрального процессора) с целью проверки любой неоднознач-
;9753    ;    ной адресации.
;9754    ;       Путь данных теста ТВА 09 является обычным для микропрограмм WRITE.TB,
;9755    ;    WRITE.UBS.MAP и соответствующих им микропрограмм чтения.
;9756    ;
;9757    ; ПРЕДПОЛОЖЕНИЯ:
;9758    ;
;9759    ;    Предполагается, что все предыдущие тесты выполнены успешно.
;9760    ;
;9761    ; ШАГИ ТЕСТА:
;9762    ;
;9763    ;    1) Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;9764    ;       (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;9765    ;       мяти. Маска ошибки разрешает проверку битов с 1 по 22.
;9766    ;    2) Запись в память фона нулей с заполнением всех адресов буфера трансляции и
;9767    ;       всех адресов памяти отображения общей шины.
;9768    ;    3) Чтение нулей, запись единиц, чтение единиц каждой ячейки буфера трансляции,
;9769    ;       пока не будут выбраны все 128 адресов.
;9770    ;    4) Продолжение чтения/записи/чтения тестовых данных в части памяти для отобра-
;9771    ;       жения адресов общей шины по возрастающим адресам, пока не будут выбраны все
;9772    ;       512 ячеек памяти отображения общей шины.
;9773    ;    5) Начиная от вершины памяти отображения общей шины, читаются единицы, запи-
;9774    ;       сываются нули, читаются нули по убывающим адресам, пока не будут выбраны
;9775    ;       все 512 ячеек.
;9776    ;    6) Продолжение чтения/записи/чтения тестовых данных в части памяти буфера транс-
;9777    ;       ляции адресов центрального процессора, пока не будут выбраны все 128 ячеек.
;9778    ;
;9779    ; ОШИБКИ:
;9780    ;
;9781    ;    ПРИМЕЧАНИЕ: Данными под OTHER является адрес буфера трансляции или памяти
;9782    ;    отображения общей шины.
;9783    ;
;9784    ;    ошибка 1 - неправильно работает "МАРШ" в области памяти буфера трансляции
;9785    ;               центрального процессора при возрастающих адресах.
;9786    ;    ошибка 2 - неправильно работает "МАРШ" в части памяти отображения адресов
;9787    ;               общей шины при возрастающих адресах.
;9788    ;    ошибка 3 - неправильно работает "МАРШ" в части памяти отображения адресов
;9789    ;               общей шины при убывающих адресах.
;9790    ;    ошибка 4 - неправильно работает "МАРШ" в части памяти буфера трансляции
;9791    ;               центрального процессора при убывающих адресах.
;9792    ;
;9793    ; НАЛАДКА:
;9794    ;
```

```
                ;9795   ;       ОШИБКИ С 1 ПО 4 - Если тест работает неправильно, прежде всего необходимо
                ;9796   ;    проверить адресную линию ТВА 09. При этой неисправности или все биты непра-
                ;9797   ;    вильные или неправильна группа из 4 битов. Это можно проверить при зацикли-
                ;9798   ;    вании по ошибке. Эта линия должна иметь высокий уровень при обращении в па-
                ;9799   ;    мять отображения общей шины и низкий уровень при обращении в буфер трансля-
                ;9800   ;    ции центрального процессора. Если неправильна группа из 4 битов, необходимо
                ;9801   ;    проверить ту линию на микросхеме, выходы которой неправильные. Если здесь
                ;9802   ;    неправильно, необходимо проверить +3VB и землю, подсоединенные к той части
                ;9803   ;    мультиплексора, которая формирует ТВА 09 Н. Если эти соединения правильные,
                ;9804   ;    по-видимому, неисправен мультиплексор.
                ;9805   ;       Другой возможностью является неисправность внутренних схем адреса в микро-
                ;9806   ;    схеме памяти. Необходимо заменить микросхему памяти, которая выдает непра-
                ;9807   ;    вильный бит.
                ;9808   ;
                ;9809    T.14:
U 0CAC, B65E,15 ;9810            MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR[0] для слова управления и
                ;9811                                                       ; состояния
U 0CAD, 3E80,15 ;9812            MOV WR[0] TO LS[CONTROL.STATUS]            ; установка бита 15 в слове управления и состояния. Бит
                ;9813                                                       ; 15 указывает конс.процессору начало теста
U 0CAE, 10E0,15 ;9814            MISC [SET.CP.ATTN]                         ; выдача сигнала CPU ATTN конс.процессору
                ;9815    WAIT.T14.0:
U 0CAF, 88CA,F4 ;9816            JMP [WAIT.T14.0]                           ; зацикливание для ожидания ответа от конс.процессора
U 0CB0, 0A1A,AC ;9817            JSR [SETUP.1]                              ; установка масок, кода модуля и др.
U 0CB1, B640,15 ;9818            MOV LS[#1] TO WR[0]                        ; установка 1 в WR0
U 0CB2, BEFC,15 ;9819            MOV WR[0] TO LS[SIZE]                      ; подготовка регистра размера для типа данных = слово
U 0CB3, B62A,15 ;9820            MOV LS[#FF000000] TO WR[0]                 ; установка старшего байта маски ошибки
U 0CB4, C76E,15 ;9821            BIS LS[BIT23] TO WR[0]                     ; также установка бита 23
U 0CB5, C740,15 ;9822            BIS LS[BIT0] TO WR[0]                      ; и бита 0
U 0CB6, 3E8A,15 ;9823            MOV WR[0] TO LS[ERROR.MASK]                ; будут проверяться биты с 1 по 22
U 0CB7, B670,15 ;9824            MOV LS[OTHER.DATA] TO WR[0]                ; подготовка для распечатки адреса под OTHER
U 0CB8, 3E80,15 ;9825            MOV WR[0] TO LS[CONTROL.STATUS]            ; установка бита 24 в слове управления для печати под
                ;9826                                                       ; OTHER при ошибке
U 0CB9, 6512,15 ;9827            CLR LS[T9]                                 ; запуск от адреса 0
                ;9828    BACK.T14.1:
U 0CBA, 9812,F5 ;9829            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]        ; подготовка записи в области буфера трансляции
                ;9830                                                       ; центрального процессора
U 0CBB, B29C,15 ;9831            WRITE.MEM LS[ZERO]                         ; запись нулей в буфер трансляции
U 0CBC, 0A18,2C ;9832            JSR [NEXT.TB.ADDRESS]                      ; выборка следующего адреса буфера трансляции
U 0CBD, 08CB,A4 ;9833            JMP [BACK.T14.1]                           ; повторение, если возврат сюда (еще не выполнено),
                ;9834                                                       ; иначе запись фона в область отображения адресов общей
                ;9835                                                       ; шины
U 0CBE, 6512,15 ;9836            CLR LS[T9]                                 ; запуск от адреса 200 (бит 9 устанавливается
                ;9837                                                       ; аппаратно)
                ;9838    BACK.T14.1A:
U 0CBF, 1813,F5 ;9839            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG]   ; подготовка записи в область отображения адресов
                ;9840                                                       ; общей шины
U 0CC0, B29C,15 ;9841            WRITE.MEM LS[ZERO]                         ; запись нулей по адресу области общей шины
U 0CC1, 8A18,CC ;9842            JSR [NEXT.UB.ADDRESS]                      ; выборка следующего адреса области общей шины
U 0CC2, 08CB,F4 ;9843            JMP [BACK.T14.1A]                          ; повторение, если возврат сюда (еще не выполнено)
U 0CC3, 6512,15 ;9844            CLR LS[T9]                                 ; запуск от адреса 0
U 0CC4, 6588,15 ;9845            CLR LS[ADDRESS.DATA]                       ; адрес для печати при ошибке
                ;9846    LOOP.T14.1:
U 0CC5, 2F82,95 ;9847            CLR WR[1]                                  ; ожидаемые данные
U 0CC6, 9C13,F5 ;9848            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]         ; подготовка чтения из области буфера трансляции
                ;9849                                                       ; центрального процессора
```

```
U 0CC7, 3022,15 ;9850            MOV MEM.DATA TO WR[0]                    ; выборка результата из буфера трансляции
U 0CC8, 0869,3C ;9851            JSR [CHECK.RESULT]                       ; проверка результата на нуль
U 0CC9, 88CC,54 ;9852            JMP [LOOP.T14.1]                         ; зацикливание при ошибке, если разрешено
U 0CCA, 369E,95 ;9853            MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 0CCB, 9812,F5 ;9854            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]      ; подготовка записи единиц
U 0CCC, 329E,15 ;9855            WRITE.MEM LS[ONES]                       ; запись единиц в буфер трансляции
U 0CCD, 9C13,F5 ;9856            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]       ; подготовка чтения результата
U 0CCE, 3022,15 ;9857            MOV MEM.DATA TO WR[0]                    ; чтение данных буфера трансляции
U 0CCF, 0869,3C ;9858            JSR [CHECK.RESULT]                       ; проверка результата на единицы
U 0CD0, 88CC,54 ;9859            JMP [LOOP.T14.1]                         ; зацикливание при ошибке, если разрешено
U 0CD1, 0A18,2C ;9860            JSR [NEXT.TB.ADDRESS]                    ; увеличение адреса
U 0CD2, 88CC,54 ;9861            JMP [LOOP.T14.1]                         ; повторение, если не выполнено, иначе подпрограмма
                ;9862                                                     ; выполнит возврат к следующему адресу (проверка области
                ;9863                                                     ; шины)
U 0CD3, FF82,15 ;9864            INC LS[ERROR.NUMBER]                     ; ошибка 2
U 0CD4, 6512,15 ;9865            CLR LS[T9]                               ; запуск от адреса 0 (аппаратно устанавливается бит 9
                ;9866                                                     ; для адреса 200 памяти)
U 0CD5, B652,15 ;9867            MOV LS[#200] TO WR[0]                    ; подготовка адреса для печати
U 0CD6, BE88,15 ;9868            MOV WR[0] TO LS[ADDRESS.DATA]            ; адрес для печати при ошибке (200 является первым
                ;9869                                                     ; адресом области общей шины)
                ;9870    LOOP.T14.2:
U 0CD7, 2F82,95 ;9871            CLR WR[1]                                ; ожидаемые данные
U 0CD8, 1C13,75 ;9872            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения из области общей шины
U 0CD9, 3022,15 ;9873            MOV MEM.DATA TO WR[0]                    ; выборка результата из области отображения общей шины
U 0CDA, 0869,3C ;9874            JSR [CHECK.RESULT]                       ; проверка результата на нуль
U 0CDB, 88CD,74 ;9875            JMP [LOOP.T14.2]                         ; зацикливание при ошибке, если разрешено
U 0CDC, 369E,95 ;9876            MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 0CDD, 1B13,F5 ;9877            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи единиц
U 0CDE, 329E,15 ;9878            WRITE.MEM LS[ONES]                       ; запись единиц в область отображения общей шины
U 0CDF, 1C13,75 ;9879            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения результата
U 0CE0, 3022,15 ;9880            MOV MEM.DATA TO WR[0]                    ; чтение данных из области отображения общей шины
U 0CE1, 0869,3C ;9881            JSR [CHECK.RESULT]                       ; проверка результата на единицы
U 0CE2, 88CD,74 ;9882            JMP [LOOP.T14.2]                         ; зацикливание при ошибке, если разрешено
U 0CE3, 8A18,CC ;9883            JSR [NEXT.UB.ADDRESS]                    ; увеличение адреса
U 0CE4, 88CD,74 ;9884            JMP [LOOP.T14.2]                         ; повторение, если не выполнено, иначе подпрограмма
                ;9885                                                     ; выполнит возврат к следующему адресу
U 0CE5, FF82,15 ;9886            INC LS[ERROR.NUMBER]                     ; ошибка 3
U 0CE6, 6512,15 ;9887            CLR LS[T9]                               ; запуск от адреса 0 (аппаратно устанавливается бит 9
                ;9888                                                     ; для адреса памяти 200)
U 0CE7, B654,15 ;9889            MOV LS[#400] TO WR[0]                    ; подготовка адреса для печати
U 0CE8, 2100,15 ;9890            DEC WR[0]                                ; сейчас подготовлен адрес 3FF(H)
U 0CE9, BE88,15 ;9891            MOV WR[0] TO LS[ADDRESS.DATA]            ; адрес для печати при ошибке
                ;9892    REPEAT.T14.3:
U 0CEA, 7D12,15 ;9893            COM LS[T9]                               ; сейчас уменьшение адреса памяти в области отображения
                ;9894                                                     ; общей шины
                ;9895    LOOP.T14.3:
U 0CEB, 369E,95 ;9896            MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 0CEC, 1C13,75 ;9897            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения из области отображения общей шины
U 0CED, 3022,15 ;9898            MOV MEM.DATA TO WR[0]                    ; выборка результата из области отображения общей шины
U 0CEE, 0869,3C ;9899            JSR [CHECK.RESULT]                       ; проверка результата на нуль
U 0CEF, 88CE,B4 ;9900            JMP [LOOP.T14.3]                         ; зацикливание при ошибке, если разрешено
U 0CF0, 2F82,95 ;9901            CLR WR[1]                                ; ожидаемые данные
U 0CF1, 1B13,F5 ;9902            MEM.REQ[WRITE.UBS.MAP] ADRS[T9] DT[LONG] ; подготовка записи нулей
U 0CF2, B29C,15 ;9903            WRITE.MEM LS[ZERO]                       ; запись нулей в области отображения общей шины
U 0CF3, 1C13,75 ;9904            MEM.REQ[READ.UBS.MAP] ADRS[T9] DT[LONG]; подготовка чтения результата
```

```
U 0CF4, 3022,15 ;9905            MOV MEM.DATA TO WR[0]                  ; чтение данных из области отображения общей шины
U 0CF5, 0869,3C ;9906            JSR [CHECK.RESULT]                     ; проверка результата на единицы
U 0CF6, 88CE,B4 ;9907            JMP [LOOP.T14.3]                       ; зацикливание при ошибке, если разрешено
U 0CF7, 7D12,15 ;9908            COM LS[T9]                             ; восстановление адреса
U 0CF8, FC88,15 ;9909            DEC LS[ADDRESS.DATA]                   ; уменьшение адреса для печати
U 0CF9, FC88,15 ;9910            DEC LS[ADDRESS.DATA]                   ; дважды, так как подпрограмма увеличивает его
U 0CFA, 8A18,CC ;9911            JSR [NEXT.UB.ADDRESS]                  ; увеличение адреса
U 0CFB, 08CE,A4 ;9912            JMP [REPEAT.T14.3]                     ; повторение, если не выполнено, иначе подпрограмма
                ;9913                                                   ; выполнит возврат к следующему адресу
U 0CFC, FF82,15 ;9914            INC LS[ERROR.NUMBER]                   ; ошибка 4
U 0CFD, 6512,15 ;9915            CLR LS[T9]                             ; запуск от адреса 0
U 0CFE, 364E,15 ;9916            MOV LS[#80] TO WR[0]                   ; подготовка адреса для печати
U 0CFF, 2100,15 ;9917            DEC WR[0]                              ; сейчас имеется верхний адрес (7F) буфера трансляции
                ;9918                                                   ; центрального процессора
U 0D00, BE88,15 ;9919            MOV WR[0] TO LS[ADDRESS.DATA]          ; адрес для печати при ошибке
                ;9920    REPEAT.T14.4:
U 0D01, 7D12,15 ;9921            COM LS[T9]                             ; сейчас уменьшение адреса памяти буфера трансляции
                ;9922    LOOP.T14.4:
U 0D02, 369E,95 ;9923            MOV LS[ONES] TO WR[1]                  ; ожидаемые данные
U 0D03, 9C13,F5 ;9924            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]     ; подготовка чтения из буфера трансляции
U 0D04, 3022,15 ;9925            MOV MEM.DATA TO WR[0]                  ; выборка результата из буфера трансляции
U 0D05, 0869,3C ;9926            JSR [CHECK.RESULT]                     ; проверка результата на единицы
U 0D06, 88D0,24 ;9927            JMP [LOOP.T14.4]                       ; зацикливание при ошибке, если разрешено
U 0D07, 2F82,95 ;9928            CLR WR[1]                              ; ожидаемые данные
U 0D08, 9812,F5 ;9929            MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]    ; подготовка записи нулей
U 0D09, B29C,15 ;9930            WRITE.MEM LS[ZERO]                     ; запись нулей в буфер трансляции
U 0D0A, 9C13,F5 ;9931            MEM.REQ[READ.TB] ADRS[T9] DT[LONG]     ; подготовка чтения результата
U 0D0B, 3022,15 ;9932            MOV MEM.DATA TO WR[0]                  ; чтение данных из буфера трансляции
U 0D0C, 0869,3C ;9933            JSR [CHECK.RESULT]                     ; проверка результата на нуль
U 0D0D, 88D0,24 ;9934            JMP [LOOP.T14.4]                       ; зацикливание при ошибке, если разрешено
U 0D0E, 7D12,15 ;9935            COM LS[T9]                             ; восстановление адреса
U 0D0F, FC88,15 ;9936            DEC LS[ADDRESS.DATA]                   ; уменьшение адреса
U 0D10, FC88,15 ;9937            DEC LS[ADDRESS.DATA]                   ; дважды, так как подпрограмма увеличивает его
U 0D11, 0A18,2C ;9938            JSR [NEXT.TB.ADDRESS]                  ; увеличение адреса
U 0D12, 88D0,14 ;9939            JMP [REPEAT.T14.4]                     ; повторение, если не выполнено, иначе подпрограмма
                ;9940                                                   ; выполнит возврат к следующему адресу (выполнено)
                ;9941    END.T14:
```

```
;9942    .PAGE "ТЕСТ 15 - тест поля признаков и промаха в буфере трансляции (модуль MCT)"
;9943    ;
;9944    ; ОПИСАНИЕ ТЕСТА:
;9945    ;
;9946    ;       Этот тест проверяет схемы признаков, включая память 256X4, компаратор, фор-
;9947    ;   мирующий сигнал промаха в буфере трансляции, и схему CSR1 для бита промаха в
;9948    ;   буфере трансляции (бит 21). Тест выполняет процедуру WRITE.TB для загрузки бу-
;9949    ;   фера трансляции и поля признаков и тогда выполняет процедуру TEST.V.RCHK для
;9950    ;   стробирования в CSR результата проверки промаха в буфере трансляции. Схемы
;9951    ;   промаха в буфере трансляции проверяются первыми. Когда они проверены, прове-
;9952    ;   ряется поле признаков. Следующая часть этого теста проверяет, что процедура
;9953    ;   WRITE.UBS.MAP предотвращает модификацию поля признаков памяти и что очищенный
;9954    ;   бит BYTE OFFSET будет разрешать установку бита промаха в буфере трансляции
;9955    ;   независимо от результатов сравнения.
;9956    ;       Данные для поля признаков поступают из LVA15 по 30 при записи в буфер тран-
;9957    ;   сляции. Данные загружаются в память по адресу, указанному LVA9 по 14 и LVA31
;9958    ;   (128 десят.ячеек). Бит 7 адреса памяти аппаратно устанавливается низким. За-
;9959    ;   пись в память признаков запрещаетя при записи в область отображения общей ши-
;9960    ;   ны. Биты признаков сравниваются с LVA15 по 30 при процедуре TEST.V.RCHK для
;9961    ;   формирования сигнала TB MISS L. Этот бит стробируется в бит 21 CSR1, если
;9962    ;   BYTE OFFSET установлен.
;9963    ;       Путь данных для процедуры WRITE.TB описан в тесте памяти буфера трансляции.
;9964    ;       Путь данных для процедуры TEST.V.RCHK следующий: выдается инструкция MEM.
;9965    ;   REQ C MF = TEST.V.RCHK (2H) и выполняется начальное ветвление к соответствую-
;9966    ;   щей микропрограмме памяти, как описано в начале этого листинга. По адресу
;9967    ;   начального ветвления выдается строб циклического сдвига (используется для
;9968    ;   подготовки OP ERR (ошибка операции), но в этом тесте не проверяется), разре-
;9969    ;   шается VAR BYPASS и выставляется BUSY. Следующий цикл просто повторяет VAR
;9970    ;   BYPASS и MEMORY BUSY. Следующий цикл стробирует в CSR1 любые биты ошибок, ко-
;9971    ;   торые установлены, и сбрасывает биты RDS и CRD. Приемо-передатчики MCT разре-
;9972    ;   шены, так что биты PA из буфера трансляции и некоторые биты регистра вирту-
;9973    ;   ального адреса могут считываться. VAR BYPASS и MEMORY BUSY все еще возбужде-
;9974    ;   ны. Следующий цикл повторяет предыдущий, за исключением того, что VAR BYPASS
;9975    ;   сброшен. Следующий цикл зацикливается в ожидании снятия CPU GRANT (процессор
;9976    ;   читает данные на шине MC инструкцией MOV MEM.DATA TO WR), поддерживая раз-
;9977    ;   решенными приемо-передатчики MCT. После снятия CPU GRANT выполняется подго-
;9978    ;   товка для цикла общей шины и микрокод возвращается в холостой цикл.
;9979    ;       Тестовыми данными, используемыми в этом тесте, являются: все единицы, все
;9980    ;   нули, сдвигаемая единица и сдвигаемый нуль для проверки схем промаха в буфере
;9981    ;   трансляции и для проверки памяти признаков; кроме того, для памяти признаков
;9982    ;   выполняется тест "БЕГУЩЕЙ ИНВЕРСИИ".
;9983    ;
;9984    ; ПРЕДПОЛОЖЕНИЯ:
;9985    ;
;9986    ;   Предполагается, что все предыдущие тесты выполнены успешно.
;9987    ;
;9988    ; ШАГИ ТЕСТА:
;9989    ;
;9990    ;   1. Установка маски ошибки, номера ошибки и номера модуля в местной памяти
;9991    ;      (для распечатки ошибок) и очистка предыдущего номера ошибки в местной па-
;9992    ;      мяти. Также в CSR1 устанавливается бит MME (разрешение диспетчера памяти).
;9993    ;   2. Очистка WR1 и выдача инструкции MEM.REQ C MF = WRITE.TB и DT = LONGWORD
;9994    ;      с использованием ячейки местной памяти, содержащей 0, в качестве адреса.
;9995    ;   3. Выдача инструкции WRITE.MEM LS с использованием ячейки местной памяти, со-
;9996    ;      держащей 1 в бите 25 (для записи BYTE OFFSET), остальные 0. Теперь поле
```

```
;9997    ;       признаков содержит нули.
;9998    ;    4. Выдача инструкции MEM.REQ C MF = TEST.V.RCHK и DT = LONGWORD, с использова-
;9999    ;       нием в качестве адреса ячейки местной памяти, содержащей нули. Затем выдает-
;10000   ;       ся инструкция MOV.MEM.DATA TO WR[0] и проверяются нули (проверка пересылки
;10001   ;       битов PA из буфера трансляции и битов LVA, главным образом, является провер-
;10002   ;       кой микрокода).
;10003   ;    5. Изменение маски ошибки для проверки только бита 21 (промах в буфере тран-
;10004   ;       сляции) и выдача READ.CSR C 1 B LVA02. Бит 21 должен быть сброшен.
;10005   ;    6. Выдача инструкции MEM.REQ C MF = TEST.V.RCHK и DT = LONGWORD с использова-
;10006   ;       нием в качестве адреса ячейки местной памяти, содержащей сдвигаемую единицу
;10007   ;       в битах с 15 по 30. Затем выдается инструкция MOV MEM.DATA TO WR[0] для
;10008   ;       завершения. Выдается READ.CSR C 1 в LVA02 и проверяется установка бита про-
;10009   ;       маха в буфере трансляции (бит 21).
;10010   ;    7. Повторение шагов 6, пока единица не сдвинется через биты с 15 по 30.
;10011   ;    8. Установка маски ошибки для проверки битов 00-22 и 24-31, установка WR1 и
;10012   ;       выдача инструкции MEM.REQ C MF = WRITE.TB и DT = LONGWORD, с использованием
;10013   ;       ячейки местной памяти, содержащей нули, в качестве адреса.
;10014   ;    9. Выдача инструкции WRITE.MEM LS с использованием ячейки местной памяти, со-
;10015   ;       держащей все единицы. Сейчас поле признаков содержит единицы.
;10016   ;   10. Выдача инструкции MEM.REQ с MF = TEST.V.RCHK и DT = LONGWORD. В качестве
;10017   ;       адреса используется ячейка местной памяти, содержащая единицы в битах с
;10018   ;       15 по 30. Затем выдача инструкции MOV MEM.DATA TO WR[0] и проверка на еди-
;10019   ;       ницы.
;10020   ;   11. Изменение маски ошибки для проверки только бита 21 (промах в буфере тран-
;10021   ;       сляции) и выдача инструкции READ.CSR C единицей в LVA02. Бит 21 должен
;10022   ;       быть сброшен.
;10023   ;   12. Выдача инструкции MEM.REQ C MF = TEST.V.RCHK и DT = LONGWORD. В качестве
;10024   ;       адреса используется ячейка местной памяти, содержащая сдвигаемый нуль в
;10025   ;       битах с 15 по 30. Тогда выдача инструкции MOV MEM.DATA TO WR[0] (для завер-
;10026   ;       шения цикла памяти). Выдача инструкции READ CSR с единицей в LVA02 и про-
;10027   ;       верка, установлен ли промах в буфере трансляции (бит 21).
;10028   ;   13. Повторение шага 6, пока нуль не сдвинется через биты с 15 по 30.
;10029   ;   14. Запись в буфер трансляции сдвигаемой единицы в поле признаков от адреса 0.
;10030   ;       Выдача инструкции TEST.V.RCHK со сдвигаемой единицей в LVA 15-30 и провер-
;10031   ;       ка в CSR1 бита промаха в буфере трансляции на нуль.
;10032   ;   15. Повторене шага 14 во всех других адресах.
;10033   ;   16. Повторение шагов 14 и 15, используя сдвигаемый нуль в качестве данных.
;10034   ;   17. Выполнение теста типа "МАРШ" со сдвигаемыми битами в адресе памяти призна-
;10035   ;       ков для проверки неисправностей адресных линий.
;10036   ;   18. Выполнение теста "БЕГУЩЕЙ ИНВЕРСИИ" для памяти признаков.
;10037   ;   19. Запись нулей в поле признаков, запись единиц в область отображения общей
;10038   ;       шины, чтение поля признаков для проверки на нуль.
;10039   ;   20. Запись нулей в поле признаков и в бит BYTE OFFSET (смещение байта). Выда-
;10040   ;       ча TEST.V.RCHK с соответствующими LVA 15-30. Проверка, что сброшенный бит
;10041   ;       BYTE OFFSET вызывает промах в буфере трансляции.
;10042   ;
;10043   ; ОШИБКИ:
;10044   ;
;10045   ;   ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными является бит промаха в буфере
;10046   ;   трансляции в CSR1.
;10047   ;
;10048   ;       ошибка 1 - неправильно работает функция TEST.V.RCHK при возвращении правиль-
;10049   ;   ных битов PA или LVA. Возможно, неисправна ПЗУ микропамяти. Под OTHER в сообще-
;10050   ;   нии об ошибке печатается адрес памяти.
;10051   ;
```

```
;10052  ;    ПРИМЕЧАНИЕ: При ошибках 2 и 3 данными под OTHER будут биты LVA С 15 по 30,
;10053  ;    которые сравниваются со всеми нулями в памяти признаков.
;10054  ;
;10055  ;       ошибка 2 - промах в буфере трансляции устанавливается при совпадении со
;10056  ;    всеми нулями. Смотри примечание выше.
;10057  ;
;10058  ;       ошибка 3 - неправильно работает установка промаха в буфере трансляции при
;10059  ;    несовпадении нулей и сдвигаемой единицы. Смотри примечание выше.
;10060  ;
;10061  ;       ошибка 4 - неправильно работает функция TEST.V.RCHK при возвращении правиль-
;10062  ;    ных битов PA или LVA. возможно, неисправна ПЗУ микропамяти. Под OTHER в сообще-
;10063  ;    нии об ошибке печатается адрес памяти.
;10064  ;
;10065  ;    ПРИМЕЧАНИЕ: При ошибках 5 и 6 данными под OTHER будут биты LVA с 15 по 30,
;10066  ;    которые сравниваются со всеми единицами в памяти признаков.
;10067  ;
;10068  ;       ошибка 5 - промах в буфере трансляции устанавливается при совпадении со все-
;10069  ;    ми единицами. Смотри примечание выше.
;10070  ;
;10071  ;       ошибка 6 - неправильно работает установка промаха в буфере трансляции при
;10072  ;    несовпадении единиц и сдвигаемого нуля. Смотри примечание выше.
;10073  ;
;10074  ;       ошибка 7 - неправильно работает поле признаков памяти в тесте сдвигаемой
;10075  ;    единицы по адресу памяти, распечатанному под OTHER. Тестовые данные при неис-
;10076  ;    правности находятся в WR2.
;10077  ;
;10078  ;       ошибка 8 - неправильно работает поле признаков памяти в тесте со сдвигаемым
;10079  ;    нулем по адресу памяти, распечатанному под OTHER. Тестовые данные при неисправ-
;10080  ;    ности находятся в WR2.
;10081  ;
;10082  ;       ошибка 9 - неправильно работает поле признаков памяти в тесте "МАРШ" со
;10083  ;    сдвигаемыми битами адреса. Адрес печатается под OTHER в сообщении об ошибке.
;10084  ;
;10085  ;       ошибка A - неправильно работает поле признаков памяти в тесте "БЕГУЩЕЙ ИН-
;10086  ;    ВЕРСИИ. Адрес находится в ячейке 9 местной памяти.
;10087  ;
;10088  ;       ошибка B - происходит запись в поле признаков памяти при записи в область
;10089  ;    отображения общей шины (неправильно работает запрет записи). Под OTHER печа-
;10090  ;    тается адрес памяти признаков.
;10091  ;
;10092  ;       ошибка C - неправильно возбуждается TB MISS (промах в буфере трансляции)
;10093  ;    при совпадении и при сброшенном BYTE OFFSET (смещение байта). Под OTHER печа-
;10094  ;    тается адрес памяти признаков.
;10095  ;
;10096  ; НАЛАДКА:
;10097  ;
;10098  ;    ПРИМЕЧАНИЕ: Когда в сообщении об ошибке под OTHER печатается адрес, биты
;10099  ;    расположены так, что получается действительный адрес памяти в шеснадцате-
;10100  ;    ричном формате.
;10101  ;
;10102  ;       ОШИБКА 1 - Эта ошибка указывает на возможную неисправность ПЗУ управляю-
;10103  ;    щей памяти. Все схемы, используемые для записи и чтения данных, были прове-
;10104  ;    рены предыдущими тестами. Если пошаговый режим MCT возможен, эти биты можно
;10105  ;    проверить на наличие ошибок в каждом цикле.
;10106  ;
```

```
;10107  ;       ОШИБКА 2 - Эта ошибка указывает, что TB MISS (промах в буфере трансляции)
;10108  ;    был возбужден, когда тестировалось совпадение со всеми нулями. Неправильно
;10109  ;    работает память признаков, компаратор или схемы CSR1. Необходимо остановить
;10110  ;    центральный процессор на инструкции MOVE MEM.DATA TO WR[0] после запроса
;10111  ;    памяти TEST.V.RCHK. МСТ должен циклиться в ожидании сигнала CPU DATA RCVD
;10112  ;    для снятия сигнала CPU GRANT. В это время может быть проверен высокий уро-
;10113  ;    вень сигнала TB MISS L. Если TB MISS L низкий, необходимо проверить входы
;10114  ;    компаратора. Все биты TAG и LVA должны быть низкими. Если входы правильные,
;10115  ;    неисправен компаратор. Если биты признаков неправильные, необходимо прове-
;10116  ;    рить, изменяются ли уровни на ножках 20 и 19 микросхем памяти поля призна-
;10117  ;    ков. Эти сигналы были проверены предыдущими тестами. Если уровни не меняются,
;10118  ;    подозревается, что неисправна память или неправильный вход LVA.
;10119  ;       Если сигнал TB MISS L имеет высокий уровень, необходимо проверить его на
;10120  ;    входе ПМЛ CSR 1A. Также необходимо проверить высокий уровень сигнала BYTE
;10121  ;    OFFSET H. Если эти сигналы правильные, скорее всего неисправна ПМЛ CSR 1A.
;10122  ;
;10123  ;       ОШИБКА 3 - Эта ошибка указывает, что сигнал TB MISS не возбуждается, ког-
;10124  ;    да появляется несовпадение одного бита. Неправильно работает память поля приз-
;10125  ;    наков или компаратор признаков или схемы CSR1. Необходимо остановить централь-
;10126  ;    ный процессор на инструкции MOVE MEM.DATA TO WR[0] после запроса памяти TEST.V.
;10127  ;    RCHK. МСТ должен циклиться в ожидании сигнала CPU DATA RCVD для снятия сиг-
;10128  ;    нала CPU GRANT. В это время может быть проверен низкий уровень сигнала TB
;10129  ;    MISS L. Если TB MISS L высокий, необходимо проверить входы компаратора. Все
;10130  ;    биты признаков должны быть низкими и все биты LVA, за исключением одного,
;10131  ;    должны быть низкими. Если так, компаратор неисправен.
;10132  ;       Если TB MISS L низкий, необходимо проверить его на входе ПМЛ CSR 1A. Если
;10133  ;    здесь правильно, скорее всего, подозревается ПМЛ CSR 1A.
;10134  ;
;10135  ;       ОШИБКА 4 - То же, что и при ошибке 1.
;10136  ;
;10137  ;       ОШИБКА 5 - То же, что и при ошибке 2. Отличается только тем, что все вхо-
;10138  ;    ды компаратора должны быть высокими (все единицы).
;10139  ;
;10140  ;       ОШИБКА 6 - То же, что и при ошибке 3. Отличается только тем, что все вхо-
;10141  ;    ды компаратора должны быть высокими на битах признаков и битах LVA, за исклю-
;10142  ;    чением одного.
;10143  ;
;10144  ;       ОШИБКА 7 - Вероятнее всего, эта ошибка указывает на несправность памяти.
;10145  ;    Центральный процессор необходимо остановить на инструкции MOV MEM.DATA TO
;10146  ;    WR[0], следующей за MEM.REQ, которая выполняет функцию TEST.V.RCHK. МСТ дол-
;10147  ;    жен циклиться в ожидании сброса CPU GRANT. В это время можно проверить выхо-
;10148  ;    ды памяти для обнаружения неисправного бита.
;10149  ;
;10150  ;       ОШИБКА 8 - То же, что и при ошибке 7.
;10151  ;
;10152  ;       ОШИБКА 9 - Эта ошибка указывает на возможную неисправность адресных линий
;10153  ;    или декодирования адреса внутри микросхемы памяти. Адресные линии могут быть
;10154  ;    проверены установкой останова по адресу микрокоманды MOV MEM.DATA TO WR[0],
;10155  ;    следующей за TEST.V.RCHK. Когда центральный процессор остановится по совпа-
;10156  ;    дению микроадреса, можно проверить адресные линии на микросхеме памяти на
;10157  ;    сдвиг единицы через поле нулей. После каждой команды продолжения, выдаваемой
;10158  ;    из терминала, бит адреса должен сдвигаться. Если адресные линии исправны,
;10159  ;    подозревается микросхема памяти.
;10160  ;
;10161  ;       ОШИБКА A - Скорее всего, эту ошибку формирует неисправная микросхема па-
```

```
                ;10162  ;   мяти. Для обнаружения неисправной микросхемы памяти необходимо выполнить
                ;10163  ;   действия, описанные для ошибки 9.
                ;10164  ;
                ;10165  ;       ОШИБКА B - Эта ошибка указывает возможную неисправность входа TBA9 H или
                ;10166  ;   самой памяти признаков. Необходимо остановить центральный процессор на инст-
                ;10167  ;   рукции MEM.REQ WRITE.UBS.MAP. Вход TBA9 H памяти поля признаков в это время
                ;10168  ;   должен быть высоким на каждой микросхеме памяти поля признаков. Если эти
                ;10169  ;   входы правильные, тогда неисправна одна из микросхем памяти.
                ;10170  ;       Для определения неисправной микросхемы памяти, необходимо остановить цент-
                ;10171  ;   ральный процессор на инструкции MOV MEM DATA TO WR[0], следующей за TEST.V.
                ;10172  ;   RCHK и проверить выходы памяти. Микросхема памяти, имеющая на выходе единицу,
                ;10173  ;   неисправна.
                ;10174  ;
                ;10175  ;       ОШИБКА C - Эта ошибка указывает на возможную неисправность ПМЛ CSR 1A или
                ;10176  ;   входа BYTE OFFSET. Необходимо остановить центральный процессор на инструкции
                ;10177  ;   MOV MEM.DATA TO WR[0], следующей за TEST.V.RCHK, и проверить низкий уровень
                ;10178  ;   сигнала BYTE OFFSET на входе ПМЛ CSR 1A. Если правильно, по-видимому, неис-
                ;10179  ;   правна ПМЛ CSR 1A.
                ;10180  ;
                ;10181  T.15:
U 0D13, B65E,15 ;10182          MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;10183                                                  ; состояния
U 0D14, 3E80,15 ;10184          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;10185                                                  ; 15 указывает начало теста консольному процессору
U 0D15, 10E0,15 ;10186          MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN консольному процессору
                ;10187  WAIT.T15.0:
U 0D16, 88D1,64 ;10188          JMP [WAIT.T15.0]                        ; зацикливание для ожидания ответа консольного
                ;10189                                                  ; процессора
U 0D17, 0A1A,AC ;10190          JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0D18, B66E,15 ;10191          MOV LS[BIT23] TO WR[0]                  ; маскирование бита 23 (PA24)
U 0D19, 3E8A,15 ;10192          MOV WR[0] TO LS[ERROR.MASK]             ; подготовка маски ошибки для проверки всех битов, за
                ;10193                                                  ; исключением бита 23
U 0D1A, B670,15 ;10194          MOV LS[OTHER.DATA] TO WR[0]             ; подготовка адреса для печати под "OTHER"
U 0D1B, 3E80,15 ;10195          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 24 в слове управления для разрешения
                ;10196                                                  ; печати адреса под "OTHER", если ошибка
U 0D1C, 0A17,DC ;10197          JSR [WRITE.CSR1.MME]                    ; разрешение диспетчера памяти
U 0D1D, B640,15 ;10198          MOV LS[#1] TO WR[0]                     ; 1 в WR0
U 0D1E, BEFC,15 ;10199          MOV WR[0] TO LS[SIZE]                   ; подготовка регистра размера для типа данных - слово
U 0D1F, 6588,15 ;10200          CLR LS[ADDRESS.DATA]                    ; очистка адреса, который выбирается при печати
U 0D20, 989C,F5 ;10201          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи по адресу 0 буфера трансляции
                ;10202                                                  ; нулей в поле признаков
U 0D21, B272,15 ;10203          WRITE.MEM LS[BIT25]                     ; запись в буфере трансляции 1 в бит BYTE OFFSET и нулей
                ;10204                                                  ; в другие
                ;10205  LOOP.T15.1:
U 0D22, B640,15 ;10206          MOV LS[#1] TO WR[0]                     ; 1 в WR0
U 0D23, BE82,15 ;10207          MOV WR[0] TO LS[ERROR.NUMBER]           ; подготовка номера ошибки в местной памяти
U 0D24, 989D,75 ;10208          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]  ; подготовка проверки буфера трансляции и
                ;10209                                                  ; стробирование CSR1 с битом TB MISS
U 0D25, 3022,15 ;10210          MOV MEM.DATA TO WR[0]                   ; чтение PA и LVA битов из буфера трансляции по адресу 0
                ;10211                                                  ; (в основном проверяется микрокод)
U 0D26, 2F82,95 ;10212          CLR WR[1]                               ; ожидаемые данные
U 0D27, 0869,3C ;10213          JSR [CHECK.RESULT]                      ; проверка битов PA и LVA на все нули
U 0D28, 08D2,24 ;10214          JMP [LOOP.T15.1]                        ; зацикливание при ошибке, если разрешено
U 0D29, FF82,15 ;10215          INC LS[ERROR.NUMBER]                    ; ошибка 2
U 0D2A, 5F6A,15 ;10216          MCOM LS[TB.MISS] TO WR[0]               ; очистка бита 21 в WR0, установка других
```

```
U 0D2B, 3E8A,15 ;10217          MOV WR[0] TO LS[ERROR.MASK]                ; подготовка проверки только бита 21 (бит TB MISS
                ;10218                                                     ; CSR1)
U 0D2C, 9D45,75 ;10219          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]      ; подготовка чтения CSR1
U 0D2D, 3022,15 ;10220          MOV MEM.DATA TO WR[0]                      ; выборка данных из CSR1
U 0D2E, 2F82,95 ;10221          CLR WR[1]                                  ; ожидаемые данные
U 0D2F, 0869,3C ;10222          JSR [CHECK.RESULT]                         ; проверка, что бит 21 сброшен (нет TB MISS)
U 0D30, 08D2,24 ;10223          JMP [LOOP.T15.1]                           ; зацикливание при ошибке, если разрешено
U 0D31, FF82,15 ;10224          INC LS[ERROR.NUMBER]                       ; ошибка 3
U 0D32, 365F,15 ;10225          MOV LS[BIT15] TO WR[2]                     ; установка бита 15 в WR
                ;10226  REPEAT.T15.3:
U 0D33, 3E13,15 ;10227          MOV WR[2] TO LS[T9]                        ; подготовка адреса в ячейке 9 местной памяти
                ;10228                                                     ; (сдвигаемая 1 из LVA 15 по LVA 30)
U 0D34, 3F89,15 ;10229          MOV WR[2] TO LS[ADDRESS.DATA]              ; подготовка печати значения LVA 15 по LVA 30 под
                ;10230                                                     ; "OTHER DATA"
                ;10231  LOOP.T15.3:
U 0D35, B66A,95 ;10232          MOV LS[TB.MISS] TO WR[1]                   ; ожидаемые данные (TB MISS установлен)
U 0D36, 9813,75 ;10233          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]     ; подготовка выдачи TEST.V.RCHK с несовпадением поля
                ;10234                                                     ; признаков
U 0D37, 3022,15 ;10235          MOV MEM.DATA TO WR[0]                      ; выдача инструкции MOV для завершения цикла памяти (нет
                ;10236                                                     ; необходимости проверять эти данные)
U 0D38, 9D45,75 ;10237          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]      ; подготовка чтения CSR1
U 0D39, 3022,15 ;10238          MOV MEM.DATA TO WR[0]                      ; выборка данных из CSR1
U 0D3A, 0869,3C ;10239          JSR [CHECK.RESULT]                         ; проверка установки TB MISS (BIT 21)
U 0D3B, 08D3,54 ;10240          JMP [LOOP.T15.3]                           ; зацикливание при ошибке, если разрешено
U 0D3C, 23C1,15 ;10241          ROL WR[2]                                  ; сдвиг бита влево
                ;10242            TST WR[2],                               ; проверка, установлен ли бит 31 и установка кодов
U 0D3D, 2005,35 ;10243          DT(LONG)&SET.ALU.CC                        ; условий
U 0D3E, 88D3,30 ;10244          JMP.IF[N.CLR] TO [REPEAT.T15.3]            ; повторение, если бит 31 еще не установлен, иначе
                ;10245                                                     ; продолжение
U 0D3F, FF82,15 ;10246          INC LS[ERROR.NUMBER]                       ; ошибка 4
U 0D40, 3683,95 ;10247          MOV LS[ERROR.NUMBER] TO WR[3]              ; сохранение номера ошибки
U 0D41, B736,15 ;10248          MOV LS[#7FFF8000] TO WR[0]                 ; выборка данных для адреса буфера трансляции
U 0D42, C726,15 ;10249          BIS LS[#FF] TO WR[0]                       ; установка битов 0-7 LVA для TEST.V. RCHK
U 0D43, 4750,15 ;10250          BIS LS[BIT8] TO WR[0]                      ; также установка LVA 8
U 0D44, BE12,15 ;10251          MOV WR[0] TO LS[T9]                        ; запоминание в ячейке 9 местной памяти
U 0D45, 9812,F5 ;10252          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]        ; подготовка записи в буфер трансляции по адресу 0
                ;10253                                                     ; единиц в поле признаков
U 0D46, 329E,15 ;10254          WRITE.MEM LS[ONES]                         ; запись всех единиц в буфер трансляции по адресу 0
                ;10255  LOOP.T15.4:
U 0D47, 6588,15 ;10256          CLR LS[ADDRESS.DATA]                       ; очистка адреса для печати
U 0D48, BE83,95 ;10257          MOV WR[3] TO LS[ERROR.NUMBER]              ; ошибка номер 4
U 0D49, B66E,15 ;10258          MOV LS[BIT23] TO WR[0]                     ; маскирование бита 23 (PA 24)
U 0D4A, 3E8A,15 ;10259          MOV WR[0] TO LS[ERROR.MASK]                ; подготовка маски ошибки для проверки всех битов, кроме
                ;10260                                                     ; бита 23
U 0D4B, 9813,75 ;10261          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]     ; подготовка проверки буфера трансляции и
                ;10262                                                     ; стробирование CSR1 битом TB MISS
U 0D4C, 3022,15 ;10263          MOV MEM.DATA TO WR[0]                      ; чтение битов PA и LVA из буфера трансляции по адресу 0
                ;10264                                                     ; (в основном проверяется микрокод)
U 0D4D, 5F7C,95 ;10265          MCOM LS[BIT30] TO WR[1]                    ; ожидаемые данные (все биты установлены, кроме LVA31,
                ;10266                                                     ; который возвращается как бит 30)
U 0D4E, 0869,3C ;10267          JSR [CHECK.RESULT]                         ; проверка битов PA и LVA на все единицы
U 0D4F, 08D4,74 ;10268          JMP [LOOP.T15.4]                           ; зацикливание при ошибке, если разрешено
U 0D50, FF82,15 ;10269          INC LS[ERROR.NUMBER]                       ; ошибка 5
U 0D51, B736,15 ;10270          MOV LS[#7FFF8000] TO WR[0]                 ; установка LVA 15 по 30
U 0D52, BE88,15 ;10271          MOV WR[0] TO LS[ADDRESS.DATA]              ; "OTHER DATA" являются LVA 15 по 30
```

```
U 0D53, 5F6A,15 ;10272          MCOM LS[TB.MISS] TO WR[0]                ; очистка бита 21 в WR0, установка других
U 0D54, 3E8A,15 ;10273          MOV WR[0] TO LS[ERROR.MASK]              ; подготовка проверки только бита 21 (бит TB MISS CSR1)
U 0D55, 9D45,75 ;10274          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; подготовка чтения CSR1
U 0D56, 3022,15 ;10275          MOV MEM.DATA TO WR[0]                    ; выборка данных CSR1
U 0D57, 2F82,95 ;10276          CLR WR[1]                                ; ожидаемые данные
U 0D58, 0869,3C ;10277          JSR [CHECK.RESULT]                       ; проверка, что бит 21 сброшен (нет TB MISS)
U 0D59, 08D4,74 ;10278          JMP [LOOP.T15.4]                         ; зацикливание при ошибке, если разрешено
U 0D5A, FF82,15 ;10279          INC LS[ERROR.NUMBER]                     ; ошибка 6
U 0D5B, 5F5F,15 ;10280          MCOM LS[BIT15] TO WR[2]                  ; очистка бита 15 в WR2
                ;10281  REPEAT.T15.6:
U 0D5C, 3E13,15 ;10282          MOV WR[2] TO LS[T9]                      ; подготовка поля признаков в местной памяти (сдвигаемый
                ;10283                                                   ; нуль из бита 15 до бита 30)
U 0D5D, 3736,95 ;10284          MOV LS[#7FFF8000] TO WR[1]               ; очистка битов адреса буфера трансляции (8-14 и бит
                ;10285                                                   ; 31)
U 0D5E, A0C2,95 ;10286          COM WR[1]                                ; сейчас биты 0-14 и 31 установлены в WR1
U 0D5F, EF12,95 ;10287          XOR WR[1] TO LS[T9]                      ; очистка битов адреса буфера трансляции в ячейке 9
                ;10288                                                   ; местной памяти
U 0D60, 3612,15 ;10289          MOV LS[T9] TO WR[0]                      ; выборка значения LVA 15-30
U 0D61, BE88,15 ;10290          MOV WR[0] TO LS[ADDRESS.DATA]            ; печать значения LVA 15 по 30 под "OTHER DATA"
                ;10291   LOOP.T15.6:
U 0D62, B66A,95 ;10292          MOV LS[TB.MISS] TO WR[1]                 ; ожидаемые данные (TB MISS установлен)
U 0D63, 9813,75 ;10293          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]   ; подготовка выдачи TEST.V.RCHK С несовпадением поля
                ;10294                                                   ; признаков
U 0D64, 3022,15 ;10295          MOV MEM.DATA TO WR[0]                    ; выдача инструкции MOV для завершения цикла памяти (нет
                ;10296                                                   ; необходимости проверять эти данные)
U 0D65, 9D45,75 ;10297          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; подготовка чтения CSR1
U 0D66, 3022,15 ;10298          MOV MEM.DATA TO WR[0]                    ; выборка данных из CSR1
U 0D67, 0869,3C ;10299          JSR [CHECK.RESULT]                       ; проверка, установлен ли TB MISS (бит 21)
U 0D68, 88D6,24 ;10300          JMP [LOOP.T15.6]                         ; зацикливание при ошибке, если разрешено
U 0D69, 23C1,15 ;10301          ROL WR[2]                                ; сдвиг бита влево
                ;10302            TST WR[2],                             ; проверка, сброшен ли бит 31 и установка кодов
U 0D6A, 2005,35 ;10303          DT(LONG)&SET.ALU.CC                      ; условий
U 0D6B, 08D5,C8 ;10304          JMP.IF[N.SET] TO [REPEAT.T15.6]          ; повторение, если бит 31 еще не сброшен, иначе
                ;10305                                                   ; продолжение
U 0D6C, FF82,15 ;10306          INC LS[ERROR.NUMBER]                     ; ошибка 7
U 0D6D, 6512,15 ;10307          CLR LS[T9]                               ; очистка начального адреса буфера трансляции
U 0D6E, 6588,15 ;10308          CLR LS[ADDRESS.DATA]                     ; очистка адреса для печати
                ;10309  REPEAT.T15.7.NEWADDRESS:
U 0D6F, 365F,15 ;10310          MOV LS[BIT15] TO WR[2]                   ; использование WR2 для запоминания тестовых данных со
                ;10311                                                   ; сдвигаемым битом
                ;10312  REPEAT.T15.7:
U 0D70, 3612,15 ;10313          MOV LS[T9] TO WR[0]                      ; выборка адреса буфера трансляции
U 0D71, AEC4,15 ;10314          BIS WR[2] TO WR[0]                       ; установка поля признаков в адресе
U 0D72, BE14,15 ;10315          MOV WR[0] TO LS[T10]                     ; запоминание в местной памяти данных для записи в
                ;10316                                                   ; буфер трансляции
                ;10317  LOOP.T15.7:
U 0D73, 9814,F5 ;10318          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]     ; подготовка записи в буфер трансляции сдвигаемой
                ;10319                                                   ; единицы в поле признаков
U 0D74, B272,15 ;10320          WRITE.MEM LS[BIT25]                      ; запись данных в буфер трансляции по адресу, указанному
                ;10321                                                   ; в ячейке 9 местной памяти (BYTE OFFSET установлен,
                ;10322                                                   ; другие сброшены)
U 0D75, 2F82,95 ;10323          CLR WR[1]                                ; ожидаемые данные (TB MISS сброшен)
U 0D76, 9815,75 ;10324          MEM.REQ[TEST.V.RCHK] ADRS[T10] DT[LONG]  ; подготовка выдачи TEST.V.RCHK с совпадением поля
                ;10325                                                   ; признаков
U 0D77, 3022,15 ;10326          MOV MEM.DATA TO WR[0]                    ; выдача инструкции MOV для завершения цикла памяти (нет
```

```
                    ;10327                                                  ; необходимости проверять эти данные)
U 0D78, 9D45,75     ;10328          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0D79, 3022,15     ;10329          MOV MEM.DATA TO WR[0]                   ; выборка данных из CSR1
U 0D7A, 0869,3C     ;10330          JSR [CHECK.RESULT]                      ; проверка, сброшен ли TB MISS (бит 21)
U 0D7B, 88D7,34     ;10331          JMP [LOOP.T15.7]                        ; зацикливание при ошибке, если разрешено
U 0D7C, 23C1,15     ;10332          ROL WR[2]                               ; сдвиг бита влево
                    ;10333            TST WR[2],                            ; проверка, установлен ли бит 31 и
U 0D7D, 2005,35     ;10334          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0D7E, 08D7,00     ;10335          JMP.IF[N.CLR] TO [REPEAT.T15.7]         ; повторение, если бит 31 еще не установлен, иначе
                    ;10336                                                  ; переход на следующий адрес и повторение
U 0D7F, 0A18,2C     ;10337          JSR [NEXT.TB.ADDRESS]                   ; следующий адрес буфера трансляции в ячейке 9 местной
                    ;10338                                                  ; памяти
U 0D80, 08D6,F4     ;10339          JMP [REPEAT.T15.7.NEWADDRESS]           ; если здесь, повторение теста с новым адресом. Если все
                    ;10340                                                  ; адреса проверены; подпрограмма возвращает на
                    ;10341                                                  ; следующую инструкцию
U 0D81, FF82,15     ;10342          INC LS[ERROR.NUMBER]                    ; ошибка 8
U 0D82, 6512,15     ;10343          CLR LS[T9]                              ; очистка начального адреса буфера трансляции
U 0D83, 6588,15     ;10344          CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати при ошибке
                    ;10345  REPEAT.T15.8.NEWADDRESS:
U 0D84, 5F5F,15     ;10346          MCOM LS[BIT15] TO WR[2]                 ; использование WR2 для запоминания тестовых данных со
                    ;10347                                                  ; сдвигаемым нулем
                    ;10348  REPEAT.T15.8:
U 0D85, 2004,15     ;10349          MOV WR[2] TO WR[0]                      ; выборка поля признаков
U 0D86, 3736,95     ;10350          MOV LS[#7FFF8000] TO WR[1]              ; очистка битов адреса буфера трансляции (8-14 и бита
                    ;10351                                                  ; 31)
U 0D87, A0C2,95     ;10352          CON WR[1]                               ; сейчас биты 0-14 и бит 31 установлены
U 0D88, AF42,15     ;10353          BIC WR[1] TO WR[0]                      ; очистка битов адреса буфера трансляции
U 0D89, 4712,15     ;10354          BIS LS[T9] TO WR[0]                     ; установка адреса буфера трансляции в данных поля
                    ;10355                                                  ; признаков
U 0D8A, BE14,15     ;10356          MOV WR[0] TO LS[T10]                    ; запоминание в местной памяти данных для записи буфера
                    ;10357                                                  ; трансляции
                    ;10358  LOOP.T15.8:
U 0D8B, 9814,F5     ;10359          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]    ; подготовка записи в буфер трансляции сдвигаемого
                    ;10360                                                  ; нуля в поле признаков
U 0D8C, B272,15     ;10361          WRITE.MEM LS[BIT25]                     ; запись данных в буфер трансляции по адресу, указанному
                    ;10362                                                  ; в ячейке 9 местной памяти (BYTE OFFSET установлен,
                    ;10363                                                  ; другие сброшены)
U 0D8D, 2F82,95     ;10364          CLR WR[1]                               ; ожидаемые данные (TB MISS сброшен)
U 0D8E, 9815,75     ;10365          MEM.REQ[TEST.V.RCHK] ADRS[T10] DT[LONG] ; подготовка выдачи TEST.V.RCHK с совпадением поля
                    ;10366                                                  ; признаков
U 0D8F, 3022,15     ;10367          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла памяти (нет
                    ;10368                                                  ; необходимости проверять эти данные)
U 0D90, 9D45,75     ;10369          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0D91, 3022,15     ;10370          MOV MEM.DATA TO WR[0]                   ; выборка данных из CSR1
U 0D92, 0869,3C     ;10371          JSR [CHECK.RESULT]                      ; проверка, сброшен ли TB MISS (бит 21)
U 0D93, 08D8,B4     ;10372          JMP [LOOP.T15.8]                        ; зацикливание при ошибке, если разрешено
U 0D94, 23C1,15     ;10373          ROL WR[2]                               ; сдвиг бита влево
                    ;10374            TST WR[2],                            ; проверка, сброшен ли бит 31 и
U 0D95, 2005,35     ;10375          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0D96, 88D8,58     ;10376          JMP.IF[N.SET] TO [REPEAT.T15.8]         ; повторение, если бит 31 еще не сброшен, иначе переход
                    ;10377                                                  ; к следующему адресу и повторение
   0D97  0A18,2C    ;10378          JSR [NEXT.TB.ADDRESS]                   ; следующий адрес буфера трансляции в ячейке 9 местной
                     10379                                                  ; памяти
   0D99  08D8         10380          JMP [REPEAT.T15.8.NEWADDRESS]           ; если здесь, повторение теста с новым адресом. Если все
                    ;10381                                                  ; адреса проверены, подпрограмма возвращает на следующую
```

```
                ;10382                                                  ; инструкцию
U 0D99, FF82,15 ;10383          INC LS[ERROR.NUMBER]                    ; ошибка 9
U 0D9A, 6512,15 ;10384          CLR LS[T9]                              ; очистка адреса буфера трансляции в местной памяти
                ;10385  BACK.T15.9:
U 0D9B, 9812,F5 ;10386          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи в буфер трансляции
U 0D9C, B272,15 ;10387          WRITE.MEM LS[BIT25]                     ; запись фона нулей, кроме бита BYTE OFFSET (который
                ;10388                                                  ; должен быть 1)
U 0D9D, 0A18,2C ;10389          JSR [NEXT.TB.ADDRESS]                   ; выборка следующего адреса буфера трансляции
U 0D9E, B8D9,B4 ;10390          JMP [BACK.T15.9]                        ; здесь, если не все адреса проверены и повторение,
                ;10391                                                  ; иначе подпрограмма возвращает на следующую инструкцию
U 0D9F, 6512,15 ;10392          CLR LS[T9]                              ; очистка адреса буфера трансляции
U 0DA0, 6588,15 ;10393          CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати при ошибке
U 0DA1, 3651,95 ;10394          MOV LS[BIT8] TO WR[3]                   ; установка бита 8 в WR3. Циклический сдвиг в WR3
                ;10395                                                  ; сформирует первый сдвинутый адрес (адрес 1(H) памяти)
U 0DA2, B67F,15 ;10396          MOV LS[BIT31] TO WR[2]                  ; установка бита 31 в WR3. Циклический сдвиг в WR2
                ;10397                                                  ; сформирует первый сдвинутый адрес для печати (1(H))
                ;10398  REPEAT.T15.9:
U 0DA3, 3612,15 ;10399          MOV LS[T9] TO WR[0]                     ; выборка адреса трансляции для обращения
U 0DA4, 3736,95 ;10400          MOV LS[#7FFF8000] TO WR[1]              ; пересылка 7FFF8000 В WR1
U 0DA5, AEC2,15 ;10401          BIS WR[1] TO WR[0]                      ; установка битов 15-30 адреса буфера трансляции. Бит 31
                ;10402                                                  ; остается таким, каким был в адресе буфера трансляции
U 0DA6, BE14,15 ;10403          MOV WR[0] TO LS[T10]                    ; запоминание в местной памяти для TEST.V.RCHK с
                ;10404                                                  ; дополнительным кодом данных
                ;10405  LOOP.T15.9:
U 0DA7, 2F82,95 ;10406          CLR WR[1]                               ; ожидаемые данные (сброшенный TB MISS указывает
                ;10407                                                  ; совпадение)
U 0DA8, 9813,75 ;10408          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]  ; подготовка проверки поля признаков для совпадения
U 0DA9, 3022,15 ;10409          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла памяти
                ;10410                                                  ; (нет необходимости проверять эти данные)
U 0DAA, 9D45,75 ;10411          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0DAB, 3022,15 ;10412          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0DAC, 0869,3C ;10413          JSR [CHECK.RESULT]                      ; проверка TB MISS на 0 (совпадение поля признаков)
U 0DAD, B8DA,74 ;10414          JMP [LOOP.T15.9]                        ; зацикливание при ошибке, если разрешено
                ;10415  LOOP.T15.9A:
U 0DAE, 2F82,95 ;10416          CLR WR[1]                               ; ожидаемые данные (сброшенный TB MISS указывает
                ;10417                                                  ; совпадение)
U 0DAF, 9814,F5 ;10418          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]    ; подготовка записи дополнительного кода данных
U 0DB0, B272,15 ;10419          WRITE.MEM LS[BIT25]                     ; запись данных в буфер трансляции по адресу, указанному
                ;10420                                                  ; в ячейке 9 местной памяти (BYTE OFFSET установлен,
                ;10421                                                  ; другие сброшены)
U 0DB1, 9815,75 ;10422          MEM.REQ[TEST.V.RCHK] ADRS[T10] DT[LONG] ; подготовка проверки поля признаков на совпадение
U 0DB2, 3022,15 ;10423          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла памяти
                ;10424                                                  ; (нет необходимости проверять эти данные)
U 0DB3, 9D45,75 ;10425          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0DB4, 3022,15 ;10426          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0DB5, 0869,3C ;10427          JSR [CHECK.RESULT]                      ; проверка TB MISS на 0 (совпадение поля признаков)
U 0DB6, B8DA,E4 ;10428          JMP [LOOP.T15.9A]                       ; зацикливание при ошибке, если разрешено
U 0DB7, A3C1,95 ;10429          ROL WR[3]                               ; сдвиг адреса для пересылки 1 в следующий бит
                ;10430            BIT LS[BIT0] WITH WR[3],              ; проверка, установлен ли бит 0
U 0DB8, 5941,B5 ;10431          DT(LONG)&SET.ALU.CC                     ; (означает, что бит 31 уже был проверен) и установка
                ;10432                                                  ; кодов условий
U 0DB9, B8DC,11 ;10433          JMP.IF[BIT.SET] TO [TEST.T15.A]         ; переход на следующий тест, если выполнено
                ;10434            BIT LS[BIT15] WITH WR[3],             ; проверка, проверены ли биты 5-0
U 0DBA, 595F,B5 ;10435          DT(LONG)&SET.ALU.CC                     ; (если последний тест был с установленным битом 14,
                ;10436                                                  ; после сдвига будет установлен бит 15) и установка
```

```
                ;10437                                                  ; кодов условий
U 0DBB, 5B00,09 ;10438          SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит 15 не
                ;10439                                                  ; установлен
U 0DBC, 367F,95 ;10440          MOV LS[BIT31] TO WR[3]                  ; иначе установка бита 31 для следующего бита адреса и
                ;10441                                                  ; сброс бита 15
U 0DBD, BE13,95 ;10442          MOV WR[3] TO LS[T9]                     ; запоминание адреса
U 0DBE, 23C1,15 ;10443          ROL WR[2]                               ; адрес информации для печати при ошибке
U 0DBF, 3E89,15 ;10444          MOV WR[2] TO LS[ADDRESS.DATA]           ; запоминание в местной памяти
U 0DC0, 08DA,34 ;10445          JMP [REPEAT.T15.9]                      ; повторение, пока все биты адреса не будут проверены
                ;10446  TEST.T15.A:
U 0DC1, FF82,15 ;10447          INC LS[ERROR.NUMBER]                    ; ошибка A
U 0DC2, B640,15 ;10448          MOV LS[#1] TO WR[0]                     ; пересылка 1 в WR0
U 0DC3, BEFC,15 ;10449          MOV WR[0] TO LS[SIZE]                   ; загрузка регистра размера кодом для типа данных -
                ;10450                                                  ; слово
U 0DC4, 6512,15 ;10451          CLR LS[T9]                              ; очистка адреса буфера трансляции в местной памяти
                ;10452  BACK.T15.A:
U 0DC5, 9812,F5 ;10453          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи фона в буфер трансляции
U 0DC6, B272,15 ;10454          WRITE.MEM LS[BIT25]                     ; запись фона нулей, кроме бита BYTE OFFSET (который
                ;10455                                                  ; должен быть 1)
U 0DC7, 0A18,2C ;10456          JSR [NEXT.TB.ADDRESS]                   ; выборка следующего адреса трансляции
U 0DC8, 08DC,54 ;10457          JMP [BACK.T15.A]                        ; если здесь, не все адреса проверены и повторение
                ;10458                                                  ; иначе подпрограмма возвращает на следующую инструкцию
U 0DC9, E580,15 ;10459          CLR LS[CONTROL.STATUS]                  ; запрет печати "OTHER"
U 0DCA, 3653,15 ;10460          MOV LS[BIT9] TO WR[2]                   ; подготовка увеличения (увеличивает бит 0 адреса на
                ;10461                                                  ; микросхеме памяти)
U 0DCB, B653,95 ;10462          MOV LS[BIT9] TO WR[3]                   ; подготовка бита для переключения с максимальной
                ;10463                                                  ; скоростью. Первым переключаемым битом является бит 0
                ;10464                                                  ; памяти
                ;10465  REPEAT.T15.A.1:
U 0DCC, 6512,15 ;10466          CLR LS[T9]                              ; начальный адрес буфера трансляции - 0
                ;10467  REPEAT.T15.A.NEXT:
U 0DCD, 3612,15 ;10468          MOV LS[T9] TO WR[0]                     ; выборка адреса буфера трансляции
U 0DCE, 3736,95 ;10469          MOV LS[#7FFF8000] TO WR[1]              ; выборка битов признака для установки в WR1
U 0DCF, AEC2,15 ;10470          BIS WR[1] TO WR[0]                      ; адрес с полем признаков - все единицы
U 0DD0, BE14,15 ;10471          MOV WR[0] TO LS[T10]                    ; запоминание в местной памяти
                ;10472  LOOP.T15.A.1:
U 0DD1, 2F82,95 ;10473          CLR WR[1]                               ; ожидаемые данные
U 0DD2, 9813,75 ;10474          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]  ; подготовка стробирования CSR1 с TB MISS
U 0DD3, 3022,15 ;10475          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла (нет
                ;10476                                                  ; необходимости проверять эти данные)
U 0DD4, 9D45,75 ;10477          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ;
U 0DD5, 3022,15 ;10478          MOV MEM.DATA TO WR[0]                   ; проверка, сброшен ли TB MISS
U 0DD6, 0869,3C ;10479          JSR [CHECK.RESULT]                      ; проверка результата
U 0DD7, 08DD,14 ;10480          JMP [LOOP.T15.A.1]                      ; зацикливание при ошибке, если разрешено
U 0DD8, 9814,F5 ;10481          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]    ; подготовка записи дополнения в поле пизнаков
U 0DD9, B272,15 ;10482          WRITE.MEM LS[BIT25]                     ; запись единицы в BYTE OFFSET
                ;10483            ADD WR[3] TO LS[T9],                  ; переключение бита в адресе и
U 0DDA, EC13,D5 ;10484          DT(SIZE)&SET.ALU.CC                     ; установка кодов условий
U 0DDB, 08DC,D0 ;10485          JMP.IF[N.CLR] TO [REPEAT.T15.A.NEXT]    ; если бит 15 не установлен, повторение со следующим
                ;10486                                                  ; адресом
U 0DDC, EC13,15 ;10487          ADD WR[2] TO LS[T9]                     ; иначе увеличение адреса ниже переключенного бита
U 0DDD, B65E,15 ;10488          MOV LS[BIT15] TO WR[0]                  ; подготовка сброса бита 15 в ячейке 9 местной памяти
U 0DDE, 6F12,15 ;10489          XOR WR[0] TO LS[T9]                     ; сброс бита 15 в адресе
                ;10490            BIT WR[3] WITH LS[T9],                ; если переключенный бит уже установлен, выполнено и
U 0DDF, D913,B5 ;10491          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
```

```
U 0DE0, 08DC,D9 ;10492          JMP.IF[BITS.CLR] TO [REPEAT.T15.A.NEXT]; повторение теста с новым адресом, если не выполнено
U 0DE1, 3612,15 ;10493          MOV LS[T9] TO WR[0]                    ; выборка адреса из местной памяти
U 0DE2, C528,15 ;10494          BIC LS[#FFFF] TO WR[0]                 ; очистка битов с 9 по 14 (вместе с другими, которые уже
                ;10495                                                 ; сброшены)
                ;10496            XOR LS[BIT31] TO WR[0],              ; переключение бита 31 и
U 0DE3, 437E,35 ;10497          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 0DE4, BE12,15 ;10498          MOV WR[0] TO LS[T9]                    ; пересылка нового адреса в ячейку 9 местной памяти
U 0DE5, 88DC,D8 ;10499          JMP.IF[N.SET] TO [REPEAT.T15.A.NEXT]   ; если бит 31 был сброшен, повторение, используя тот
                ;10500                                                 ; самый переключенный бит, но с установленным битом 31,
                ;10501                                                 ; иначе подготовка запуска уменьшения адресов
U 0DE6, 6512,15 ;10502          CLR LS[T9]                             ; запуск от адреса 0 буфера трансляции (будет дополнение
                ;10503                                                 ; для запуска от верхнего адреса)
                ;10504  REPEAT.T15.A.NEXT1:
U 0DE7, 7D12,15 ;10505          COM LS[T9]                             ; дополнение адреса для получения уменьшающихся адресов
U 0DE8, 3612,15 ;10506          MOV LS[T9] TO WR[0]                    ; выборка текущего адреса
U 0DE9, 3736,95 ;10507          MOV LS[#7FFF8000] TO WR[1]             ; выборка в WR1 битов признака для очистки
U 0DEA, AF42,15 ;10508          BIC WR[1] TO WR[0]                     ; очистка битов признака в пределах адреса буфера
                ;10509                                                 ; трансляции
U 0DEB, BE14,15 ;10510          MOV WR[0] TO LS[T10]                   ; запоминание в местной памяти
                ;10511  LOOP.T15.A.2:
U 0DEC, 2F82,95 ;10512          CLR WR[1]                              ; ожидаемые данные
U 0DED, 9813,75 ;10513          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG] ; подготовка стробирования CSR с TB MISS
U 0DEE, 3022,15 ;10514          MOV MEM.DATA TO WR[0]                  ; выдача инструкции MOV для завершения цикла (нет
                ;10515                                                 ; необходимости проверять эти данные)
U 0DEF, 9D45,75 ;10516          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; подготовка чтения CSR1
U 0DF0, 3022,15 ;10517          MOV MEM.DATA TO WR[0]                  ; выборка результата из CSR1
U 0DF1, 0869,3C ;10518          JSR [CHECK.RESULT]                     ; проверка, сброшен ли TB MISS
U 0DF2, 88DE,C4 ;10519          JMP [LOOP.T15.A.2]                     ; зацикливание при ошибке, если разрешено
U 0DF3, 9814,F5 ;10520          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]   ; подготовка записи дополнения в признак
U 0DF4, B272,15 ;10521          WRITE.MEM LS[BIT25]                    ; запись 1 в BYTE OFFSET
U 0DF5, 7D12,15 ;10522          COM LS[T9]                             ; восстановление адреса для получения оригинала
                ;10523            ADD WR[3] TO LS[T9],                 ; переключение бита в адресе и
U 0DF6, EC13,D5 ;10524          DT(SIZE)&SET.ALU.CC                    ; установка кодов условий
U 0DF7, 88DE,70 ;10525          JMP.IF[N.CLR] TO [REPEAT.T15.A.NEXT1]  ; если бит 15 не установлен, повторение со следующим
                ;10526                                                 ; адресом
U 0DF8, EC13,15 ;10527          ADD WR[2] TO LS[T9]                    ; иначе увеличение адреса ниже переключенного бита
U 0DF9, B65E,15 ;10528          MOV LS[BIT15] TO WR[0]                 ; подготовка сброса бита 15 в ячейке 9 местной памяти
U 0DFA, 6F12,15 ;10529          XOR WR[0] TO LS[T9]                    ; сброс бита 15 в адресе
                ;10530            BIT WR[3] WITH LS[T9],               ; если переключенный бит уже установлен, выполнено и
U 0DFB, D913,B5 ;10531          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 0DFC, 88DE,79 ;10532          JMP.IF[BITS.CLR] TO [REPEAT.T15.A.NEXT1] ; повторение теста с новым адресом, если не выполнено
U 0DFD, 3612,15 ;10533          MOV LS[T9] TO WR[0]                    ; выборка адреса из местной памяти
U 0DFE, C528,15 ;10534          BIC LS[#FFFF] TO WR[0]                 ; очистка битов с 9 по 14 (вместе с другими, которые уже
                ;10535                                                 ; сброшены)
                ;10536            XOR LS[BIT31] TO WR[0],              ; переключение бита 31 и
U 0DFF, 437E,35 ;10537          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 0E00, BE12,15 ;10538          MOV WR[0] TO LS[T9]                    ; пересылка нового адреса в ячейку 9 местной памяти
U 0E01, 08DE,78 ;10539          JMP.IF[N.SET] TO [REPEAT.T15.A.NEXT1]  ; если бит 31 сброшен, повторение, используя тот самый
                ;10540                                                 ; переключенный бит, но с установленным битом 31
U 0E02, A3C1,95 ;10541          ROL WR[3]                              ; следующий бит для переключения с максимальной
                ;10542                                                 ; скоростью
                ;10543            BIT LS[BIT15] WITH WR[3],            ; проверка, установлен ли бит 15 и
U 0E03, 595F,B5 ;10544          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 0E04, 88DC,C9 ;10545          JMP.IF[BITS.CLR] TO [REPEAT.T15.A.1]   ; повторение теста с новым пеключенным битом, если бит
                ;10546                                                 ; 15 не был установлен, иначе подготовка переключения
```

```
                ;10547                                          ; следующего бита 31
U 0E05, 367F,95 ;10548          MOV LS[BIT31] TO WR[3]           ; подготовка бита для переключения с максимальной
                ;10549                                          ; скоростью. Сейчас переключаемым битом является бит 6
                ;10550                                          ; памяти
                ;10551  REPEAT.T15.A1.1:
U 0E06, 6512,15 ;10552          CLR LS[T9]                       ; начальный адрес буфера трансляции - 0
                ;10553  REPEAT.T15.A1.NEXT:
U 0E07, 3612,15 ;10554          MOV LS[T9] TO WR[0]              ; выборка адреса буфера трансляции
U 0E08, 3736,95 ;10555          MOV LS[#7FFF8000] TO WR[1]       ; выборка битов признака для установки в WR1
U 0E09, AEC2,15 ;10556          BIS WR[1] TO WR[0]               ; адрес с полем признаков - все единицы
U 0E0A, BE14,15 ;10557          MOV WR[0] TO LS[T10]             ; запоминание в местной памяти
                ;10558  LOOP.T15.A1.1:
U 0E0B, 2F82,95 ;10559          CLR WR[1]                        ; ожидаемые данные
U 0E0C, 9813,75 ;10560          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG] ; подготовка стробирования CSR1 с TB MISS
U 0E0D, 3022,15 ;10561          MOV MEM.DATA TO WR[0]            ; выдача инструкции MOV для завершения цикла (нет
                ;10562                                          ; необходимости проверять эти данные)
U 0E0E, 9D45,75 ;10563          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG] ;
U 0E0F, 3022,15 ;10564          MOV MEM.DATA TO WR[0]            ; проверка, сброшен ли TB MISS
U 0E10, 0869,3C ;10565          JSR [CHECK.RESULT]               ; проверка результата
U 0E11, 88E0,B4 ;10566          JMP [LOOP.T15.A1.1]              ; зацикливание при ошибке, если разрешено
U 0E12, 9814,F5 ;10567          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG] ; подготовка записи дополнения в признак
U 0E13, B272,15 ;10568          WRITE.MEM LS[BIT25]              ; запись 1 в BYTE OFFSET
                ;10569            XOR WR[3] TO LS[T9],           ; переключение бита в адресе и
U 0E14, EF13,B5 ;10570          DT(LONG)&SET.ALU.CC              ; установка кодов условий
U 0E15, 88E0,78 ;10571          JMP.IF[N.SET] TO [REPEAT.T15.A1.NEXT] ; если бит 31 не сброшен, повторение со следующим
                ;10572                                          ; адресом
                ;10573            ADD WR[2] TO LS[T9],           ; иначе увеличение адреса ниже переключаемого
U 0E16, 6C13,55 ;10574          DT(SIZE)&SET.ALU.CC              ; бита и установка кодов условий
U 0E17, 08E0,70 ;10575          JMP.IF[N.CLR] TO [REPEAT.T15.A1.NEXT] ; повторение теста с новым адресом, если не выполнено
U 0E18, 6512,15 ;10576          CLR LS[T9]                       ; запуск от адреса буфера трансляции (будет дополнение
                ;10577                                          ; для формирования верхнего адреса)
                ;10578  REPEAT.T15.A1.NEXT1:
U 0E19, 7D12,15 ;10579          COM LS[T9]                       ; дополнение адреса для формирования уменьшающихся
                ;10580                                          ; адресов
U 0E1A, 3612,15 ;10581          MOV LS[T9] TO WR[0]              ; выборка текущего адреса
U 0E1B, 3736,95 ;10582          MOV LS[#7FFF8000] TO WR[1]       ; выборка в WR1 битов признака для очистки
U 0E1C, AF42,15 ;10583          BIC WR[1] TO WR[0]               ; очистка битов признака в пределах адреса буфера
                ;10584                                          ; трансляции
U 0E1D, BE14,15 ;10585          MOV WR[0] TO LS[T10]             ; запоминание в местной памяти
                ;10586  LOOP.T15.A1.2:
U 0E1E, 2F82,95 ;10587          CLR WR[1]                        ; ожидаемые данные
U 0E1F, 9813,75 ;10588          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG] ; подготовка стробирования CSR с TB MISS
U 0E20, 3022,15 ;10589          MOV MEM.DATA TO WR[0]            ; выдача инструкции MOV для завершения цикла (нет
                ;10590                                          ; необходимости проверять эти данные)
U 0E21, 9D45,75 ;10591          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG] ; подготовка чтения CSR1
U 0E22, 3022,15 ;10592          MOV MEM.DATA TO WR[0]            ; выборка результата из CSR1
U 0E23, 0869,3C ;10593          JSR [CHECK.RESULT]               ; проверка, сброшен ли TB MISS
U 0E24, 08E1,E4 ;10594          JMP [LOOP.T15.A1.2]              ; зацикливание при ошибке, если разрешено
U 0E25, 9814,F5 ;10595          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG] ; подготовка записи дополнения в признак
U 0E26, B272,15 ;10596          WRITE.MEM LS[BIT25]              ; запись 1 в BYTE OFFSET
U 0E27, 7D12,15 ;10597          COM LS[T9]                       ; восстановление адреса для получения оригинала
                ;10598            XOR WR[3] TO LS[T9],           ; переключение бита в адресе и
U 0E28, EF13,B5 ;10599          DT(LONG)&SET.ALU.CC              ; установка кодов условий
U 0E29, 88E1,98 ;10600          JMP.IF[N.SET] TO [REPEAT.T15.A1.NEXT1] ; если бит 31 не сброшен, повторение со следующим
                ;10601                                          ; адресом
```

```
                 ;10602             ADD WR[2] TO LS[T9],                    ; иначе увеличение адреса ниже переключаемого
U 0E2A, 6C13,55  ;10603           DT(SIZE)&SET.ALU.CC                      ; бита и установка кодов условий
U 0E2B, 08E1,90  ;10604           JMP.IF[N.CLR] TO [REPEAT.T15.A1.NEXT1]   ; повторение теста с новым адресом, если не
                 ;10605                                                    ; выполнено, иначе выполнено
U 0E2C, FF82,15  ;10606           INC LS[ERROR.NUMBER]                     ; ошибка B
U 0E2D, B670,15  ;10607           MOV LS[OTHER.DATA] TO WR[0]              ; установка бита в WR0 для печати под OTHER
U 0E2E, 3E80,15  ;10608           MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита в слове управления
U 0E2F, 6588,15  ;10609           CLR LS[ADDRESS.DATA]                     ; очистка адреса для печати при ошибке
U 0E30, 989C,F5  ;10610           MEM.REQ[WRITE.TB] ADRS[ZERO] DT[LONG]    ; подготовка записи в буфер трансляции по адресу 0
                 ;10611                                                    ; нулей в поле признаков
U 0E31, B272,15  ;10612           WRITE.MEM LS[BIT25]                      ; запись в буфер трансляции 1 в BYTE OFFSET
                 ;10613   LOOP.T15.B:
U 0E32, 9B63,F5  ;10614           MEM.REQ[WRITE.UBS.MAP] ADRS[BIT17] DT[LONG] ; подготовка записи в UBS MAP (память отображения
                 ;10615                                                    ; общей шины). Признак в памяти признаков по адресу 0
                 ;10616                                                    ; должен иметь установленый признак 2, если запрет записи
                 ;10617                                                    ; в память отображения общей шины работает неправильно
U 0E33, B272,15  ;10618           WRITE.MEM LS[BIT25]                      ; установка BYTE OFFSET в буфере трансляции
U 0E34, 2F82,95  ;10619           CLR WR[1]                                ; ожидаемые данные (TB MISS сброшен)
U 0E35, 989D,75  ;10620           MEM.REQ[TEST.V.RCHK] ADRS[ZERO] DT[LONG] ; стробирование CSR результирующим TB MISS
U 0E36, 3022,15  ;10621           MOV MEM.DATA TO WR[0]                    ; выдача инструкции MOV для завершения цикла (нет
                 ;10622                                                    ; необходимости проверять эти данные)
U 0E37, 9D45,75  ;10623           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; чтение CSR1
U 0E38, 3022,15  ;10624           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 0E39, 0869,3C  ;10625           JSR [CHECK.RESULT]                       ; проверка, сброшен ли TB MISS
U 0E3A, 88E3,24  ;10626           JMP [LOOP.T15.B]                         ; зацикливание при ошибке, если разрешено
U 0E3B, FF82,15  ;10627           INC LS[ERROR.NUMBER]                     ; ошибка C
U 0E3C, 989C,F5  ;10628           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]      ; подготовка записи в буфер трансляции по адресу 0
                 ;10629                                                    ; нулей в поле признаков
U 0E3D, B29C,15  ;10630           WRITE.MEM LS[#0]                         ; запись в буфер трансляции нулей во все биты, включая и
                 ;10631                                                    ; бит BYTE OFFSET
                 ;10632   LOOP.T15.C:
U 0E3E, 989D,75  ;10633           MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]   ; подготовка стробирования CSR1 с TB MISS
U 0E3F, 3022,15  ;10634           MOV MEM.DATA TO WR[0]                    ; выдача инструкции MOV для завершения цикла (нет
                 ;10635                                                    ; необходимости проверять эти данные)
U 0E40, B66A,95  ;10636           MOV LS[TB.MISS] TO WR[1]                 ; ожидаемые данные (TB MISS установлен, так как BYTE
                 ;10637                                                    ; OFFSET сброшен)
U 0E41, 9D45,75  ;10638           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; подготовка чтения CSR1
U 0E42, 3022,15  ;10639           MOV MEM.DATA TO WR[0]                    ; выборка данных CSR1
U 0E43, 0869,3C  ;10640           JSR [CHECK.RESULT]                       ; проверка, установлен ли бит 21 (TB MISS возбужден)
U 0E44, 88E3,E4  ;10641           JMP [LOOP.T15.C]                         ; зацикливание при ошибке, если разрешено
                 ;10642   END.T15:
```

```
;10643  .PAGE "ТЕСТЫ ПАРИТЕТА ТВ И БИТА VALID CSR1"
;10644  .TOC "ТЕСТ 16 - тест паритета буфера трансляции, включая бит ТВ P0 (модуль МСТ)"
;10645  ;
;10646  ; ОПИСАНИЕ ТЕСТА:
;10647  ;
;10648  ;       Этот тест проверяет схемы паритета буфера трансляции и непроверенный бит
;10649  ;    памяти буфера трансляции - ТВ P0 L. Бит ТВ P0 L в памяти не может быть прочи-
;10650  ;    тан непосредственно, как другие биты, так что он должен быть проверен установ-
;10651  ;    кой или сбросом в каждой ячейке и проверкой, что ошибка паритета не появляет-
;10652  ;    ся при чтении буфера трансляции. Схемы паритета должны проверяться первыми.
;10653  ;    Бит ТВ P0 L формирует ПМЛ UB CSR2 при записи в буфер трансляции. Для формиро-
;10654  ;    вания ТВ P0 L используются входы VALID H, GEN P0 H и PROT PAR H. Вход VALID H
;10655  ;    поступает непосредственно из памяти буфера трансляции. Вход GEN P0 H поступает
;10656  ;    из пары генераторов паритета, входами которых являются биты PA из памяти, BYTE
;10657  ;    OFFSET из памяти и ТВ PAR DIAG H (бит диагностики для принудительной установки
;10658  ;    ошибки паритета). Вход PROT PAR H поступает из ПЗУ, входами которой являются
;10659  ;    PROT A по PROT D и MODIFY H. Вход ТВ PAR ERR формирует ПМЛ UB CSR2 из GEN P0,
;10660  ;    PROT PAR, VALID и ТВ P0 и передает на ПМЛ CSR 1A. Функция TEST.V.RCHK использу-
;10661  ;    ется для стобирования CSR и, таким образом, этот бит может быть проверен. Путь
;10662  ;    данных функции TEST.V.RCHK описан в предыдущем тесте промаха в буфере трансля-
;10663  ;    ции.
;10664  ;       В первую очередь неоьходимо проверить сигнал ТВ PAR ERR записью в буфер
;10665  ;    трансляции всех нулей и выдачей инструкции MEM.REQ с MF=TEST.V.RCHK. Тогда чи-
;10666  ;    тается CSR1 и проверяется низкий уровень бита 15 (ТВ PAR ERR H). Затем повторя-
;10667  ;    ется функция TEST.V.RCHK с установленным битом 29 (ТВ PAR DIAG H) в CSR1 и прове-
;10668  ;    ряется высокий уровень ТВ PAR ERR H. Это повторяется со всеми единицами и тогда с
;10669  ;    данными, содержащими единицу, сдвигаемую через все биты памяти буфера трансляции.
;10670  ;      Бит памяти ТВ P0 должен быть проверен во всех ячейках записью 1 или 0 и про-
;10671  ;    веркой, что ошибка паритета не появляется. Простой тест "МАРШ" заканчивает
;10672  ;    проверку.
;10673  ;
;10674  ; ПРЕДПОЛОЖЕНИЯ:
;10675  ;
;10676  ;    Предполагается, что все предыдущие тесты выполнены успешно.
;10677  ;
;10678  ; ШАГИ ТЕСТА:
;10679  ;
;10680  ;    1) Установка номера ошибки и номера модуля в местной памяти (для распечатки
;10681  ;       ошибок) и очистка предыдущего номера ошибки в местной памяти. Также уста-
;10682  ;       новка бита MME в CSR1 и очистка других битов.
;10683  ;    2) Установка маски ошибки для проверки только бита 15 (CSR TB PAR ERR).
;10684  ;    3) Выполнение инструкции MEM.REQ с MF=WRITE.TB и DT=LONGWORD. В качестве
;10685  ;       адреса используется ячейка местной памяти, содержащая все нули. Выдача
;10686  ;       WRITE.MEM LS с данными - все нули.
;10687  ;    4) Очистка WR1.
;10688  ;    5) Выполнение MEM.REQ с MF=TEST.V.RCHK и DT=LONGWORD. В качестве адреса ис-
;10689  ;       пользуется ячека местной памяти, содержащая все нули. Выдача инструкции
;10690  ;       MOV.MEM.DATA TO WR[0] для завершения функции TEST.V.RCHK (нет необходимос-
;10691  ;       ти проверять данные, поступающие обратно).
;10692  ;    6) Выдача инструкции MEM.REQ с MF=READ.CSR и MOV MEM.DATA TO WR[0] для чте-
;10693  ;       ния CSR1. Проверка бита 15 на 0 (нет ошибки паритета).
;10694  ;    7) Запись 1 в бит 29 CSR1 (ТВ PAR DIAG H) для принудительной установки непра-
;10695  ;       вильного паритета.
;10696  ;    8) Повторение шагов 5 и 6, проверяя бит 15 на 1 (ошибка паритета).
;10697  ;    9) Повторение шагов с 3 по 8 с данными буфера трансляции - все единицы в ша-
```

```
;10698  ;       ге 3.
;10699  ;   10) Повторение шагов с 3 по 8 с данными буфера трансляции - сдвигаемая единица
;10700  ;       в шаге 3, пока сдвигаемая единица не побывает на всех выходах бухера тран-
;10701  ;       сляции.
;10702  ;   11) Повторение шагов с 3 по 8 с данными буфера трансляции - 1 в бите 09 PA и 1
;10703  ;       в бите PROT A.
;10704  ;   12) Очистка диагностического бита паритета буфера трансляции в CSR1. Загрузка
;10705  ;       нулями ячейки временного хранения местной памяти, которая используется
;10706  ;       в качестве адреса памяти буфера трансляции. Очистка WR1 (ожидаемые данные).
;10707  ;   13) Выполнение инструкции MEM.REQ с MF=WRITE.TB и DT=LONGWORD. Использование
;10708  ;       в качестве адреса ячейки временного хранения местной памяти. Выдача инст-
;10709  ;       рукции WRITE.MEM LS с данными - все нули.
;10710  ;   14) Выполнение MEM.REQ с MF=TEST.V.RCHK и DT=LONGWORD. Использование в качестве
;10711  ;       адреса ячейки временного хранения местной памяти. Выдача инструкции MOV MEM.
;10712  ;       DATA TO WR[0] для завершения функции TEST.V.RCHK (нет необходимости про-
;10713  ;       верять данные, поступающие обратно).
;10714  ;   15) Выдача инструкции MEM.REQ с MF=READ.CSR и MOV MEM.DATA TO WR[0] для чтения
;10715  ;       CSR1. Проверка бита 15 на нуль (нет ошибки паритета).
;10716  ;   16) Повторение шагов с 13 по 15, увеличивая ячейку временного хранения местной
;10717  ;       памяти для обращения ко всем адресам.
;10718  ;   17) Повторение шагов с 13 по 16 с 1 в бите 09 PA в качестве тестовых данных для
;10719  ;       записи в память. Этим TB P0 L принудительно устанавливается в единицу.
;10720  ;   18) Запуск простого набора "МАРШ", используя шаги с 13 по 16. Выполнение пос-
;10721  ;       ледовательности чтение/запись/чтение тестового набора с проверкой чтения
;10722  ;       на отсутствие ошибки паритета.
;10723  ;
;10724  ;
;10725  ; ОШИБКИ:
;10726  ;
;10727  ;     ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными является бит TB PAR ERR в CSR1.
;10728  ;
;10729  ;     ошибка 1 - TB PAR ERR был возбужден, когда не было ошибки паритета.
;10730  ;                Данные буфера трансляции - все нули.
;10731  ;     ошибка 2 - неправильно работает TB PAR ERR при ошибке паритета буфера тран-
;10732  ;                сляции. Данные буфера трансляции - все нули.
;10733  ;     ошибка 3 - TB PAR ERR был возбужден, когда не было ошибки паритета.
;10734  ;                Данные буфера трансляции - все единицы.
;10735  ;     ошибка 4 - неправильно работает TB PAR ERR при ошибке паритета буфера тран-
;10736  ;                сляции. Данные буфера трансляции - все единицы.
;10737  ;
;10738  ;     ПРИМЕЧАНИЕ: При ошибках с 5 по a данные, записанные в буфер трансляции, в
;10739  ;                сообщении об ошибке печатаются под OTHER. Биты 0-14 поступают
;10740  ;                на биты 09-23 PA соответственно. Биты 25-30 поступают на биты
;10741  ;                BYTE OFFSET, MODIFY, PROT A, PROT B, PROT C, и PROT D соответст-
;10742  ;                венно.
;10743  ;
;10744  ;     ошибка 5 - TB PAR ERR был возбужден, когда не было ошибки паритета.
;10745  ;                Данные буфера трансляции - сдвигаемая единица в битах PA и BYTE
;10746  ;                OFFSET. См. примечание выше.
;10747  ;     ошибка 6 - неправильно работает TB PAR ERR при ошибке паритета буфера тран-
;10748  ;                сляции. Данные буфера трансляции - сдвигаемая единица в битах PA
;10749  ;                и BYTE OFFSET. См. примечание выше.
;10750  ;     ошибка 7 - TB PAR ERR был возбужден, когда не было ошибки паритета.
;10751  ;                Данные буфера трансляции - сдвигаемая единица в битах PROT и
;10752  ;                MODIFY. См. примечание выше.
```

```
;10753  ;     ошибка В - неправильно работает TB PAR ERR при ошибке паритета буфера тран-
;10754  ;                сляции. Данные буфера трансляции - сдвигаемая единица в битах
;10755  ;                PROT, MODIFY. См. примечание выше.
;10756  ;     ошибка 9 - ТВ PAR ERR был возбужден, когда не было ошибки паритета.
;10757  ;                Данные буфера трансляции - 1 в одном бите РА и 1 в одном бите
;10758  ;                PROT. См. примечание выше.
;10759  ;     ошибка А - неправильно работает TB PAR ERR при ошибке паритета буфера тран-
;10760  ;                сляции. Данные буфера трансляции - 1 в одном бите РА и 1 в одном
;10761  ;                бите PROT. См. примечание выше.
;10762  ;     ошибка В - неправильно работает память буфера трансляции при попытке запи-
;10763  ;                сать 0 в ТВ Р0 L. Адрес памяти печатается под OTHER.
;10764  ;     ошибка С - неправильно работает память буфера трансляции при попытке запи-
;10765  ;                сать 1 в ТВ Р0 L. Адрес памяти печатается под OTHER.
;10766  ;     ошибка D - неправильно работает память буфера трансляции при тесте "МАРШ".
;10767  ;                Адрес памяти печатается под OTHER.
;10768  ;
;10769  ; НАЛАДКА:
;10770  ;
;10771  ;        ОШИБКА 1 - Эта ошибка указывает на возможную неисправность ПМЛ UB CSR2,
;10772  ;     ПМЛ CSR 1 или других схем паритета. Проверку лучше всего начать от ПМЛ CSR2.
;10773  ;     центральный процессор необходимо остановить на инструкции MOV MEM.DATA TO
;10774  ;     WR[0], которая следует за запросом памяти с функцией TEST.V.RCHK. Микропрог-
;10775  ;     рамма памяти должна циклиться в ожидании снятия CPU GRANT. Необходимо прове-
;10776  ;     рить высокий уровень сигнала TB PAR ERR L на выходе ПМЛ UB CSR2. Если высо-
;10777  ;     кий, необходимо проверить на входе ПМЛ CSR 1A. Если высокий на входе ПМЛ CSR
;10778  ;     1A, по-видимому, неисправна ПМЛ CSR 1A.
;10779  ;        Если сигнал TB PAR ERR L низкий, необходимо проверить низкие уровни сиг-
;10780  ;     налов GEN P0 H, PROT PAR H, TB P0 L и VALID H на входах ПМЛ UB CSR2. Если
;10781  ;     здесь правильно, по-видимому, неисправна ПМЛ UB CSR2.
;10782  ;       Если сигнал GEN P0 H высокий, необходимо проверить его на генераторе пари-
;10783  ;     тета. Если здесь высокий, тогда необходимо проверить низкие уровни на всех
;10784  ;     входах. Если вход от другово генератора паритета высокий, необходимо прове-
;10785  ;     рить низкие уровни на всех его входах. Если входы правильные, подозревается
;10786  ;     сам генератор паритета.
;10787  ;        Если сигнал PROT PAR H высокий, необходимо проверить его на выходе ПЗУ,
;10788  ;     которое формирует этот сигнал. Если здесь высокий, необходимо проверить
;10789  ;     низкие уровни на входах PROT A по PROT D и MODIFY H. Если входы низкие, по-
;10790  ;     видимому, неисправна ПЗУ.
;10791  ;        Если ТВ Р0 L высокий, необходимо проверить его на выходе памяти. Неисправ-
;10792  ;     ность здесь может означать, что неисправна память или ПМЛ UB CSR2, которая
;10793  ;     формирует этот бит при записи. Если пошаговый режим МСТ возможен, централь-
;10794  ;     ный процессор необходимо остановить на MEM.REQ с MF=WRITE.TB, разрешить поша-
;10795  ;     говый режим МСТ и остановить центральный процессор на инструкции WRITE.MEM LS.
;10796  ;     Далее остановить МСТ на цикле, который возбуждает TB WE L и проверить низ-
;10797  ;     кий уровень сигнала ТВ Р0 L на входе памяти. Если здесь правильно, тогда не-
;10798  ;     исправна память. Если нет, проверяется, что TB WE L низкий на ПМЛ UB CSR2
;10799  ;     вместе с низкими VALID H, GEN P0 H и PROT PAR H. Если здесь правильно, выход
;10800  ;     ТВ Р0 L должен быть низким. Иначе неисправна ПМЛ.
;10801  ;
;10802  ;        ОШИБКА 2 - Эта ошибка указывает на возможную неисправность ПМЛ UB CSR2,
;10803  ;     входа генерации паритета GEN P0 H или входа TB PAR DIAG H. Центральный про-
;10804  ;     цессор необходимо остановить на инструкции MOV MEM.DATA TO WR[0], которая
;10805  ;     следует за запросом памяти с функцией TEST.V.RCHK. Микропрограмма памяти долж-
;10806  ;     на циклиться в ожидании снятия CPU GRANT. Проверяется низкий уровень сигнала
;10807  ;     TB PAR ERR L на выходе UB CSR 2. Если низкий, необходимо проверить его на вхо-
```

```
                ;10808  ;    де ПМЛ CSR 1A. Если на ПМЛ CSR 1A высокий, по-видимому, неисправна ПМЛ CSR 1A.
                ;10809  ;       Если TB PAR ERR L высокий, необходимо проверить высокий уровень на входе
                ;10810  ;    GEN P0 H ПМЛ UB CSR 2. Другие входы такие, как и при ошибке 1, так, что они
                ;10811  ;    по-видимому, правильные. Если GEN P0 L высокий, подозревается ПМЛ UB CSR2.
                ;10812  ;       Если GEN P0 L низкий, необходимо проверить высокий уровень TB PAR DIAG H,
                ;10813  ;    поступающего на генератор паритета. Если здесь правильно, выход этого ге-
                ;10814  ;    нератора должен быть высоким и выход второго генератора паритета GEN P0 H
                ;10815  ;    должен быть высоким. Иначе генератор паритета, по-видимому, неисправен.
                ;10816  ;
                ;10817  ;      ОШИБКА 3 - Эта ошибка аналогична ошибке 1 с тем отличием, что все входы
                ;10818  ;    генератора паритета (за исключением TB PAR DIAG H) и ПЗУ PROT PAR высокие.
                ;10819  ;    Другими сигналами, которые отличаются, является PROT PAR H, который должен
                ;10820  ;    быть высоким и VALID H, который должен быть высоким.
                ;10821  ;
                ;10822  ;       ОШИБКА 4 - Скорее всего, подозревается ПМЛ UB CSR 2.
                ;10823  ;
                ;10824  ;       ОШИБКА 5 - Эта ошибка аналогична ошибке 1 с тем отличием, что один вход
                ;10825  ;    PA или вход BYTE OFFSET генератора паритета должны быть высокими. Сигналами
                ;10826  ;    которые отличаются, являются GEN P0 H, который должен быть высоким и TB P0 L,
                ;10827  ;    который должен быть высоким.
                ;10828  ;
                ;10829  ;      ОШИБКА 6 - Скорее всего, подозревается ПМЛ UB CSR2.
                ;10830  ;
                ;10831  ;      ОШИБКА 7 - Скорее всего, подозревается ПЗУ защиты.
                ;10832  ;
                ;10833  ;      ОШИБКА B ПО A - Скорее всего, подозревается ПМЛ UB CSR2.
                ;10834  ;
                ;10835  ;      ОШИБКИ C В ПО D - Скорее всего подозревается память буфера трансляции.
                ;10836  ;    Неисправный адрес печатается под OTHER.
                ;10837  T.16:
U 0E45, B65E,15 ;10838          MOV LS[BEGIN.TEST] TO WR[0]          ; установка бита 15 в WR[0] для слова управления и
                ;10839                                               ; состояния
U 0E46, 3E80,15 ;10840          MOV WR[0] TO LS[CONTROL.STATUS]       ; установка бита 15 в слове управления и состояния. Бит
                ;10841                                               ; 15 указывает начало теста консольному процессору
U 0E47, 10E0,15 ;10842          MISC [SET.CP.ATTN]                   ; выдача сигнала CPU ATTN консольному процессору
                ;10843  WAIT.T16.0:
U 0E48, 08E4,84 ;10844          JMP [WAIT.T16.0]                     ; зацикливание для ожидания ответа консольного
                ;10845                                               ; процессора
U 0E49, 0A1A,AC ;10846          JSR [SETUP.1]                        ; установка масок, кода модуля и др.
U 0E4A, DF5E,15 ;10847          MCOM LS[TB.PAR.ERR] TO WR[0]          ; сброс бита 15 и установка других битов рабочего
                ;10848                                               ; регистра
U 0E4B, 3E8A,15 ;10849          MOV WR[0] TO LS[ERROR.MASK]           ; проверка только бита 15 (ошибка паритета буфера
                ;10850                                               ; трансляции)
U 0E4C, B640,15 ;10851          MOV LS[#1] TO WR[0]                  ; 1 в WR0
U 0E4D, BEFC,15 ;10852          MOV WR[0] TO LS[SIZE]                ; подготовка регистра размера для типа данных=слово
U 0E4E, 6512,15 ;10853          CLR LS[T9]                           ; адрес буфера трансляции для записи данных
                ;10854  T16.1:
U 0E4F, 6514,15 ;10855          CLR LS[T10]                          ; данные для записи в буфер трансляции - все нули
U 0E50, 08EA,BC ;10856          JSR [T16.NO.PAR.ERR]                 ; переход для проверки отсутствия ошибки паритета
                ;10857  T16.2:
U 0E51, FF82,15 ;10858          INC LS[ERROR.NUMBER]                 ; ошибка 2
U 0E52, 08EB,6C ;10859          JSR [T16.PAR.ERR]                    ; переход для проверки наличия ошибки паритета
                ;10860  T16.3:
U 0E53, FF82,15 ;10861          INC LS[ERROR.NUMBER]                 ; ошибка 3
U 0E54, 7D14,15 ;10862          COM LS[T10]                          ; тестовые данные для записи в буфер трансляции - все
```

```
                ;10863                                                  ; единицы
U 0E55, 08EA,BC ;10864          JSR [T16.NO.PAR.ERR]                    ; переход для проверки отсутствия ошибки паритета
                ;10865  T16.4:
U 0E56, FF82,15 ;10866          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 0E57, 08EB,6C ;10867          JSR [T16.PAR.ERR]                       ; переход для проверки наличия ошибки паритета
U 0E58, FF82,15 ;10868          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 0E59, B670,15 ;10869          MOV LS[OTHER.DATA] TO WR[0]             ; установка бита для указания консольному процессору
                ;10870                                                  ; печатать под OTHER
U 0E5A, 3E80,15 ;10871          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка этого бита в слове управления и состояния
U 0E5B, 3641,15 ;10872          MOV LS[#1] TO WR[2]                     ; пересылка 1 в WR2 (поступит на PA 09)
                ;10873  REPEAT.T16.5:
U 0E5C, 3E89,15 ;10874          MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка данных для печати под OTHER в сообщении об
                ;10875                                                  ; ошибке
U 0E5D, BE21,15 ;10876          MOV WR[2] TO LS[10]                     ; загрузка ячейки временного хранения местной памяти
                ;10877                                                  ; тестовые данные со сдвигаемой единицей (начинаются с LVA 0
                ;10878                                                  ; который поступает на PA 9)
U 0E5E, 08EA,BC ;10879          JSR [T16.NO.PAR.ERR]                    ; переход для проверки отсутствия ошибки паритета
U 0E5F, 23C1,15 ;10880          ROL WR[2]                               ; следующие тестовые данные
                ;10881            BIT LS[BIT26] WITH WR[2],             ; проверка, установлен ли бит 26 (означает, что проверен
                ;10882                                                  ; BYTE OFFSET (бит 25))
U 0E60, 5975,35 ;10883          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E61, 08E6,61 ;10884          JMP.IF[BIT.SET] TO [T16.6]              ; переход на следующую проверку, если проверен BYTE
                ;10885                                                  ; OFFSET, иначе проверка, что прошли все биты PA
                ;10886            BIT LS[BIT15] WITH WR[2],             ; проверка, установлен ли бит 15 (если установлен,
                ;10887                                                  ; означает, что PA с 09 по 23 проверены)
U 0E62, D95F,35 ;10888          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E63, 5B00,09 ;10889          SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит 15 не
                ;10890                                                  ; установлен,
U 0E64, B673,15 ;10891          MOV LS[BIT25] TO WR[2]                  ; иначе установка бита BYTE OFFSET
U 0E65, 08E5,C4 ;10892          JMP [REPEAT.T16.5]                      ; повторение с новыми тестовыми данными
                ;10893  T16.6:
U 0E66, 3641,15 ;10894          MOV LS[#1] TO WR[2]                     ; пересылка 1 в WR2 (поступает на PA 09)
U 0E67, FF82,15 ;10895          INC LS[ERROR.NUMBER]                    ; ошибка 6
                ;10896  REPEAT.T16.6:
U 0E68, 3E89,15 ;10897          MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка данных для печати под OTHER в сообщении об
                ;10898                                                  ; ошибке
U 0E69, BE21,15 ;10899          MOV WR[2] TO LS[10]                     ; загрузка ячейки временного хранения местной памяти
                ;10900                                                  ; набором со сдвигаемой единицей (начинается от LVA 0,
                ;10901                                                  ; который поступает на PA 9)
U 0E6A, 9814,F5 ;10902          MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]    ; подготовка записи тестовых данных в буфер трансляции
U 0E6B, B214,15 ;10903          WRITE.MEM LS[T10]                       ; запись тестовых данных в буфер трансляции
U 0E6C, 08EB,6C ;10904          JSR [T16.PAR.ERR]                       ; переход для проверки наличия ошибки паритета
U 0E6D, 23C1,15 ;10905          ROL WR[2]                               ; следующие тестовые данные
                ;10906            BIT LS[BIT26] WITH WR[2],             ; проверка, установлен ли бит 26 (означает, что проверен
                ;10907                                                  ; BYTE OFFSET (бит 25))
U 0E6E, 5975,35 ;10908          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E6F, 08E7,41 ;10909          JMP.IF[BIT.SET] TO [T16.7]              ; переход на следующую проверку, если проверен BYTE
                ;10910                                                  ; OFFSET, иначе проверка, что прошли все биты PA
                ;10911            BIT LS[BIT15] WITH WR[2],             ; проверка, установлен ли бит 15 (если установлен,
                ;10912                                                  ; означает, что PA с 09 PO 23 проверены)
U 0E70, D95F,35 ;10913          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E71, 5B00,09 ;10914          SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит 15 не
                ;10915                                                  ; установлен, иначе установка бита BYTE OFFSET
U 0E72, 3671,15 ;10916          MOV LS[BIT24] TO WR[2]                  ;
U 0E73, 88E6,84 ;10917          JMP [REPEAT.T16.6]                      ; повторение с новыми тестовыми данными
```

```
                    ;10918   T16.7:
U 0E74,  FF82,15    ;10919           INC LS[ERROR.NUMBER]                    ; ошибка 7
U 0E75,  B675,15    ;10920           MOV LS[BIT26] TO WR[2]                  ; установка бита MODIFY
                    ;10921   REPEAT.T16.7:
U 0E76,  3E89,15    ;10922           MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка данных для печати под OTHER в сообщении об
                    ;10923                                                   ; ошибке
U 0E77,  BE21,15    ;10924           MOV WR[2] TO LS[10]                     ; загрузка ячейки временного хранения местной памяти
                    ;10925                                                   ; тестовыми данными со сдвигаемой единицей
                    ;10926                                                   ; (начинаются от LVA 26, который поступает на бит MODIFY)
U 0E78,  08EA,BC    ;10927           JSR [T16.NO.PAR.ERR]                    ; переход на проверку отсутствия ошибки паритета
U 0E79,  23C1,15    ;10928           ROL WR[2]                               ; следующие тестовые данные
                    ;10929             BIT LS[BIT31] WITH WR[2],             ; проверка, установлен ли бит 31 (означает, что проверен
                    ;10930                                                   ; PROT D)
U 0E7A,  597F,35    ;10931           DI(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E7B,  08E7,69    ;10932           JMP.IF[BITS.CLR] TO [REPEAT.T16.7]      ; повторение, если PROT D еще не установлен, иначе
                    ;10933                                                   ; переход на следующую проверку
U 0E7C,  FF82,15    ;10934           INC LS[ERROR.NUMBER]                    ; ошибка 8
U 0E7D,  B675,15    ;10935           MOV LS[BIT26] TO WR[2]                  ; установка бита MODIFY
                    ;10936   REPEAT.T16.8:
U 0E7E,  3E89,15    ;10937           MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка данных для печати под OTHER в сообщении об
                    ;10938                                                   ; ошибке
U 0E7F,  BE21,15    ;10939           MOV WR[2] TO LS[10]                     ; загрузка ячейки временного хранения местной памяти
                    ;10940                                                   ; набором со сдвигаемой единицей (начинается от LVA 26,
                    ;10941                                                   ; который поступает на бит MODIFY)
U 0E80,  9814,F5    ;10942           MEM.REQ[WRITE.TB] ADRS[T10] DT[LONG]    ; подготовка записи тестовых данных в буфер трансляции
U 0E81,  B214,15    ;10943           WRITE.MEM LS[T10]                       ; запись тестовых данных в буфер трансляции
U 0E82,  08E8,6C    ;10944           JSR [T16.PAR.ERR]                       ; переход для проверки наличия ошибки паритета
U 0E83,  23C1,15    ;10945           ROL WR[2]                               ; следующие тестовые данные
                    ;10946             BIT LS[BIT31] WITH WR[2],             ; проверка, установлен ли бит 31 (означает, что проверен
                    ;10947                                                   ; PROT D)
U 0E84,  597F,35    ;10948           DI(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0E85,  88E7,E9    ;10949           JMP.IF[BITS.CLR] TO [REPEAT.T16.8]      ; повторение, если PROT D не установлен, иначе переход
                    ;10950                                                   ; на следующую проверку
                    ;10951   T16.9:
U 0E86,  FF82,15    ;10952           INC LS[ERROR.NUMBER]                    ; ошибка 9
U 0E87,  B643,15    ;10953           MOV LS[BIT1] TO WR[2]                   ; установка PA 09
U 0E88,  4777,15    ;10954           BIS LS[BIT27] TO WR[2]                  ; установка PROT A
U 0E89,  3E15,15    ;10955           MOV WR[2] TO LS[T10]                    ; запоминание в местной памяти
U 0E8A,  3E89,15    ;10956           MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка данных для печати под OTHER в сообщении об
                    ;10957                                                   ; ошибке
U 0E8B,  08EA,BC    ;10958           JSR [T16.NO.PAR.ERR]                    ; переход для проверки отсутствия ошибки паритета
                    ;10959   T16.A:
U 0E8C,  FF82,15    ;10960           INC LS[ERROR.NUMBER]                    ; ошибка A
U 0E8D,  08E8,6C    ;10961           JSR [T16.PAR.ERR]                       ; переход для проверки наличия ошибки паритета
U 0E8E,  FF82,15    ;10962           INC LS[ERROR.NUMBER]                    ; ошибка B
U 0E8F,  6588,15    ;10963           CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати при ошибке
U 0E90,  6514,15    ;10964           CLR LS[T10]                             ; тестовые данные - все нули
                    ;10965   REPEAT.T16.B:
U 0E91,  08EA,BC    ;10966           JSR [T16.NO.PAR.ERR]                    ; переход для проверки отсутствия ошибки паритета
U 0E92,  0A18,2C    ;10967           JSR [NEXT.TB.ADDRESS]                   ; выборка следующего адреса буфера трансляции
U 0E93,  88E9,14    ;10968           JMP [REPEAT.T16.B]                      ; повторение, пока все адреса в буфере трансляции не
                    ;10969                                                   ; будут выбраны
U 0E94,  FF82,15    ;10970           INC LS[ERROR.NUMBER]                    ; ошибка C
U 0E95,  6512,15    ;10971           CLR LS[T9]                              ; очистка адреса в буфере трансляции для выборки
U 0E96,  6588,15    ;10972           CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати при ошибке
```

```
U 0E97, 3642,15 ;10973          MOV LS[BIT1] TO WR[0]                   ; установка LVA 00 для установки PA 09
U 0E98, BE14,15 ;10974          MOV WR[0] TO LS[T10]                    ; тестовыми данными является один установленный бит в
                ;10975                                                  ; буфере трансляции для дополнения бита паритета буфера
                ;10976                                                  ; трансляции
                ;10977  REPEAT.T16.C:
U 0E99, 08EA,BC ;10978          JSR [T16.NO.PAR.ERR]                    ; переход для проверки отсутствия ошибки паритета
U 0E9A, 0A18,2C ;10979          JSR [NEXT.TB.ADDRESS]                   ; выборка следующего адреса буфера трансляции
U 0E9B, 08E9,94 ;10980          JMP [REPEAT.T16.C]                      ; повторение, пока все адреса в буфере трансляции не
                ;10981                                                  ; будут выбраны
U 0E9C, FF82,15 ;10982          INC LS[ERROR.NUMBER]                    ; ошибка D
U 0E9D, 6512,15 ;10983          CLR LS[T9]                              ; очистка адреса
                ;10984  BACK.T16.D:
U 0E9E, 9812,F5 ;10985          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи в буфер трансляции
U 0E9F, B29C,15 ;10986          WRITE.MEM LS[ZERO]                      ; запись нулей в буфер трансляции
U 0EA0, 0A18,2C ;10987          JSR [NEXT.TB.ADDRESS]                   ; выборка следующего адреса
U 0EA1, 88E9,E4 ;10988          JMP [BACK.T16.D]                        ; повторение, пока все адреса не будут записаны
U 0EA2, 6512,15 ;10989          CLR LS[T9]                              ; очистка начального адреса
U 0EA3, 6588,15 ;10990          CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати при ошибке
U 0EA4, 3642,15 ;10991          MOV LS[BIT1] TO WR[0]                   ; установка LVA 00 в WR0
U 0EA5, BE14,15 ;10992          MOV WR[0] TO LS[T10]                    ; установка одного бита в буфере трансляции для
                ;10993                                                  ; переключения паритета
                ;10994  REPEAT.T16.D:
U 0EA6, 08EA,EC ;10995          JSR [LOOP.T16.NO.PAR.ERR]               ; проверка отсутствия ошибки паритета в текущем адресе
U 0EA7, 08EA,BC ;10996          JSR [T16.NO.PAR.ERR]                    ; запись в буфер трансляции для переключения
                ;10997                                                  ; паритета и проверка отсутствия ошибки паритета
U 0EA8, 0A18,2C ;10998          JSR [NEXT.TB.ADDRESS]                   ; выборка нового адреса буфера трансляции
U 0EA9, 08EA,64 ;10999          JMP [REPEAT.T16.D]                      ; повторение, пока все адреса буфера трансляции не будут
                ;11000                                                  ; проверены
U 0EAA, 88EC,14 ;11001          JMP [END.T16]                           ; выполнено
                ;11002  T16.NO.PAR.ERR:
U 0EAB, 0A17,DC ;11003          JSR [WRITE.CSR1.MME]                    ; разрешение диспетчера памяти
U 0EAC, 9812,F5 ;11004          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; подготовка записи в буфер трансляции по адресу в
                ;11005                                                  ; ячейке 9 местной памяти
U 0EAD, B214,15 ;11006          WRITE.MEM LS[T10]                       ; запись тестовых данных в буфер трансляции
                ;11007  LOOP.T16.NO.PAR.ERR:
U 0EAE, 2F82,95 ;11008          CLR WR[1]                               ; ожидаемые данные (ошибка паритета буфера трансляции
                ;11009                                                  ; сброшена)
U 0EAF, 9813,75 ;11010          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]  ; выполнение TEST.V.RCHK для стробирования CSR
U 0EB0, 3022,15 ;11011          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла (нет
                ;11012                                                  ; необходимости проверять эти данные)
U 0EB1, 9D45,75 ;11013          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0EB2, 3022,15 ;11014          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0EB3, 0869,3C ;11015          JSR [CHECK.RESULT]                      ; проверка ошибки паритета буфера трансляции на 0
U 0EB4, 88EA,E4 ;11016          JMP [LOOP.T16.NO.PAR.ERR]               ; зацикливание при ошибке, если разрешено
U 0EB5, 5B00,14 ;11017          RETURN                                  ; возврат в основную микропрограмму
                ;11018  T16.PAR.ERR:
U 0EB6, B676,15 ;11019          MOV LS[MME] TO WR[0]                    ; установка бита разрешения диспетчера в WR0
U 0EB7, C77A,15 ;11020          BIS LS[TB.PAR.DIAG] TO WR[0]            ; установка диагностического бита паритета буфера
                ;11021                                                  ; трансляции
U 0EB8, 8A17,FC ;11022          JSR [WRITE.CSR1]                        ; установка битов MME и TB PAR DIAG в CSR1
                ;11023  LOOP.T16.PAR.ERR:
U 0EB9, 369E,95 ;11024          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные (ошибка паритета буфера трансляции
                ;11025                                                  ; установлена)
U 0EBA, 9813,75 ;11026          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]  ; выполнение TEST.V.RCHK для стробирования CSR
U 0EBB, 3022,15 ;11027          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла (нет
```

```
                ;11028                                                  ; необходимости проверять эти данные)
U 0EBC, 9D45,75 ;11029          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0EBD, 3022,15 ;11030          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0EBE, 0869,3C ;11031          JSR [CHECK.RESULT]                      ; проверка ошибки паритета буфера трансляции на 1
  0EC?, 88EB,94 ;11032          JMP [LOOP.T16.PAR.ERR]                  ; зацикливание при ошибке, если разрешено
U 0EC0, 5B00,14 ;11033          RETURN                                  ; возврат в основную микропрограмму
                ;11034  END.T16:
```

```
;11035  .PAGE  "ТЕСТ 17 - тест бита VALID в CSR1 (модуль MCT)"
;11036  ;
;11037  ;  ОПИСАНИЕ ТЕСТА:
;11038  ;
;11039  ;        Это беглая проверка бита VALID (действительности данных) (бит 14) CSR1.
;11040  ;      Сам бит VALID был проверен предыдущим тестом памяти буфера трансляции,
;11041  ;      но бит в CSR не был проверен. Тест использует те же самые микропрограммы,
;11042  ;      как и предыдущий тест паритета буфера трасляции. Тест записывает в буфер
;11043  ;      трансляции значение установленного или сброшенного бита VALID в буфере
;11044  ;      трансляции. Тогда выполняется функция TEST.V.RCHK для стробирования CSR.
;11045  ;      Потом CSR1 читается и проверяется бит VALID. В этом тесте проверяется
;11046  ;      только бит 14; другие биты маскируются.
;11047  ;
;11048  ;        ПРИМЕЧАНИЕ: Бит VALID, который является выходом CSR, представляет со-
;11049  ;      бой инверсию бита VALID в буфере трансляции, то есть BUS MC D14 (VALID H)
;11050  ;      на самом деле является сигналом VALID ERR H.
;11051  ;
;11052  ;  ПРЕДПОЛОЖЕНИЯ:
 11053
;11054  ;        Предполагается, что все предыдущие тесты выполнены успешно.
;11055  ;
;11056  ;  ШАГИ ТЕСТА:
;11057  ;
;11058  ;        1. Установка номера ошибки и номера модуля в местной памяти (для распе-
;11059  ;           чатки ошибок) и очистка предыдущего номера ошибки в местной памяти.
;11060  ;           Также устанавливается бит PME в CSR1.
;11061  ;        2. Установка маски ошибки для проверки только бита 14 (VALID H).
;11062  ;        3. Выполнение инструкции MEM.REQ с MF=TEST.V.RCHK и DT=LONGWORD. В ка-
;11063  ;           честве адреса используется ячейка местной памяти, содержащая все нули.
;11064  ;           Выдача инструкции WRITE.MEM LS с данными - все единицы.
;11065  ;        4. Выполнение инструкции MEM.REQ с MF=WRITE.TB и DT=LONGWORD. В качестве
;11066  ;           адреса используется ячейка местной памяти, содержащая все нули. Выдача
;11067  ;           инструкции MOV MEM.DATA TO WR[0] для завершения функции TEST.V.RCHK
;11068  ;           (нет необходимости проверять данные, поступающие обратно).
;11069  ;        5. Выдача инструкций MEM.REQ с MF=READ.CSR и MOV.DATA TO WR[0] для чте-
;11070  ;           ния CSR1. Проверка бита 14 на 1 (инверсия VALID в буфере трансляции).
;11071  ;        6. Повторение шагов с 3 по 5 [illegible] буфер трансляции - все
;11072  ;           нули и проверка бита 14 (CSR1 на 1 (бит VALID в буфере трансляции сбро-
;11073  ;           шен).
;11074  ;
;11075  ;  ОШИБКИ:
;11076  ;
;11077  ;        ошибка 1 - неправильно работает [illegible] бита VALID ERR в CSR1, когда VALID
;11078  ;                   в буфере трансляции был установлен.
;11079  ;        ошибка 2 - неправильно работает [illegible] бита VALID ERR в CSR1, когда
;11080  ;                   в буфере трансляции VALID был сброшен.
;11081  ;
;11082  ;  НАЛАДКА:
;11083  ;
;11084  ;        ОШИБКА 1 - Эта ошибка указывает на возможную неисправность ПМЛ CSR 1B
;11085  ;      или одного из его входов. Центральный процессор необходимо остановить на
;11086  ;      инструкции MOV MEM.DATA TO [illegible] после TEST.V.RCHK. Микропрограмма MCT
;11087  ;      циклится в ожидании сигнала DATA RCVD. Если должным образом проверить высокий
;11088  ;      уровень на входе ПМЛ CSR 1B [illegible] ми здесь правильно, необходимо проверить,
;11089  ;      что вход CSR ENA WRM CLK H [illegible] ляется. Если все входы правильные, выход
```

```
                    ;11090  ;       BUS MC D14 H (VALID) должен быть низким. Если нет, по-видимому, ПМЛ неис-
                    ;11091  ;       правна.
                    ;11092  ;
                    ;11093  ;          ОШИБКА 2 - Так же, как при ошибке 1, с тем отличием, что VALID H дол-
                    ;11094  ;       жен быть низким и BUS MC D14 H должен быть высоким.
                    ;11095  ;
                    ;11096  ;
                    ;11097  ;
                    ;11098  T.17:
U 0EC1, B65E,15     ;11099          MOV LS[BEGIN.TEST] TO WR[0]                 ; установка бита 15 в WR[0] для слова управления и
                    ;11100                                                      ; состояния
U 0EC2, 3E80,15     ;11101          MOV WR[0] TO LS[CONTROL.STATUS]              ; установка бита 15 в слове управления и состония. Бит 15
                    ;11102                                                      ; указывает начало теста консольному процессору
U 0EC3, 10E0,15     ;11103          MISC [SET.CP.ATTN]                          ; выдача сигнала CPU ATTN консольному процессору
                    ;11104  WAIT.T17.0:
U 0EC4, 88EC,44     ;11105          JMP [WAIT.T17.0]                            ; зацикливание для ожидания ответа консольного
                    ;11106                                                      ; процессора
U 0EC5, 0A1A,AC     ;11107          JSR [SETUP.1]                               ; установка масок, кода модуля и др.
U 0EC6, 5F5C,15     ;11108          MCOM LS[VALID.ERR] TO WR[0]                 ; сброс бита 14 в WR0 и установка других
U 0EC7, 3E8A,15     ;11109          MOV WR[0] TO LS[ERROR.MASK]                  ; проверка только бита VALID ERR (бит 14)
U 0EC8, 0A17,DC     ;11110          JSR [WRITE.CSR1.MME]                        ; разрешение диспетчера памяти
U 0EC9, 989C,F5     ;11111          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]          ; подготовка записи в буфер трансляции
U 0ECA, 329E,15     ;11112          WRITE.MEM LS[ONES]                          ; запись единиц в буфер трансляции по адресу 0
                    ;11113  LOOP.T17.1:
U 0ECB, 2F82,95     ;11114          CLR WR[1]                                   ; ожидаемые данные (VALID ERR сброшен)
U 0ECC, 989D,75     ;11115          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]      ; подготовка стробирования битов ошибок CSR
U 0ECD, 3022,15     ;11116          MOV MEM.DATA TO WR[0]                       ; выдача инструкции MOV для завершения цикла
U 0ECE, 9D45,75     ;11117          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; подготовка чтения CSR1
U 0ECF, 3022,15     ;11118          MOV MEM.DATA TO WR[0]                       ; выборка содержимого CSR1
U 0ED0, 0869,3C     ;11119          JSR [CHECK.RESULT]                          ; проверка VALID ERR на 0
U 0ED1, 88EC,B4     ;11120          JMP [LOOP.T17.1]                            ; зацикливание при ошибке, если разрешено
U 0ED2, FF82,15     ;11121          INC LS[ERROR.NUMBER]                        ; ошибка 2
U 0ED3, 989C,F5     ;11122          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]         ; подготовка записи в буфер трансляции
U 0ED4, B29C,15     ;11123          WRITE.MEM LS[ZERO]                          ; запись нулей в буфер трансляции по адресу 0
                    ;11124  LOOP.T17.2:
U 0ED5, 369E,95     ;11125          MOV LS[ONES] TO WR[1]                       ; ожидаемые данные (бит VALID ERR установлен)
U 0ED6, 989D,75     ;11126          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]      ; подготовка стробирования битов ошибок CSR
U 0ED7, 3022,15     ;11127          MOV MEM.DATA TO WR[0]                       ; выдача инструкции MOV для завершения цикла
U 0ED8, 9D45,75     ;11128          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; подготовка чтения CSR1
U 0ED9, 3022,15     ;11129          MOV MEM.DATA TO WR[0]                       ; выборка содержимого CSR1
U 0EDA, 0869,3C     ;11130          JSR [CHECK.RESULT]                          ; проверка VALID ERR на 1
U 0EDB, 88ED,54     ;11131          JMP [LOOP.T17.2]                            ; зацикливание при ошибке, если разрешено
                    ;11132  END.T17:
                    ;11133
```

```
;11134  .PAGE "ТЕСТЫ ПЗУ ЗАЩИТЫ"
;11135  .TOC "ТЕСТ 1B - тест функции TEST.V.WCHK (для следующего теста (модуль MCT))"
;11136  ;
;11137  ;  ОПИСАНИЕ ТЕСТА:
;11138  ;
;11139  ;        Это короткий тест, составленный для проверки микропрограммы инструкции
;11140  ;      памяти TEST.V.WCHK. Инструкция похожа (и использует часть тех же микро-
;11141  ;      программ) на функцию TEST.V.RCHK, использованную раньше. Разница только
;11142  ;      в том, что она возбуждает RD CSR (C47) до стробирования CSR, так что
;11143  ;      WR ACROSS PG ERR будет возбужден при PAGE BOUNDARY. Сигнал WR ACROSS PG
;11144  ;      ERR будет проверяться позже. Этот тест проверяет только стробирование CSR.
;11145  ;      Тест загружает буфер трансляции для установки бита BYTE OFFSET и выдает
;11146  ;      инструкцию MEM.REQ с MF=TEST.V.WCHK. Выдается инструкция MOV MEM.DATA TO
;11147  ;      WR, но эти данные игнорируются. Читается CSR и проверяется бит TB PAR ERR
;11148  ;      на 0. Далее устанавливается бит CSR TB PAR DIAG и тест повторяется.
;11149  ;      TB PAR ERR проверяется на 1. Этим проверяется, что TEST.V.WCHK стробиру-
;11150  ;      ет CSR правильно и нет пере.ода в неопределенное место микропрограммы.
;11151  ;      Этим проверяется и последовательность микроинструкций.
;11152  ;        Микропограмма TEST.V.WCHK следующая: начальное ветвление по инструкции
;11153  ;      MEM.REQ выполняется, как описано в начале этого листинга. По адресу на-
;11154  ;      чального ветвления выдается ROT CLK и разрешается VAR BYPASS (обход ре-
;11155  ;      гистра виртуального адреса) и MEMORY BUSY. Следующий цикл поддерживает
;11156  ;      возбужденными VAR BYPASS и MEMORY BUSY и возбуждает RD CSR. Следующий
;11157  ;      цикл повторяет предыдущий, а также стробирует CSR. Следующий цикл поддер-
;11158  ;      живает возбужденным MEMORY BUSY и разрешает биты PA на шину MC BUS для
;11159  ;      чтения центральным процессором. Следующий цикл снимает MEMORY BUSY и
;11160  ;      поддерживает разрешение данных на MC BUS, пока не снимется CPU GRANT.
;11161  ;      Микропрограмма подготавливается для цикла общей шины и возвращается
;11162  ;      в холостой цикл.
;11163  ;
;11164  ;  ПРЕДПОЛОЖЕНИЯ:
;11165  ;
;11166  ;      Предполагается, что все предыдущие тесты выполнены успешно.
;11167  ;
;11168  ;  ШАГИ ТЕСТА:
;11169  ;
;11170  ;      1.Установка номера ошибки и номера модуля в местной памяти (для распе-
;11171  ;        чатки ошибок) и очистка предыдущего номера ошибки в местной памяти.
;11172  ;        Также установка бита MME в CSR1 и очистка других битов.
;11173  ;      2.Установка маски ошибки для проверки только бита 15 (CSR TB PAR ERR).
;11174  ;      3.Выполнение инструкции MEM.REQ с MF=TEST.V.WCHK и DT=LONGWORD. В ка-
;11175  ;        честве адреса используется ячейка местной памяти, содержащая все нули.
;11176  ;        Выдача инструкции WRITE.MEM LS с данными - все нули.
;11177  ;      4.Очистка WR1.
;11178  ;      5.Выполнение инструкции MEM.REQ с MF=TEST.V.WCHK и DT=LONGWORD. В ка-
;11179  ;        честве адреса используется ячейка местной памяти, содержащая все нули.
;11180  ;        Выдача инструкции MOV MEM.DATA TO WR[0] для завершения функции TEST.V.
;11181  ;        RCHK (нет необходимости проверять данные, поступающие обратно).
;11182  ;      6.Выдача инструкции MEM.REQ с MF=READ.CSR. Выдача инструкции MOV
;11183  ;        MEM.DATA TO WR[0] для чтения CSR1. Проверка бита 15 на 0 (нет ошибки
;11184  ;        паритета).
;11185  ;      7.Запись 1 в бит 29 (TB PAR DIAG H) CSR1 для принудительной установки
;11186  ;        неправильного паритета.
;11187  ;      8.Повторение шагов с 5 по 6, проверяя бит 15 на 1 (ошибка паритета).
;11188  ;
```

```
                ;11189  ;   ОШИБКИ:
                ;11190  ;
                ;11191  ;       ошибка 1 - функция TEST.V.WCHK работает неправильно при стробировании
                ;11192  ;                  CSR1 для сброса паритета буфера трансляции.
                ;11193  ;       ошибка 2 - функция TEST.V.WCHK работает неправильно при стробировании
                ;11194  ;                  CSR1 для установки паритета буфера трансляции.
                ;11195  ;
                ;11196  ;   НАЛАДКА:
                ;11197  ;
                ;11198  ;         ОШИБКА 1 И 2 - Эта ошибка указывает на возможную неисправность микро-
                ;11199  ;       последовательности, хотя вся аппаратура, используемая в этом тесте была
                ;11200  ;       проверена раньше. Если пошаговый режим MCT возможен, микропоследователь-
                ;11201  ;       ность можно проследить. Иначе необходим логический анализатор.
                ;11202  ;
                ;11203  ;
                ;11204
                ;11205  T.18:
U 0EDC, B65E,15 ;11206          MOV LS[BEGIN.TEST] TO WR[0]             ; установка бита 15 в WR[0] для слова управления и
                ;11207                                                  ; состояния
U 0EDD, 3E80,15 ;11208          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;11209                                                  ; 15 указывает начало теста консольному процессору
U 0EDE, 10E0,15 ;11210          MISC [SET.CP.ATTN]                      ; выдача сигнала CPU ATTN консольному процессору
                ;11211  WAIT.T18.0:
U 0EDF, 88ED,F4 ;11212          JMP [WAIT.T18.0]                        ; зацикливание для ожидания ответа консольного
                ;11213                                                  ; процессора
U 0EE0, 0A1A,AC ;11214          JSR [SETUP.1]                           ; установка масок, кода модуля и др.
U 0EE1, DF5E,15 ;11215          MCOM LS[TB.PAR.ERR] TO WR[0]            ; очистка бита 15 и установка других битов WR0
U 0EE2, 3E8A,15 ;11216          MOV WR[0] TO LS[ERROR.MASK]             ; проверка только бита 15 (ошибка паритета буфера
                ;11217                                                  ; трансляции)
U 0EE3, 0A17,DC ;11218          JSR [WRITE.CSR1.MME]                    ; разрешение диспетчера памяти
U 0EE4, 989C,F5 ;11219          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; подготовка записи в буфер трансляции по адресу 0
U 0EE5, B29C,15 ;11220          WRITE.MEM LS[ZERO]                      ; запись тестовых данных в буфер трансляции
                ;11221  LOOP.T18.1:
U 0EE6, 2F82,95 ;11222          CLR WR[1]                               ; ожидаемые данные (ошибка паритета буфера трансляции
                ;11223                                                  ; сброшена)
U 0EE7, 199C,F5 ;11224          MEM.REQ[TEST.V.WCHK] ADRS[#0] DT[LONG]  ; выполнение TEST.V.WCHK для стробирования CSR
U 0EE8, 3022,15 ;11225          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла (нет
                ;11226                                                  ; необходимости проверять эти данные)
U 0EE9, 9D45,75 ;11227          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0EEA, 3022,15 ;11228          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0EEB, 0869,3C ;11229          JSR [CHECK.RESULT]                      ; проверка ошибки паритета буфера трансляции на 0
U 0EEC, 88EE,64 ;11230          JMP [LOOP.T18.1]                        ; зацикливание при ошибке, если разрешено
U 0EED, FF82,15 ;11231          INC LS[ERROR.NUMBER]                    ; ошибка 2
U 0EEE, 8676,15 ;11232          MOV LS[MME] TO WR[0]                    ; установка бита разрешения диспетчера памяти в WR0
U 0EEF, C77A,15 ;11233          BIS LS[TB.PAR.DIAG] TO WR[0]            ; установка бита диагностики паритета буфера трансляции
U 0EF0, 8A17,FC ;11234          JSR [WRITE.CSR1]                        ; установка битов MME и TB PAR DIAG в CSR1
                ;11235  LOOP.T18.2:
U 0EF1, 369E,95 ;11236          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные (ошибка паритета буфера трансляции
                ;11237                                                  ; установлена)
U 0EF2, 199C,F5 ;11238          MEM.REQ[TEST.V.WCHK] ADRS[#0] DT[LONG]  ; выполнение функции TEST.V.WCHK для стробирования CSR
U 0EF3, 3022,15 ;11239          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла (нет
                ;11240                                                  ; необходимости проверять эти данные)
U 0EF4, 9D45,75 ;11241          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; подготовка чтения CSR1
U 0EF5, 3022,15 ;11242          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0EF6, 0869,3C ;11243          JSR [CHECK.RESULT]                      ; проверка ошибки паритета буфера трансляции на 1
```

```
U 0EF7, 88EF,14 ;11244           JMP [LOOP.T18.2]                    ; зацикливание при ошибке, если разрешено
                ;11245   END.T18:
```

```
;11246  .PAGE "ТЕСТ 19 - тест ПЗУ защиты и битов CSR1 ACCESS/MODIFY REFUSED (модуль MCT)"
;11247  ;
;11248  ; ОПИСАНИЕ ТЕСТА:
;11249  ;
;11250  ;       Этот тест проверяет ПЗУ защиты и связанные с ним биты ошибок CSR 1 ACCESS
;11251  ;    REFUSED (отклонение обращения) и MODIFY REFUSED (отклонение записи). Этот тест
;11252  ;    проверяет каждый адрес ПЗУ защиты, по которому может происходить обращение,
;11253  ,    с использованием всех режимов процессора и трех из четырех возможных типов
;11254  ;    функции памяти. Один тип функции памяти не использует выходов ПЗУ защиты
;11255  ;    (CSR не стробируется вместе с информацией). К этому типу функций памяти при-
 11256       надлежат те функции, для которых MF0 и MF1 являются высокими.
 11257          ПЗУ защиты использует в качестве пяти адресных битов четыре бита PROT и бит
;11258       MODIFY. Еще два адресных бита поступают из центрального процессора - биты
;11259  ,    текущего режима (ядро, исполнитель, супервизор или пользователь). Оставшиеся
;11260       два адресных бита поступают из двух младших битов поля функции памяти (MF),
;11261       передаваемого и буферизируемого во время инструкции MEM.REQ.
 11262          Выходы ПЗУ защиты ACCESS REFUSED L и MODIFY REFUSED L поступают в ПМЛ CSR
 11263       14 и стробируются микрокодом памяти в биты 22 и 23 соответственно. Для про-
 11264       верки этих битов считывается CSR.
 11265          Первой проверяемой частью ПЗУ является участок NO CHECK. Вначале централь-
 11266  ;    ный процессор записывает в буфер трансляции (TB) биты PROT и MODIFY и уста-
 11267       навливает в PSL текущий режим. В TB устанавливается бит TB PAR DIAG, чтобы
 11268       возник сигнал суммарной ошибки системы памяти во время выполнения следующей
 11269       инструкции MEM.REQ. Центральный процессор посылает инструкцию MEM.REQ с фун-
 11270       кцией MF=WRITE.V.NOCHK. Память осуществляет начальное ветвление, как описано
 11271       в начале этого листинга. Путь данных следующий: по адресу ветвления возбуж-
 11272       даются сигналы VAR BYPASS и MEMORY BUSY, данные из центрального процессора
 11273       пропускаются на шину MC и возбуждается разрешение для RAS. В следующем
 11274       цикле поддерживается возбужденным сигнал MEMORY BUSY (память занята) и
 11275       происходит попытка чтения из матрицы. В следующем цикле стробируется CSR 1
 11276       и устанавливаются другие сигналы для чтения из матрицы. Следующий цикл про-
 11277  ,    должает чтение из матрицы и выполняет ветвление по установленному сигналу
;11278  ;    суммарной ошибки ERR SUM. В следующих циклах поддерживается в установленном
;11279  ;    состоянии сигнал MEMORы BUSY и, наконец, происходит ветвление по CPU DATA
;11280  ;    REQ (посланному из центрального процессора по инструкции WRITE.MEM LS). Сле-
;11281  ;    дующий цикл переходит в подготовительный цикл для общей шины, затем в холос-
;11282  ;    той цикл. Теперь считывается CSR. Биты ACCESS REFUSED и MODIFY REFUSED про-
;11283  ;    веряются на 0. Это повторяется для каждого кода защиты - наборов битов MODIFY
;11284  ;    и текущего режима. Для функции WRITE.V.NOCHK MF0=0 и MF1=0.
;11285  ;       В следующих тестах (проверка защиты по чтению и проверка защиты по записи)
;11286  ;    для определения результата будут использоваться константы, загруженные в LS.
;11287  ;    Вначале центральный процессор устанавливает биты PROT, MODIFY и текущего
;11288  ;    режима. Затем выполняется инструкция MEM.REQ с функцией MF=TEST.V.RCHK для
;11289  ;    загрузки CSR битами ACCESS REFUSED и MODIFY REFUSED. Проверяется правильность
;11290  ;    значения CSR. Это повторяется для каждого кода защиты и каждого состояния
;11291  ;    бита MODIFY и каждого состояния битов текущего режима. Для TEST.V.RCHK поле
;11292  ;    MF0=0 и MF1=1.
;11293  ;      Последний тест аналогичен предыдущему за исключением того, что выдается
;11294  ;    инструкция MEM.REQ с функцией MF=TEST.V.WCHK. Для TEST.V.WCHK поля MF0=1 и
;11295  ;    MF1=0.
;11296  ;
;11297  ; ПРЕДПОЛОЖЕНИЯ:
;11298  ;
;11299  ;    Принимается, что все предыдущие тесты прошли успешно.
;11300  ;
```

```
;11301  ; ШАГИ ТЕСТА:
;11302  ;
;11303  ;    1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок) и
;11304  ;       очистка в LS номера предыдущей ошибки.
;11305  ;    2) Подготовка маски ошибки для проверки только битов 22 и 23. Установка в
;11306  ;       CSR 1 бита 29 (TB PAR DIAG) и бита 2 (MME) для создания суммарной ошибки
;11307  ;       в операции WRITE.V.NOCHK.
;11308  ;    3) Очистка WR1 (ожидаемые данные) и очистка в PSL битов текущего режима (те-
;11309  ;       кущий режим=KERNEL).
;11310  ;    4) Запись в буфер трансляции (TB) с нулями в битах PROT и MODIFY.
;11311  ;    5) Выдача инструкции MEM.REQ с MF=WRITE.V.NOCHK и типом данных = длинное
;11312  ;       слово. Выдача инструкции WRITE.MEM LS для завершения функции WRITE.V.
;11313  ;       NOCHK.
;11314  ;    6) Чтение CSR1 и проверка на 0 битов 22 и 23 (ACCESS REFUSED и MODIFY
;11315  ;       REFUSED).
;11316  ;    7) Инкрементирование данных в битах TB PROT и MODIFY и повторение шагов 5 и
;11317  ;       6. Повторение до тех пор, пока все биты PROT и бит MODIFY будут единицы.
;11318  ;    8) Инкрементирование битов текущего режима PSL и повторение шагов с 3 по 7.
;11319  ;       Повторение до тех пор, пока не будут проверены все остальные режимы
;11320  ;       (EXECUTIVE, SUPERVISOR и USER).
;11321  ;    9) Очистка в PSL битов текущего режима (текущий режим = KERNEL).
;11322  ;   10) Запись в TB с нулями в битах PROT и MODIFY.
;11323  ;   11) Вычисление ожидаемых данных путем чтения ячеек LS[ACCESS.RCHK.KERNEL] и
;11324  ;       LS[MODIFY.RCHK.KERNAL]. Запоминание их в WR2 и WR3. Младшие биты в WR2
;11325  ;       являются соответственно ожидаемемыми данными для ACCESS REFUSED и MODIFY
;11326  ;       REFUSED. Загрузка ожидаемых данных в WR1.
;11327  ;   12) Выдача инструкции MEM.REQ с MF=TEST.V.RCHK и DT=длинное слово. Выдача
;11328  ;       инструкции MOV MEM.DATA TO WR[0] для завершения функции TEST.V.RCHK, но
;11329  ;       эта информация не нуждается в проверке.
;11330  ;   13) Чтение CSR1 и проверка битов 22 и 23 (ACCESS REFUSED и MODIFY REFUSED)
;11331  ;       на правильность данных.
;11332  ;   14) Инкрементирование данных в битах TB PROT и MODIFY. Сдвиг WR[2] и WR[3]
;11333  ;       вправо на 1 бит для получения в младших битах следующих ожидаемых дан-
;11334  ;       ных. Повторение шагов с 11 по 13. Повторение до тех пор, пока все биты
;11335  ;       PROT и бит MODIFY не будут 1.
;11336  ;   15) Увеличение битов текущего режима в PSL и повторение шагов с 10 по 14. В
;11337  ;       шаге 11 использование ячеек LS[ACCESS.RCHK.EXEC] и LS[MODIFY.RCHK.EXEC]
;11338  ;       для режима EXECUTIVE. Использование ячеек LS[ACCESS.RCHK.SUP] и LS[MODIFY.
;11339  ;       RCHK.SUP] для режима супервизора. Использование LS[ACCESS.RCHK.USER]
;11340  ;       и LS[MODIFY.RCHK.USER] для режима пользователя. Повторение, пока все ре-
;11341  ;       жимы не будут проверены.
;11342  ;   16) Повторение шагов с 9 по 15 с заменой RCHK на WCHK.
;11343  ;
;11344  ; ОШИБКИ:
;11345  ;
;11346  ;   ПРИМЕЧАНИЕ 1: В сообщении об ошибке под OTHER (другие данные) будет печатать-
;11347  ;   ся адрес ПЗУ, по которому было обращение, поэтому при наладке можно прове-
;11348  ;   рить адресные линии. Напечатанный адрес использует бит PROT A в качестве
;11349  ;   младшего бита (бит адреса 0) и L CM1 в качестве старшего бита (бит адреса 8).
;11350  ;   Такой порядок является обратным в отношении адресных линий на плате, но со-
;11351  ;   ответствует листингу ПЗУ.
;11352  ;
;11353  ;   ПРИМЕЧАНИЕ 2: Ожидаемыми и полученными данными являются биты CSR 1 22 и 23
;11354  ;   (ACC или MOD REFUSED).
;11355  ;
```

```
;11356  ;   ошибка 1 - бит ACCESS REFUSED и/или MODIFY REFUSED установлен в операции
;11357  ;              NOCHK
;11358  ;   ошибка 2 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11359  ;              RCHK. режим ядра (KERNAL)
;11360  ;   ошибка 3 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11361  ;              RCHK. Режим исполнителя (EXECUTIVE)
;11362  ;   ошибка 4 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11363  ;              RCHK. Режим супервизора (SUPERVISOR)
;11364  ;   ошибка 5 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11365  ;              RCHK. Режим пользователя (USER)
;11366  ;   ошибка 6 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11367  ;              WCHK. Режим ядра (KERNAL)
;11368  ;   ошибка 7 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11369  ;              WCHK. Режим исполнителя (EXECUTIVE)
;11370  ;   ошибка 8 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11371  ;              WCHK. Режим супервизора (SUPERVISOR)
;11372  ;   ошибка 9 - бит ACCESS REFUSED и/или MODIFY REFUSED неправильный в операции
;11373  ;              WCHK. Режим пользователя (USER)
;11374  ;
;11375  ; НАЛАДКА:
;11376  ;
;11377  ;      ОШИБКА 1 - Эта ошибка указывает на неисправность ПЗУ защиты, его входов MF,
;11378  ;   или логических схем CSR. Подведите центральный процессор в пошаговом режиме на
;11379  ;   инструкцию MEM.REQ, которая выдает WRITE.V.NOCHK. Микрокод памяти будет выпол-
;11380  ;   нять цикл ожидания, пока не будет выставлен сигнал CPU DATA REQ. Проверьте на
;11381  ;   высокий уровень выходы ПЗУ защиты ACCESS REFUSED L и MODIFY REFUSED L. Если
;11382  ;   они неправильные, проверьте на низкие уровни оба входа MF0 и MF1 H. Кроме того,
;11383  ;   проверьте, что сигнал разрешения на ножке 13 соединен с "ЗЕМЛЕЙ". Если эти вхо-
;11384  ;   ды правильные то, вероятно, неисправна микросхема ПЗУ. Если один из входов MF
;11385  ;   имеет высокий уровень, проверьте его на выходе буфера-защелки, имеющего в ка-
;11386  ;   честве входов CSR 07 H и CSR 08 H. Оба эти входа должны быть низкими.
;11387  ;     Если сигнал ACCESS REFUSED L и MODIFY REFUSED L около ПЗУ защиты являются
;11388  ;   высокими, проверьте их на входе к ПМЛ CSR 1A. Если они высокие и около ПМЛ,
;11389  ;   то вероятно, что ПМЛ неисправна.
;11390  ;
;11391  ;      ОШИБКА 2 - Эта ошибка указывает на неисправность в ПЗУ защиты, его входов
;11392  ;   или ПМЛ CSR 1A. Подведите центральный процессор в пошаговом режиме на инст-
;11393  ;   рукцию MEM.REQ, которая выдает TEST.V.RCHK. Микрокод памяти будет выполнять
;11394  ;   цикл в ожидании снятия сигнала CPU GRANT. В это время можно просматривать
;11395  ;   ПЗУ защиты. Вначале проверьте ожидаемые результаты (биты 22 и 23 под EXP
;11396  ;   (ожидаемые данные)). Если они неправильные, проверьте входы на правильность
;11397  ;   адреса на адресных линиях PROT и MODIFY. Остальные адресные линии следующие:
;11398  ;   L CM0 H, L CM1 H и L MF0 H должны быть все низкими. Сигнал L MF1 H должен
;11399  ;   быть высоким. Сигнал разрешения на ножке 13 должен быть низким. Если все они
;11400  ;   правильные, вероятно, неисправна сама микросхема ПЗУ защиты. Входы MF и CM
;11401  ;   можно проверить на буфере-защелке, из которого они поступают. Входы буфера-
;11402  ;   защелки должны также находиться в правильном состоянии (такие же, как и вы-
;11403  ;   ходы).
;11404  ;     Если выходы правильные, проверьте их на входе к ПМЛ CSR 1A. Если там пра-
;11405  ;   вильно, то вероятно, неисправна ПМЛ.
;11406  ;
;11407  ;      ОШИБКА 3 - Эта ошибка указывает на неисправность ПЗУ защиты или его входов.
;11408  ;   Подведите центральный процессор в пошаговом режиме на инструкцию MEM.REQ, ко-
;11409  ;   торая выдает TEST.V.RCHK. Микрокод памяти будет выполнять цикл, и можно про-
;11410  ;   верить ПЗУ защиты. Проверьте входы на правильность адреса на линиях PROT и
```

```
                ;11411  ;   MODIFY. Другие входы следующие: L CM1 H и L MF0 H должны быть низкими. Входы
                ;11412  ;   L CM0 H и L MF1 H должны быть высокими. Входы MF и CM можно проверить около
                ;11413  ;   буфера-защелки, из которого они поступают. Входы к буферу-защелке должны так-
                ;11414  ;   же находиться в правильном состоянии (такие же, как и выходы). Если входы
                ;11415  ;   правильные, подозревается сама микросхема ПЗУ.
                ;11416  ;
                ;11417  ;      ОШИБКА 4 - То же, что и для ошибки 3, за исключением того, что L CM1 H дол-
                ;11418  ;   жен быть высоким, а L CM0 H должен быть низким.
                ;11419  ;
                ;11420  ;      ОШИБКА 5 - То же что и для ошибки 3, за исключением L CM1 H, который должен
                ;11421  ;   быть высоким.
                ;11422  ;
                ;11423  ;      ОШИБКА 6 - Эта ошибка указывает на неисправность ПЗУ защиты, его входов или
                ;11424  ;   ПМЛ CSR 1A. Подведите центральный процессор в пошаговом режиме на инструкцию
                ;11425  ;   MEM.REQ, которая выдает TEST.V.WCHK. Микрокод памяти будет выполнять цикл в
                ;11426  ;   ожидании снятия CPU GRANT. В это время можно просмотреть ПЗУ защиты. Внача-
                ;11427  ;   ле проверьте выходы на соответствие ожидаемым результатам (биты 22 и 23 под
                ;11428  ;   EXP (ожидаемые данные)). Если они неправильные, проверьте входы на правиль-
                ;11429  ;   ность адреса на линиях PROT и MODIFY. Другие адресные линии являются следую-
                ;11430  ;   щими: L CM0 H, L CM1 H и L MF1 H должен быть низкими. Сигнал L MF0 H должен
                ;11431  ;   иметь высокий уровень. Сигнал разрешения на ножке 13 должен быть низким. Ес-
                ;11432  ;   ли все они правильные, вероятно неисправна сама микросхема ПЗУ. Входы MF и
                ;11433  ;   CM можно проверить около буфера-защелки, из которого они поступают. Входы к
                ;11434  ;   этому буферу-защелке должны также находиться в правильном состоянии (такие же,
                ;11435  ;   как выходы).
                ;11436  ;     Если выходы правильные, проверьте их на входе к ПМЛ CSR 1A. Если здесь сиг-
                ;11437  ;   налы правильные, то вероятно, что неисправна сама ПМЛ.
                ;11438  ;
                ;11439  ;      ОШИБКИ С 7 ПО 9 - С наибольшей вероятностью можно подозревать ПЗУ защиты.
                ;11440  ;
                ;11441  ;---
                ;11442   T.19:
U 0EF8, B65E,15 ;11443           MOV LS[BEGIN.TEST] TO WR[0]            ; установка в WR0 бита 15 для слова управления и
                ;11444                                                  ; состояния
U 0EF9, 3E80,15 ;11445           MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                ;11446                                                  ; 15 указывает для консольного процессора начало теста
U 0EFA, 10E0,15 ;11447           MISC [SET.CP.ATTN]                     ; выдача для консольного процессора CPU ATTN
                ;11448   WAIT.T19.0:
U 0EFB, 88EF,B4 ;11449           JMP [WAIT.T19.0]                       ; цикл для ожидания овета консольного процессора
U 0EFC, 0A1A,AC ;11450           JSR [SETUP.1]                          ; установка масок, кода модуля и т.д.
U 0EFD, 5F6C,15 ;11451           MCOM LS[ACCESS.REFUSED] TO WR[0]       ; очистка в WR0 бита 22
U 0EFE, 456E,15 ;11452           BIC LS[MODIFY.REFUSED] TO WR[0]        ; а также, очистка бита 23
U 0EFF, 3E8A,15 ;11453           MOV WR[0] TO LS[ERROR.MASK]            ; подготока маски ошибки для проверки только битов 22 и
                ;11454                                                  ; 23 (ACCESS и MODIFY REFUSED)
U 0F00, B670,15 ;11455           MOV LS[OTHER.DATA] TO WR[0]            ; установка в рабочем регистре бита 24
U 0F01, 3E80,15 ;11456           MOV WR[0] TO LS[CONTROL.STATUS]        ; разрешение печати под OTHER в сообщении об ошибке
                ;11457                                                  ; (адрес, по которому было обращение к ПЗУ защиты)
U 0F02, B676,15 ;11458           MOV LS[MME] TO WR[0]                   ; установка в WR0 бита 27
U 0F03, C77A,15 ;11459           BIS LS[TB.PAR.DIAG] TO WR[0]           ; а также установка бита 29
U 0F04, 8A17,FC ;11460           JSR [WRITE.CSR1]                       ; установка бита разрешения диспетчера памяти и бита
                ;11461                                                  ; ошибки паритета буфера трансляции (TB)
U 0F05, 6514,15 ;11462           CLR LS[T10]                            ; данные для записи в TB
U 0F06, 2F80,15 ;11463           CLR WR[0]                              ; очистка WR0 для установки режима (текущий режим=ядро
                ;11464                                                  ; (KERNAL))
U 0F07, E516,15 ;11465           CLR LS[T11]                            ; запоминание набора битов текущего режима
```

```
                ;11466  REPEAT.T19.1.NEXTMODE:
U 0F0B, 88F5,4C ;11467          JSR [T19.PROM.ADDRESS]                  ; переход к вычислению адреса ПЗУ для печати
                ;11468  REPEAT.T19.1:
U 0F09, 989C,F5 ;11469          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в ТВ с текущими значениями PROT и
                ;11470                                                  ; MODIFY
U 0F0A, B214,15 ;11471          WRITE.MEM LS[T10]                       ; запись данных в ТВ
                ;11472  LOOP.T19.1:
U 0F0B, 2F82,95 ;11473          CLR WR[1]                               ; ожидаемые данные
U 0F0C, 1B9C,75 ;11474          MEM.REQ[WRITE.V.NOCHK] ADRS[#0] DT[LONG] ; запрос для занесения в CSR 1 информации по защите
U 0F0D, B29C,15 ;11475          WRITE.MEM LS[ZERO]                      ; выдача данных для записи в память
U 0F0E, 9D45,75 ;11476          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0F0F, 3022,15 ;11477          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0F10, 0869,3C ;11478          JSR [CHECK.RESULT]                      ; проверка, что оба бита ACCESS REFUSED и MODIFY REFUSED
                ;11479                                                  ; очищены
U 0F11, 08F0,B4 ;11480          JMP [LOOP.T19.1]                        ; цикл при ошибке, если разрешено
U 0F12, 88F6,8C ;11481          JSR [T19.INC.PROT]                      ; вычисление следующего набора битов PROT и MODIFY
U 0F13, 88F0,94 ;11482          JMP [REPEAT.T19.1]                      ; подготовка возвращает управление сюда, если еще не
                ;11483                                                  ; выполнен тест для всех наборов PROT и MODIFY
U 0F14, B616,15 ;11484          MOV LS[T11] TO WR[0]                    ; иначе выборка текущего кода режима
U 0F15, 4070,15 ;11485          ADD LS[BIT24] TO WR[0]                  ; увеличение значения режима
U 0F16, 3E16,15 ;11486          MOV WR[0] TO LS[T11]                    ; запоминание в LS
                ;11487             BIT LS[BIT26] WITH WR[0],            ; проверка, что бит 26 установлен (означает, что тест
                ;11488                                                  ; выполнен для всех режимов)
U 0F17, D974,35 ;11489          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F18, 88F0,89 ;11490          JMP.IF[BITS.CLR] TO [REPEAT.T19.1.NEXTMODE] ; повторение, если еще не выполнено для всех
                ;11491                                                  ; режимов
U 0F19, E5F8,15 ;11492          CLR LS[OS]                              ; очистка индекса
U 0F1A, FDF8,15 ;11493          COM LS[OS]                              ; инвертирование индекса. Теперь при первом увеличении
                ;11494                                                  ; индекса произойдет переход в 0
U 0F1B, 2F80,15 ;11495          CLR WR[0]                               ; очистка WR0 для режима (текущий режим=KERNAL)
U 0F1C, E516,15 ;11496          CLR LS[T11]                             ; запоминание кода текущего режима
                ;11497  REPEAT.T19.2.NEXTMODE:
U 0F1D, FF82,15 ;11498          INC LS[ERROR.NUMBER]                    ; ошибка от 2 до 5
U 0F1E, 6514,15 ;11499          CLR LS[T10]                             ; данные для записи в ТВ
U 0F1F, 88F5,4C ;11500          JSR [T19.PROM.ADDRESS]                  ; переход к вычислению адреса ПЗУ для печати
U 0F20, B64C,15 ;11501          MOV LS[BIT6] TO WR[0]                   ; установка в рабочем регистре бита 6
U 0F21, 6F88,15 ;11502          XOR WR[0] TO LS[ADDRESS.DATA]           ; установка бита 6 в адресе для печати
U 0F22, 7FF8,15 ;11503          INC LS[OS]                              ; указатель для следующего набора данных
U 0F23, 37F7,15 ;11504          MOV LS[USER.INDEX(4-0)] TO WR[2]        ; загрузка в WR2 ожидаемых данных для ACCESS REFUSED,
                ;11505                                                  ; хранящихся во второй половине индексируемой части LS
U 0F24, 7FF8,15 ;11506          INC LS[OS]                              ; увеличение индекса
U 0F25, B7F7,95 ;11507          MOV LS[USER.INDEX(4-0)] TO WR[3]        ; загрузка в WR3 ожидаемых данных для MODIFY REFUSED,
                ;11508                                                  ; хранящихся в следующей ячейке LS
                ;11509  REPEAT.T19.2:
U 0F26, 989C,F5 ;11510          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в ТВ с текущими значениями PROT и
                ;11511                                                  ; MODIFY
U 0F27, B214,15 ;11512          WRITE.MEM LS[T10]                       ; запись данных в ТВ
U 0F28, 88F5,EC ;11513          JSR [T19.COMPUTE.RESULT]                ; вычисление ожидаемых данных
                ;11514  LOOP.T19.2:
U 0F29, B618,95 ;11515          MOV LS[T12] TO WR[1]                    ; ожидаемые данные
U 0F2A, 989D,75 ;11516          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]  ; запрос для занесения в CSR1 информации по защите
U 0F2B, 3022,15 ;11517          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения цикла памяти (эти
                ;11518                                                  ; данные не нуждаются в проверке)
U 0F2C, 9D45,75 ;11519          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0F2D, 3022,15 ;11520          MOV MEM.DATA TO WR[0]                   ; выдача содержимого CSR1
```

```
U 0F2E, 0869,3C ;11521          JSR [CHECK.RESULT]                      ; проверка битов ACCEESS REFUSED и MODIFY REFUSED на
                ;11522                                                  ; соответствие ожидаемым
U 0F2F, 08F2,94 ;11523          JMP [LOOP.T19.2]                        ; цикл при ошибке, если разрешено
U 0F30, 88F6,8C ;11524          JSR [T19.INC.PROT]                      ; вычисление следующего набора битов PROT и MODIFY
U 0F31, 08F2,64 ;11525          JMP [REPEAT.T19.2]                      ; подпрограмма выполняет возврат сюда, если еще не
                ;11526                                                  ; проверены все наборы PROT и MODIFY
U 0F32, B616,15 ;11527          MOV LS[T11] TO WR[0]                    ; иначе выборка текущего кода режима
U 0F33, 4070,15 ;11528          ADD LS[BIT24] TO WR[0]                  ; увеличение значение режима
U 0F34, 3E16,15 ;11529          MOV WR[0] TO LS[T11]                    ; запоминание в LS
                ;11530             BIT LS[BIT26] WITH WR[0],            ; проверка, что бит 26 устаеновлен (означает, что
                ;11531                                                  ; проверены все режимы)
U 0F35, D974,35 ;11532          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F36, 08F1,D9 ;11533          JMP.IF[BITS.CLR] TO [REPEAT.T19.2.NEXTMODE] ; повторение, если тест выполнен еще не для всех
                ;11534                                                  ; режимов
U 0F37, 2F80,15 ;11535          CLR WR[0]                               ; очистка WR0 для режима (текущий режим=KERNAL)
U 0F38, E516,15 ;11536          CLR LS[T11]                             ; запоминание кода текущего режима
                ;11537  REPEAT.T19.6.NEXTMODE:
U 0F39, FF82,15 ;11538          INC LS[ERROR.NUMBER]                    ; ошибка от 6 до 9
U 0F3A, 6514,15 ;11539          CLR LS[T10]                             ; данные для записи в TB
U 0F3B, 88F5,4C ;11540          JSR [T19.PROM.ADDRESS]                  ; переход к вычислению адреса ПЗУ для печати
U 0F3C, B64A,15 ;11541          MOV LS[BIT5] TO WR[0]                   ; установка в рабочем регистре бита 5
U 0F3D, 6F88,15 ;11542          XOR WR[0] TO LS[ADDRESS.DATA]           ; установка бита 5 в адресе для печати
U 0F3E, 7FF8,15 ;11543          INC LS[OS]                              ; указатель к следующему набору данных защиты
U 0F3F, 37F7,15 ;11544          MOV LS[USER.INDEX(4-0)] TO WR[2]        ; загрузка WR2 ожидаемыми данными для ACCESS REFUSED,
                ;11545                                                  ; хранящимися во второй части индексируемой LS
U 0F40, 7FF8,15 ;11546          INC LS[OS]                              ; увеличение индекса
U 0F41, B7F7,95 ;11547          MOV LS[USER.INDEX(4-0)] TO WR[3]        ; загрузка в WR3 ожидаемых данных для MODIFY REFUSED,
                ;11548                                                  ; хранящихся в следующей ячейке
                ;11549  REPEAT.T19.6:
U 0F42, 989C,F5 ;11550          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в TB с текущими значениями PROT и
                ;11551                                                  ; MODIFY
U 0F43, B214,15 ;11552          WRITE.MEM LS[T10]                       ; запись данных в TB
U 0F44, 88F5,EC ;11553          JSR [T19.COMPUTE.RESULT]                ; вычисление ожидаемых данных
                ;11554  LOOP.T19.6:
U 0F45, B618,95 ;11555          MOV LS[T12] TO WR[1]                    ; ожидаемые данные
U 0F46, 199C,F5 ;11556          MEM.REQ[TEST.V.WCHK] ADRS[#0] DT[LONG]  ; запрос для занесения в CSR1 информации по защите
U 0F47, 3022,15 ;11557          MOV MEM.DATA TO WR[0]                   ; выдача MOV для завершения цикла памяти (эти данные не
                ;11558                                                  ; нуждаются в проверке)
U 0F48, 9D45,75 ;11559          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0F49, 3022,15 ;11560          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0F4A, 0869,3C ;11561          JSR [CHECK.RESULT]                      ; проверка ACCESS REFUSED и MODIFY REFUSED на
                ;11562                                                  ; соответствие ожидаемым
U 0F4B, 08F4,54 ;11563          JMP [LOOP.T19.6]                        ; цикл при ошибке, если разрешено
U 0F4C, 88F6,8C ;11564          JSR [T19.INC.PROT]                      ; вычисление следующего набора битов PROT и MODIFY
U 0F4D, 88F4,24 ;11565          JMP [REPEAT.T19.6]                      ; подпрограмма выполнит возврат сюда, если проверены еще
                ;11566                                                  ; не все комбинации PROT и MODIFY
U 0F4E, B616,15 ;11567          MOV LS[T11] TO WR[0]                    ; иначе выборка текущего кода режима
U 0F4F, 4070,15 ;11568          ADD LS[BIT24] TO WR[0]                  ; увеличение значения режима
U 0F50, 3E16,15 ;11569          MOV WR[0] TO LS[T11]                    ; запоминание режима в LS
                ;11570            BIT LS[BIT26] WITH WR[0],             ; проверка, что бит 26 установлен (означает, что тест
                ;11571                                                  ; выполнен для всех режимов)
U 0F51, D974,35 ;11572          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F52, 08F3,99 ;11573          JMP.IF[BITS.CLR] TO [REPEAT.T19.6.NEXTMODE] ; повторение, если тест выполнен еще не для всех
                ;11574                                                  ; режимов
U 0F53, 88F7,24 ;11575          JMP [END.T19]                           ; конец теста
```

```
                ;11576   T19.PROM.ADDRESS:
U 0F54, BFFE,15 ;11577           MOV WR[0] TO LS[PSL.HW]                 ; подготовка в центральном процессоре битов PSL (текущий
                ;11578                                                   ; режим)
U 0F55, 2F80,15 ;11579           CLR WR[0]                               ; начинается адресом 0
U 0F56, 3616,95 ;11580           MOV LS[T11] TO WR[1]                    ; выборка текущего режима
                ;11581             BIT LS[BIT24] WITH WR[1],             ; проверка, что бит CM0 установлен
U 0F57, D970,B5 ;11582           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F58, 5B00,09 ;11583           SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если CM0 очищен
U 0F59, 474E,15 ;11584           BIS LS[BIT7] TO WR[0]                   ; иначе установка бита 7 в адресе
                ;11585             BIT LS[BIT25] WITH WR[1],             ; проверка, что бит CM1 установлен
U 0F5A, 5972,B5 ;11586           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F5B, 5B00,09 ;11587           SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если CM1 очищен
U 0F5C, 4750,15 ;11588           BIS LS[BIT8] TO WR[0]                   ; иначе установка бита адреса 8
                ;11589             MOV WR[0] TO LS[ADDRESS.DATA],        ; запоминание результата в LS для печати в случае
                ;11590                                                   ; ошибки
U 0F5D, 3E88,14 ;11591           RETURN                                  ; возврат в основную программу (в основном коде
                ;11592                                                   ; установлены биты MF0 и MF1, биты адреса 5 и 6)
                ;11593   T19.COMPUTE.RESULT:
U 0F5E, 2F82,95 ;11594           CLR WR[1]                               ; вычисление начинается очисткой ожидаемых данных
                ;11595             BIT LS[BIT0] WITH WR[2],              ; проверка, должен ли бит ACCESS иметь значение 1 или 0
U 0F5F, D941,35 ;11596           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F60, 5B00,09 ;11597           SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит ACCESS очищен
U 0F61, C76C,95 ;11598           BIS LS[ACCESS.REFUSED] TO WR[1]         ; иначе установка в ожидаемых данных бита 22
                ;11599             BIT LS[BIT0] WITH WR[3],              ; проверка, должен ли бит MODIFY иметь значение 1 или 0
U 0F62, 5941,B5 ;11600           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F63, 5B00,09 ;11601           SKIP.IF[BITS.CLR]                       ; пропуск следующей инструкции, если бит MODIFY очищен
U 0F64, 476E,95 ;11602           BIS LS[MODIFY.REFUSED] TO WR[1]         ; иначе установка в ожидаемых данных бита 23
U 0F65, 3E18,95 ;11603           MOV WR[1] TO LS[T12]                    ; запоминание в LS
U 0F66, A341,15 ;11604           ROR WR[2]                               ; занесение следующего бита данных для ACCESS в позицию 0
                ;11605             ROR WR[3],                            ; занесение следующего бита данных для MODIFY в позицию 0
                ;11606
U 0F67, A341,94 ;11607           RETURN                                  ; возврат в основную программу
                ;11608   T19.INC.PROT:
U 0F68, FF88,15 ;11609           INC LS[ADDRESS.DATA]                    ; увеличение адреса для печати в случае ошибки
U 0F69, B676,15 ;11610           MOV LS[BIT27] TO WR[0]                  ; подготовкка для увеличения данных в TB (биты PROT)
                ;11611             ADD WR[0] TO LS[T10],                 ; увеличение битов PROT в данных TB
U 0F6A, EC14,35 ;11612           DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 0F6B, 88F7,00 ;11613           JMP.IF[N.CLR] TO [T19.INC.RETURN] .     ; повторение, если бит 31 (VALID) еще не установлен,
                ;11614                                                   ; иначе проверка, был ли установлен бит MODIFY
U 0F6C, 3674,15 ;11615           MOV LS[BIT26] TO WR[0]                  ; установка в рабочем регистре бита 26
                ;11616             BIT WR[0] WITH LS[T10],               ; проверка, был ли установлен бит MODIFY
U 0F6D, D914,35 ;11617           DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0F6E, 88F7,11 ;11618           JMP.IF[BIT.SET] TO [T19.INC.RETURN+1]   ; переход к выполнению для следующего режима, если был
                ;11619                                                   ; установлен бит MODIFY
U 0F6F, BE14,15 ;11620           MOV WR[0] TO LS[T10]                    ; иначе установка в данных TB бита MODIFY
                ;11621   T19.INC.RETURN:
U 0F70, 5B00,14 ;11622           RETURN                                  ; и продолжение
                ;11623   T19.INC.RETURN+1:
U 0F71, DB00,16 ;11624           RETURN+1                                ; выполнение для следующего режима
                ;11625   END.T19:
```

```
;11626  .PAGE "ТЕСТЫ ДРУГИХ БИТОВ CSR1"
;11627  .TOC "ТЕСТ 1A - проверка битов NXM и ADAPTER REG SEL (модуль MCT)"
;11628  ;
;11629  ;  ОПИСАНИЕ ТЕСТА:
;11630  ;
;11631  ;        Этот тест проверяет бит NXM посредством попытки записи по адресу 0
;11632  ;      и записей по несуществующим адресам. Первая запись должна завершаться
;11633  ;      без бита NXM, в то время, как другие записи должны вызвать появление
;11634  ;      NXM. Кроме того, проверяется бит ADAP REG SEL (бит 18), чтобы убедиться,
;11635  ;      что он появляется при записи в пространство адресов регистров адаптера
;11636  ;      общей шины.
;11637  ;        Путь данных следующий: в ячейке LS подготавливается адрес и выдается
;11638  ;      инструкция MEM.REQ, содержащая в качестве функции памяти WRITE.P (запись
;11639  ;      по физическому адресу) при типе данных DT=длинное слово. Для завершения
;11640  ;      цикла памяти выполняется инструкция WRITE.MEM LS[ZERO] и считывается
;11641  ;      CSR1 для проверки бита NXM (бит 16) и бита ADAP REG SEL (бит 18). Нет
;11642  ;      необходимости описывать микропрограмму функции WRITE.P, так как все, что
;11643  ;      здесь требуется, это сигнал CSR ERR SUM CLOCK. Это в значительной мере
;11644  ;      проверялось тестом ПЗУ защиты. Биты физического адреса 18-23, посылае-
;11645  ;      мые в качестве адреса инструкцией MEM.REQ, поступают вместе с сигналами
;11646  ;      FP ("ОТПЕЧАТКИ ПАЛЬЦЕВ") плат матриц памяти в две ПМЛ для формирования
;11647  ;      правильного состояния бита NXM (NXM L). Этот сигнал поступает в ПМЛ CSR
;11648  ;      1A и заносится в регистр микрокодом. Сигнал UB ADAPTER REG SEL L генери-
;11649  ;      руется в ПМЛ дешифрации в соответствии со значением битов адреса PA
;11650  ;      23-18 и посылается в ПМЛ CSR 1B. Микрокод также заносит значение этого
;11651  ;      бита в CSR.
;11652  ;
;11653  ;  ПРЕДПОЛОЖЕНИЯ:
;11654  ;
;11655  ;        Принимается, что все предыдущие тесты прошли успешно.
;11656  ;
;11657  ;  ШАГИ ТЕСТА:
;11658  ;
;11659  ;        1.Установка в LS номера ошибки и номера модуля (для распечатки ошибок)
;11660  ;          и очистка в LS номера предыдущей ошибки.
;11661  ;        2.Подготовка маски ошибок для проверки только битов 16 (NXM) и 18
;11662  ;          (UB ADAP REG).
;11663  ;        3.Выполнение инструкции MEM.REQ с функцией MF=WRITE.P и типом данных
;11664  ;          DT=длинное слово. Использование в качестве адреса ячейки LS, содер-
;11665  ;          жащей все нули.
;11666  ;        4.Выполнение инструкции WRITE.MEM.LS с данными, состоящими из нулей.
;11667  ;          Затем чтение CSR1 и проверка, что бит NXM очищен и бит ADAP REG SEL
;11668  ;          очищен.
;11669  ;        5.Загрузка ячейки LS значением 540000(H). Этот адрес непосредственно
;11670  ;          выше наибольшего адреса памяти, возможного в системе.
;11671  ;        6.Выполнение MEM.REQ с MF=WRITE.P и DT=длинное слово. Использование
;11672  ;          в качестве физического адреса только что загруженной ячейки LS.
;11673  ;        7.Выполнение инструкции WRITE.MEM LS с данными из всех нулей. Затем
;11674  ;          чтение CSR1 и проверка, что бит NXM установлен, а бит ADAP REG SEL
;11675  ;          очищен.
;11676  ;        8.Увеличение ячейки LS, содержащей физический адрес, на 40000(H).
;11677  ;          повторение шагов с 6 по 8, пока адрес не станет = F0000(H).
;11678  ;        9.Повторение шагов с 6 по 8 до адреса = FC0000(H), проверяя, что бит
;11679  ;          NXM очищен, а бит ADAP REG SEL установлен.
;11680  ;       10.Повторение шагов 6 и 7, проверяя, что по адресу FFFFFF(H) бит NXM
```

```
;11681  ;          очищен и бит UB ADAP REG SEL очищен.
;11682  ;
;11683  ;  ОШИБКИ:
;11684  ;
;11685  ;       ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными являются биты CSR1 18 и 16
;11686  ;       (UB ADAP REG и NXM).
;11687  ;
;11688  ;       ошибка 1 - бит NXM или бит UB ADAP REG SEL не очищается при записи по
;11689  ;                  физическому адресу в существующую память (адрес 0).
;11690  ;       ошибка 2 - бит NXM не устанавливается или бит UB ADAP REG SEL не очищает-
;11691  ;                  ся при записи по физическому адресу в несуществующую память
;11692  ;                  (адрес = от 540000(H) до EC0000(H)).
;11693  ;       ошибка 3 - бит NXM не очищается или бит UB ADAP REG SEL не устанавливает-
;11694  ;                  ся при записи по физическому дресу в регистр адаптера общей
;11695  ;                  шины (адрес = от F00000(H) до F80000(H)). Адрес печатается под
;11696  ;                  OTHER (другие данные).
;11697  ;       ошибка 4 - бит NXM или бит UB ADAP REG SEL не очищается при записи по фи-
;11698  ;                  зическому адресу из пространства адресов общей шины (адрес=
;11699  ;                  FFFFFF(H)). Адрес печатается под OTHER (другие данные).
;11700  ;
;11701  ;
;11702  ;  НАЛАДКА:
;11703  ;
;11704  ;       ПРИМЕЧАНИЕ: Адрес, по которому происходило обращение, печатается в сооб-
;11705  ;       шении об ошибке под OTHER (другие данные).
;11706  ;
;11707  ;          ОШИБКА 1. Эта ошибка указывает на неисправность, связанную с ПМЛ ДЕШ.
;11708  ;       ФИЗИЧЕСКОГО АДР.А, ПМЛ ДЕШ. ФИЗИЧЕСКОГО АДР.В, ПМЛ CSR 1А, ПМЛ CSR 1В,
;11709  ;       или их входами. Входы к ПМЛ дешифрации можно проверить, когда централь-
;11710  ;       ный процессор остановлен в пошаговом режиме на инструкции MEM.REQ с функ-
;11711  ;       цией памяти MF=WRITE.P. Микрокод памяти будет выполнять цикл в ожидании
;11712  ;       CPU DATE REQ. Выход P NXM L из ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А должен иметь
;11713  ;       высокий уровень. Если нет, проверьте входы ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А на
;11714  ;       низкие уровни для сигналов с PA 23 по PA 18 и на низкий уровень для сиг-
;11715  ;       нала FP3A L. Если эти входы правильные, ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А неисп-
;11716  ;       равна.
;11717  ;          Если сигнал P NXM L на выходе ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А является вы-
;11718  ;       соким, то ПМЛ исправна. Проверьте выходы из ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.В на
;11719  ;       высокий уровень для сигналов NXM L и UB ADAPTER REG SEL L. Если хотя бы
;11720  ;       один из этих сигналов низкий, проверьте на высокий уровень вход P NXM L и
;11721  ;       на низкие уровни входы с PA 23 H по PA 18 H. Если эти входы правильные,
;11722  ;       то неисправна ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.В.
;11723  ;          Если выходы NXM L и UB ADAPTER REG SEL L оба высокие, проверьте их на
;11724  ;       входах к ПМЛ CSR 1А и CSR 1В. Если там они также правильные, неисправна
;11725  ;       ПМЛ CSR1.
;11726  ;
;11727  ;          ОШИБКА 2. Эта ошибка указывает на неисправность, связанную с микросхе-
;11728  ;       мами ПМЛ дешифрации или логическими схемами CSR. Входы к ПМЛ дешифрации
;11729  ;       можно проверить, когда центральный процессор остановлен в пошаговом режиме
;11730  ;       на инструкции MEM.REQ с функцией памяти MF=WRITE.P. Микрокод памяти будет
;11731  ;       выполнять цикл в ожидании CPU DATA REQ. Выход P NXM L из ПМЛ ДЕШ.ФИЗИЧЕС-
;11732  ;       КОГО АДР.А должен быть низким. Если нет, проверьте входы от PA 23 H до
;11733  ;       PA 18.H на значения 010101(B) соответственно или на более высокое двоич-
;11734  ;       ное значение (действительный адрес представлен в распечатке под OTHER
;11735  ;       (другие данные), но любое значение PA 23 - PA 18, больше, чем 010101,
```

```
                ;11736  ;       должно генерировать сигнал NXM). Если эти входы правильные, следует по-
                ;11737  ;       дозревать ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А.
                ;11738  ;          Если сигнал P NXM L низкий, проверьте выходы NXM L и UB ADAPTER REG SEL
                ;11739  ;       L из ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.В. Сигнал NXM L должен быть низким, а UB
                ;11740  ;       ADAPTER REG SEL L должен быть высоким. Если какой-либо из этих выходов
                ;11741  ;       неправильный, проверьте входы к ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.В на значение
                ;11742  ;       между 010101(B) и 111011(B) на PA 23 H до PA 18 H соответственно. Дейст-
                ;11743  ;       вительный адрес дается в распечатке под OTHER (другие данные). Кроме того,
                ;11744  ;       убедитесь, что сигнал P NXM L низкий. Если они правильные, то ПМЛ неиспр-
                ;11745  ;       равна. Если сигналы NXM L и UB ADAPTER REG SEL L правильные, тогда про-
                ;11746  ;       верьте их на входах ПМЛ CSR. Если эти входы правильные, то неисправна
                ;11747  ;       ПМЛ CSR.
                ;11748  ;
                ;11749  ;          ОШИБКА 3. Выполняйте проверку в такой же последовательности, как и в
                ;11750  ;       случае ошибки 2, за исключением того, что сигналы P NXM и NXM L должны
                ;11751  ;       быть высокими, а UB ADAPTER REG SEL L должен быть низким. Входы PA 23 -
                ;11752  ;       PA 18 должны иметь значения между 111100(B) и 111110(B).
                ;11753  ;
                ;11754  ;          ОШИБКА 4. Выполняйте проверку в такой же последовательности, как и в
                ;11755  ;       случае ошибки 2, за исключением того, что сигналы P NXM и NXM L должны
                ;11756  ;       быть высокими, а адрес на PA 23 - PA 18 должен быть 111111(B). С наи-
                ;11757  ;       большей вероятностью можно подозревать микросхемы ПМЛ дешифрации.
                ;11758  ;
                ;11759  ;
                ;11760
                ;11761  T.1A:
U 0F72, B65E,15 ;11762          MOV LS[BEGIN.TEST]  TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;11763                                                   ; состояния
U 0F73, 3E80,15 ;11764          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 15 с слове управления и состояния. Бит
                ;11765                                                   ; 15 указывает для консольного процессора начало теста
U 0F74, 10E0,15 ;11766          MISC [SET.CP.ATTN]                       ; выдача для консольного процессора CPU ATTN
                ;11767  WAIT.T1A.0:
U 0F75, 08F7,54 ;11768          JMP [WAIT.T1A.0]                         ; цикл для ожидания ответа консольного процессора
U 0F76, 0A1A,9C ;11769          JSR [SETUP]                              ; установка маски, кода модуля и т.д. и очистка CSR1
                ;11770                                                   ; МСТ
U 0F77, 5F60,15 ;11771          MCOM LS[NXM] TO WR[0]                    ; очистка в рабочем регистре бита 16
U 0F78, 4564,15 ;11772          BIC LS[ADP.REG] TO WR[0]                 ; очистка в рабочем регистре бита 18
U 0F79, 3E8A,15 ;11773          MOV WR[0] TO LS[ERROR.MASK]              ; маска ошибки для проверки только битов CSR1 NXM и UB
                ;11774                                                   ; ADAP REG
U 0F7A, B670,15 ;11775          MOV LS[OTHER.DATA] TO WR[0]              ; подготовка разрешения печати под OTHER (другие
                ;11776                                                   ; данные)
U 0F7B, 3E80,15 ;11777          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка этого бита в слове управления
U 0F7C, 6588,15 ;11778          CLR LS[ADDRESS.DATA]                     ; очистка адреса для печати
                ;11779  LOOP.T1A.1:
U 0F7D, 2F82,95 ;11780          CLR WR[1]                                ; ожидаемые данные
U 0F7E, 999C,75 ;11781          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]       ; запрос для записи по адресу 0
U 0F7F, B29C,15 ;11782          WRITE.MEM LS[ZERO]                       ; запись по заданному адресу
U 0F80, 9D45,75 ;11783          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; запрос для чтения CSR1
U 0F81, 3022,15 ;11784          MOV MEM.DATA TO WR[0]                    ; прием данных из CSR1
U 0F82, 0869,3C ;11785          JSR [CHECK.RESULT]                       ; проверка, что биты CSR1 NXM и UB ADAP REG очищены
U 0F83, 88F7,D4 ;11786          JMP [LOOP.T1A.1]                         ; цикл при ошибке, если разрешено
U 0F84, FF82,15 ;11787          INC LS[ERROR.NUMBER]                     ; ошибка 2
U 0F85, 3738,15 ;11788          MOV LS[#540000] TO WR[0]                 ; наибольший возможный адрес памяти +1
U 0F86, BE12,15 ;11789          MOV WR[0] TO LS[T9]                      ; запоминание адреса в LS
                ;11790  REPEAT.T1A.2:
```

```
U 0F87, 3612,15 ;11791          MOV LS[T9] TO WR[0]                     ; занесение в рабочий регистр ранее подготовленного
                ;11792                                                  ; адреса
U 0F88, BE88,15 ;11793          MOV WR[0] TO LS[ADDRESS.DATA]           ; выборка адреса для печати в случае ошибки
                ;11794  LOOP.T1A.2:
U 0F89, B660,95 ;11795          MOV LS[NXM] TO WR[1]                    ; ожидаемые данные (установленный бит NXM)
U 0F8A, 9912,75 ;11796          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи по несуществующему адресу
U 0F8B, B29C,15 ;11797          WRITE.MEM LS[ZERO]                      ; запись по заданному адресу
U 0F8C, 9D45,75 ;11798          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0F8D, 3022,15 ;11799          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0F8E, 0869,3C ;11800          JSR [CHECK.RESULT]                      ; проверка, что бит NXM в CSR1 установлен, бит UB ADAP
                ;11801                                                  ; REG очищен
U 0F8F, 08F8,94 ;11802          JMP [LOOP.T1A.2]                        ; цикл при ошибке, если разрешено
U 0F90, 3612,15 ;11803          MOV LS[T9] TO WR[0]                     ; выборка старого адреса
U 0F91, 4064,15 ;11804          ADD LS[#40000] TO WR[0]                 ; увеличение на 40000(H)
U 0F92, BE88,15 ;11805          MOV WR[0] TO LS[ADDRESS.DATA]           ; запоминание в LS для печати, если разрешено
U 0F93, BE12,15 ;11806          MOV WR[0] TO LS[T9]                     ; запоминание в LS нового адреса
U 0F94, B73A,15 ;11807          MOV LS[#F00000] TO WR[0]                ; выборка из LS значения для сравнения
                ;11808            CMP WR[0] WITH LS[T9],                ; уже достигнут адрес F00000(H)?
U 0F95, 4F12,35 ;11809          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0F96, 88F8,71 ;11810          JMP.IF[NEQ] TO [REPEAT.T1A.2]           ; повторение, если еще не адрес F00000(H)
U 0F97, FF82,15 ;11811          INC LS[ERROR.NUMBER]                    ; ошибка 3
                ;11812  REPEAT.T1A.3:
U 0F98, 3612,15 ;11813          MOV LS[T9] TO WR[0]                     ; выборка адреса для печати
U 0F99, BE88,15 ;11814          MOV WR[0] TO LS[ADDRESS.DATA]           ; запоминание адреса для печати в случае ошибки
                ;11815  LOOP.T1A.3:
U 0F9A, 3664,95 ;11816          MOV LS[ADP.REG] TO WR[1]                ; ожидаемые данные (бит UB ADAP REG установлен)
U 0F9B, 9912,75 ;11817          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи по несуществующему адресу
U 0F9C, B29C,15 ;11818          WRITE.MEM LS[ZERO]                      ; запись по заданному адресу
U 0F9D, 9D45,75 ;11819          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0F9E, 3022,15 ;11820          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0F9F, 0869,3C ;11821          JSR [CHECK.RESULT]                      ; проверка, что бит NXM в CSR1 очищен, бит UB ADAP REG
                ;11822                                                  ; установлен
U 0FA0, 88F9,A4 ;11823          JMP [LOOP.T1A.3]                        ; цикл при ошибке, если разрешено
U 0FA1, 3612,15 ;11824          MOV LS[T9] TO WR[0]                     ; выборка старого адреса
U 0FA2, 4064,15 ;11825          ADD LS[#40000] TO WR[0]                 ; увеличение на 40000(H)
U 0FA3, BE12,15 ;11826          MOV WR[0] TO LS[T9]                     ; запоминание нового адреса в LS
U 0FA4, B73C,15 ;11827          MOV LS[#FC0000] TO WR[0]                ; выборка из LS значения для сравнения
                ;11828            CMP WR[0] WITH LS[T9],                ; еще не достигнут адрес FC0000(H)?
U 0FA5, 4F12,35 ;11829          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 0FA6, 08F9,81 ;11830          JMP.IF[NEQ] TO [REPEAT.T1A.3]           ; повторение, если не адрес FC0000(H)
U 0FA7, FF82,15 ;11831          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 0FA8, DF2A,15 ;11832          MCOM LS[#FF000000] TO WR[0]             ; занесение в WR0 адреса FFFFFF(H)
U 0FA9, BE12,15 ;11833          MOV WR[0] TO LS[T9]                     ; занесение его в LS для инструкции MEM.REQ
U 0FAA, BE88,15 ;11834          MOV WR[0] TO LS[ADDRESS.DATA]           ; занесение его в LS для печати адреса при ошибке
                ;11835  LOOP.T1A.4:
U 0FAB, 2F82,95 ;11836          CLR WR[1]                               ; ожидаемые данные
U 0FAC, 9912,75 ;11837          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи по несуществующему адресу
U 0FAD, B29C,15 ;11838          WRITE.MEM LS[ZERO]                      ; запись по заданному адресу
U 0FAE, 9D45,75 ;11839          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FAF, 3022,15 ;11840          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0FB0, 0869,3C ;11841          JSR [CHECK.RESULT]                      ; проверка, что бит NXM В CSR1 очищен, бит UB ADAP REG
                ;11842                                                  ; очищен
U 0FB1, 08FA,B4 ;11843          JMP [LOOP.T1A.4]                        ; цикл при ошибке, если разрешено
                ;11844  END.T1A:
```

```
;11845  .PAGE "ТЕСТ 1B - проверка бита ILL UB OPER (бит 20 CSR)(модуль MCT или модуль DAP)"
;11846  ;
;11847  ;
;11848  ; ОПИСАНИЕ ТЕСТА:
;11849  ;
;11850  ;       Этот тест проверяет бит 20 CSR1. Это бит неразрешенной операции общей ши-
;11851  ;   ны, который устанавливается при попытке записи невыравненного слова или при
;11852  ;   записи любого длинного слова в устройство на общей шине. Используемыми адре-
;11853  ;   сами являются все единицы или единицы за исключением бита 0. Они всегда при-
;11854  ;   водят к несуществующему устройству.
;11855  ;       Путь данных следующий: выдается инструкция MEM.REQ с функцией памяти MF=
;11856  ;   WRITE.P и типом данных DT=длинное слово. Используемым адресом будут все еди-
;11857  ;   ницы, за исключением бита 0. ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти будет генериро-
;11858  ;   вать сигнал OP ERR L, который стробируется и заносится в CSR микрокодом. Вы-
;11859  ;   дается инструкция WRITE.MEM LS для завершения цикла памяти и выполняется
;11860  ;   чтение CSR1. Бит ILL UB OPER (бит 20 CSR1) будет проверяться на единицу. Дру-
;11861  ;   гие тесты проверяют MEM.REQ при DT=слово и DT=байт со значениями бита 0, рав-
;11862  ;   ными 0 и 1. Проверяется правильность значения бита ILL UB OPER. В этом тесте
;11863  ;   проверяется только бит 20. Остальные биты маскируются.
;11864  ;       Кроме того, этот тест проверяет, что запись в матрицу памяти невыравнен-
;11865  ;   ного слова в режиме совместимости генерирует ошибку ILL UB OPER.
;11866  ;
;11867  ; ПРЕДПОЛОЖЕНИЯ :
;11868  ;
;11869  ;       Принимается, что все предыдущие тесты прошли успешно.
;11870  ;       Этот тест использует некоторые сигналы, которые генерируются модулем DAP
;11871  ;   (DATA TYPE 0, DATA TYPE 1 и COMPAT MODE). Эти сигналы проверялись только
;11872  ;   в самом модуле DAP, но не на выходах модуля. Поэтому имеется некоторая
;11873  ;   возможность того, что неисправность может быть вызвана модулем DAP. Эти
;11874  ;   случаи перечислены в разделе наладки.
;11875  ;
;11876  ; ШАГИ ТЕСТА :
;11877  ;
;11878  ;     1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок) и
;11879  ;        очистка в LS номера предыдущей ошибки.
;11880  ;     2) Подготовка маски ошибки для проверки только бита 20 (ILL UB OPER).
;11881  ;        очистка в PSL бита COMPAT MODE (бит 31).
;11882  ;     3) Выполнение инструкции MEM.REQ с функцией памяти MF=WRITE.P и типом дан-
;11883  ;        ных DT=длинное слово. Использование в качестве адреса ячейки LS, содер-
;11884  ;        жащей все единицы, за исключением бита 0.
;11885  ;     4) Выполнение инструкции WRITE.MEM.LS с данными, состоящими из нулей. За-
;11886  ;        тем чтение CSR и проверка на 1 бита ILL UB OPER.
;11887  ;     5) Повторение шагов 3 и 4 при DT=слово (в шаге 3) и проверка бита ILL UB
;11888  ;        OPER на 0. Эта операция представляет собой разрешенное обращение к общей
;11889  ;        шине.
;11890  ;     6) Повторение шагов 3 и 4 при DT=слово с адресом = все 1. Это неразрешенное
;11891  ;        обращение к общей шине (невыравненное слово).
;11892  ;     7) Повторение шагов 3 и 4 при DT=байт с адресом = все 1. Это допустимое об-
;11893  ;        ращение к общей шине (данные типа байта не требуют выравнивания).
;11894  ;     8) Повторение шагов 3 и 4 при DT= байт с адресом, содержащим все единицы,
;11895  ;        за исключением бита 0. Это допустимое обращение к общей шине (выравнен-
;11896  ;        ный байт).
;11897  ;     9) Установка в PSL бита режима совместимости COMPAT MODE (бит 31).
;11898  ;    10) Выполнение инструкции MEM.REQ с функцией памяти MF=WRITE.P и типом данных
;11899  ;        DT=слово. Использование в качестве адреса ячейки LS, содержащей все 0,
```

```
;11900  ;       за исключением бита 0.
;11901  ;   11) Выполнение инструкции WRITE.MEM LS с данными, содержащими 0. Затем чте-
;11902  ;       ние CSR и проверка на 1 бита ILL UB OPER (запись невыравненного слова в
;11903  ;       режиме совместимости).
;11904  ;   12) Очистка в PSL бита COMPAT MODE и повторение шагов 9 и 10. Проверка бита
;11905  ;       ILL UB OPER на 0 (запись невыравненного слова в собственом режиме допус-
;11906  ;       кается).
;11907  ;
;11908  ; ОШИБКИ:
;11909  ;
;11910  ;   ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными является бит 20 CSR 1 (OP ERR).
;11911  ;
;11912  ;   ошибка 1 - в CSR не установлен бит ILL UB OPER при записи в общую шину дан-
;11913  ;              ных типа длинного слова.
;11914  ;   ошибка 2 - в CSR не очищен бит ILL UB OPER при записи в общую шину выравнен-
;11915  ;              ного слова.
;11916  ;   ошибка 3 - в CSR не установлен бит ILL UB OPER при записи в общую шину
;11917  ;              невыравненного слова.
;11918  ;   ошибка 4 - в CSR не очищен бит ILL UB OPER при записи в общую шину данных
;11919  ;              типа байта (нечетный адрес).
;11920  ;   ошибка 5 - в CSR не очищен бит ILL UB OPER при записи в общую шину данных
;11921  ;              типа байта (четный адрес).
;11922  ;   ошибка 6 - в CSR не установлен бит ILL UB OPER при записи в матрицу памяти
;11923  ;              невыравненного слова в режиме совместимости.
;11924  ;   ошибка 7 - в CSR не очищен бит ILL UB OPER при записи в матрицу памяти невы-
;11925  ;              равненного слова в собственном режиме.
;11926  ;
;11927  ; НАЛАДКА:
;11928  ;
;11929  ;       ОШИБКА 1 - Эта ошибка указывает на неисправность, связанную с ПМЛ ДЕШ.
;11930  ;   ФИЗИЧЕСКОГО АДР.А, с ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ, ПМЛ УПР.РЕГИСТРОМ ДАННЫХ
;11931  ;   памяти или ПМЛ CSR 1A. Остановите центральный процессор в пошаговом режиме
;11932  ;   на инструкции MEM.REQ с MF=WRITE.P. Микрокод памяти будет выполнять цикл в
;11933  ;   ожидании CPU DATA REQ. Проверьте на низкий уровень сигнал UB PH ADDR SEL L
;11934  ;   на выходе ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А. Если он неправильный, проверьте на вы-
;11935  ;   сокие уровни входы от PA 23 H до PA 18 H. Если эти входы правильные, то ПМЛ
;11936  ;   неисправна.
;11937  ;       Если сигнал UB PH ADDR SEL L низкий, проверьте его на входе ПМЛ УПР.РЕ-
;11938  ;   ГИСТРОМ ДАННЫХ памяти. Выходящий из этой ПМЛ сигнал OP ERR L должен быть
;11939  ;   низким. Если нет, проверьте низкий уровень на входах RS1 L и UB PH ADDR
;11940  ;   SEL L. Если эти входы правильные, то ПМЛ неисправна.
;11941  ;       Если сигнал RS1 L является высоким, проверьте его на выходе ПМЛ УПР. СДВИ-
;11942  ;   ГАТЕЛЕМ ДАННЫХ. Если там сигнал неправильный, проверьте высокий уровень на
;11943  ;   входе CPH/UBL и высокий уровень на L DT1 H. Сигнал L DT1 H, если он непра-
;11944  ;   вильный, можно проследить назад через буфер-защелку до модуля DAP. Если вхо-
;11945  ;   ды правильные, то вероятно, что неисправна ПМЛ.
;11946  ;       Если сигнал OP ERR L на выходе ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти является
;11947  ;   низким, то проверьте его на входе к ПМЛ CSR 1A. Если там правильно, то веро-
;11948  ;   ятно, что неисправна ПМЛ CSR 1A.
;11949  ;
;11950  ;       ОШИБКА 2 - Эта ошибка указывает на неисправность, связанную с ПМЛ УПР.СДВИ-
;11951  ;   ГАТЕЛЕМ ДАННЫХ, ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти или ПМЛ CSR 1A. Остановите
;11952  ;   центральный процессор в пошаговом режиме на инструкции MEM.REQ с MF=WRITE.P.
;11953  ;   микрокод памяти будет выполнять цикл в ожидании CPU DATA REQ. Проверьте на вы-
;11954  ;   сокий уровень выходной сигнал OP ERR L из ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти. Ес-
```

;11955 ; ли он неправильный, проверьте высокий уровень на входах RS1 L и A0 L. Если они
;11956 ; правильные, то вероятно, что неисправна ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти.
;11957 ; Если какой-либо из сигналов RS1 L или A0 L имеет низкий уровень, проверьте
;11958 ; их на выходе из ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если эти сигналы там неправиль-
;11959 ; ные, проверьте на низкий уровень входы L DT1 H и LVA 00 H. Сигнал L DT1 H,
;11960 ; если он неправильный, можно проследить назад к 8-битовому буферу-защелке,
;11961 ; который его генерирует.
;11962 ; Если сигнал OP ERR L на выходе ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти является
;11963 ; высоким, проверьте его на входе к ПМЛ CSR 1A. Если этот сигнал там правиль-
;11964 ; ный то вероятно, что неисправна ПМЛ CSR 1A.
;11965 ;
;11966 ; ОШИБКА 3 - Эта ошибка указывает на неисправность, связанную либо с ПМЛ
;11967 ; УПР.СДВИГАТЕЛЕМ ДАННЫХ, либо с ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти. Остановите
;11968 ; центральный процессор в пошаговом режиме на инструкции MEM.REQ с MF=WRITE.P.
;11969 ; Микрокод памяти будет выполнять цикл в ожидании CPU DATA REQ. Проверьте на
;11970 ; низкий уровень сигнал OP ERR L на выходе ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти.
;11971 ; Если он неправильный, проверьте низкие уровни на входах RS0 L и A0 L. Если
;11972 ; они правильные, тогда вероятно, что неисправна ПМЛ УПР.РЕГИСТРОМ ДАННЫХ па-
;11973 ; мяти.
;11974 ; Если либо сигнал RS0 L, либо A0 L является высоким, проверьте их на выхо-
;11975 ; де ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если сигналы там неправильные, проверьте высо-
;11976 ; кие уровни на входах LVA0 H и L DT0 H. Если они правильные, то неисправна
;11977 ; ПМЛ УПР СДВИГАТЕЛЕМ ДАННЫХ. Сигнал L DT0 H, если он неправильный, можно про-
;11978 ; следить назад к 8-битовому буферу-защелке, который его генерирует.
;11979 ;
;11980 ОШИБКА - Эта ошибка указывает на неисправность, связанную либо с ПМЛ УПР.
;11981 СДВИГАТЕЛЕМ ДАННЫХ, либо с ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти. Остановите цент-
;11982 ; ральный процессор в пошаговом режиме на инструкции MEM.REQ с MF=WRITE.P. Мик-
;11983 ; рокод памяти будет выполнять цикл в ожидании CPU DATA REQ. Проверьте на высо-
;11984 ; кий уровень выходной сигнал OP ERR L из ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти. Если
;11985 ; он неправильный, проверьте высокий уровень на входах RS1 L и RS0 L. Если они
;11986 ; правильные, тогда вероятно, что неисправна ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти.
;11987 ; Если хотя бы один из сигналов RS1 L или RS0 L является низким, проверьте
;11988 их на выходе из ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если сигналы там неправильные,
;11989 ; проверьте низкий уровень на входах L DT0 H и L DT1 H. Если они правильные,
;11990 ; то вероятно, что неисправна ПМЛ. Сигнал L DT0 H, если он неправильный, можно
;11991 ; проследить назад к 8-мибитовому буферу-защелке, который генерирует его.
;11992 ;
;11993 ; ОШИБКА 5 - То же, что и при ошибке . Если это первая ошибка, следует подо-
;11994 ; зревать ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти.
;11995 ;
;11996 ОШИБКА 6 - Эта ошибка указывает на неисправность, связанную с ПМЛ УПР.
;11997 РЕГИСТРОМ ДАННЫХ памяти или ее входов. COMPAT MODE H. Остановите центральный
;11998 процессор в пошаговом режиме на инструкции MEM.REQ с MF=WRITE.P. Микрокод
;11999 памяти будет выполнять цикл в ожидании CPU DATA REQ. Проверьте на низкий уро-
;12000 вень сигнал OP ERR L на выходе из ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти. Если он
;12001 ; неправильный, проверьте на высокий уровень вход COMPAT MODE H. Если он пра-
;12002 ; вильный, тогда вероятно, что ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти неисправна (дру-
;12003 ; гие входы проверялись ранее).
;12004 Если сигнал COMPAT MODE H низкий, это можно проследить назад до модуля DAP.
;12005 Он ранее проверялся в модуле DAP. на выходе из модуля
;12006 ;
;12007 ОШИБКА 7 - Эта ошибка указывает неисправность, связанную с ПМЛ УПР.
;12008 РЕГИСТРОМ ДАННЫХ памят ее входами ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А. Останови-
;12009 те цен ный процесс на инст =WRITE.P.

```
                ;12010  ;   Микрокод памяти будет выполнять цикл в ожидании CPU DATA REQ. Проверьте выход-
                ;12011  ;   ной сигнал OP ERR L из ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти на высокий уровень.
                ;12012  ;   Если он неправильный, проверьте входы на низкий уровень для COMPAT MODE H и
                ;12013  ;   на высокий уровень для UB PH ADDR SEL L. Если они правильные, то вероятно,
                ;12014  ;   что неисправна ПМЛ УПР.РЕГИСТРОМ ДАННЫХ памяти.
                ;12015  ;      Если сигнал COMPAT MODE H высокий, его можно проследить назад к модулю DAP.
                ;12016  ;   Он был проверен ранее в модуле DAP, но не на выходе из модуля.
                ;12017  ;      Если сигнал UB PH ADDR SEL L низкий, проверьте его на выходе ПМЛ ДЕШ.ФИЗИ-
                ;12018  ;   ЧЕСКОГО АДР.А. Если сигнал там неправильный, проверьте на низкий уровень любую
                ;12019  ;   из линий от PA 23 H до PA 18 H. Если эти линии правильные, то ПМЛ неисправна
                ;12020  ;   (низкий уровень на любой из линий от PA 23 до PA 18 сформирует высокий уро-
                ;12021  ;   вень на UB PH ADDR SEL L).
                ;12022
                ;12023  T.1B:
U 0FB2, B65E,15 ;12024          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;12025                                                  ; состояния
U 0FB3, 3E80,15 ;12026          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;12027                                                  ; 15 указывает для консольного процессора начало теста
U 0FB4, 10E0,15 ;12028          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;12029  WAIT.T1B.0:
U 0FB5, 08FB,54 ;12030          JMP [WAIT.T1B.0]                        ; цикл для ожидания ответа консольного процессора
U 0FB6, 0A1A,AC ;12031          JSR [SETUP.1]                           ; установка масок, кода модуля и т.д.
U 0FB7, 4742,15 ;12032          BIS LS[CPU] TO WR[0]                    ; а также установка бита 1 для указания модуля DAP
U 0FB8, 3E8C,15 ;12033          MOV WR[0] TO LS[MODULE.NUM]             ; запоминание в LS кода модуля для индикации плат MCT и
                ;12034                                                  ; DAP
U 0FB9, DF68,15 ;12035          MCOM LS[OP.ERR] TO WR[0]                ; очистка в WR0 BITA 20
U 0FBA, 3E8A,15 ;12036          MOV WR[0] TO LS[ERROR.MASK]             ; установка маски ошибки для проверки только бита 20
U 0FBB, 2F80,15 ;12037          CLR WR[0]                               ; используется для загрузки PSL
U 0FBC, BFFE,15 ;12038          MOV WR[0] TO LS[PSL.HW]                 ; очистка PSL для гарантии, что бит COMPAT MODE очищен
U 0FBD, DF40,15 ;12039          MCOM LS[BIT0] TO WR[0]                  ; занесение в WR0 значения FFFFFFFE(H)
U 0FBE, BE12,15 ;12040          MOV WR[0] TO LS[T9]                     ; LS 9 содержит адрес
                ;12041  LOOP.T1B.1:
U 0FBF, 3668,95 ;12042          MOV LS[OP.ERR] TO WR[1]                 ; ожидаемые данные (ошибка ILL UB OP)
U 0FC0, 9912,75 ;12043          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи длинного слова в устройство на общей
                ;12044                                                  ; шине
U 0FC1, B29C,15 ;12045          WRITE.MEM LS[ZERO]                      ; запись в общую шину неразрешенного длинного слова
U 0FC2, 9D45,75 ;12046          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FC3, 3022,15 ;12047          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FC4, 0869,3C ;12048          JSR [CHECK.RESULT]                      ; проверка на единицу бита ошибки ILL UB OP
U 0FC5, 08FB,F4 ;12049          JMP [LOOP.T1B.1]                        ; цикл при ошибке, если разрешено
U 0FC6, FF82,15 ;12050          INC LS[ERROR.NUMBER]                    ; ошибка 2
                ;12051  LOOP.T1B.2:
U 0FC7, 2F82,95 ;12052          CLR WR[1]                               ; ожидаемые данные (бит ILL UB OP очищен)
U 0FC8, 1912,35 ;12053          MEM.REQ[WRITE.P] ADRS[T9] DT[WORD]      ; запрос для записи слова в общую шину
U 0FC9, B29C,15 ;12054          WRITE.MEM LS[ZERO]                      ; запись в общую шину допустимого слова
U 0FCA, 9D45,75 ;12055          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FCB, 3022,15 ;12056          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FCC, 0869,3C ;12057          JSR [CHECK.RESULT]                      ; проверка на 0 бита ошибки ILL UB OP
U 0FCD, 08FC,74 ;12058          JMP [LOOP.T1B.2]                        ; цикл при ошибке, если разрешено
U 0FCE, FF82,15 ;12059          INC LS[ERROR.NUMBER]                    ; ошибка 3
                ;12060  LOOP.T1B.3:
U 0FCF, 3668,95 ;12061          MOV LS[OP.ERR] TO WR[1]                 ; ожидаемые данные (ошибка ILL UB OP)
U 0FD0, 999E,35 ;12062          MEM.REQ[WRITE.P] ADRS[ONES] DT[WORD]    ; запрос для записи слова в общую шину
U 0FD1, B29C,15 ;12063          WRITE.MEM LS[ZERO]                      ; запись в общую шину неразрешенного слова
                ;12064                                                  ; (невыравненного на границе слова)
```

```
U 0FD2, 9D45,75 ;12065          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FD3, 3022,15 ;12066          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FD4, 0869,3C ;12067          JSR [CHECK.RESULT]                      ; проверка на единицу бита ошибки ILL UB OP
U 0FD5, 88FC,F4 ;12068          JMP [LOOP.T1B.3]                        ; цикл при ошибке, если разрешено
U 0FD6, FF82,15 ;12069          INC LS[ERROR.NUMBER]                    ; ошибка 4
                ;12070  LOOP.T1B.4:
U 0FD7, 2F82,95 ;12071          CLR WR[1]                               ; ожидаемые данные (бит ILL UB OP очищен )
U 0FD8, 199E,15 ;12072          MEM.REQ[WRITE.P] ADRS[ONES] DT[BYTE]    ; запрос для записи байта в общую шину
U 0FD9, B29C,15 ;12073          WRITE.MEM LS[ZERO]                      ; запись в общую шину разрешенного байта (не требуется
                ;12074                                                  ; выравнивание на границе слова)
U 0FDA, 9D45,75 ;12075          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FDB, 3022,15 ;12076          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FDC, 0869,3C ;12077          JSR [CHECK.RESULT]                      ; проверка на 0 бита ошибки ILL UB OP
U 0FDD, 88FD,74 ;12078          JMP [LOOP.T1B.4]                        ; цикл при ошибке, если разрешено
U 0FDE, FF82,15 ;12079          INC LS[ERROR.NUMBER]                    ; ошибка 5
                ;12080  LOOP.T1B.5:
U 0FDF, 2F82,95 ;12081          CLR WR[1]                               ; ожидаемые данные (бит ILL UB OP очищен)
U 0FE0, 9912,15 ;12082          MEM.REQ[WRITE.P] ADRS[T9] DT[BYTE]      ; запрос для записи байта в общую шину
U 0FE1, B29C,15 ;12083          WRITE.MEM LS[ZERO]                      ; запись в общую шину разрешенного байта (четный адрес)
U 0FE2, 9D45,75 ;12084          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FE3, 3022,15 ;12085          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 0FE4, 0869,3C ;12086          JSR [CHECK.RESULT]                      ; проверка на 0 бита ошибки ILL UB ERR
U 0FE5, 08FD,F4 ;12087          JMP [LOOP.T1B.5]                        ; цикл при ошибке, если разрешено
U 0FE6, FF82,15 ;12088          INC LS[ERROR.NUMBER]                    ; ошибка 6
                ;12089  LOOP.T1B.6:
U 0FE7, 3668,95 ;12090          MOV LS[OP.ERR] TO WR[1]                 ; ожидаемые данные (бит ILL UB ERR установлен)
U 0FE8, 367E,15 ;12091          MOV LS[BIT31] TO WR[0]                  ; установка бита 31
U 0FE9, BFFE,15 ;12092          MOV WR[0] TO LS[PSL.HW]                 ; установка режима совместимости
U 0FEA, 9940,35 ;12093          MEM.REQ[WRITE.P] ADRS[#1] DT[WORD]      ; запрос для записи слова по нечетному адресу памяти
U 0FEB, B29C,15 ;12094          WRITE.MEM LS[ZERO]                      ; запись в память неразрешенного слова (нечетный адрес в
                ;12095                                                  ; режиме совместимости)
U 0FEC, 2F80,15 ;12096          CLR WR[0]                               ; очистка рабочего регистра
U 0FED, BFFE,15 ;12097          MOV WR[0] TO LS[PSL.HW]                 ; очистка в PSL режима совместимости
U 0FEE, 9D45,75 ;12098          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FEF, 3022,15 ;12099          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FF0, 0869,3C ;12100          JSR [CHECK.RESULT]                      ; проверка на 1 бита ошибки ILL UB OP
U 0FF1, 88FE,74 ;12101          JMP [LOOP.T1B.6]                        ; цикл при ошибке, если разрешено
U 0FF2, FF82,15 ;12102          INC LS[ERROR.NUMBER]                    ; ошибка 7
                ;12103  LOOP.T1B.7:
U 0FF3, 2F82,95 ;12104          CLR WR[1]                               ; ожидаемые данные (бит ошибки ILL UB OP очищен)
U 0FF4, 9940,35 ;12105          MEM.REQ[WRITE.P] ADRS[#1] DT[WORD]      ; запрос для записи слова по нечетному адресу
U 0FF5, B29C,15 ;12106          WRITE.MEM LS[ZERO]                      ; разрешенная запись слова в память (нечетный адрес в
                ;12107                                                  ; собственном режиме)
U 0FF6, 9D45,75 ;12108          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 0FF7, 3022,15 ;12109          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 0FF8, 0869,3C ;12110          JSR [CHECK.RESULT]                      ; проверка на 0 бита ошибки ILL UB OP
U 0FF9, 88FF,34 ;12111          JMP [LOOP.T1B.7]                        ; цикл при ошибке, если разрешено
                ;12112  END.T1B:
U 0FFA, 0900,04 ;12113          JMP [T.1C]                              ; во избежание ошибок истинности, следующий тест
                ;12114                                                  ; начинается адресом 1000
```

```
;12115  .PAGE "ТЕСТ 1С - проверка бита ошибки перехода границы при записи (модуль МСТ)"
;12116  ;
;12117  ;  ОПИСАНИЕ ТЕСТА:
;12118  ;
;12119  ;        Этот тест проверяет бит CSR1 WR ACCROSS PG ERR (бит 19). Этот бит ус-
;12120  ;      танавливается, когда запись по определенному виртуальному адресу требует
;12121  ;      двух циклов памяти и пересекает либо границу между страницами (установ-
;12122  ;      лены все биты виртуального адреса с 2 по 8), либо пересекает границу
;12123  ;      системы (установлены все биты с 2 по 29)  Для проверки этого сигнала
;12124  ;      используется микропрограмма функции TEST.V.WCHK. Путь данных следующий:
;12125  ;      выдается инструкция MEM.REQ с функцией памяти MF=TEST.V.WCHK и типом
;12126  ;      данных DT=длинное слово. Микропрограмма для TEST.V.WCHK генерирует сиг-
;12127  ;      нал ROT CLK, который стробирует ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Эта ПМЛ вы-
;12128  ;      водит сигнал 2 MEM CYCLES, который возбуждается, если при записи длин-
;12129  ;      ного слова являются не нулевыми два младших бита (LVA 00 и LVA 01) или
;12130  ;      если при записи слова оба младших бита имеют высокий уровень. Сигнал
;12131  ;      2 MEM CYCLES L поступает в ПМЛ CSR 1B. Вход PAGE BOUNDARY H поступает
;12132  ;      из регистра виртуального адреса VAR. Если установлены все биты с VAR 02
;12133  ;      по VAR 08, сигнал PAGE BOUNDARY H становится высоким, и вызовет установ-
;12134  ;      ку бита ошибки WR ACCROSS PG ERR в случае, когда возбужден сигнал 2 MEM
;12135  ;      CYCLES L. После инструкции MEM.REQ выдается MOV MEM.DATA TO WR для за-
;12136  ;      вершения цикла памяти и считывется CSR для проверки бита WR ACCROSS PG
;12137  ;      ERR.
;12138  ;        Бит WR ACCROSS PG ERR устанавливается также, если запись вызывает из-
;12139  ;      менение битов адреса 30 или 31. Это происходит, когда возбуждены сигналы
;12140  ;      2 MEM CYCLES L и SYS ADDR VIOL H. Сигнал SYS ADDR VIOL H поступает из
;12141  ;      логических схем регистра виртуального адреса и возбуждается, если уста-
;12142  ;      новлены все биты с 02 по 29.
;12143  ;        Тест проверяет этот бит при обращении с данными типа байта, слова и
;12144  ;      длинного слова при различных комбинациях LVA 00 и LVA 01 с целью удосто-
;12145  ;      вериться, что он срабатывает правильно.
;12146  ;
 12147  ;  ПРЕДПОЛОЖЕНИЯ:
;12148  ;
;12149  ;      Принимается, что все предыдущие тесты прошли успешно.
;12150  ;
;12151  ;  ШАГИ ТЕСТА:
;12152  ;
;12153  ;      1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок)
;12154  ;         и очистка в LS номера предыдущей ошибки.
;12155  ;      2) Подготовка маски ошибок для проверки только бита 19 (WR ACCROSS PG
;12156  ;         ERR) и запись в CSR1 для установки MME (разрешение диспетчера памяти).
;12157  ;      3) Выполнение инструкции MEM.REQ с функцией памяти MF=TEST.V.WCHK и
;12158  ;         DT=длинное слово. Используется ячейка LS, содержащая единицы в битах
;12159  ;         с 2 по 8 (вызывает установку сигнала PAGE BOUNDARY H) и нули в битах
;12160  ;         0 и 1.
;12161  ;      4) Выполнение инструкции MOV MEM.DATA TO WR[0] для завершения цикла
;12162  ;         TEST.V.WCHK, но без проверки данных.
;12163  ;      5) Чтение CSR и проверка на низкий уровень бита WR ACCROSS PG ERR.
;12164  ;      6) Повторение шагов 3 и 4 с тем отличием, что биты 0 и 1 содержат 01(B).
;12165  ;         чтение CSR и проверка бита WR ACCROSS PG ERR на высокий уровень
;12166  ;         (установлен).
;12167  ;      7) Повторение шага 6 со значением 10(B) в битах 1 и 0 и со значением тех
;12168  ;         же битов 11(B).
;12169  ;      8) Повторение шагов с 3 по 5 с типом данных DT=слово при 10(B) в битах
```

```
;12170  ;          1 и 0.
;12171  ;       9) Повторение шагов с 3 по 5 с типом данных DT=байт при 11(B) в битах
;12172  ;          1 и 0.
;12173  ;      10) Выполнение инструкции MEM.REQ с функцией памяти MF=TEST.V.WCHK и ти-
;12174  ;          пом данных DT=длинное слово. Используется ячейка LS, содержащая еди-
;12175  ;          ницы в битах с 02 по 0B, за исключением "БЕГУЩЕГО" 0 в одном бите.
;12176  ;          Биты 1 и 0 содержат 11(B) для возбуждения двухцикловой записи
;12177  ;          (2 MEM CYCLES).
;12178  ;      11) Выполнение инструкции MOV MEM.DATA TO WR[0] для завершения цикла
;12179  ;          функции TEST.V.WCHK, но без проверки данных.
;12180  ;      12) Чтение CSR и проверка бита WR ACCROSS PG ERR на низкий уровень (очи-
;12181  ;          щенный). Повторение шагов с 10 по 12,  пока 0 не будет сдвинут через
;12182  ;          все биты VAR с 02 по 0B.
;12183  ;      13) Запись 1 в бит TB PAR DIAG в CSR1 для принудительной установки ошибки
;12184  ;          паритета при каждой попытке обращения, использующей буфер трансляции
;12185  ;          TB.
;12186  ;      14) Выполнение инструкции MEM.REQ с функцией памяти MF=WRITE.V.NOCHK
;12187  ;          при типе данных DT=длинное слово. Используется ячейка LS, содержащая
;12188  ;          единицы в битах с 0 по 8.
;12189  ;      15) Выполнение инструкции WRITE.MEM.LS для завершения цикла WRITE.V.NOCHK.
;12190  ;      16) Чтение CSR и проверка бита WR ACCROSS PG ERR на низкий уровень (очи-
;12191  ;          щенный). (Функция NOCHK не должна вызывать бита WR ACCROSS PG ERR).
;12192  ;      17) Повторение шагов 14 и 15 с единицами в битах с 0 по 29 в шаге 14
;12193  ;          (этим вызывается установка сигнала SYS ADDR VIOL H). Проверка бита
;12194  ;          WR ACCROSS PG ERR на высокий уровень (установленный).
;12195  ;
;12196  ;  ОШИБКИ:
;12197  ;
;12198  ;       ПРИМЕЧАНИЕ: Ожидаемыми и полученными данными является бит 19 CSR1
;12199  ;       (WR XPG ERR). Данные под OTHER содержат адрес обращения.
;12200  ;
;12201  ;       ошибка 1 - бит WR ACCROSS PG ERR установлен после команды TEST.V.WCHK при
;12202  ;                  проверке записи выравненного длинного слова.
;12203  ;       ошибка 2 - бит WR ACCROSS PG ERR не установлен после команды TEST.V.WCHK
;12204  ;                  при проверке записи невыравненного длинного слова.
;12205  ;       ошибка 3 - бит WR ACCROSS PG ERR установлен после команды TEST.V.WCHK при
;12206  ;                  проверке записи выравненного слова.
;12207  ;       ошибка 4 - бит WR ACCROSS PG ERR установлен после команды TEST.V.WCHK при
;12208  ;                  проверке записи байта.
;12209  ;       ошибка 5 - бит WR ACCROSS PG ERR установлен после команды TEST.V.WCHK при
;12210  ;                  проверке записи невыравненного длинного слова без пересечения
;12211  ;                  границы страниц.
;12212  ;       ошибка 6 - бит WR ACCROSS PG ERR установлен после команды WRITE.V.NOCHK
;12213  ;                  для невыравненного длинного слова.
;12214  ;       ошибка 7 - бит WR ACCROSS PG ERR не установлен после команды TEST.V.NOCHK
;12215  ;                  при пересечении границ системы (SYS ADDR VIOL).
;12216  ;
;12217  ;  НАЛАДКА:
;12218  ;
;12219  ;       ПРИМЕЧАНИЕ: Поле OTHER (другие данные) содержит адрес, используемый
;12220  ;       с командой TEST.V.WCHK.
;12221  ;
;12222  ;          ОШИБКА 1 - Эта ошибка указывает на неисправность в ПМЛ УПР.СДВИГАТЕ-
;12223  ;       ЛЕМ ДАННЫХ или ПМЛ CSR 1B. Остановите центральный процессор в пошаговом
;12224  ;       режиме на инструкции MEM.REQ с MF=TEST.V.WCHK. Микрокод памяти будет вы-
```

```
;12225  ;       полнять цикл в ожидании снятия CPU GRANT (это произойдет после выполне-
;12226  ;       ния инструкции MOV MEM.DATA TO WR). Проверьте на высокий уровень сигнал
;12227  ;       2 MEM CYCLES L на входе ПМЛ CSR 1B. Если он правильный, выход BUS MC D19
;12228  ;       H должен иметь низкий уровень. Если нет, то вероятно, что ПМЛ CSR 1B не-
;12229  ;       исправна.
;12230  ;          Если сигнал 2 MEM CYCLES L на входе ПМЛ CSR 1B низкий, проверьте его
;12231  ;       на выходе ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если там неправильный (низкий),
;12232  ;       проверьте на низкий уровень входы LVA 01 H и LVA 00 H. Если они правиль-
;12233  ;       ные, то вероятно, что неисправна ПМЛ УПР.СДИГАТЕЛЕМ ДАННЫХ. Если нет,
;12234  ;       проследите их назад к схемам регистра виртуального адреса VAR.
;12235  ;
;12236  ;          ОШИБКА 2 – Эта ошибка указывает на неисправность ПМЛ УПР.СДВИГАТЕЛЕМ
;12237  ;       ДАННЫХ, ПМЛ CSR 1B или логических схем регистра VAR, которые генерируют
;12238  ;       сигнал PAGE BOUNDARY H. Остановите центральный процессор в пошаговом ре-
;12239  ;       жиме на инструкции MEM.REQ с MF=TEST.V.WCHK. Микрокод памяти будет выпол-
;12240  ;       нять цикл в ожидании снятия CPU GRANT (это произойдет после выполнения
;12241  ;       инструкции MOV MEM.DATA TO WR). Проверьте на высокий уровень выход BUS
;12242  ;       MC D19 H из ПМЛ CSR 1B. Если он неправильный, проверьте входы 2 MEM
;12243  ;       CYCLES L на низкий уровень и PAGE BOUNDARY H на высокий уровень. Кроме
;12244  ;       того, проверьте, что сигнал RD CSR L не меняется (в этой точке он будет
;12245  ;       высоким, но должен стать низким во время микропрограммы TEST.V.WCHK).
;12246  ;       Если эти входы правильные, то вероятно, что ПМЛ неисправна. Сигнал RD
;12247  ;       CSR L можно проверить на низкий уровень при пошаговой работе MCT (если
;12248  ;       пошаговая работа MCT возможна). Подведите центральный процессор в поша-
;12249  ;       говом режиме к инструкции, предшествующей инструкции MEM.REQ, и перева-
;12250  ;       дите MCT на пошаговый режим. Затем выполните шаг центрального процессо-
;12251  ;       ра к инструкции MEM.REQ и выполните шаги MCT до второго цикла после ад-
;12252  ;       реса начального ветвления. Сигнал RD CSR L должен быть низким в одно и
;12253  ;       то же время с высоким уровнем сигнала CSR ERR SUM H. Если пошаговая ра-
;12254  ;       бота MCT недоступна, сигнал можно проверить во время зацикливания на
;12255  ;       ошибке. Если сигнал 2 MEM CYCLES L высокий, проверьте его на выходе из
;12256  ;       ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Если там он неправильный (высокий), проверь-
;12257  ;       те на высокий уровень входы LVA 00 H или LVA 01 H. Если хотя бы один из
;12258  ;       них высокий, то вероятно, что ПМЛ неисправна. Если эти входы неправиль-
;12259  ;       ные, то их можно проследить назад к регистру виртуального адреса VAR.
;12260  ;       Если сигнал PAGE BOUNDARY H является низким на входе ПМЛ CSR 1B, проверь-
;12261  ;       те его на выходе VAR. Если он там неправильный (низкий), можно подозре-
;12262  ;       вать ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА.
;12263  ;
;12264  ;          ОШИБКА 3 – Эта ошибка с наибольшей вероятностью указывает на неисправ-
;12265  ;       ность в самой ПМЛ УПР.СДВИГАТЕЛЕМ ДАННЫХ. Входы были проверены предыдущи-
;12266  ;       ми тестами.
;12267  ;
;12268  ;          ОШИБКА 4 – То же, что и при ошибке 3.
;12269  ;
;12270  ;          ОШИБКА 5 – Эта ошибка указывает на неисправность ПМЛ CSR 1B или логи-
;12271  ;       ческих схем регистра виртуального адреса VAR, которы генерируют сигналы
;12272  ;       PAGE BOUNDARY H и SYS ADDR VIOL H. Остановите центральный процессор в по-
;12273  ;       шаговом режиме на инструкции MEM.REQ с MF=TEST.V.WCHK. Микрокод памяти
;12274  ;       будет выполнять цикл в ожидании снятия CPU GRANT (это произойдет после
;12275  ;       выполнения инструкции MOV MEM.DATA TO WR). Проверьте на низкие уровни
;12276  ;       входы ПМЛ CSR 1B PAGE BOUNDARY H и SYS ADDR VIOL H. Если они правильные,
;12277  ;       то вероятно, что неисправна ПМЛ CSR 1B.
;12278  ;          Если сигнал PAGE BOUNDARY H высокий, то следует подозревать ПМЛ РЕГ.
;12279  ;       ВИРТУАЛЬНОГО АДРЕСА, которая генерирует этот сигнал. Если сигнал SYS ADDR
```

```
                    ;12280  ;       VIOL H является высоким, тогда проверьте его на выходе ПМЛ РЕГ.ВИРТУАЛЬ-
                    ;12281  ;       НОГО АДРЕСА, которая генерирует этот сигнал. Если там этот сигнал высо-
                    ;12282  ;       кий, подозревается ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА.
                    ;12283  ;
                    ;12284  ;          ОШИБКА 6 - Эта ошибка указывает на неисправность ПМЛ CSR 1B или ее вхо-
                    ;12285  ;       да RD CSR L. Остановите центральный процессор в пошаговом режиме на инст-
                    ;12286  ;       рукции MEM.REQ с MF=WRITE.V.NOCHK. Микрокод памяти будет выполнять цикл в
                    ;12287  ;       ожидании снятия CPU GRANT (это произойдет после выполнения инструкции
                    ;12288  ;       WRITE.MEM LS). Проверьте на высокий уровень сигнал RD CSR L. Если он
                    ;12289  ;       правильный, то вероятно, что неисправна ПМЛ CSR 1B.
                    ;12290  ;
                    ;12291  ;          ОШИБКА 7 - Эта ошибка указывает на неисправность ПМЛ CSR 1B или логи-
                    ;12292  ;       ческих схем регистра VAR, которые генерируют сигнал SYS ADDR VIOL H. Ос-
                    ;12293  ;       тановите центральный процессор в пошаговом режиме на инструкции MEM.REQ
                    ;12294  ;       С MF=WRITE.V.NOCHK. Микрокод памяти будет выполнять цикл в ожидании сня-
                    ;12295  ;       тия CPU GRANT (это произойдет после выполнения инструкции WRITE.MEM LS).
                    ;12296  ;       Проверьте на высокий уровень вход SYS ADDR VIOL H ПМЛ CSR 1B. Если он
                    ;12297  ;       правильный, то вероятно, что неисправна ПМЛ.
                    ;12298  ;          Если сигнал SYS ADDR VIOL H низкий, проверьте его на выходе из ПМЛ
                    ;12299  ;       РЕГ.ВИРТУАЛЬНОГО АДРЕСА, которая его генерирует. Если выход имеет низкий
                    ;12300  ;       уровень, то вероятно, что ПМЛ РЕГ.ВИРТУАЛЬНОГО АДРЕСА неисправна.
                    ;12301  ;
                    ;12302  ;
                    ;12303  ;
                    ;12304   1000:                                  ; запуск с адреса 1000 для предотвращения
                    ;12305                                          ; ошибки истинности
                    ;12306   T.1C:
U 1000, B65E,15     ;12307           MOV LS[BEGIN.TEST] TO WR[0]    ; установка в WR0 бита 15 для слова управления и
                    ;12308                                          ; состояния
U 1001, 3EB0,15     ;12309           MOV WR[0] TO LS[CONTROL.STATUS] ; установка бита 15 в слове управления и состояния. Бит
                    ;12310                                          ; 15 указывает для консольного процессора начало теста
U 1002, 10E0,15     ;12311           MISC [SET.CP.ATTN]             ; выдача для консольного процессора CPU ATTN
                    ;12312   WAIT.T1C.0:
U 1003, 0900,34     ;12313           JMP [WAIT.T1C.0]               ; цикл для ожидания ответа консольного процессора
U 1004, 0A1A,AC     ;12314           JSR [SETUP.1]                  ; установка маски, кода модуля и т.д.
U 1005, 5F66,15     ;12315           MCOM LS[WR.ACROSS.PG] TO WR[0] ; очистка в рабочем регистре бита 19
U 1006, 3E8A,15     ;12316           MOV WR[0] TO LS[ERROR.MASK]    ; установка маски для проверки бита ошибки перехода
                    ;12317                                          ; границы страниц при записи
U 1007, 0A17,DC     ;12318           JSR [WRITE.CSR1.MME]           ; разрешение диспетчера памяти
U 1008, B670,15     ;12319           MOV LS[OTHER.DATA] TO WR[0]    ; установка бита для печати в сообщении об ошибке под
                    ;12320                                          ; OTHER (другие данные)
U 1009, 3EB0,15     ;12321           MOV WR[0] TO LS[CONTROL.STATUS] ; запись слова управления и состояния
U 100A, B626,15     ;12322           MOV LS[#FF] TO WR[0]           ; установка в рабочем регистре битов 0-7
U 100B, 4750,15     ;12323           BIS LS[BIT8] TO WR[0]          ; установка бита 8
U 100C, C5C0,15     ;12324           BIC LS[#3(H)] TO WR[0]         ; очистка битов 0 и 1
U 100D, BE12,15     ;12325           MOV WR[0] TO LS[T9]            ; загрузка в LS адреса для инструкции MEM.REQ
U 100E, BE88,15     ;12326           MOV WR[0] TO LS[ADDRESS.DATA]  ; адрес для печати под OTHER в сообщении об ошибке
U 100F, 2F87,95     ;12327           CLR WR[3]                      ; ожидаемые данные (бит WR ACCROSS PG ERR очищен)
U 1010, 0905,EC     ;12328           JSR [LOOP.T1C.1.2]             ; выполнение теста с адресом 1FC
U 1011, FF82,15     ;12329           INC LS[ERROR.NUMBER]           ; ошибка 2
U 1012, FF12,15     ;12330           INC LS[T9]                     ; адрес теперь содержит 01(B) в двух младших битах
U 1013, FF88,15     ;12331           INC LS[ADDRESS.DATA]           ; адрес для печати под OTHER (другие данные)
U 1014, 3667,95     ;12332           MOV LS[WR.ACROSS.PG] TO WR[3]  ; ожидаемые данные (бит WR ACCROSS PG ERR установлен)
U 1015, 0905,EC     ;12333           JSR [LOOP.T1C.1.2]             ; выполнение теста с адресом 1FD
U 1016, FF12,15     ;12334           INC LS[T9]                     ; адрес теперь содержит 10(B) в младших двух битах
```

```
U 1017, FF88,15 ;12335          INC LS[ADDRESS.DATA]                    ; адрес для печати под OTHER (другие данные)
U 1018, 0905,EC ;12336          JSR [LOOP.T1C.1.2]                      ; выполнение теста с адресом 1FE
U 1019, FF12,15 ;12337          INC LS[T9]                              ; адрес теперь содержит 11(B) в двух младших битах
U 101A, FF88,15 ;12338          INC LS[ADDRESS.DATA]                    ; адрес для печати под OTHER
U 101B, 0905,EC ;12339          JSR [LOOP.T1C.1.2]                      ; выполнение теста с адресом 1FF
U 101C, FF82,15 ;12340          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 101D, FC12,15 ;12341          DEC LS[T9]                              ; адрес теперь содержит 10(B) в младших двух битах
U 101E, FC88,15 ;12342          DEC LS[ADDRESS.DATA]                    ; адрес для печати под OTHER
                ;12343  LOOP.T1C.3:
U 101F, 2F82,95 ;12344          CLR WR[1]                               ; ожидаемые данные (бит WR ACCROSS PAGE ERR очищен)
U 1020, 9912,B5 ;12345          MEM.REQ[TEST.V.WCHK] ADRS[T9] DT[WORD]  ; запрос для проверки записи на границе выравненного
                ;12346                                                  ; слова
U 1021, 3022,15 ;12347          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла, но
                ;12348                                                  ; без выборки результата
U 1022, 9D45,75 ;12349          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1023, 3022,15 ;12350          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 1024, 0869,3C ;12351          JSR [CHECK.RESULT]                      ; проверка, что бит CSR1 WR ACCROSS PG ERR очищен
U 1025, 8901,F4 ;12352          JMP [LOOP.T1C.3]                        ; цикл при ошибке, если разрешено
U 1026, FF82,15 ;12353          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 1027, FF12,15 ;12354          INC LS[T9]                              ; адрес теперь содержит 11(B) в двух младших битах
U 1028, FF88,15 ;12355          INC LS[ADDRESS.DATA]                    ; адрес для печати под OTHER
                ;12356  LOOP.T1C.4:
U 1029, 2F82,95 ;12357          CLR WR[1]                               ; ожидаемые данные (бит WR ACCROSS PG ERR очищен)
U 102A, 1912,95 ;12358          MEM.REQ[TEST.V.WCHK] ADRS[T9] DT[BYTE]  ; запрос для проверки записи байта на границе
U 102B, 3022,15 ;12359          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла, но
                ;12360                                                  ; без проверки результата
U 102C, 9D45,75 ;12361          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 102D, 3022,15 ;12362          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 102E, 0869,3C ;12363          JSR [CHECK.RESULT]                      ; проверка, что бит CSR1 WR ACCROSS PG ERR очищен
U 102F, B902,94 ;12364          JMP [LOOP.T1C.4]                        ; цикл при ошибке, если разрешено
U 1030, FF82,15 ;12365          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 1031, B613,15 ;12366          MOV LS[T9] TO WR[2]                     ; установлены биты 0-8
U 1032, 4543,15 ;12367          BIC LS[BIT1] TO WR[2]                   ; очистка бита 1
                ;12368  REPEAT.T1C.5:
U 1033, 23C1,15 ;12369          ROL WR[2]                               ; сдвиг тестовых данных на 1 бит влево (сдвиг нуля
                ;12370                                                  ; через биты 2-8)
U 1034, C553,15 ;12371          BIC LS[BIT9] TO WR[2]                   ; очистка бита 9, если установлен
U 1035, 4741,15 ;12372          BIS LS[BIT0] TO WR[2]                   ; установка бита 0
U 1036, 3E13,15 ;12373          MOV WR[2] TO LS[T9]                     ; занесение адреса в LS для инструкции MEM.REQ
U 1037, 3E89,15 ;12374          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати под OTHER (другие данные)
                ;12375  LOOP.T1C.5:
U 1038, 2F82,95 ;12376          CLR WR[1]                               ; ожидаемые данные (бит WR ACCROSS PG ERR очищен)
U 1039, 1912,F5 ;12377          MEM.REQ[TEST.V.WCHK] ADRS[T9] DT[LONG]  ; запрос для проверки на границе невыравненного слова
U 103A, 3022,15 ;12378          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла, но
                ;12379                                                  ; без проверки результата
U 103B, 9D45,75 ;12380          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 103C, 3022,15 ;12381          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 103D, 0869,3C ;12382          JSR [CHECK.RESULT]                      ; проверка, что бит CSR1 WR ACCROSS PG ERR очищен
U 103E, 8903,84 ;12383          JMP [LOOP.T1C.5]                        ; цикл при ошибке, если разрешено
                ;12384            BIT LS[BIT8] WITH WR[2],              ; проверка, был ли очищен бит 8
U 103F, 5951,35 ;12385          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 1040, 0903,31 ;12386          JMP.IF[BIT.SET] TO [REPEAT.T1C.5]       ; если бит 8 установлен, проверка с новыми тестовыми данными
U 1041, FF82,15 ;12387          INC LS[ERROR.NUMBER]                    ; ошибка 6
U 1042, B626,15 ;12388          MOV LS[#FF] TO WR[0]                    ; установка битов 0-7
U 1043, 4750,15 ;12389          BIS LS[BIT8] TO WR[0]                   ; теперь установлены биты 0-8
```

```
U 1044, BE12,15 ;12390          MOV WR[0] TO LS[T9]                      ; запоминание в LS для записи
U 1045, BE88,15 ;12391          MOV WR[0] TO LS[ADDRESS.DATA]            ; адрес для печати под OTHER (другие данные)
U 1046, B676,15 ;12392          MOV LS[MME] TO WR[0]                     ; установка бита MME (разрешение диспетчера памяти)
U 1047, B67A,15 ;12393          MOV LS[TB.PAR.DIAG] TO WR[0]             ; и бита TB PARITY DIAG
U 1048, 8A17,FC ;12394          JSR [WRITE.CSR1]                         ; установка этих битов в CSR1
                ;12395  LOOP.T1C.6:
U 1049, 2F82,95 ;12396          CLR WR[1]                                ; ожидаемые данные (бит WR ACCROSS PG ERR очищен)
U 104A, 1B12,75 ;12397          MEM.REQ[WRITE.V.NOCHK] ADRS[T9] DT[LONG] ; запрос для записи на границе невыравненного длинного
                ;12398                                                   ; слова с функцией NOCHK
U 104B, B29C,15 ;12399          WRITE.MEM LS[ZERO]                       ; запись нулей для завершения цикла
U 104C, 9D45,75 ;12400          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; запрос для чтения CSR1
U 104D, 3022,15 ;12401          MOV MEM.DATA TO WR[0]                    ; прием содержимого CSR1
U 104E, 0869,3C ;12402          JSR [CHECK.RESULT]                       ; проверка, что бит CSR1 WR ACCROSS PG ERR очищен
U 104F, B904,94 ;12403          JMP [LOOP.T1C.6]                         ; цикл при ошибке, если есть разрешение
U 1050, FF82,15 ;12404          INC LS[ERROR.NUMBER]                     ; ошибка 7
U 1051, B69E,15 ;12405          MOV LS[ONES] TO WR[0]                    ; занесение в рабочий регистр всех единиц
U 1052, 457C,15 ;12406          BIC LS[BIT30] TO WR[0]                   ; очистка бита 30
U 1053, C57E,15 ;12407          BIC LS[BIT31] TO WR[0]                   ; и бита 31. Теперь установлены биты 0-29
U 1054, BE12,15 ;12408          MOV WR[0] TO LS[T9]                      ; занесение адреса в LS для записи
U 1055, BE88,15 ;12409          MOV WR[0] TO LS[ADDRESS.DATA]            ; адрес для печати под OTHER
                ;12410  LOOP.T1C.7:
U 1056, B666,95 ;12411          MOV LS[WR.ACROSS.PG] TO WR[1]            ; ожидаемые данные (бит WR ACCROSS PG ERR установлен)
U 1057, 1B12,75 ;12412          MEM.REQ[WRITE.V.NOCHK] ADRS[T9] DT[LONG] ; запрос для записи на границе невыравненного длинного
                ;12413                                                   ; слова с функцией NOCHK при нарушении границы системы
U 1058, B29C,15 ;12414          WRITE.MEM LS[ZERO]                       ; запись нулей для завершения цикла
U 1059, 9D45,75 ;12415          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; запрос для чтения CSR1
U 105A, 3022,15 ;12416          MOV MEM.DATA TO WR[0]                    ; прием содержимого CSR1
U 105B, 0869,3C ;12417          JSR [CHECK.RESULT]                       ; проверка, что бит CSR1 WR ACCROSS PG ERR установлен
U 105C, 0905,64 ;12418          JMP [LOOP.T1C.7]                         ; цикл при ошибке, если разрешен
U 105D, 0906,64 ;12419          JMP [END.T1C]                            ; конец теста
                ;12420  LOOP.T1C.1.2:
U 105E, 2006,95 ;12421          MOV WR[3] TO WR[1]                       ; ожидаемое состояние бита WR ACCROSS PG ERR
U 105F, 1912,F5 ;12422          MEM.REQ[TEST.V.WCHK] ADRS[T9] DT[LONG]   ; запрос для проверки на границе выравненного длинного
                ;12423                                                   ; слова
U 1060, 3022,15 ;12424          MOV MEM.DATA TO WR[0]                    ; выдача инструкции MOV для завершения микроцикла, но
                ;12425                                                   ; без проверки результатов
U 1061, 9D45,75 ;12426          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]    ; запрос для чтения CSR1
U 1062, 3022,15 ;12427          MOV MEM.DATA TO WR[0]                    ; прием содержимого CSR1
U 1063, 0869,3C ;12428          JSR [CHECK.RESULT]                       ; проверка в CSR1 бита WR ACCROSS PG ERR
U 1064, 8905,E4 ;12429          JMP [LOOP.T1C.1.2]                       ; цикл при ошибке, если разрешено
U 1065, 5800,14 ;12430          RETURN                                   ; конец проверки, проверка следующего условия
                ;12431  END.T1C:
```

```
;12432  .PAGE "ТЕСТЫ МАТРИЦЫ ОСНОВНОЙ ПАМЯТИ"
;12433  .TOC "ТЕСТ 1D - определение обьема и инициализация (модуль МСТ или модуль памяти)"
;12434  ;
;12435  ; ОПИСАНИЕ ТЕСТА:
;12436  ;
;12437  ;      Этот тест заносит во всю основную память нули, выравненные по границам
;12438  ;   длинного слова. Инициализация необходима для того, чтобы иметь гарантии, что
;12439  ;   контрольные биты записаны правильно, и при чтении из основной памяти будет
;12440  ;   работать коррекция ошибок. По ходу записи в память проверяется бит ERR SUM.
;12441  ;   когда возникает суммарная ошибка системы памяти, считывается CSR1 с целью
;12442  ;   убедиться, что причиной ошибки является NXM (несуществующая память). Этот
;12443  ;   случай соответствует верхнему адресу основной памяти (вершине) и этот обьем
;12444  ;   запоминается в ячейке LS для использования в последующих тестах. Монитор,
;12445  ;   кроме того, печатает обьем памяти в блоках по 256 кбайт к сведению операто-
;12446  ;   ра. Если выполнением управляет APT, тест сравнивает вычисленный обьем с тем,
;12447  ;   который был передан из APT, для гарантии, что логика определения обьема па-
;12448  ;   мяти функционирует правильно.
;12449  ;    Этот тест проверяет, кроме того, "ПРОВАЛЫ" в памяти, продолжая разыскивать
;12450  ;   NXM для блоков по 256 кбайт после того, как был первый раз получен бит NXM,
;12451  ;   до достижения адреса EC0000(H) (следующий блок из 256 кбайт будет иметь ад-
;12452  ;   рес F00000(H), который принадлежит пространству регистров адаптера общей шины).
;12453  ;    ПРИМЕЧАНИЕ: "ПРОВАЛ" в памяти не вызовет неуспешного выполнения последующих
;12454  ;   тестов, однако будет тестироваться только непрерывная память. Любая память
;12455  ;   выше "ПРОВАЛА" игнорируется.
;12456  ;
;12457  ; ПРЕДПОЛОЖЕНИЯ:
;12458  ;
;12459  ;   Принимается, что все предыдущие тесты прошли успешно, в частности, тест логи-
;12460  ;   ки NXM. Если логика NXM не срабатывает, этот тест будет "ЗАВИСАТЬ". Если тес-
;12461  ;   ты логики NXM проходят, этот тест "ЗАВИСАТЬ" не должен.
;12462  ;
;12463  ; ШАГИ ТЕСТА:
;12464  ;
;12465  ;   1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок) и
;12466  ;      очистка в LS номера предыдущей ошибки.
;12467  ;   2) Установка маски ошибки для проверки только бита 16 (бит NXM в CSR1).
;12468  ;   3) Очистка ячейки LS, используемой для хранения размера памяти (MM.LASTADDR+1).
;12469  ;   4) Выполнение инструкции MEM.REQ с функцией памяти WRITE.P и DT=длинное сло-
;12470  ;      во. В качестве адреса используется ячейка LS MM.LASTADDR+1.
;12471  ;   5) Выполнение инструкции WRITE.MEM LS, с использованием в качестве данных
;12472  ;      ячейки LS, содержащей все нули.
;12473  ;   6) Проверка суммарной ошибки (ERROR SUM) и переход к шагу 8, если ошибка об-
;12474  ;      наружена.
;12475  ;   7) Увеличение на 4 значения MM.LASTADDR+1 и переход к шагу 4.
;12476  ;   8) Чтение CSR1 и проверка, что установлен бит NXM. Если нет, загрузка адреса,
;12477  ;      по которому происходило обращение, в поле OTHER (другие данные), сообщение
;12478  ;      об ошибке, очистка MM.LASTADDR+1 и прекращение этого теста. Иначе продол-
;12479  ;      жение.
;12480  ;   9) Проверка, что MM.LASTADDR+1 находится на границе 256 кбайт. Если нет,
;12481  ;      сообщение об ошибке и прекращение теста. Иначе продолжение.
;12482  ;  10) Преобразование значения в MM.LASTADDR+1 в число блоков по 256 кбайт и вы-
;12483  ;      вод в LS 7. Выдача консольному процессору CPU.ATTN для печати этого зна-
;12484  ;      чения.
;12485  ;  11) Если тестом управляет APT, сравнение заданного обьема памяти с обьемом,
;12486  ;      вычисленным этим тестом. В случае несовпадения сообщение об ошибке.
```

```
;12487  ;  12) Изменение кода модуля для индикации платы ОЗУ в случае ошибки.
;12488  ;  13) Запоминание MM.LASTADDR+1 в промежуточной ячейке LS. Повторение шагов
;12489  ;      4 и 5 с тем отличием, что в качестве адреса используется промежуточная
;12490  ;      ячейка LS.
;12491  ;  14) Чтение CSR1 и проверка, что установлен бит NXM. Если нет, загрузка адреса,
;12492  ;      по которому было обращение, в поле OTHER (другие данные) и сообщение об
;12493  ;      ошибке (ниже этого адреса имеется "ПРОВАЛ").
;12494  ;  15) Увеличение промежуточной ячейки на 40000(H) (256 кбайт) и повторение ша-
;12495  ;      гов с 11 по 13, пока не произойдет обращение по адресу EC0000(H).
;12496  ;
;12497  ; ОШИБКИ:
;12498  ;
;12499  ;   ПРИМЕЧАНИЕ: Данные под OTHER (другие данные) представляют собой адрес основ-
;12500  ;   ной памяти, обращение по которому было неуспешным.
;12501  ;
;12502  ;   ошибка 1 - суммарная ошибка получена раньше, чем NXM. Ожидаемыми данными яв-
;12503  ;              ляется установленный бит NXM в CSR1, полученными данными - дейст-
;12504  ;              вительное значение бита NXM в CSR1.
;12505  ;   ошибка 2 - ошибка NXM получена, но не на границе 256 кбайт. Ожидаемыми данными
;12506  ;              являются все нули, полученными данными является адрес с очищенными
;12507  ;              битами 18-31.
;12508  ;   ошибка 3 - обьем памяти не совпадает с тем, какой был задан из APT (эта ошибка
;12509  ;              фиксируется только если выполнением управляет APT). Ожидаемыми дан-
;12510  ;              ными является размер в блоках по 256 кбайт, заданный из APT. Полу-
;12511  ;              ченными данными является размер, определенный тестом.
;12512  ;   ошибка 4 - память содержит "ПРОВАЛ". Бит NXM не устанавливается по адресу,
;12513  ;              превышающему адрес, по которому этот бит уже устанавливался. Ожи-
;12514  ;              даемыми данными является установленный бит NXM в CSR1. Полученными
;12515  ;              данными является действительное значение бита NXM в CSR1.
;12516  ;
;12517  ; НАЛАДКА:
;12518  ;
;12519  ;   ПРИМЕЧАНИЕ: Если МСТ работает в пошаговом режиме, не происходит регенерация
;12520  ;   матрицы ОЗУ. Поэтому данные в модуле памяти будут потеряны.
;12521  ;
;12522  ;      ОШИБКА 1 - Эта ошибка указывает на неисправность в логических схемах ошибок
;12523  ;   CSR. Чтобы попытаться уточнить проблему, выполните снова предыдущие тесты. Мож-
;12524  ;   но прочитать CSR с помощью команды консоли EX MC 1 с целью просмотреть, какой
;12525  ;   бит ошибки был установлен. Адрес, по которому происходило обращение в момент
;12526  ;   возникновения ошибки, печатается в сообщении об ошибке под OTHER (другие данные).
;12527  ;
;12528  ;      ОШИБКА 2 - Эта ошибка указывает, что бит NXM установлен раньше, чем дос-
;12529  ;   тигнута граница 256 кбайт. Одной из возможностей являются помехи в логичес-
;12530  ;   ких схемах NXM. Проверьте наличие помех во время зацикливания на ошибке. По-
;12531  ;   ле OTHER (другие данные) в распечатке ошибки содержит адрес, по которому ус-
;12532  ;   тановлен бит NXM.
;12533  ;
;12534  ;      ОШИБКА 3 - Эта ошибка может произойти только под управлением из APT. Она
;12535  ;   указывает, что обьем, заданный программами APT в области "ПОЧТОВОГО ЯЩИКА"
;12536  ;   не совпадает с обьемом, вычисленным этим тестом. Данные под EXP (ожидаемые)
;12537  ;   представляют собой число блоков по 256 кбайт, которое должно быть найдено,
;12538  ;   как задано из APT. Число под REC (полученные) представляет собой число бло-
;12539  ;   ков по 256 кбайт, обнаруженное этим тестом. Если этот тест проходит неуспеш-
;12540  ;   но, следует подозревать ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А и ДЕШ.ФИЗИЧЕСКОГО АДР.В.
;12541  ;
```

```
                    ;12542  ;       ОШИБКА 4 - Эта ошибка указывает, что в основной памяти существует "ПРОВАЛ".
                    ;12543  ;   адрес под OTHER (другие данные) в сообщении об ошибке представляет собой ад-
                    ;12544  ;   рес, по которому не удалось установить бит NXM, хотя он устанавливался для
                    ;12545  ;   меньших адресов. Наиболее вероятно, что на плате матрицы ОЗУ имеется непра-
                    ;12546  ;   вильная перемычка, или что один из старших битов физического адреса (PA)
                    ;12547  ;   "ЗАЛИП" на низком уровне.
                    ;12548  ;
                    ;12549  T.1D:
U 1066, B65E,15     ;12550          MOV LS[BEGIN.TEST] TO WR[0]                 ; установка в WR0 бита 15 для слова управления и
                    ;12551                                                      ; состояния
U 1067, 3E80,15     ;12552          MOV WR[0] TO LS[CONTROL.STATUS]              ; установка бита 15 в слове управления и состояния. Бит
                    ;12553                                                      ; 15 указывает для конс.процессора начало теста
U 1068, 10E0,15     ;12554          MISC [SET.CP.ATTN]                          ; выдача для конс.процессора CPU ATTN
                    ;12555  WAIT.T1D.0:
U 1069, 0906,94     ;12556          JMP [WAIT.T1D.0]                            ; цикл для ожидания ответа конс.процессора
U 106A, 0A1A,9C     ;12557          JSR [SETUP]                                 ; установка маски, номера модуля и т.д. и очистка CSR1
                    ;12558                                                      ; MCT
U 106B, B670,15     ;12559          MOV LS[OTHER.DATA] TO WR[0]                  ; подготовка для записи слова управления
U 106C, 3E80,15     ;12560          MOV WR[0] TO LS[CONTROL.STATUS]              ; под OTHER (другие данные) консоль будет печатать
                    ;12561                                                      ; адрес
U 106D, 6524,15     ;12562          CLR LS[MM.LASTADDR+1]                       ; очистка ячейки LS (ячейка 12(H)) для хранения адреса
U 106E, 6588,15     ;12563          CLR LS[ADDRESS.DATA]                        ; адрес для печати в случае ошибки
U 106F, B645,15     ;12564          MOV LS[#4] TO WR[2]                         ; увеличение значения для адреса памяти
                    ;12565  LOOP.T1D.1:
U 1070, 5F60,15     ;12566          MCOM LS[NXM] TO WR[0]                       ; очистка в рабочем регистре бита 16
U 1071, 3E8A,15     ;12567          MOV WR[0] TO LS[ERROR.MASK]                  ; проверка в CSR1 только бита NXM
U 1072, B640,15     ;12568          MOV LS[#1] TO WR[0]                          ; занесение 1 в рабочий регистр
U 1073, BE82,15     ;12569          MOV WR[0] TO LS[ERROR.NUMBER]                ; установка в LS номера ошибки.
                    ;12570  REPEAT.T1D.1:
U 1074, 9924,75     ;12571          MEM.REQ[WRITE.P] ADRS[MM.LASTADDR+1] DT[LONG] ; запрос для записи нулей по всем адресам памяти
U 1075, B29C,15     ;12572          WRITE.MEM LS[ZERO]                          ; запись в память всех нулей
U 1076, 5B00,1D     ;12573          SKIP.IF[MEM.REF.OK]                         ; пропуск следующей инструкции, если суммарная ошибка
                    ;12574                                                      ; отсутствует
U 1077, B907,A4     ;12575          JMP [T1D.CHECK.NXM]                         ; переход к проверке, что суммарная ошибка вызвана битом
                    ;12576                                                      ; NXM
U 1078, EC25,15     ;12577          ADD WR[2] TO LS[MM.LASTADDR+1]               ; увеличение адреса памяти на 4
U 1079, 0907,44     ;12578          JMP [REPEAT.T1D.1]                          ; повторение до получения суммарной ошибки
                    ;12579  T1D.CHECK.NXM:
U 107A, 3624,15     ;12580          MOV LS[MM.LASTADDR+1] TO WR[0]               ; выборка адреса
U 107B, BE88,15     ;12581          MOV WR[0] TO LS[ADDRESS.DATA]                ; подготовка для печати в случае ошибки
U 107C, B660,95     ;12582          MOV LS[NXM] TO WR[1]                         ; ожидаемые данные (установленный бит NXM)
U 107D, 9D45,75     ;12583          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; запрос для чтения CSR1
U 107E, 3022,15     ;12584          MOV MEM.DATA TO WR[0]                        ; прием содержимого CSR1
U 107F, 0869,3C     ;12585          JSR [CHECK.RESULT]                          ; проверка, что бит NXM установлен
U 1080, 0907,44     ;12586          JMP [REPEAT.T1D.1]                          ; цикл при ошибке, если есть разрешение
U 1081, FF82,15     ;12587          INC LS[ERROR.NUMBER]                        ; ошибка 2
U 1082, E58A,15     ;12588          CLR LS[ERROR.MASK]                          ; проверка всех битов
U 1083, 3624,15     ;12589          MOV LS[MM.LASTADDR+1] TO WR[0]               ; выборка значения последнего адреса+1
U 1084, DF28,95     ;12590          MCOM LS[#FFFF] TO WR[1]                      ; установка в рабочем регистре битов 16-31
U 1085, 4560,95     ;12591          BIC LS[BIT16] TO WR[1]                       ; очистка бита 16
U 1086, C562,95     ;12592          BIC LS[BIT17] TO WR[1]                       ; очистка бита 17. Теперь установлены биты 18-31
U 1087, AF42,15     ;12593          BIC WR[1] TO WR[0]                           ; очистка битов 18-31 последнего адреса+1
U 1088, 2F82,95     ;12594          CLR WR[1]                                   ; ожидаемые данные (все нули указывают, что последний
                    ;12595                                                      ; адрес находится на границе 256 кбайт)
U 1089, 0869,3C     ;12596          JSR [CHECK.RESULT]                          ; проверка, что адрес находится на границе 256 кбайт
```

```
U 108A, B907,04 ;12597          JMP [LOOP.T1D.1]                        ; цикл при ошибке, если разрешено
U 108B, B680,15 ;12598          MOV LS[CONTROL.STATUS] TO WR[0]         ; выборка слова управления и состояния
                ;12599            BIT LS[ERROR] WITH WR[0],             ; проверка, произошла ли ошибка во время определения
                ;12600                                                  ; обьема
U 108C, D950,35 ;12601          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 108D, B909,09 ;12602          JMP.IF[BITS.CLR] TO [T1D.PRINT.SIZE]    ; переход к печати обьема памяти, если нет ошибки при
                ;12603                                                  ; определении
U 108E, 6524,15 ;12604          CLR LS[MM.LASTADDR+1]                   ; иначе очистка MM.LASTADDR+1, поэтому следующие тесты
                ;12605                                                  ; памяти будут вынуждены повторять определение обьема
U 108F, B908,A4 ;12606          JMP [END.T1D]                           ; и прекращение остальной части этого теста
                ;12607  T1D.PRINT.SIZE:
U 1090, B625,15 ;12608          MOV LS[MM.LASTADDR+1] TO WR[2]          ; занесение последнего адреса+1 в рабочий регистр
U 1091, 3E13,15 ;12609          MOV WR[2] TO LS[T9]                     ; запоминание последнего адреса+1 в промежуточной ячейке
                ;12610                                                  ; LS для теста на "ПРОВАЛЫ"
U 1092, 3648,15 ;12611          MOV LS[#10] TO WR[0]                    ; WR0 содержит 16 десятичн.
U 1093, C042,15 ;12612          ADD LS[#2] TO WR[0]                     ; WR0 содержит 18 десятичн.
                ;12613  T1D.SHIFT.LOOP:
U 1094, A341,15 ;12614          ROR WR[2]                               ; сдвиг вправо последнего адреса+1
                ;12615            DEC WR[0],                            ; уменьшение счетчика
U 1095, A100,35 ;12616          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 1096, B909,41 ;12617          JMP.IF[NEQ] TO [T1D.SHIFT.LOOP]         ; повторение, пока не произойдет сдвиг вправо на 18
                ;12618                                                  ; мест
U 1097, BE0F,15 ;12619          MOV WR[2] TO LS[MEMORY.SIZE]            ; занесение в LS обьема в блоках по 256 кбайт для
                ;12620                                                  ; печати
U 1098, 3656,15 ;12621          MOV LS[PRINT.MEM.SIZE] TO WR[0]         ; установка в рабочем регистре бита 11
U 1099, 3E80,15 ;12622          MOV WR[0] TO LS[CONTROL.STATUS]         ; информируется консоль о необходимости печати размера
                ;12623                                                  ; памяти
U 109A, 10E0,15 ;12624          MISC [SET.CP.ATTN]                      ; выдача для конс.процессора CPU ATTN
                ;12625  WAIT.T1D.1:
U 109B, B909,B4 ;12626          JMP [WAIT.T1D.1]                        ; цикл для ожидания ответа конс.процессора
U 109C, FF82,15 ;12627          INC LS[ERROR.NUMBER]                    ; ошибка 3 (только для APT)
U 109D, 3690,15 ;12628          MOV LS[ERROR.CONTROL] TO WR[0]          ; выборка слова управления ошибками
                ;12629            BIT LS[APT.PRESENT] WITH WR[0],       ; проверка, управляет ли выполнением теста APT
U 109E, 594A,35 ;12630          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 109F, B90A,59 ;12631          JMP.IF[BITS.CLR] TO [T1D.HOLE.CHECK]    ; если не APT, переход к проверке памяти на "ПРОВАЛЫ"
U 10A0, 3692,95 ;12632          MOV LS[APT.PARAM] TO WR[1]              ; выборка параметров APT
U 10A1, C52C,95 ;12633          BIC LS[#FFFFFF00] TO WR[1]              ; очистка всех байтов, кроме младшего (параметр SIZE)
                ;12634                                                  ; (ожидаемые данные)
U 10A2, 2004,15 ;12635          MOV WR[2] TO WR[0]                      ; обьем в блоках по 256 К, обнаруженный тестом
U 10A3, 0869,3C ;12636          JSR [CHECK.RESULT]                      ; проверка, соответствует ли обьем значению, заданному
                ;12637                                                  ; из APT
U 10A4, DB00,15 ;12638          NOP                                     ; под управлением из APT ошибка не зацикливается
                ;12639  T1D.HOLE.CHECK:
U 10A5, FF82,15 ;12640          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 10A6, 5F60,15 ;12641          MCOM LS[NXM] TO WR[0]                   ; очистка в рабочем регистре бита 16
U 10A7, 3E8A,15 ;12642          MOV WR[0] TO LS[ERROR.MASK]             ; проверка в CSR1 только бита NXM
U 10A8, B670,15 ;12643          MOV LS[OTHER.DATA] TO WR[0]             ; подготовка для записи слова управления
U 10A9, 3E80,15 ;12644          MOV WR[0] TO LS[CONTROL.STATUS]         ; под OTHER (другие данные) консоль будет печатать
                ;12645                                                  ; адрес
U 10AA, 3648,15 ;12646          MOV LS[ARRAY] TO WR[0]                  ; выборка кода модуля для платы матрицы ОЗУ
U 10AB, 3E8C,15 ;12647          MOV WR[0] TO LS[MODULE.NUM]             ; печать в качестве неисправного модуля матрицы ОЗУ в
                ;12648                                                  ; случае ошибки
                ;12649  LOOP.T1D.4:
U 10AC, B660,95 ;12650          MOV LS[NXM] TO WR[1]                    ; ожидаемые данные (установленный бит NXM)
U 10AD, 9912,75 ;12651          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи в несуществующую память
```

```
U 10AE,  B29C,15 ;12652          WRITE.MEM LS[ZERO]                          ; запись нулей
U 10AF,  9D45,75 ;12653          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; запрос для чтения CSR1
U 10B0,  3022,15 ;12654          MOV MEM.DATA TO WR[0]                       ; прием содержимого CSR1
U 10B1,  0869,3C ;12655          JSR [CHECK.RESULT]                          ; проверка, что бит NXM установлен
U 10B2,  090A,C4 ;12656          JMP [LOOP.T1D.4]                            ; цикл при ошибке, если разрешено
U 10B3,  3612,15 ;12657          MOV LS[T9] TO WR[0]                         ; выборка текущего адреса
U 10B4,  4064,15 ;12658          ADD LS[#40000] TO WR[0]                     ; прибавление 256 кбайт для обращения к следующему
                 ;12659                                                      ; блоку
U 10B5,  BE12,15 ;12660          MOV WR[0] TO LS[T9]                         ; запоминание нового адреса в LS,
U 10B6,  BE88,15 ;12661          MOV WR[0] TO LS[ADDRESS.DATA]               ; а также адреса для печати в случае ошибки
U 10B7,  373A,95 ;12662          MOV LS[#F00000] TO WR[1]                    ; проверка, что достигнуто пространство адаптера ОЗУ
                 ;12663            CMP WR[1] WITH WR[0],                     ; проверка, что адрес=F00000(H)
U 10B8,  2642,35 ;12664          DT(LONG)&SET.ALU.CC                         ; и установка кодов условий
                 ;12665
                 ;12666
U 10B9,  090A,C1 ;12667          JMP.IF[NEQ] TO [LOOP.T1D.4]                 ; повторение, пока не будет выполнено
                 ;12668
                 ;12669  END.T1D:
```

```
;12670  .PAGE "ТЕСТ 1E - тест данных (все единицы и все нули) (модуль памяти или модуль МСТ)"
;12671  ;
;12672  ; ОПИСАНИЕ ТЕСТА:
;12673  ;
;12674  ;       Этот тест проверяет всю доступную память на возможность записи и чтения
;12675  ;   всех единиц и возможность записи и чтения всех нулей. Кроме того, тест выпол-
;12676  ;   нчется с тестовыми данными 80000000(H), так как этот набор генерирует в памя-
;12677  ;   ти инвертированные контрольные биты. Неисправимые ошибки (в двух или более
;12678  ;   битах одного длинного слова) вызывают обычное сообщение об ошибке с адресом
;12679  ;   неуспешного обращения под OTHER (другие данные) в сообщении об ошибке. Тест
;12680  ;   записывает все единицы и считывает все единицы, используя функции памяти
;12681  ;   WRITE.P и READ.P при типе данных DT = длинное слово. После проверки ошибок тест
;12682  ;   повторяет эту последовательность, используя как данные все нули и 80000000(H).
;12683  ;   это повторяется в последовательном порядке для каждого адреса. Поскольку
;12684  ;   тест выполняется при разрешенной коррекции ошибок, при сравнении данных будут
;12685  ;   обнаружены только неисправимые ошибки. Такие ошибки будут вызывать сообщение
;12686  ;   об ошибке.
;12687  ;       Заметим, что этот тест не является тестом адреса. Неисправность в адресных
;12688  ;   линиях не вызовет неуспешного выполнения этого теста.
;12689  ;
;12690  ; ПРЕДПОЛОЖЕНИЯ:
;12691  ;
;12692  ;   Принимается, что все предыдущие тесты прошли успешно.
;12693  ;
;12694  ; ШАГИ ТЕСТА:
;12695  ;
;12696  ;     1) Установка в LS маски ошибок, номера ошибки и номера модуля (для распе-
;12697  ;        чатки ошибок) и очистка в LS номера предыдущей ошибки.
;12698  ;     2) Очистка управляющих битов CSR1, загрузка нулями промежуточной ячейки LS
;12699  ;        (будет использоваться в качестве адреса) и загрузка в WR2 из LS
;12700  ;        MM.LASTADDR.
;12701  ;     3) Проверка, что WR2 (последний адрес памяти+1) не нуль. Если определение
;12702  ;        обьема ранее не выполнено, определение обьема памяти и сообщение о ре-
;12703  ;        зультатах.
;12704  ;     4) Выполнение инструкции MEM.REQ с функцией памяти MF=WRITE.P и типом
;12705  ;        данных DT = длинное слово. В качестве адреса используется ранее загружен-
;12706  ;        ная ячейка LS.
;12707  ;     5) Выполнение инструкции WRITE.MEM LS с использованием единиц в качестве
;12708  ;        данных.
;12709  ;     6) Выполнение инструкции MEM.REQ с функцией MF=READ.P и DT=длинное слово.
;12710  ;        В качестве адреса используется промежуточная ячейка LS.
;12711  ;     7) Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка единиц. Сообщение
;12712  ;        об ошибке, если неуспешно. Переход к шагу 10.
;12713  ;     8) Повторение шагов с 4 по 7 с данными из всех нулей. Если в данных ошибок
;12714  ;        не обнаружено, проверка в шаге 7 суммарной ошибки. При наличии суммарной
;12715  ;        ошибки чтение CSR1 и проверка бита RDS. Сообщение об ошибке, если этот
;12716  ;        бит установлен (сбой контрольных битов).
;12717  ;     9) Повторение шага 8 с данными 80000000(H).
;12718  ;    10) Увеличение на 4 ячейки LS, содержащей адрес.
;12719  ;    11) Сравнение нового адреса с последним адресом памяти+1. Если не равно, пе-
;12720  ;        реход к шагу 4. Иначе - завершение теста.
;12721  ;
;12722  ; ОШИБКИ:
;12723  ;
;12724  ;   ПРИМЕЧАНИЕ: В следующих ошибках данными под OTHER является адрес неуспешного
```

```
                ;12725  ;   обращения к памяти.
                ;12726  ;
                ;12727  ;   ошибка 1 - неисправимая ошибка данных при данных, содержащих все единицы.
                ;12728  ;   ошибка 2 - неисправимая ошибка данных при данных из всех нулей.
                ;12729  ;   ошибка 3 - неисправимая ошибка в контрольных битах, обнаруженная по установ-
                ;12730  ;               ленному в CSR 1 биту RDS. Данные в контрольных битах 0111100(B).
                ;12731  ;   ошибка 4 - неисправимая ошибка памяти при данных 80000000(H).
                ;12732  ;   ошибка 5 - неисправимая ошибка в контрольных битах, обнаруженная по установ-
                ;12733  ;               ленному в CSR 1 биту RDS. Данные в контрольных битах 1000011(B).
                ;12734  ;
                ;12735  ; НАЛАДКА:
                ;12736  ;
                ;12737  ;   ПРИМЕЧАНИЕ: Если МСТ работает в пошаговом режиме, регенерация в модуле мат-
                ;12738  ;   риц ОЗУ не состоится. Таким образом, данные в матрице памяти будут потеряны.
                ;12739  ;
                ;12740  ;      ОШИБКА 1 - Эта ошибка указывает на существование 2 или более дефектных би-
                ;12741  ;   тов данных или неисправности логических схем записи/чтения в модуле МСТ. Если
                ;12742  ;   не срабатывает адрес 0, проверьте сигнал MEM SEL A L на выходе ПМЛ ДЕШ.ФИЗИ-
                ;12743  ;   ЧЕСКОГО АДР.А в модуле МСТ. Его можно проверить после того, как центральный
                ;12744  ;   процессор в пошаговом режиме остановлен на инструкции MEM.REQ с MF=WRITE.P.
                ;12745  ;   Сигнал MEM SEL A L должен быть низким. Если нет, то вероятно, что неисправной
                ;12746  ;   является ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР.А. Сказанное применимо также к адресам, ко-
                ;12747  ;   торые пересекают границу модуля в первый раз (т.е. адрес 40000 для матрицы
                ;12748  ;   ОЗУ по 16К или адрес 100000 для матрицы ОЗУ по 64К).
                ;12749  ;      Другими сигналами, которые следует проверить, если неуспешно обращение по
                ;12750  ;   адресу 0, являются CAS TIM L, RAS TIM L, WRT TIM L (во время цикла записи).
                ;12751  ;   Их можно проверить при зацикливании на ошибке. Сами линии данных также сле-
                ;12752  ;   дует проверить, чтобы убедиться, что они выходят из платы. Если похоже, что
                ;12753  ;   все эти сигналы работают правильно, то неисправность, по всей вероятности,
                ;12754  ;   находится в модуле матрицы памяти.
                ;12755  ;      Если ошибки возникают на адресах, отличных от первого адреса модуля, с
                ;12756  ;   наибольшей вероятностью можно подозревать микросхему ОЗУ. Установить место-
                ;12757  ;   нахождение микросхемы можно из адреса ошибки (находится под OTHER (другие
                ;12758  ;   данные) в сообщении об ошибке) и из позиции несработавшего бита по ожида-
                ;12759  ;   емым и полученным данным в сообщении об ошибке.
                ;12760  ;
                ;12761  ;      ОШИБКА 2 - То же, что и при ошибке 1.
                ;12762  ;
                ;12763  ;      ОШИБКА 3 - Эта ошибка указывает на неисправимую ошибку в контрольных битах.
                ;12764  ;   Полученные данные являются правильными, но установлен бит RDS, указывающий,
                ;12765  ;   что была обнаружена двойная ошибка. С наибольшей вероятностью можно подозре-
                ;12766  ;   вать микросхему ЗУ на участке контрольных битов 39-битового слова. Контроль-
                ;12767  ;   ные биты можно проверить при зацикливании на ошибке. Они должны быть 0111100
                ;12768  ;   (B) соответственно от CT до C1.
                ;12769  ;
                ;12770  ;      ОШИБКА 4 - То же, что и для ошибки 1. Эта ошибка не должна происходить по
                ;12771  ;   адресам, отличающимся от тех, в которых возникала ошибка 1 или 2.
                ;12772  ;
                ;12773  ;      ОШИБКА 5 - То же, что и для ошибки 3, только правильными контрольными
                ;12774  ;   битами являются 1000011(B).
                ;12775  ;
                ;12776   T.1E:
U 10BA, B65E,15 ;12777          MOV LSI[BEGIN.TEST] TO WRI[0]           ; установка в WR0 бита 15 для слова управления и
                ;12778                                                  ; состояния
U 10BB, 3E80,15 ;12779          MOV WRI[0] TO LSI[CONTROL.STATUS]       ; установка бита 15 в слове управления и состояния. Бит
```

```
                ;12780
                                                    ; 15 указывает для консольного процессора начало теста
U 10BC, 10E0,15 ;12781          MISC [SET.CP.ATTN]  ; выдача для консольного процессора CPU ATTN
                ;12782  WAIT.T1E.0:
U 10BD, 090B,D4 ;12783          JMP [WAIT.T1E.0]    ; цикл для ожидания ответа консольного процессора
U 10BE, 0A1A,9C ;12784          JSR [SETUP]         ; установка маски, кода модуля и т. д. и очистка CSR1
                ;12785                              ; MCT
U 10BF, 4748,15 ;12786          BIS LS[ARRAY] TO WR[0] ; добавление к коду модуля платы матрицы памяти
U 10C0, 3E8C,15 ;12787          MOV WR[0] TO LS[MODULE.NUM] ; запоминание в LS для индикации платы MCT или
                ;12788                              ; платы матрицы памяти
                ;12789            MOV LS[MM.LASTADDR+1] TO WR[2], ; выборка из LS последнего
U 10C1, 3625,35 ;12790          DT(LONG)&SET.ALU.CC ; адреса+1 и установка кодов условий
U 10C2, 890C,41 ;12791          JMP.IF[NEQ] TO [T1E.CONTINUE] ; продолжение теста, если последний адрес+1 не нуль
U 10C3, 8A19,4C ;12792          JSR [SIZE.MEMORY]   ; иначе определение обьема памяти, печать результата и
                ;12793                              ; продолжение
                ;12794  T1E.CONTINUE:
U 10C4, 3645,95 ;12795          MOV LS[#4] TO WR[3] ; запоминание в WR3 значения инкремента
U 10C5, B670,15 ;12796          MOV LS[OTHER.DATA] TO WR[0] ; подготовка печати под OTHER (другие данные)
U 10C6, 3E80,15 ;12797          MOV WR[0] TO LS[CONTROL.STATUS] ; установка слова управления
U 10C7, 6512,15 ;12798          CLR LS[T9]          ; используется в качестве физического адреса основной
                ;12799                              ; памяти
U 10C8, 6588,15 ;12800          CLR LS[ADDRESS.DATA] ; очистка адреса для печати в случае ошибки
                ;12801  LOOP.T1E.1:
U 10C9, 369E,95 ;12802          MOV LS[ONES] TO WR[1] ; ожидаемые данные
U 10CA, 9912,75 ;12803          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG] ; запрос для записи данных в основную память.
U 10CB, 0A1B,4C ;12804          JSR [NOP.DELAY]     ; используется для предотвращения таймаута, если матрица
                ;12805                              ; разрушена (задержка на 5 циклов центрального
                ;12806                              ; процессора)
U 10CC, 329E,15 ;12807          WRITE.MEM LS[ONES]  ; запись единиц в текущую ячейку памяти
U 10CD, 1813,F5 ;12808          MEM.REQ[READ.P] ADRS[T9] DT[LONG] ; запрос для чтения записанных данных
U 10CE, 3022,15 ;12809          MOV MEM.DATA TO WR[0] ; выборка результата
U 10CF, 0869,3C ;12810          JSR [CHECK.RESULT]  ; проверка данных на все единицы
U 10D0, 090C,94 ;12811          JMP [LOOP.T1E.1]    ; цикл при ошибке, если разрешено
U 10D1, 6C13,95 ;12812          ADD WR[3] TO LS[T9] ; увеличение текущего адреса на 4
U 10D2, 6CB9,95 ;12813          ADD WR[3] TO LS[ADDRESS.DATA] ; увеличение адреса для печати
                ;12814            CMP WR[2] WITH LS[T9], ; проверка, что новый адрес равен последнему
U 10D3, CF13,35 ;12815          DT(LONG)&SET.ALU.CC ; адресу+1 и установка кодов условий
U 10D4, 090C,91 ;12816          JMP.IF[NEQ] TO [LOOP.T1E.1] ; повторение, пока не будет проверена вся доступная
                ;12817                              ; память
U 10D5, E580,15 ;12818          CLR LS[CONTROL.STATUS] ; очистка слова управления и состояния (NOP)
U 10D6, 10E0,15 ;12819          MISC [SET.CP.ATTN]  ; выдача консольному процессору CPU ATTN для
                ;12820                              ; информирования, что центральный процессор продолжает
                ;12821                              ; работать
                ;12822  WAIT.T1E.1:
U 10D7, 090D,74 ;12823          JMP [WAIT.T1E.1]    ; цикл для ожидания ответа консольного процессора
U 10D8, B670,15 ;12824          MOV LS[OTHER.DATA] TO WR[0] ; подготовка для печати под OTHER (другие данные)
U 10D9, 3E80,15 ;12825          MOV WR[0] TO LS[CONTROL.STATUS] ; установка слова управления
U 10DA, FF82,15 ;12826          INC LS[ERROR.NUMBER] ; ошибка 2
U 10DB, 6512,15 ;12827          CLR LS[T9]          ; первым будет адрес памяти 0
U 10DC, 6588,15 ;12828          CLR LS[ADDRESS.DATA] ; совпадает с адресом для печати в случае ошибки
U 10DD, 5B00,1E ;12829          SKIP                ; следующая инструкция нужна только для зацикливания на
                ;12830                              ; ошибке 3
                ;12831  LOOP.T1E.3:
U 10DE, FC82,15 ;12832          DEC LS[ERROR.NUMBER] ; вход цикла при ошибке 4. Номер ошибки уменьшается до
                ;12833                              ; ошибки 2
U 10DF, E58A,15 ;12834          CLR LS[ERROR.MASK]  ; проверка всех битов
```

```
                    ;12835    LOOP.T1E.2:
U 10E0,  2F82,95    ;12836            CLR WR[1]                          ; ожидаемыми данными памяти являются нули
U 10E1,  9912,75    ;12837            MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в память
U 10E2,  B29C,15    ;12838            WRITE.MEM LS[ZERO]                 ; запись нулей по текущему адресу
U 10E3,  1813,F5    ;12839            MEM.REQ[READ.P] ADRS[T9] DT[LONG]  ; запрос для чтения из памяти
U 10E4,  0A1B,4C    ;12840            JSR [NOP.DELAY]                    ; используется для предотвращения таймаута, если матрица
                    ;12841                                               ; памяти разрушена (5 циклов центрального процессора)
U 10E5,  3022,15    ;12842            MOV MEM.DATA TO WR[0]               ; прием данных из памяти
U 10E6,  0869,3C    ;12843            JSR [CHECK.RESULT]                 ; проверка результата на нули
U 10E7,  890E,04    ;12844            JMP [LOOP.T1E.2]                   ; цикл при ошибке, если разрешено
U 10E8,  5800,1D    ;12845            SKIP.IF[MEM.REF.OK]                ; пропуск следующей инструкции, если нет суммарной
                    ;12846                                               ; ошибки
U 10E9,  890E,F4    ;12847            JMP [TEST.T1E.3]                   ; иначе переход к проверке в CSR бита RDS
                    ;12848    TEST.T1E.2.INC:
U 10EA,  6C13,95    ;12849            ADD WR[3] TO LS[T9]                 ; увеличение текущего адреса на 4
U 10EB,  6C89,95    ;12850            ADD WR[3] TO LS[ADDRESS.DATA]       ; а также увеличение адреса для печати
                    ;12851              CMP WR[2] WITH LS[T9],            ; сравнение нового адреса с последним
U 10EC,  CF13,35    ;12852            DT(LONG)&SET.ALU.CC                ; адресом+1 и установка кодов условий
U 10ED,  890E,01    ;12853            JMP.IF[NEQ] TO [LOOP.T1E.2]        ; повторение, пока не будут проверены все адреса
U 10EE,  090F,A4    ;12854            JMP [TEST.T1E.4]                   ; переход к следующему тесту, если все адреса проверены
                    ;12855    TEST.T1E.3:
U 10EF,  FF82,15    ;12856            INC LS[ERROR.NUMBER]               ; ошибка 3
U 10F0,  5F7E,15    ;12857            MCOM LS[RDS] TO WR[0]              ; очистка в рабочем регистре бита RDS (бит 31)
U 10F1,  3E8A,15    ;12858            MOV WR[0] TO LS[ERROR.MASK]        ; проверка в CSR1 только бита RDS
U 10F2,  2F82,95    ;12859            CLR WR[1]                          ; ожидаемые данные (бит RDS очищен)
U 10F3,  9D45,75    ;12860            MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG] ; запрос для чтения CSR1
U 10F4,  3022,15    ;12861            MOV MEM.DATA TO WR[0]               ; прием содержимого CSR1
U 10F5,  0869,3C    ;12862            JSR [CHECK.RESULT]                 ; проверка, установлен ли бит RDS
U 10F6,  090D,E4    ;12863            JMP [LOOP.T1E.3]                   ; цикл при ошибке, если разрешен
U 10F7,  FC82,15    ;12864            DEC LS[ERROR.NUMBER]               ; восстановление номера ошибки до ошибки 2
U 10F8,  E58A,15    ;12865            CLR LS[ERROR.MASK]                 ; восстановление маски для проверки всех битов
U 10F9,  890E,A4    ;12866            JMP [TEST.T1E.2.INC]               ; переход к генерации следующего адреса
                    ;12867    TEST.T1E.4:
U 10FA,  E580,15    ;12868            CLR LS[CONTROL.STATUS]             ; очистка слова управления и состояния (NOP)
U 10FB,  10E0,15    ;12869            MISC [SET.CP.ATTN]                 ; выдача консольному процессору CPU ATTN для
                    ;12870                                               ; информирования, что центральный процессор продолжает
                    ;12871                                               ; работать
                    ;12872    WAIT.T1E.2:
U 10FC,  090F,C4    ;12873            JMP [WAIT.T1E.2]                   ; цикл для ожидания ответа консольного процессора
U 10FD,  8670,15    ;12874            MOV LS[OTHER.DATA] TO WR[0]         ; подготовка для печати под OTHER (другие данные)
U 10FE,  3E80,15    ;12875            MOV WR[0] TO LS[CONTROL.STATUS]     ; установка слова управления
U 10FF,  FF82,15    ;12876            INC LS[ERROR.NUMBER]               ; теперь номер ошибки 3
U 1100,  FF82,15    ;12877            INC LS[ERROR.NUMBER]               ; номер текущей ошибки 4
U 1101,  6512,15    ;12878            CLR LS[T9]                         ; первым будет адрес памяти 0
U 1102,  6588,15    ;12879            CLR LS[ADDRESS.DATA]               ; тот же адрес для печати в случае ошибки
U 1103,  5800,1E    ;12880            SKIP                               ; следующая инструкция нужна только для зацикливания на
                    ;12881                                               ; ошибке 5
                    ;12882    LOOP.T1E.5:
U 1104,  FC82,15    ;12883            DEC LS[ERROR.NUMBER]               ; вход цикла при ошибке. Уменьшение номера ошибки до
                    ;12884                                               ; ошибки 4
U 1105,  E58A,15    ;12885            CLR LS[ERROR.MASK]                 ; проверка всех битов
                    ;12886    LOOP.T1E.4:
U 1106,  867E,95    ;12887            MOV LS[#80000000] TO WR[1]          ; ожидаемые данные
U 1107,  9912,75    ;12888            MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в память
U 1108,  B27E,15    ;12889            WRITE.MEM LS[#80000000]             ; запись по текущему адресу данных 80000000(H)
```

```
U 1109, 1B13,F5 ;12890          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 110A, 3022,15 ;12891          MOV MEM.DATA TO WR[0]                   ; прием данных из памяти
U 110B, 0869,3C ;12892          JSR [CHECK.RESULT]                      ; проверка, что результат 80000000(H)
U 110C, 8910,64 ;12893          JMP [LOOP.T1E.4]                        ; цикл при ошибке, если разрешен
U 110D, 5B00,1D ;12894          SKIP.IF[MEM.REF.OK]                     ; пропуск следующей инструкции, если нет суммарной
                ;12895                                                  ; ошибки
U 110E, 8911,44 ;12896          JMP [TEST.T1E.5]                        ; иначе переход к проверке в CSR бита RDS
                ;12897  TEST.T1E.4.INC:
U 110F, 6C13,95 ;12898          ADD WR[3] TO LS[T9]                     ; увеличение текущего адреса на 4
U 1110, 6C89,95 ;12899          ADD WR[3] TO LS[ADDRESS.DATA]           ; а также, увеличение адреса для печати
                ;12900            CMP WR[2] WITH LS[T9],                ; сравнение нового адреса с последним
U 1111, CF13,35 ;12901          DT(LONG)&SET.ALU.CC                     ; адресом+1 и установка кодов условий
U 1112, 8910,61 ;12902          JMP.IF[NEQ] TO [LOOP.T1E.4]             ; повторение, пока не будут проверены все адреса
U 1113, 0911,F4 ;12903          JMP [END.T1E]                           ; выполнено. Переход к концу теста
                ;12904  TEST.T1E.5:
U 1114, FF82,15 ;12905          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 1115, 5F7E,15 ;12906          MCOM LS[RDS] TO WR[0]                   ; очистка в рабочем регистре бита RDS (бит 31)
U 1116, 3E8A,15 ;12907          MOV WR[0] TO LS[ERROR.MASK]             ; проверка в CSR1 только бита RDS
U 1117, 2F82,95 ;12908          CLR WR[1]                               ; ожидаемые данные (бит RDS очищен)
U 1118, 9D45,75 ;12909          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1119, 3022,15 ;12910          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 111A, 0869,3C ;12911          JSR [CHECK.RESULT]                      ; проверка, установлен ли бит RDS
U 111B, 0910,44 ;12912          JMP [LOOP.T1E.5]                        ; цикл при ошибке, если разрешен
U 111C, FC82,15 ;12913          DEC LS[ERROR.NUMBER]                    ; восстановление номера ошибки до ошибки 3
U 111D, E58A,15 ;12914          CLR LS[ERROR.MASK]                      ; восстановление маски для проверки всех битов
U 111E, 8910,F4 ;12915          JMP [TEST.T1E.4.INC]                    ; переход к генерации следующего адреса
                ;12916  END.T1E:
```

```
                ;12917  .PAGE "ТЕСТ 1F - тест тракта данных (модуль матрицы памяти)"
                ;12918  ;
                ;12919  ; ОПИСАНИЕ ТЕСТА:
                ;12920  ;
                ;12921  ;       Этот тест проверяет тракт данных на платах матриц ОЗУ на короткие замыка-
                ;12922  ;   ния между битами. Этот тест проверяет один адрес в каждом блоке из 256 кбайт,
                ;12923  ;   чтобы иметь гарантию, что проверены все платы матриц памяти. Этот тест запи-
                ;12924  ;   сывает и считывает тестовые данные с "БЕГУЩЕЙ 1" в поле нулей через все 32
                ;12925  ;   бита.
                ;12926  ;
                ;12927  ; ПРЕДПОЛОЖЕНИЯ:
                ;12928  ;
                ;12929  ;   Принимается, что все предыдущие тесты прошли успешно.
                ;12930  ;
                ;12931  ; ШАГИ ТЕСТА:
                ;12932  ;
                ;12933  ;   1) Установка в LS маски ошибки, номера ошибки и номера модуля (для распечатки
                ;12934  ;      ошибок) и очистка в LS номера предыдущей ошибки.
                ;12935  ;   2) Очистка управляющих битов CSR1, загрузка нулями временной ячейки LS (бу-
                ;12936  ;      дет использоваться в качестве адреса) и загрузка в WR2 из LS MM.LASTADDR.
                ;12937  ;   3) Проверка, что WR2 не нулевой (последний адрес памяти+1). Если размер па-
                ;12938  ;      мяти ранее не определен, определение размера памяти и печать результата.
                ;12939  ;   4) Очистка регистра OS (используется для индексации) и загрузка в WR1 первого
                ;12940  ;      тестового набора данных.
                ;12941  ;   5) Выполнение инструкции MEM.REQ с функцией памяти MF=WRITE.P и типом данных
                ;12942  ;      DT=длинное слово. В качестве адреса используется промежуточная ячейка LS.
                ;12943  ;   6) Выполнение инструкции WRITE.MEM LS с использованием индексного режима.
                ;12944  ;   7) Выполнение инструкции MEM.REQ с функцией памяти MF=READ.P и типом данных
                ;12945  ;      DT=длинное слово. В качестве адреса используется промежуточная ячейка LS.
                ;12946  ;   8) Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата.
                ;12947  ;   9) Увеличение OS и проверка шагов с 3 по 7, пока не будут проверены все 32
                ;12948  ;      бита.
                ;12949  ;  10) Увеличение на 40000(H)(256 Кбайт) промежуточной ячейки LS, используемой
                ;12950  ;      в качестве адреса памяти, и сравнение с LS MM.LASTADDR+1. Если они рав-
                ;12951  ;      ны, тест выполнен. иначе повторение шагов с 3 по 7 с новым адресом.
                ;12952  ;
                ;12953  ; ОШИБКИ:
                ;12954  ;
                ;12955  ;   ошибка 1 - тест платы матрицы ОЗУ со сдвигаемыми данными выполнен неправиль-
                ;12956  ;   но. Адрес находится под OTHER (другие данные).
                ;12957  ;
                ;12958  ; НАЛАДКА:
                ;12959  ;
                ;12960  ;   ПРИМЕЧАНИЕ: если МСТ работает в пошаговом режиме, не происходит регенерация
                ;12961  ;   матрицы памяти. Таким образом, данные в матрице памяти будут потеряны.
                ;12962  ;
                ;12963  ;       ОШИБКА 1 - Проверьте на замыкание между неправильно работающими битами на
                ;12964  ;   плате матрицы памяти, используя адрес под OTHER (другие данные) для определе-
                ;12965  ;   ния неисправного модуля.
                ;12966  T.1F:
U 111F, 865E,15 ;12967          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;12968                                                  ; состояния
U 1120, 3EB0,15 ;12969          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;12970                                                  ; 15 указывает для конс.процессора начало теста
U 1121, 10E0,15 ;12971          MISC [SET.CP.ATTN]                      ; выдача для конс.процессора CPU ATTN
```

```
                ;12972   WAIT.T1F.0:
U 1122, 8912,24 ;12973           JMP [WAIT.T1F.0]                ; цикл для ожидания ответа конс. процессора
U 1123, 0A1A,9C ;12974           JSR [SETUP]                     ; установка маски, кода модуля и т.д. и очистка CSR1
                ;12975                                           ; МСТ
U 1124, 3648,15 ;12976           MOV LS[ARRAY] TO WR[0]          ; установка бита для индикации модуля памяти
U 1125, 3EBC,15 ;12977           MOV WR[0] TO LS[MODULE.NUM]     ; запоминание в LS кода модуля для индикации платы
                ;12978                                           ; матрицы памяти
                ;12979             MOV LS[MM.LASTADDR+1] TO WR[2], ; выборка из LS последнего адреса+1
U 1126, 3625,35 ;12980           DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 1127, 0912,91 ;12981           JMP.IF[NEQ] TO [T1F.CONTINUE]   ; продолжение теста, если последний адрес+1 не нуль
U 1128, 8A19,4C ;12982           JSR [SIZE.MEMORY]               ; определение размера памяти, если не выполнено ранее, и
                ;12983                                           ; печать результата
                ;12984   T1F.CONTINUE:
U 1129, B670,15 ;12985           MOV LS[OTHER.DATA] TO WR[0]     ; установка бита для печати данных под OTHER (другие
                ;12986                                           ; данные)
U 112A, 3EB0,15 ;12987           MOV WR[0] TO LS[CONTROL.STATUS] ; установка слова управления и состояния для печати
                ;12988                                           ; адреса под OTHER в случае ошибки
U 112B, 6512,15 ;12989           CLR LS[T9]                      ; адрес для обращения к основной памяти
U 112C, 6588,15 ;12990           CLR LS[ADDRESS.DATA]            ; а также очистка адреса для печати
                ;12991   REPEAT.T1F.1:
U 112D, E5F8,15 ;12992           CLR LS[OS]                      ; очистка индекса
                ;12993   LOOP.T1F.1:
U 112E, B6F6,95 ;12994           MOV LS[SHIFT.OS(4-0)] TO WR[1]  ; ожидаемые данные (сдвигаемая единица)
U 112F, 9912,75 ;12995           MEM.REQ[WRITE.P] ADRS[T9] DT[LONG] ; запрос для записи в основную память
U 1130, B2F6,15 ;12996           WRITE.MEM LS[SHIFT.OS(4-0)]     ; запись в память сдвигаемых тестовых данных
U 1131, 1813,F5 ;12997           MEM.REQ[READ.P] ADRS[T9] DT[LONG] ; запрос для чтения результата
U 1132, 3022,15 ;12998           MOV MEM.DATA TO WR[0]           ; прием результата
U 1133, 0869,3C ;12999           JSR [CHECK.RESULT]              ; поверка наличия ошибки
U 1134, 8912,E4 ;13000           JMP [LOOP.T1F.1]                ; цикл при ошибке, если разрешено
U 1135, 7FF8,15 ;13001           INC LS[OS]                      ; увеличение индекса для следующего набора
U 1136, B6F8,15 ;13002           MOV LS[OS] TO WR[0]             ; подготовка проверки, все ли наборы использованы
                ;13003             BIT LS[BIT5] WITH WR[0],      ; проверка, увеличен ли регистр OS до 20(H)
U 1137, 594A,35 ;13004           DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 1138, 0912,E9 ;13005           JMP.IF[BITS.CLR] TO [LOOP.T1F.1] ; если не конец, повторение с новым набором
U 1139, E5F8,15 ;13006           CLR LS[OS]                      ; очистка индекса
U 113A, B664,15 ;13007           MOV LS[#40000] TO WR[0]         ; выборка инкремента на блок из 256 кбайт
U 113B, 6C12,15 ;13008           ADD WR[0] TO LS[T9]             ; сложение с текущим адресом
U 113C, 6C88,15 ;13009           ADD WR[0] TO LS[ADDRESS.DATA]   ; то же с адресом для печати
                ;13010             CMP WR[2] WITH LS[T9],        ; проверка, имеются ли еще блоки (сравнение с
                ;13011                                           ; последующим адресом+1)
U 113D, CF13,35 ;13012           DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 113E, 8912,D1 ;13013           JMP.IF[NEQ] TO [REPEAT.T1F.1]   ; повторение если нет, иначе конец теста
                ;13014   END.T1F:
```

```
;13015  .PAGE "ТЕСТ 20 - проверка регенерации (модуль памяти, модуль WCS)"
;13016  ;
;13017  ; ОПИСАНИЕ ТЕСТА:
;13018  ;
;13019  ;       Этот тест аналогичен тесту для всех 1 и всех 0, выполняемому перед этим
;13020  ;   тестом. Разница состоит в том, что вся доступная память записывается фоном
;13021  ;   из всех единиц или всех нулей, а считывается после задержки, превышающей
;13022  ;   минимальный уровень регенерации, который необходим для удержания данных в
;13023  ;   микросхеме ОЗУ. Если не срабатывают логические схемы регенерации, данные
;13024  ;   разрушаются, и будет получено сообщение об ошибке. Этот тест выполняется
;13025  ;   дважды, один раз со всеми единицами в качестве фоновых данных, и один раз со
;13026  ;   всеми нулями. Это создает уверенность в том, что разрушение любого состояния
;13027  ;   будет (в большинстве случаев) обнаружено.
;13028  ;
;13029  ; ПРЕДПОЛОЖЕНИЯ:
;13030  ;
;13031  ;   Принимается, что все предыдущие тесты прошли успешно. Этот тест обнаружит
;13032  ;   только полное отсутствие регенерации и обладает приемлемой точностью только
;13033  ;   при температурах 50 градусов по цельсию и выше.
;13034  ;
;13035  ; ШАГИ ТЕСТА:
;13036  ;
;13037  ;   1) Установка в LS маски ошибок, номера ошибки и номера модуля (для распечат-
;13038  ;      ки ошибок) и очистка в LS номера предыдущей ошибки.
;13039  ;   2) Очистка в CSR1 управляющих битов, загрузка нулей в промежуточную ячейку
;13040  ;      LS (будет использоваться в качестве адреса) и загрузка в WR2 из LS
;13041  ;      MM.LASTADDR.
;13042  ;   3) Проверка, что WR2 (последний адрес памяти+1) не содержит 0. Проверка объ-
;13043  ;      ема памяти и сообщение о результате, если обьем памяти не был ранее опре-
;13044  ;      делен.
;13045  ;   4) Выполнение инструкции MEM.REQ с MF=WRITE.P и DT=LONGWORD. В качестве ад-
;13046  ;      реса основной памяти используется промежуточная ячейка LS.
;13047  ;   5) Выполнение инструкции записи WRITE.MEM LS с данными из всех 1.
;13048  ;   6) Увеличение на 4 содержимого ячейки LS и повторение  шагов с 4 по 6 до тех
;13049  ;      пор, пока не будет превзойден последний адрес памяти.
;13050  ;   7) Задержка около 5 сек при помощи зацикливания микрокодов.
;13051  ;   8) Считывание каждой ячейки и проверка результата.
;13052  ;   9) Повторение шагов с 4 по 8 с данными из всех нулей.
;13053  ;
;13054  ; ОШИБКИ:
;13055  ;
;13056  ;   ошибка 1 - неисправность логических схем регенерации (неправильные результа-
;13057  ;   ты только в паре адресов могут указывать на дефектную микросхему ОЗУ). Ад-
;13058  ;   рес находится под OTHER (другие данные).
;13059  ;
;13060  ; НАЛАДКА:
;13061  ;
;13062  ;   ПРИМЕЧАНИЕ: если МСТ работает в режиме микрошага, будет происходить потеря
;13063  ;   регенерации в модуле памяти. Таким образом, данные в матрице будут потеряны.
;13064  ;
;13065  ;       ОШИБКА 1 - Эта ошибка указывает на неправильную работу логических схем ре-
;13066  ;   генерации в модуле WCS или модуле памяти. В небольшом числе случаев она может
;13067  ;   указывать на неисправность в самой микросхеме ОЗУ из-за утечки в ячейке.
;13068  ;   Первыми сигналами, которые следует проверить, являются ARRAY REF CYC L и
;13069  ;   ARRAY RAS TIM L на выходе из модуля WCS. Их можно проверить, когда машина
```

```
                ;13070  ;   выполняет холостые циклы, так как регенерация выполняется всегда. Если сиг-
                ;13071  ;   нал ARRAY REF CYC L образует импульсы, следует проверить адресные линии
                ;13072  ;   (с BUS ARRAY A0 H по BUS ARRAY A7 H) с целью убедиться, что они наращивают-
                ;13073  ;   ся правильно. Каждая из этих линий должна переключаться с частотой вдвое
                ;13074  ;   меньшей, чем предыдущая, если проверка начинается с линии адреса 0 и продол-
                ;13075  ;   жается до линии адреса 7. Если все эти сигналы вырабатываются, неисправность
                ;13076  ;   содержится в модуле памяти.
                ;13077  ;      Если ARRAY REF CYC L или ARRY RAS TIM L не пульсируют, следует проверить
                ;13078  ;   входы ПМЛ УПР.РЕГЕНЕРАЦИЕЙ на высокие уровни на обоих входах ALLOW REF H и
                ;13079  ;   INH REF CYC L. Сигнал ALLOW REF H является битом управляющей памяти контрол-
                ;13080  ;   лера ОЗУ. Если они правильные, то необходимо проверить, что сигналы MAIN
                ;13081  ;   MEM REFR L и 90 NS CP H являются пульсирующими. Если они правильные, то ве-
                ;13082  ;   роятно, что неисправна микросхема ПМЛ.
                ;13083  ;      Если неправильно работают линии BUS ARRAY A0 H до BUS ARRAY A7 H, следует
                ;13084  ;   проверить входы в четырехвходовые шинные формирователи и сигнал BIN CTR.
                ;13085  ;
                ;13086   T.20:
U 113F, B65E,15 ;13087          MOV LS[BEGIN.TEST] TO WR[0]          ; установка в WR0 бита 15 для слова управления и
                ;13088                                               ; состояния
U 1140, 3E80,15 ;13089          MOV WR[0] TO LS[CONTROL.STATUS]       ; установка бита 15 в слове управления и состояния. Бит
                ;13090                                               ; 15 указывает для конс.процессора начало теста
U 1141, 10E0,15 ;13091          MISC [SET.CP.ATTN]                   ; выдача для конс.процессора CPU ATTN
                ;13092   WAIT.T20.0:
U 1142, B914,24 ;13093          JMP [WAIT.T20.0]                     ; цикл для ожидания ответа конс.процессора
U 1143, 0A1A,9C ;13094          JSR [SETUP]                          ; установка масок, кода модуля и т. д. и очистка CSR1
                ;13095                                               ; МСТ
U 1144, B640,15 ;13096          MOV LS[WCS] TO WR[0]                 ; установка кода модуля. В рабочем регистре установлен
                ;13097                                               ; бит 1
U 1145, 4748,15 ;13098          BIS LS[ARRAY] TO WR[0]               ; установка бита для платы модуля памяти
U 1146, 3E8C,15 ;13099          MOV WR[0] TO LS[MODULE.NUM]          ; запоминание в LS кода модуля для указания платы WCS
                ;13100                                               ; или модуля памяти
                ;13101            MOV LS[MM.LASTADDR+1] TO WR[2],    ; выборка из LS последнего адреса+1
U 1147, 3625,35 ;13102          DT(LONG)&SET.ALU.CC                  ; и установка кодов условий
U 1148, 0914,A1 ;13103          JMP.IF[NEQ] TO [T20.CONTINUE]        ; продолжение теста, если последний адрес+1 не нулевой
U 1149, 8A19,4C ;13104          JSR [SIZE.MEMORY]                    ; определение обьема памяти, если не выполнено раньше, и
                ;13105                                               ; сообщение о результате
                ;13106   T20.CONTINUE:
U 114A, 3645,95 ;13107          MOV LS[#4] TO WR[3]                  ; значение инкремента для адреса
U 114B, 6512,15 ;13108          CLR LS[T9]                           ; адрес для обращения к основной памати
                ;13109   BACK.T20.1:
U 114C, 9912,75 ;13110          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]   ; запрос для записи фона из единиц
U 114D, 329E,15 ;13111          WRITE.MEM LS[ONES]                   ; запись единиц
U 114E, 6C13,95 ;13112          ADD WR[3] TO LS[T9]                  ; увеличение адреса
                ;13113            CMP WR[2] WITH LS[T9],             ; проверка, что вся память заполнена
U 114F, CF13,35 ;13114          DT(LONG)&SET.ALU.CC                  ; и установка кодов условий
U 1150, 0914,C1 ;13115          JMP.IF[NEQ] TO [BACK.T20.1]          ; повторение, пока память не будет записана полностью
U 1151, 6512,15 ;13116          CLR LS[T9]                           ; адрес для обращения к основной памяти
U 1152, 9912,75 ;13117          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]   ; запрос для повторной записи единиц в ячейку 0
U 1153, 329E,15 ;13118          WRITE.MEM LS[ONES]                   ; для установки адресных линий на низкий уровень
U 1154, 0917,DC ;13119          JSR [DELAY.T20]                      ; выполнение задержки на 5 секунд
U 1155, B670,15 ;13120          MOV LS[OTHER.DATA] TO WR[0]          ; установка бита для распечатки поля OTHER
U 1156, 3E80,15 ;13121          MOV WR[0] TO LS[CONTROL.STATUS]       ; подготовка слова управления для печати под OTHER
                ;13122                                               ; (другие данные) адреса в случае ошибки
U 1157, 6512,15 ;13123          CLR LS[T9]                           ; запись с адреса 0
U 1158, 6588,15 ;13124          CLR LS[ADDRESS.DATA]                 ; а также очистка адреса, подлежащего печати
```

```
                      LOOP.T20.1:
U 1159, 369E,95               MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 115A, 1813,F5               MEM.REQ[READ.P] ADRS[T9] DT[LONG]        ; запрос для считывания данных
U 115B, 3022,15               MOV MEM.DATA TO WR[0]                    ; прием результата
U 115C, 0869,3C               JSR [CHECK.RESULT]                       ; проверка на все единицы
U 115D, 8915,94               JMP [LOOP.T20.1]                         ; цикл при ошибке, если есть разрешение
U 115E, 6C13,95               ADD WR[3] TO LS[T9]                      ; увеличение адреса
U 115F, 6C89,95               ADD WR[3] TO LS[ADDRESS.DATA]            ; а также увеличение адреса для печати
                                CMP WR[2] WITH LS[T9],                 ; проверка, что тест выполнен для всей памяти
U 1160, CF13,35               DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1161, 8915,91               JMP.IF[NEQ] TO [LOOP.T20.1]              ; повторение до тех пор, пока чтение не пройдет для всей
                                                                       ; памяти
U 1162, E580,15               CLR LS[CONTROL.STATUS]                   ; очистка слова управления и состояния для случая, если
                                                                       ; была предыдущая ошибка
U 1163, 10E0,15               MISC [SET.CP.ATTN]                       ; выдача для консоли CPU ATTN (для предотвращения
                                                                       ; таймаута в больших системах памяти)
                      WAIT.T20.1:
U 1164, 0916,44               JMP [WAIT.T20.1]                         ; цикл для ожидания ответа конс.процессора
U 1165, 6512,15               CLR LS[T9]                               ; адрес для обращения к основной памяти
                      BACK.T20.1A:
U 1166, 9912,75               MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]       ; запрос для записи фона нулей
U 1167, B29C,15               WRITE.MEM LS[ZERO]                       ; запись нулей
U 1168, 6C13,95               ADD WR[3] TO LS[T9]                      ; увеличение адреса
                                CMP WR[2] WITH LS[T9],                 ; проверка, что память записана полностью
U 1169, CF13,35               DT(LONG)&SET.ALU.CC                        и установка кодов условий
U 116A, 8916,61               JMP.IF[NEQ] TO [BACK.T20.1A]               повторение, пока память не будет полностью записана
U 116B, 6512,15               CLR LS[T9]                                 адрес для обращения к основной памяти
U 116C, 9912,75               MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]         запрос для повторной записи 0 в ячейку 0
U 116D, B29C,15               WRITE.MEM LS[ZERO]                       ; для установки адресных линий на низкий уровень
U 116E, 0917,DC               JSR [DELAY.T20]                          ; задержка на 5 секунд
U 116F, B670,15               MOV LS[OTHER.DATA] TO WR[0]              ; установка бита на [illegible] поля OTHER (другие данные)
U 1170, 3E80,15               MOV WR[0] TO LS[CONTROL.STATUS]          ; установка слова управления на печать адреса под OTHER
                                                                       ; в случае ошибки
U 1171, 6512,15               CLR LS[T9]                               ; запуск с адреса 0
U 1172, 6588,15               CLR LS[ADDRESS.DATA]                       а также очистка адреса, подлежащего печати
                      LOOP.T20.1A:
U 1173, 2F82,95               CLR WR[1]                                  ожидаемые данные
U 1174, 1813,F5               MEM.REQ[READ.P] ADRS[T9] DT[LONG]          запрос для считыв[illegible] записанных данных
U 1175, 3022,15               MOV MEM.DATA TO WR[0]                      прием результата
U 1176, 0869,3C               JSR [CHECK.RESULT]                         [illegible]
U 1177, 0917,34               JMP [LOOP.T20.1A]                          цикл при ошибке [illegible] есть разрешение
U 1178, 6C13,95               ADD WR[3] TO LS[T9]                        увел[illegible]
U 1179, 6C89,95               ADD WR[3] TO LS[ADDRESS.DATA]              а [illegible] подлежащего печати
                                CMP WR[2] WITH LS[T9],                   проверка [illegible] считывание из всей памяти
U 117A, CF13,35               DT(LONG)&SET.ALU.CC                        и установка кодов [illegible]
U 117B, 0917,31               JMP.IF[NEQ] TO [LOOP.T20.1A]             ; повторение, [illegible] считана вся память
U 117C, 0918,64               JMP [END.T20]                            ; [illegible]
                        ;
                        ;      ПОДПРОГРАММА ЗАДЕРЖКИ
                        ;
                      DELAY.T20:
U 117D, B66E,15               MOV LS[BIT23] TO WR[0]                   ; счетчик [illegible]
U 117E, E580,15               CLR LS[CONTROL.STATUS]                   ; очистка [illegible] состояния
U 117F, 10E0,15               MISC [SET.CP.ATTN]                         выда[illegible] CPU ATTN (для предотвращения
                                                                         таймаута [illegible] памяти)
```

```
                ;13180   DELAY.T20.0:
                ;13181            DEC WR[0],                          ; уменьшение счетчиика
U 1180, A100,35 ;13182           DT(LONG)&SET.ALU.CC                  ; и установка кодов условий
U 1181, 0918,01 ;13183           JMP.IF[NEQ] TO [DELAY.T20.0]         ; повторение, пока не будет достигнута задержка на 5
                ;13184                                                ; секунд
U 1182, E580,15 ;13185           CLR LS[CONTROL.STATUS]               ; очистка слова управления и состояния
U 1183, 10E0,15 ;13186           MISC [SET.CP.ATTN]                   ; выдача для конс.процессора CPU ATTN (для
                ;13187                                                ; предотвращения таймаута в больших системах памяти)
                ;13188   DELAY.T20.1:
U 1184, 8918,44 ;13189           JMP [DELAY.T20.1]                    ; цикл для ожидания ответа конс.процессора
U 1185, 5800,14 ;13190           RETURN                               ; конец выполнения
                ;13191   END.T20:
```

```
;13192  .PAGE "ТЕСТ 21 - тест *МАРШ* для основной памяти (модуль МСТ или модуль памяти)"
;13193  ;
;13194  ; ОПИСАНИЕ ТЕСТА:
;13195  ;
;13196  ;       Этот тест выполняет проверку всей доступной памяти по алгоритму "МАРША".
;13197  ;   Вначале он проверяет, что обьем памяти уже определен, путем проверки ячейки
;13198  ;   LS MM.LASTADDR+1. Если она содержит 0, определяется обьем памяти и выводится
;13199  ;   сообщение о результатах. После этого в память записывается фон из нулей. На-
;13200  ;   чиная с адреса 0, каждая ячейка памяти размером в длинное слово, считывается
;13201  ;   с проверкой на нули, записывается единицами и считыватся с проверкой единиц.
;13202  ;   Если данные правильные, проверяется бит суммарной ошибки памяти. Если бит
;13203  ;   суммарной ошибки системы памяти установлен, считывается CSR1 и проверяется
;13204  ;   бит CRD (ошибка в одиночном бите). Если он является единственным битом ошиб-
;13205  ;   ки, установленным в CSR1, проверяются полученные биты синдрома, чтобы убе-
;13206  ;   диться, что неудача вызвана не контрольными битами. Увеличивается счетчик
;13207  ;   индикации числа ошибок, обнаруженных в одиночных битах. Если в CSR1 установ-
;13208  ;   лены какие-либо биты ошибок, сообщается об ошибке (CRD является единственной
;13209  ;   ошибкой, которая может произойти, если считанные данные правильные). Для
;13210  ;   каждых 256 кбайт памяти предусмотрен отдельный счетчик ошибок в одиночных
;13211  ;   битах. При матрице из ОЗУ по 16 К это составляет 1 модуль памяти.
;13212  ;       Адрес увеличивается и последовательность чтение/запись/чтение повторяется
;13213  ;   снова до тех пор, пока не будет достигнута верхняя граница памяти. Тогда тест
;13214  ;   повторяется, начиная с верхней границы памяти с уменьшением адреса до тех
;13215  ;   пор, пока не будет проверена ячейка 0. Убывающий "МАРШ" выполняет чтение
;13216  ;   единиц, запись нулей и чтение нулей. Сообщается только об ошибках RDS (т.е.,
;13217  ;   неисправимых ошибках). Ошибки в одиночных битах во время убывающего прохода
;13218  ;   не подсчитываются.
;13219  ;       После завершения теста "МАРШ" со всеми единицами и всеми нулями выполняет-
;13220  ;   ся другой тест "МАРШ" с тестовыми данными из нулей и 80000000(H). Набор с
;13221  ;   установленным битом 31 дает инверсию контрольных битов и, таким образом,
;13222  ;   обеспечиват выполнение теста "МАРШ" для них. Используется та же модель чте-
;13223  ;   ние/запись/чтение, как и раньше, за исключением того, что подсчитываются то-
;13224  ;   лько одиночные ошибки в контрольных битах (отказы в одиночных битах данных
;13225  ;   были подсчитаны в первом тесте "МАРШ"). Счетчики для одиночных ошибок в дан-
;13226  ;   ных и одиночных ошибок в контрольных битах складываются и образуют общий
;13227  ;   счет ошибок CRD на блок обьемом 256 кбайт.
;13228  ;
;13229  ; ПРЕДПОЛОЖЕНИЯ:
;13230  ;
;13231  ;   Принимается, что все предыдущие тесты прошли успешно. Ошибки, которые возни-
;13232  ;   кают в этом тесте, с наибольшей вероятностью относятся к адресу (за исключе-
;13233  ;   нием ошибок в одиночных битах, которые не вызывают нормального сообщения об
;13234  ;   ошибке).
;13235  ;
;13236  ; ШАГИ ТЕСТА:
;13237  ;
;13238  ;   1) Установка в LS маски ошибок, номера ошибки и номера модуля (для распечат-
;13239  ;      ки ошибок) и очистка в LS номера предыдущей ошибки.
;13240  ;   2) Загрузка в WR2 обьема памяти (ячейка LS MM.LASTADDR+1) и проверка, что
;13241  ;      обьем не 0. Если значение обьема памяти 0, повторное определение обьема
;13242  ;      памяти, сообщение о результате и продолжение.
;13243  ;   3) Очистка CSR1 и очистка промежуточной ячейки LS (для адреса).
;13244  ;   4) Запись фона нулей во всей основной памяти. Очистка промежуточной ячейки
;13245  ;      LS после завершения записи фона.
;13246  ;   5) Загрузка нулей в WR1 (ожидаемые данные) и выполнение инструкции MEM.REQ
```

```
;13247  ;       с MF=READ.P и DT=длинное слово. Использование в качестве адреса ячейки LS.
;13248  ;    6) Выполнение инструкции MOV MEM.DATA TO WR[0] и проверка результата. В слу-
;13249  ;       чае ошибки, сообщение об ошибке и переход к шагу 9.
;13250  ;    7) Проверка суммарной ошибки системы памяти (ERROR SUM). Если ERROR SUM от-
;13251  ;       сутствует, переход к шагу 9, иначе чтение CSR1 и проверка, что установлен
;13252  ;       только бит CRD.
;13253  ;    8) Если установлен бит CRD, чтение CSR0 (биты синдрома) и проверка, что оши-
;13254  ;       бка содержится в битах данных, но не в контрольных битах. Если ошибка со-
;13255  ;       держится в битах данных, увеличение в LS счетчика одиночных ошибок. Если
;13256  ;       работа под управлением APT, или установлено разрешение сообщений об оди-
;13257  ;       ночных ошибках, чтение ячейки при запрещенной коррекции (ECC) и сообщение
;13258  ;       об ошибке.
;13259  ;    9) Загрузка единиц в WR1 и выполнение инструкции MEM.REQ с MF=WRITE.P и DT=
;13260  ;       длинное слово. Выполнение инструкции WRITE.MEM LS с единицами в LS.
;13261  ;   10) Повторение шагов с 6 по 8 с проверкой данных на все единицы.
;13262  ;   11) Увеличение на 4 ячейки LS, содержашей адрес. Сравнение с ячейкой LS MM.
;13263  ;       LAST ADDR+4. Если не равны, переход к шагу 5, иначе переход к следующему
;13264  ;       шагу.
;13265  ;   12) Повторение шагов с 5 по 10, но с обходом шагов 7 и 8. Использование ин-
;13266  ;       вертированных тестовых данных и убывающих адресов.
;13267  ;   13) Повторение шагов с 5 по 11. В шаге 8 производится проверка, что ошибка
;13268  ;       содержится в контрольких битах, и в шаге 9 используются тестовые данные
;13269  ;       80000000(H). Если работа под управлением APT, или разрешены сообщения об
;13270  ;       одиночных ошибках, распечатка инвертированных битов синдрома, связанных с
;13271  ;       ошибкой.
;13272  ;
;13273  ; ОШИБКИ:
;13274  ;
;13275  ;   ПРИМЕЧАНИЕ: для распечатки ошибок в одиночных битах так же, как и обычных
;13276  ;               ошибок, следует напечатать на терминале SE SE (установка разреше-
;13277  ;               ния сообщений об одиночных ошибках) и запустить этот тест. Когда
;13278  ;               будут обнаружены ошибки в одиночных битах, адреса и неправильные
;13279  ;               данные будут печататься с номерами ошибок 1 или 4.
;13280  ;
;13281  ;   ошибка 1 - тест "МАРШ" выполнен неуспешно при восходящих адресах с данными
;13282  ;              все 0 и все 1. Если выполнением управляет APT или разрешены со-
;13283  ;              общения об одиночных ошибках, о наличии ошибок в одиночных битах
;13284  ;              будет также сообщаться здесь.
;13285  ;   ошибка 2 - в CSR установлен бит ошибки; отличный от CRD, хотя данные были
;13286  ;              правильными.
;13287  ;   ошибка 3 - тест "МАРШ" выполнен неуспешно при нисходящих адресах с данными
;13288  ;              все 0 и все 1.
;13289  ;   ошибка 4 - тест "МАРШ" выполнен неуспешно при восходящих адресах с данными
;13290  ;              все 0 и 80000000(H). Если выполнением управляет APT или установ-
;13291  ;              лено разрешение сообщений об одиночных ошибках, полученные биты
;13292  ;              синдрома будут здесь выводиться (биты синдрома считываются инвер-
;13293  ;              тированными).
;13294  ;   ошибка 5 - в CSR установлен бит ошибки, отличный от CRD, хотя данные были
;13295  ;              правильными.
;13296  ;
;13297  ; НАЛАДКА:
;13298  ;
;13299  ;   ПРИМЕЧАНИЕ: если МСТ работает в пошаговом режиме, регенерация матрицы памяти
;13300  ;   пропускается. Таким образом, данные в матрице будут потеряны.
;13301  ;
```

;13302 ; ПРИМЕЧАНИЕ: ошибка 1 может вызываться, если любая из матриц "САЖАЕТ" в низ-
;13303 ; кий уровень адресный бит. Выделению неисправности такого типа поможет удале-.
;13304 ; ние всех плат матриц и замена заведомо исправными матрицами.
;13305 ;
;13306 ; ОШИБКА 1 - Эта ошибка может вызываться неисправностями в логических схемах
;13307 ; адреса в модуле матрицы памяти, или внутренней неисправностью микросхемы ОЗУ.
;13308 ; Неисправности в логических схемах адресации будут проявляться на адресах, в
;13309 ; которых установлен новый бит (например, на адресе 10(H), в котором бит 4 ус-
;13310 ; тановлен впервые) и все биты будут неправильными. Неисправности в микросхе-
;13311 ; мах ОЗУ могут проявляться на любых адресах и неправильными будут несколько
;13312 ; битов (вероятно 2).
;13313 ; Если ошибка имеет вид адресной ошибки (все биты неправильные), тогда пер-
;13314 ; выми сигналами, подлежащими проверке, будут адресные линии BUS ARRAY A0 H по
;13315 ; BUS ARRAY A6 H, выходящие из мультиплексоров адреса на модуле МСТ. При раз-
;13316 ; решенном зацикливании на ошибке следует просмотреть линии BUS ARRAY AX H.
;13317 ; Сигналы должны показывать адрес, по которому было обращение (напечатан под
;13318 ; заголовком OTHER (другие данные)), причем 7 младших битов попадают на выход,
;13319 ; когда низким является сигнал COLAD H, а 7 старших битов имеются на выходе,
;13320 ; когда COLAD H является высоким. Также следует проверить сигнал A7 H в моду-
;13321 ; лях ОЗУ на 64K. Он должен соответствовать битам адреса 16 и 17. Осциллограф
;13322 ; можно синхронизировать сигналом, поступающим на ножку 1 линии задержки и сфор-
;13323 ; мированным из RAS EN H, когда REF IN PROG L является высоким. Синхронизация
;13324 ; этим сигналом исключает переключение в момент, когда происходит регенерация
;13325 ; (которая выключает сигналы разрешения в мультиплексорах), а также сигнализи-
;13326 ; рует начало обращения к памяти.
;13327 ; Если ошибка содержится в адресных линиях, следует проверить входы к муль-
;13328 ; типлексорам. Если они правильные, то, вероятно, что неисправны мультиплексо-
;13329 ; ры.
;13330 ; Если эти адресные линии правильные, то необходимо проверить BUS ARRAY B SEL
;13331 ; 0 H и BUS ARRAY B SEL 1 H. Они выбирают правильный банк микросхем на плате мат-
;13332 ; рицы. Если ошибка происходит на границе банка (для матриц из ОЗУ по 16K это
;13333 ; происходит когда изменяются биты 16 или 17, для матриц из ОЗУ по 64K, это
;13334 ; происходит когда изменяются биты 18 или 19), подозрительной является эта зо-
;13335 ; на. Используйте ту же точку синхронизации, как и раньше, и проверте, что ли-
;13336 ; нии BUS ARRAY B SEL X соответствуют адресам, при которых тест проходит неус-
;13337 ; пешно.
;13338 ; Если в линии выборки существует ошибка, проверьте входы, включая входы вы-
;13339 ; борки SEL, на правильность их значений. Если входы правильные, то, вероятно,
;13340 ; что неисправен мультиплексор.
;13341 ; Если линии выборки правильные, единственным вероятным оставшимся местом
;13342 ; неправности на модуле МСТ являются линии MEM SEL. Здесь неисправность возмож-
;13343 ; на, если ошибка произошла на адресе, пересекающем границу матрицы. Проверьте
;13344 ; на низкий уровень сигнал MEM SEL X L на линии соответствующей матрицы, ко-
;13345 ; торая должна была быть выбранной. Если неправильный, несбходимо проверить на
;13346 ; правильность значений для использованных модулей матрицы сигналы FP. Если
;13347 ; сигналы FP правильные, то вероятно, что ПМЛ ДЕШ.ФИЗИЧЕСКОГО АДР., которая
;13348 ; генерирует нужный сигнал MEM SEL X L, неисправна.
;13349 ; Если все логические схемы адресации на плате МСТ выглядят работоспособ-
;13350 ; ными, то вероятно, что неисправность содержится в самой плате матрицы.
;13351 ;
;13352 ; ОШИБКА 2 - Для определения, какой бит ошибки является неправильным, исполь-
;13353 ; зуйте команду EX MC 01 и проследите этот бит назад. Эту ошибку может вызвать
;13354 ; двойная ошибка в контрольных битах (в этом случае в CSR1 будет установлен
;13355 ; бит RDS).
;13356 ;

```
                ;13357  ;       ОШИБКА 3 - С наибольшей вероятностью эта ошибка указывает на неисправность
                ;13358  ;    микросхемы ОЗУ. Неисправную микросхему можно локализовать путем наблюдения
                ;13359  ;    адреса и ошибочных битов.
                ;13360  ;
                ;13361  ;       ОШИБКА 4 - Эта ошибка может происходить только на адресах, которые не сра-
                ;13362  ;    батывали, вызывая ошибку 1 выше, иначе она указывает на неправильные контроль-
                ;13363  ;    ные биты. Если маской ошибки является FFFFFF80, то эта ошибка сообщает полу-
                ;13364  ;    ченные биты синдрома после ошибки в одиночном бите.
                ;13365  ;
                ;13366  ;       ОШИБКА 5 - Для определения, какой бит ошибки является неправильным, исполь-
                ;13367  ;    зуйте команду EX MC 01 и проследите этот бит назад. Если устанавливается бит
                ;13368  ;    RDS, он может означать ошибку на участке контрольных битов в ОЗУ. Она может
                ;13369  ;    вызываться обрывом адресной линии или неисправной микросхемой ОЗУ.
                ;13370  ;
                ;13371   T.21:
U 1186, B65E,15 ;13372           MOV LS[BEGIN.TEST] TO WR[0]            ; установка в WR0 бита для слова управления и состояния
U 1187, 3E80,15 ;13373           MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                ;13374                                                  ; 15 указывает для конс.процессора начало теста
U 1188, 10E0,15 ;13375           MISC [SET.CP.ATTN]                     ; выдача для конс.процессора CPU ATTN
                ;13376   WAIT.T21.0:
U 1189, 0918,94 ;13377           JMP [WAIT.T21.0]                       ; цикл для ожидания ответа конс.процессора
U 118A, 0A1A,9C ;13378           JSR [SETUP]                            ; установка масок, кода модуля и т.д. и очистка CSR1
                ;13379                                                  ; MCT
U 118B, 4748,15 ;13380           BIS LS[ARRAY] TO WR[0]                 ; установка бита для платы матрицы
U 118C, 3E8C,15 ;13381           MOV WR[0] TO LS[MODULE.NUM]            ; запоминание в LS кода модуля, указывающего плату MCT
                ;13382                                                  ; или матрицы
                ;13383             MOV LS[MM.LASTADDR+1] TO WR[2],      ; выборка из ячейки LS LAST.ADDRESS+1
U 118D, 3625,35 ;13384           DT(LONG)&SET.ALU.CC                    ; и установка кодов условий
U 118E, 8919,01 ;13385           JMP.IF[NEQ] TO [T21.CONTINUE]          ; продолжение теста, если LAST.ADDRESS+1 не нулевой
U 118F, 8A19,4C ;13386           JSR [SIZE.MEMORY]                      ; определение обьема памяти, если это не было выполнено
                ;13387                                                  ; ранее, и сообщение о результате
                ;13388   T21.CONTINUE:
U 1190, 3645,95 ;13389           MOV LS[#4] TO WR[3]                    ; увеличение на 4
U 1191, 6512,15 ;13390           CLR LS[T9]                             ; адреса для обращения к основной памяти
                ;13391   BACK.T21.1:
U 1192, 9912,75 ;13392           MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]     ; запрос для выполнения записи во всей доступной памяти
U 1193, B29C,15 ;13393           WRITE.MEM LS[ZERO]                     ; запись нулей
U 1194, 6C13,95 ;13394           ADD WR[3] TO LS[T9]                    ; следующий адрес
                ;13395             CMP WR[2] WITH LS[T9],               ; выполнено для всех адресов?
U 1195, CF13,35 ;13396           DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 1196, 0919,21 ;13397           JMP.IF[NEQ] TO [BACK.T21.1]            ; повторение, пока     будет выполнена запись по всем
                ;13398                                                  ; адресам
U 1197, 6514,15 ;13399           CLR LS[T10]                            ; очистка ошибок в      ночных битах
U 1198, 6512,15 ;13400           CLR LS[T9]                             ; очистка начальн     дреса для теста "МАРШ"
U 1199, 651E,15 ;13401           CLR LS[T15]                            ; очистка номера м      цы для распечатки одиночных
                ;13402                                                  ; ошибок
U 119A, FF1E,15 ;13403           INC LS[T15]                            ; запуск с номером матрицы 1
U 119B, 6588,15 ;13404           CLR LS[ADDRESS.DATA]                   ; а также очистка адреса для печати
U 119C, 3665,15 ;13405           MOV LS[#40000] TO WR[2]                ; "МАРШ" для одного блока на 256 кбайт (это последний
                ;13406                                                  ; адрес+1 для первого блока)
                ;13407   LOOP.T21.2:
U 119D, E5BA,15 ;13408           CLR LS[ERROR.MASK]                     ; будут проверяться все биты
U 119E, B670,15 ;13409           MOV LS[OTHER.DATA] TO WR[0]            ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13410                                                  ; (OTHER)
U 119F, 3E80,15 ;13411           MOV WR[0] TO LS[CONTROL.STATUS]         ; подготовка слова управления для печати под OTHER
```

```
                ;13412                                                  ; адреса в случае ошибки
U 11A0, B640,15 ;13413          MOV LS[#1] TO WR[0]                     ; номер ошибки
U 11A1, BE82,15 ;13414          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 1
                ;13415  LOOP.T21.1:
U 11A2, 2F82,95 ;13416          CLR WR[1]                               ; ожидаемые данные
U 11A3, E516,15 ;13417          CLR LS[T11]                             ; запоминание ожидаемых данных в LS для сообщений об
                ;13418                                                  ; одиночных ошибках
U 11A4, 1813,F5 ;13419          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для нулей
U 11A5, 3022,15 ;13420          MOV MEM.DATA TO WR[0]                   ; прием результата из памяти
U 11A6, 0869,3C ;13421          JSR [CHECK.RESULT]                      ; проверка на нули
U 11A7, 091A,24 ;13422          JMP [LOOP.T21.1]                        ; цикл при ошибке, если есть разрешение
U 11A8, B680,15 ;13423          MOV LS[CONTROL.STATUS] TO WR[0]         ; выборка слова управления и состояния
                ;13424            BIT LS[BIT8] WITH WR[0],              ; проверка, была ли ошибка (программа CHECK.RESULT при
                ;13425                                                  ; ошибке установит бит 8)
U 11A9, D950,35 ;13426          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 11AA, C550,15 ;13427          BIC LS[BIT8] TO WR[0]                   ; очистка бита 8, если он установлен
U 11AB, 3E80,15 ;13428          MOV WR[0] TO LS[CONTROL.STATUS]         ; восстановление слова управления с очищенным битом 8
U 11AC, 091B,61 ;13429          JMP.IF[BIT.SET] TO [T21.WRITE.ONES]     ; если ошибка, не будет проверяться суммарная ошибка
                ;13430                                                  ; системы памяти
U 11AD, 5B00,1D ;13431          SKIP.IF[MEM.REF.OK]                     ; иначе проверка суммарной ошибки
U 11AE, B922,5C ;13432          JSR [T21.2.ERRORSUM]                    ; переход к проверке, что установлен бит CRD или
                ;13433                                                  ; произошла другая ошибка
U 11AF, 091B,64 ;13434          JMP [T21.WRITE.ONES]                    ; переход сюда, если нет разрешения цикла при ошибке или
                ;13435                                                  ; отсутствует суммарная ошибка
U 11B0, 0919,D4 ;13436          JMP [LOOP.T21.2]                        ; цикл при ошибке, если есть разрешение
                ;13437  LOOP.T21.2A:
U 11B1, E58A,15 ;13438          CLR LS[ERROR.MASK]                      ; будут проверяться все биты
U 11B2, B670,15 ;13439          MOV LS[OTHER.DATA] TO WR[0]             ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13440                                                  ; (OTHER)
U 11B3, 3E80,15 ;13441          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка слова управления для печати адреса под OTHER
                ;13442                                                  ; в случае ошибки
U 11B4, B640,15 ;13443          MOV LS[#1] TO WR[0]                     ; номер ошибки
U 11B5, BE82,15 ;13444          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 1
                ;13445  T21.WRITE.ONES:
U 11B6, 369E,95 ;13446          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 11B7, BE16,95 ;13447          MOV WR[1] TO LS[T11]                    ; запоминание в LS ожидаемых данных для сообщений об
                ;13448                                                  ; одиночных ошибках
U 11B8, 9912,75 ;13449          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи в память инвертированных данных
U 11B9, 329E,15 ;13450          WRITE.MEM LS[ONES]                      ; запись единиц в память
U 11BA, 1813,F5 ;13451          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения вновь записанных данных
U 11BB, 3022,15 ;13452          MOV MEM.DATA TO WR[0]                   ; прием результата
U 11BC, 0869,3C ;13453          JSR [CHECK.RESULT]                      ; проверка данных на все единицы
U 11BD, 091B,64 ;13454          JMP [T21.WRITE.ONES]                    ; цикл при ошибке, если есть разрешение
U 11BE, B680,15 ;13455          MOV LS[CONTROL.STATUS] TO WR[0]         ; выборка слова управления и состояния
                ;13456            BIT LS[BIT8] WITH WR[0],              ; проверка, произошла ли ошибка (програма CHECK.RESULT
                ;13457                                                  ; должна установить при ошибке бит 8)
U 11BF, D950,35 ;13458          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 11C0, C550,15 ;13459          BIC LS[BIT8] TO WR[0]                   ; очистка бита 8, если он установлен
U 11C1, 3E80,15 ;13460          MOV WR[0] TO LS[CONTROL.STATUS]         ; восстановление слова управления с очищенным битом 8
U 11C2, 091C,71 ;13461          JMP.IF[BIT.SET] TO [T21.1.INC]          ; если ошибка, не будет проверяться суммарная ошибка
                ;13462                                                  ; системы памяти. Переход к следующему адресу
U 11C3, 5B00,1D ;13463          SKIP.IF[MEM.REF.OK]                     ; иначе проверка суммарной ошибки
U 11C4, B922,5C ;13464          JSR [T21.2.ERRORSUM]                    ; переход к проверке, что установлен бит CRD или
                ;13465                                                  ; произошла другая ошибка
U 11C5, 5B00,1E ;13466          SKIP                                    ; переход сюда, если нет цикла при ошибке или
```

```
                ;13467                                          ; отсутствует суммарная ошибка системы памяти
U 11C6, B91B,14 ;13468          JMP [LOOP.T21.2A]               ; цикл по ошибке CBR1, если есть разрешение
                ;13469  T21.1.INC:
U 11C7, 6C13,95 ;13470          ADD WR[3] TO LS[T9]             ; увеличение адреса на 4
U 11C8, 6C89,95 ;13471          ADD WR[3] TO LS[ADDRESS.DATA]   ; а также увеличение адреса для печати
                ;13472            CMP WR[2] WITH LS[T9],        ; проверка, что тест выполнен для всех адресов в текущем
                ;13473                                          ; блоке
U 11C9, CF13,35 ;13474          DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 11CA, 091A,21 ;13475          JMP.IF[NEQ] TO [LOOP.T21.1]     ; повторение теста с нового адреса, если не закончен,
                ;13476                                          ; иначе запуск с убывающими адресами
U 11CB, 36C0,15 ;13477          MOV LS[#3(H)] TO WR[0]          ; рабочий регистр содержит 3
U 11CC, BE82,15 ;13478          MOV WR[0] TO LS[ERROR.NUMBER]   ; ошибка 3 (убывающие адреса)
U 11CD, 3E13,15 ;13479          MOV WR[2] TO LS[T9]             ; выборка верхнего адреса+1 для текущего блока
U 11CE, 3E89,15 ;13480          MOV WR[2] TO LS[ADDRESS.DATA]   ; адрес+1 для печати
U 11CF, 4165,15 ;13481          SUB LS[#40000] FROM WR[2]       ; вычисление нижнего блока
                ;13482  REPEAT.T21.3:
U 11D0, E813,95 ;13483          SUB WR[3] FROM LS[T9]           ; уменьшение адреса на 4
U 11D1, E889,95 ;13484          SUB WR[3] FROM LS[ADDRESS.DATA] ; а также уменьшение адреса для печати
                ;13485  LOOP.T21.3:
U 11D2, 369E,95 ;13486          MOV LS[ONES] TO WR[1]           ; ожидаемые данные
U 11D3, 1813,F5 ;13487          MEM.REQ[READ.P] ADRS[T9] DT[LONG] ; запрос для чтения единиц
U 11D4, 3022,15 ;13488          MOV MEM.DATA TO WR[0]           ; прием результата из памяти
U 11D5, 0869,3C ;13489          JSR [CHECK.RESULT]              ; проверка на все единицы
U 11D6, B91D,24 ;13490          JMP [LOOP.T21.3]                ; цикл при ошибке, если есть разрешение
                ;13491  LOOP.T21.3.COMP:
U 11D7, 2F82,95 ;13492          CLR WR[1]                       ; ожидаемые данные
U 11D8, 9912,75 ;13493          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG] ; запрос для записи в память инвертированных данных
U 11D9, B29C,15 ;13494          WRITE.MEM LS[ZERO]              ; запись нулей в память
U 11DA, 1813,F5 ;13495          MEM.REQ[READ.P] ADRS[T9] DT[LONG] ; запрос для считывания вновь записанных данных
U 11DB, 3022,15 ;13496          MOV MEM.DATA TO WR[0]           ; прием результата
U 11DC, 0869,3C ;13497          JSR [CHECK.RESULT]              ; проверка данных на нули
U 11DD, B91D,74 ;13498          JMP [LOOP.T21.3.COMP]           ; цикл при ошибке, если есть разрешение
                ;13499            CMP WR[2] WITH LS[T9],        ; проверка, что тест выполнен для всех адресов в текущем
                ;13500                                          ; блоке
U 11DE, CF13,35 ;13501          DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 11DF, 091D,01 ;13502          JMP.IF[NEQ] TO [REPEAT.T21.3]   ; повторение  теста с новым адресом, если не закончен,
                ;13503                                          ; иначе запуск с новыми тестовыми данными
U 11E0, 3E13,15 ;13504          MOV WR[2] TO LS[T9]             ; подготовка адреса запуска
U 11E1, 3E89,15 ;13505          MOV WR[2] TO LS[ADDRESS.DATA]   ; а также установка адреса для печати
U 11E2, C065,15 ;13506          ADD LS[#40000] TO WR[2]         ; тест "МАРШ" для одного блока из 256 кбайт (это
                ;13507                                          ; последний адрес+1 для текущего блока)
                ;13508  LOOP.T21.5:
U 11E3, E58A,15 ;13509          CLR LS[ERROR.MASK]              ; будут проверяться все биты
U 11E4, B670,15 ;13510          MOV LS[OTHER.DATA] TO WR[0]     ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13511                                          ; (OTHER)
U 11E5, 3EB0,15 ;13512          MOV WR[0] TO LS[CONTROL.STATUS] ; подготовка слова управления для печати под "OTHER"
                ;13513                                          ; адреса в случае ошибки
U 11E6, 3644,15 ;13514          MOV LS[#4] TO WR[0]             ; рабочий регистр содержит 4
U 11E7, BE82,15 ;13515          MOV WR[0] TO LS[ERROR.NUMBER]   ; ошибка 4
                ;13516  LOOP.T21.4:
U 11E8, 2F82,95 ;13517          CLR WR[1]                       ; ожидаемые данные
U 11E9, 1813,F5 ;13518          MEM.REQ[READ.P] ADRS[T9] DT[LONG] ; запрос для чтения нулей
U 11EA, 3022,15 ;13519          MOV MEM.DATA TO WR[0]           ; выборка результата из памяти
U 11EB, 0869,3C ;13520          JSR [CHECK.RESULT]              ; проверка на нули
U 11EC, B91E,84 ;13521          JMP [LOOP.T21.4]                ; цикл при ошибке, если есть резрешение
```

```
U 11ED, B680,15 ;13522          MOV LS[CONTROL.STATUS] TO WR[0]          ; выборка слова управления и состояния
                ;13523            BIT LS[BIT8] WITH WR[0],               ; проверка, произошла ли ошибка (программа CHECK.RESULT
                ;13524                                                   ; должна была установить при ошибке бит 8)
U 11EE, D950,35 ;13525          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 11EF, C550,15 ;13526          BIC LS[BIT8] TO WR[0]                    ; очистка бита 8, если он установлен
U 11F0, 3E80,15 ;13527          MOV WR[0] TO LS[CONTROL.STATUS]          ; восстановление слова управления, если бит 8 очищен
U 11F1, 091F,B1 ;13528          JMP.IF[BIT.SET] TO [T21.WRITE.4.COMP]    ; если ошибка, не будет проверяться суммарная ошибка
                ;13529                                                   ; системы памяти
U 11F2, 5B00,1D ;13530          SKIP.IF[MEM.REF.OK]                      ; иначе проверка суммарной ошибки
U 11F3, 0923,9C ;13531          JSR [T21.5.ERRORSUM]                     ; проверка, что установлен бит CRD или произошла другая
                ;13532                                                   ; ошибка
U 11F4, 091F,B4 ;13533          JMP [T21.WRITE.4.COMP]                   ; переход сюда, если нет цикла при ошибке или
                ;13534                                                   ; отсутствует суммарная ошибка системы памяти
U 11F5, 091E,34 ;13535          JMP [LOOP.T21.5]                         ; цикл при ошибке, если есть разрешение
                ;13536  LOOP.T21.5A:
U 11F6, E58A,15 ;13537          CLR LS[ERROR.MASK]                       ; будут проверяться все биты
U 11F7, B670,15 ;13538          MOV LS[OTHER.DATA] TO WR[0]              ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13539                                                   ; (OTHER)
U 11F8, 3E80,15 ;13540          MOV WR[0] TO LS[CONTROL.STATUS]           ; подготовка слова управления для печати под OTHER
                ;13541                                                   ; адреса в случае ошибки
U 11F9, 3644,15 ;13542          MOV LS[#4] TO WR[0]                      ; рабочий регистр содержит 4
U 11FA, BE82,15 ;13543          MOV WR[0] TO LS[ERROR.NUMBER]            ; ошибка 4
                ;13544  T21.WRITE.4.COMP:
U 11FB, B67E,95 ;13545          MOV LS[#80000000] TO WR[1]               ; ожидаемые данные
U 11FC, 9912,75 ;13546          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]       ; запрос для записи в память инвертированных контрольных
                ;13547                                                   ; битов
U 11FD, B27E,15 ;13548          WRITE.MEM LS[#80000000]                  ; запись в память 80000000(H)
U 11FE, 1813,F5 ;13549          MEM.REQ[READ.P] ADRS[T9] DT[LONG]        ; запрос для чтания вновь записанных данных
U 11FF, 3022,15 ;13550          MOV MEM.DATA TO WR[0]                    ; прием результата
U 1200, 0869,3C ;13551          JSR [CHECK.RESULT]                       ; проверка, что данные равны 80000000
U 1201, 091F,B4 ;13552          JMP [T21.WRITE.4.COMP]                   ; цикл при ошибке, если есть разрешение
U 1202, B680,15 ;13553          MOV LS[CONTROL.STATUS] TO WR[0]          ; выборка слова управления и состояния
                ;13554            BIT LS[BIT8] WITH WR[0],               ; проверка, не произошла ли ошибка (программа
                ;13555                                                   ; CHECK.RESULT должна быть установить в случае ошибки
                ;13556                                                   ; бит 8)
U 1203, D950,35 ;13557          DT(LONG)&SET.ALU.CC                      ; и установка кодов усл
U 1204, C550,15 ;13558          BIC LS[BIT8] TO WR[0]                    ; очистка бита 8, если    установлен
U 1205, 3E80,15 ;13559          MOV WR[0] TO LS[CONTROL.STATUS]           ; восстановление слова   вления с очищенным битом 8
U 1206, 0920,B1 ;13560          JMP.IF[BIT.SET] TO [T21.4.INC]           ; если ошибка, не будет   ряться суммарная ошибка
                ;13561                                                   ; системы памяти. Перех    следующему адресу
U 1207, 5B00,1D ;13562          SKIP.IF[MEM.REF.OK]                      ; иначе проверка суммар    ошибки
U 1208, 0923,9C ;13563          JSR [T21.5.ERRORSUM]                     ; переход к проверке,     тановлен бит CRD или
                ;13564                                                   ; произошла другая оши
U 1209, 5B00,1E ;13565          SKIP                                     ; переход сюда, если на    ла при ошибке или
                ;13566                                                   ; отсутствует суммарна     ка
U 120A, B91F,64 ;13567          JMP [LOOP.T21.5A]                        ; цикл по ошибке CSR1,     есть разрешение
                ;13568  T21.4.INC:
U 120B, 6C13,95 ;13569          ADD WR[3] TO LS[T9]                      ; увеличение адреса на
U 120C, 6C89,95 ;13570          ADD WR[3] TO LS[ADDRESS.DATA]            ; а также увеличение адр    для печати
                ;13571            CMP WR[2] WITH LS[T9],                 ; проверка, что тест бы    н для всех адресов текущего
                ;13572                                                   ; блока
U 120D, CF13,35 ;13573          DT(LONG)&SET.ALU.CC                      ; и установка кодов услови
U 120E, B91E,B1 ;13574          JMP.IF[NEQ] TO [LOOP.T21.4]              ; повторение теста с нов    адресом, если не закончен,
                ;13575                                                   ; иначе печать числа оши    в одиночных битах для
                ;13576                                                   ; данного блока и запуск     та с новым блоком из 256
```

```
                ;13577                                          ; кбайт
U 120F, 3614,15 ;13578          MOV LS[T10] TO WR[0]            ; выборка счетчика ошибок в одиночных битах
U 1210, 3E0E,15 ;13579          MOV WR[0] TO LS[T7.SUB]         ; занесение в LS7 числа ошибок для печати. Биты 24-31
                ;13580                                          ; содержат номер блока. Ячейка LS T15 содержит номер
                ;13581                                          ; матрицы (для ОЗУ по 64 К)
U 1211, 3674,15 ;13582          MOV LS[PRINT.CRD] TO WR[0]      ; установка в рабочем регистре бита 26
U 1212, 3E80,15 ;13583          MOV WR[0] TO LS[CONTROL.STATUS] ; указание консоли, что следует печатать число одиночных
                ;13584                                          ; ошибок для текущего блока
U 1213, 10E0,15 ;13585          MISC [SET.CP.ATTN]              ; выдача для конс.процессора CPU ATTN
                ;13586  WAIT.T21.1:
U 1214, 8921,44 ;13587          JMP [WAIT.T21.1]                ; цикл для ожидания ответа конс.процессора
U 1215, 3624,15 ;13588          MOV LS[MM.LASTADDR+1] TO WR[0]  ; проверка, проведен ли тест всей доступной памяти
                ;13589            CMP WR[0] WITH LS[T9],        ; текущий адрес=последний адрес+1?
U 1216, 4F12,35 ;13590          DT(LONG)&SET.ALU.CC             ; установка кодов условий
U 1217, 8927,99 ;13591          JMP.IF[EQL] TO [END.T21]        ; переход к окончанию, если тест выполнен
U 1218, 3614,15 ;13592          MOV LS[T10] TO WR[0]            ; иначе выборка счетчика ошибок в одиночных битах из
                ;13593                                          ; предыдущего блока
U 1219, 5F2A,95 ;13594          MCOM LS[#FF000000] TO WR[1]     ; загрузка в рабочий регистр FFFFFF (H)
U 121A, AF42,15 ;13595          BIC WR[1] TO WR[0]              ; очистка счетчика ошибок, оставляя немодифицированным
                ;13596                                          ; номер блока
U 121B, 4070,15 ;13597          ADD LS[BIT24] TO WR[0]          ; увеличение номера блока в счетчике ошибок в одиночных
                ;13598                                          ; битах
U 121C, 3670,95 ;13599          MOV LS[BIT24] TO WR[1]          ; установка в рабочем регистре бита 24
U 121D, C772,95 ;13600          BIS LS[BIT25] TO WR[1]          ; и установка в рабочем регистре бита 25
                ;13601            BIT WR[1] WITH WR[0],         ; проверка, что достигнута четная граница, равная 4 (оба
                ;13602                                          ; бита очищены)
U 121E, A602,35 ;13603          DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 121F, 8922,21 ;13604          JMP.IF[BIT.SET] TO [STORE.T21]  ; переход, если один из битов или оба установлены
U 1220, FF1E,15 ;13605          INC LS[T15]                     ; иначе увеличение номера матрицы; примечание: нельзя
                ;13606                                          ; использовать другой ячейки LS кроме T15!!! монитор
                ;13607                                          ; отыскивает в этой ячейке LS номер матрицы
U 1221, 2F80,15 ;13608          CLR WR[0]                       ; повторный запуск с блоком 0
                ;13609  TORE.T21:
U 1222, BE14,15 ;13610          MOV WR[0] TO LS[T10]            ; запоминание в LS
U 1223, C065,15 ;13611          ADD LS[#40000] TO WR[2]         ; последний адрес следующего блока из 256 кбайт
U 1224, 0919,D4 ;13612          JMP [LOOP.T21.2]                ; запуск теста "МАРШ" для другого блока
                ;13613  ;
                ;13614  ; подпрограммы для этого теста
                ;13615  ;
                ;13616  T21.2.ERRORSUM:
U 1225, FF82,15 ;13617          INC LS[ERROR.NUMBER]            ; ошибка 2
U 1226, E580,15 ;13618          CLR LS[CONTROL.STATUS]          ; запрет печати поля OTHER (другие данные)
U 1227, 373E,15 ;13619          MOV LS[#3F003FFF] TO WR[0]      ; установка маски для проверки в CSR только битов
                ;13620                                          ; ошибок
U 1228, 3E8A,15 ;13621          MOV WR[0] TO LS[ERROR.MASK]     ; установка маски ошибок
U 1229, 367C,95 ;13622          MOV LS[CRD] TO WR[1]            ; ожидаемые данные (бит CRD установлен, другие очищены)
U 122A, 9D45,75 ;13623          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG] ; запрос для чтения CSR
U 122B, 3022,15 ;13624          MOV MEM.DATA TO WR[0]           ; прием содержимого CSR1
U 122C, 0869,3C ;13625          JSR [CHECK.RESULT]              ; проверка, что бит CRD установлен, другие очищены
U 122D, DB00,16 ;13626          RETURN+1                        ; возврат+1 для выполнения цикла при ошибке
U 122E, E58A,15 ;13627          CLR LS[ERROR.MASK]              ; проверка всех битов
U 122F, FC82,15 ;13628          DEC LS[ERROR.NUMBER]            ; ошибка 1
U 1230, B670,15 ;13629          MOV LS[OTHER.DATA] TO WR[0]     ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13630                                          ; (OTHER)
U 1231, 3E80,15 ;13631          MOV WR[0] TO LS[CONTROL.STATUS] ; установка слова управления на печать под "OTHER" в
```

```
                ;13632                                                  ; случае ошибки
U 1232, 0924,DC ;13633          JSR [CNT.SYNDROME]                      ; переход к определению, имеется ли одиночная ошибка в
                ;13634                                                  ; битах данных
U 1233, 0923,B4 ;13635          JMP [NO.CRD.RETURN]                     ; возврат сюда, если не ошибка в одиночном бите
U 1234, B923,B9 ;13636          JMP.IF[EQL] TO [NO.CRD.RETURN]          ; обход увеличения числа ошибок в данных, если был
                ;13637                                                  ; установлен только один синдром (указывает ошибку в
                ;13638                                                  ; контрольных битах)
U 1235, FF14,15 ;13639          INC LS[T10]                             ; иначе увеличение промежуточной ячейки LS (счетчик
                ;13640                                                  ; ошибок в одиночных битах данных)
U 1236, 8925,EC ;13641          JSR [CHECK.SER]                         ; проверка, управляет ли тестом
                ;13642                                                  ; APT или установлен
                ;13643                                                  ; признак SER и распечатка ошибки, если один из этих
                ;13644                                                  ; случаев
U 1237, 0926,5C ;13645          JSR [REPORT.SINGLE.DATA]                ; возврат сюда, если одиночные ошибки должны
                ;13646                                                  ; генерировать сообщения также, как и регулярные ошибки
                ;13647                                                  ; теста
                ;13648  NO.CRD.RETURN:
U 1238, 5B00,14 ;13649          RETURN                                  ; конец выполнения
                ;13650  T21.5.ERRORSUM:
U 1239, FF82,15 ;13651          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 123A, E580,15 ;13652          CLR LS[CONTROL.STATUS]                  ; запрет печати поля OTHER (ДРУГИЕ ДАННЫЕ)
U 123B, 373E,15 ;13653          MOV LS[#3F003FFF] TO WR[0]              ; маска для проверки только битов ошибок в CSR
U 123C, 3E8A,15 ;13654          MOV WR[0] TO LS[ERROR.MASK]             ; установка маски ошибок
U 123D, 367C,95 ;13655          MOV LS[CRD] TO WR[1]                    ; ожидаемые данные (бит CRD установлен, другие очищены)
U 123E, 9D45,75 ;13656          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR
U 123F, 3022,15 ;13657          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 1240, 0869,3C ;13658          JSR [CHECK.RESULT]                      ; проверка, что бит CRD установлен, другие очищены
U 1241, DB00,16 ;13659          RETURN+1                                ; возврат+1 для выполнения цикла при ошибке
U 1242, E58A,15 ;13660          CLR LS[ERROR.MASK]                      ; будут проверяться все биты
U 1243, FC82,15 ;13661          DEC LS[ERROR.NUMBER]                    ; ошибка 4
U 1244, B670,15 ;13662          MOV LS[OTHER.DATA] TO WR[0]             ; установка бита для печати поля "ДРУГИЕ ДАННЫЕ"
                ;13663                                                  ; (OTHER)
U 1245, 3E80,15 ;13664          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка слова управления для печати под OTHER адреса
                ;13665                                                  ; в случае ошибки
U 1246, 0924,DC ;13666          JSR [CNT.SYNDROME]                      ; переход для определения, была ли одиночная ошибка в
                ;13667                                                  ; битах данных
U 1247, 0924,C4 ;13668          JMP [NO.CRD.RETURN.5]                   ; возврат сюда, если не ошибка в одиночном бите
U 1248, 0924,C1 ;13669          JMP.IF[NEQ] TO [NO.CRD.RETURN.5]        ; обход увеличения числа ошибок в контрольных битах,
                ;13670                                                  ; если был установлен более, чем один синдром (указывает
                ;13671                                                  ; ошибку в данных)
U 1249, FF14,15 ;13672          INC LS[T10]                             ; иначе увеличение промежуточной ячейки LS (счетчик
                ;13673                                                  ; одиночных ошибок в контрольных битах)
U 124A, 8925,EC ;13674          JSR [CHECK.SER]                         ; проверка управляет ли выполнением теста APT или
                ;13675                                                  ; установлен признак SER и распечатка ошибки в любом из
                ;13676                                                  ; этих случаев
U 124B, 8926,EC ;13677          JSR [REPORT.SINGLE.CKBT]                ; возврат сюда, если одиночные ошибки должны
                ;13678                                                  ; генерировать сообщения также, как и регулярные ошибки
                ;13679                                                  ; теста
                ;13680  NO.CRD.RETURN.5:
U 124C, 5B00,14 ;13681          RETURN                                  ; конец выполнения
                ;13682  CNT.SYNDROME:
U 124D, 3650,15 ;13683          MOV LS[ERROR] TO WR[0]                  ; установка в рабочем регистре бита 8
                ;13684            BIT WR[0] WITH LS[CONTROL.STATUS],    ; проверка, был ли установлен в управляющем слове бит
                ;13685                                                  ; ошибки
U 124E, 5980,55 ;13686          DT(SIZE)&SET.ALU.CC                     ; и установка кодов условий
```

```
U 124F, 0925,D1 ;13687          JMP.IF[BIT.SET] TO [NO.CRD]                 ; переход, если в CSR1 не был установлен бит CRD
U 1250, 9D9D,75 ;13688          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]       ; иначе запрос для чтения битов синдрома
U 1251, 3022,15 ;13689          MOV MEM.DATA TO WR[0]                       ; прием содержимого CSR0
U 1252, 6518,15 ;13690          CLR LS[T12]                                 ; счетчик установленных  синдромов
U 1253, E5F8,15 ;13691          CLR LS[OS]                                  ; очистка индекса
                ;13692  REPEAT.SYN.CNT:
                ;13693          BIT LS[SHIFT.OS(4-0)] WITH WR[0],           ; проверка, установлен ли текущий бит синдрома
U 1254, D9F6,35 ;13694          DT(LONG)&SET.ALU.CC                         ; и установка кодов условий
U 1255, DB00,01 ;13695          SKIP.IF[BIT.SET]                            ; пропуск, если текущий бит синдрома установлен (биты
                ;13696                                                      ; синдрома считываются инвертированными)
U 1256, FF18,15 ;13697          INC LS[T12]                                 ; иначе увеличение счетчика установленных битов синдрома
U 1257, 7FF8,15 ;13698          INC LS[OS]                                  ; проверка сдедующего бита синдрома
U 1258, B64E,95 ;13699          MOV LS[BIT7] TO WR[1]                       ; установка в рабочем регистре бита 7
                ;13700            BIT WR[1] WITH LS[SHIFT.OS(4-0)],         ; проверка, что все 7 битов синдрома просмотрены
U 1259, 59F6,B5 ;13701          DT(LONG)&SET.ALU.CC                         ; и установка кодов условий
U 125A, 8925,49 ;13702          JMP.IF[BITS.CLR] TO [REPEAT.SYN.CNT]        ; повторение счета синдромов, если не закончено
                ;13703            DEC LS[T12],                              ; вычитание 1 из счетчика синдромов
U 125B, 7C18,35 ;13704          DT(LONG)&SET.ALU.CC                         ; и установка кодов условий
U 125C, DB00,16 ;13705          RETURN+1                                    ; возврат+1 в программу счета одиночных ошибок
                ;13706  NO.CRD:
U 125D, 5B00,14 ;13707          RETURN                                      ; возврат, если не ошибка CRD (одиночная)
                ;13708  CHECK.SER:
U 125E, 3690,15 ;13709          MOV LS[ERROR.CONTROL] TO WR[0]              ; выборка слова управления ошибками
U 125F, B648,95 ;13710          MOV LS[SER] TO WR[1]                        ; установка в рабочем регистре бита SER
U 1260, 474A,95 ;13711          BIS LS[APT.PRESENT] TO WR[1]                ; установка в рабочем регистре бита "АРТ ПРИСУСТВУЕТ"
                ;13712            BIT WR[1] WITH WR[0],                     ; проверка, что какой-либо из этих битов установлен в
                ;13713                                                      ; слове управления ошибками
U 1261, A602,35 ;13714          DT(LONG)&SET.ALU.CC                         ; и установка кодов условий
U 1262, 5B00,09 ;13715          SKIP.IF[BITS.CLR]                           ; пропуск на возврат+1, если не установлены ни SER, ни
                ;13716                                                      ; APT
U 1263, 5B00,14 ;13717          RETURN                                      ; возврат для печати сообщения об ошибке
U 1264, DB00,16 ;13718          RETURN+1                                    ; возврат без сообщения об ошибке
                ;13719  REPORT.SINGLE.DATA:
U 1265, 3672,15 ;13720          MOV LS[ECC.DIS] TO WR[0]                    ; установка в рабочем регистре бита запрета коррекции
                ;13721                                                      ; (ECC)
U 1266, BA17,FC ;13722          JSR [WRITE.CSR1]                            ; установка бита ECC DIS в CSR1
                ;13723  LOOP.SINGLE:
U 1267, 3616,95 ;13724          MOV LS[T11] TO WR[1]                        ; ожидаемые данные
U 1268, 1813,F5 ;13725          MEM.REQ[READ.P] ADRS[T9] DT[LONG]           ; запрос для чтения ячейки, содержащей ошибку, при
                ;13726                                                      ; запрещенной коррекции
U 1269, 3022,15 ;13727          MOV MEM.DATA TO WR[0]                       ; выборка из памяти неисправленных данных
U 126A, 0869,3C ;13728          JSR [CHECK.RESULT]                          ; печать сообщения об ошибке
U 126B, 0926,74 ;13729          JMP [LOOP.SINGLE]                           ; цикл при ошибке, если есть разрешение
U 126C, 0A17,EC ;13730          JSR [CLEAR.CSR1]                            ; очистка CSR1 для восстановления разрешения коррекции
U 126D, 5B00,14 ;13731          RETURN                                      ; конец выполнения
                ;13732  REPORT.SINGLE.CKBT:
U 126E, B712,15 ;13733          MOV LS[#FFFFFF80] TO WR[0]                  ; установка в рабочем регистре битов 7-31
U 126F, 3E8A,15 ;13734          MOV WR[0] TO LS[ERROR.MASK]                 ; будут проверяться только биты с 0 по 6
                ;13735  LOOP.SINGLE.CKBT:
U 1270, 1813,F5 ;13736          MEM.REQ[READ.P] ADRS[T9] DT[LONG]           ; запрос для чтения ячейки, содержащей ошибку
U 1271, 3022,15 ;13737          MOV MEM.DATA TO WR[0]                       ; завершение цикла данных
U 1272, 369E,95 ;13738          MOV LS[ONES] TO WR[1]                       ; принудительный вызов ошибки (ожидаемыми данными не
                ;13739                                                      ; могут быть все единицы, так как синдромы считываются
                ;13740                                                      ; инвертированными)
U 1273, 9D9D,75 ;13741          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]       ; запрос для чтения битов синдрома
```

```
U 1274,  3022,15 ;13742          MOV MEM.DATA TO WR[0]                  ; выборка синдромов
U 1275,  0869,3C ;13743          JSR [CHECK.RESULT]                     ; сообщение, содержащее полученные синдромы
U 1276,  0927,04 ;13744          JMP [LOOP.SINGLE.CKBT]                 ; цикл при ошибке, если есть разрешение
U 1277,  E58A,15 ;13745          CLR LS[ERROR.MASK]                     ; восстановление проверки всех битов
U 1278,  5800,14 ;13746          RETURN                                 ; конец выполнения
                 ;13747  END.T21:
```

```
;13748  .PAGE "ТЕСТ 22 - NXM при обращении к устройству ОШ (модуль МСТ)"
;13749  ;
;13750  ; ОПИСАНИЕ ТЕСТА:
;13751  ;
;13752  ;       Этот тест проверяет логические схемы, связанные с принудительной установ-
;13753  ;   кой бита NXM, когда происходит таймаут на общей шине. Бит NXM будет устанав-
;13754  ;   ливаться при чтении или записи, направленной к несуществующему устройству на
;13755  ;   общей шине или к устройству, которое не выдает ответа. Вначале этот тест вы-
;13756  ;   дает запись по адресу FFFFFE, который является несуществующим адресом общей
;13757  ;   шины (нет устройств, которые отзываются на этот адрес). Бит NXM проверяется,
;13758  ;   чтобы удостовериться, что он установлен. Затем выполняется чтение по тому же
;13759  ;   адресу и бит NXM проверяется, чтобы убедиться, что он снова установлен.
;13760  ;       Бит NXM устанавливается в циклах чтения или записи микрокодом памяти,
;13761  ;   возбуждающим ROT C0 и стробирующим CSR. Сигнал ROT C0 объединяется по "ИЛИ"
;13762  ;   с битом NXM в ПМЛ CSR1A. Микрокод возбуждает сигнал ROT C0 только, если тай-
;13763  ;   маут общей шины происходит дважды, прежде, чем появится SSYN. Сигнал TIMOUT
;13764  ;   L генерируется каждый раз, когда REF IN PROG L переключается с низкого уров-
;13765  ;   ня на высокий, и остается возбужденным в течение 1 цикла памяти. Сигнал
;13766  ;   TIMEOUT L поступает в логические схемы ветвления, чтобы было вызвано ветвле-
;13767  ;   ние в микрокодах памяти. Если сигнал таймаута поступает дважды перед прихо-
;13768      дом сигнала SSYN, устанавливается бит NXM.
;13769  ;
;13770  ; ПРЕДПОЛОЖЕНИЯ:
;13771  ;
;13772  ;   Принимается, что все предыдущие тесты прошли успешно.
;13773  ;
;13774  ; ШАГИ ТЕСТА:
;13775  ;
;13776  ;   1) Установка в LS номера ошибки, номера модуля (для распечатки ошибок) и
;13777  ;      очистка в LS номера предыдущей ошибки.
;13778  ;   2) Установка маски для проверки бита NXM (бит 16).
;13779  ;   3) Выдача операции записи по адресу FFFFFE и чтение CSR1 для проверки, что
;13780  ;      бит NXM установлен.
;13781  ;   4) Выдача операции чтения по адресу FFFFFE и чтение CSR1 для проверки, что
;13782  ;      бит NXM установлен.
;13783  ;
;13784  ; ОШИБКИ:
;13785  ;
;13786  ;   ошибка 1 - бит NXM не устанавливается при записи в несуществующее устройство
;13787  ;              общей шины
;13788  ;   ошибка 2 - бит NXM не устанавливается при чтении из несуществующего устройства
;13789  ;              общей шины
;13790  ;
;13791  ; НАЛАДКА:
;13792  ;
;13793  ;       ОШИБКА 1 - Эта ошибка указывает на проблему, связанную с сигналом ROT C0
;13794  ;   или логическими схемами таймаута, которые также управляют логическими схе-
;13795  ;   мами ветвления. Сигнал TIMEOUT L можно проверить в то время, когда память
;13796  ;   работает в холостом цикле, так как он поступает из логических схем регенера-
;13797  ;   ции. Сигнал необходимо проверить на входе к мультиплексору ветвления BEN0.
;13798  ;   Он должен образовать импульсы низкого уровня на 1 цикл памяти один раз в 12
;13799  ;   мкс (приблизительно). Если сигнал TIMOUT L не пульсирующий, проверьте его на
;13800  ;   выходе из ПМЛ СИГН. КОНТР.ПИТАНИЯ И ИНИЦИАЦИИ. Если сигнал здесь неправиль-
;13801  ;   ный, проверьте на входах наличие импульсов сигнала REF IN PROG L. Если он
;13802  ;   правильный, то ПМЛ неисправна.
```

```
                ;13803  ;       Если сигнал TIMEOUT L правильный, тогда убедитесь, что во время зацикливa-
                ;13804  ;   ния на ошибке сигнал ROT C0 становится высоким. Если нет, то может быть де-
                ;13805  ;   фектным мультиплексор BEN 0, поэтому ветвление по сигналу TIMEOUT L не про-
                ;13806  ;   исходит.
                ;13807  ;       Если сигнал ROT C0 правильный, то убедитесь, что он проходит через инвер-
                ;13808  ;   тор и схему "ИЛИ" к ПМЛ CSR 1A.
                ;13809  ;
                ;13810  ;       ОШИБКА 2 - То же, что и для ошибки 1.
                ;13811  ;
                ;13812   T.22:
U 1279, B65E,15 ;13813           MOV LS[BEGIN.TEST] TO WR[0]            ; установка в WR0 бита 15 для слова управления и
                ;13814                                                  ; состояния
U 127A, 3E80,15 ;13815           MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                ;13816                                                  ; 15 указывает для.консольного процессора начало теста
U 127B, 10E0,15 ;13817           MISC [SET.CP.ATTN]                     ; выдача для консольного процессора CPU ATTN
                ;13818   WAIT.T22.0:
U 127C, 0927,C4 ;13819           JMP [WAIT.T22.0]                       ; цикл для ожидания ответа консольного процессора
U 127D, 0A1A,9C ;13820           JSR [SETUP]                            ; установка масок, кода модуля и т.д. и очистка в МСТ
                ;13821                                                  ; CSR 1
U 127E, 5F60,15 ;13822           MCOM LS[NXM] TO WR[0]                  ; очистка в рабочем регистре бита 16
U 127F, 3E8A,15 ;13823           MOV WR[0] TO LS[ERROR.MASK]            ; будет проверяться только бит NXM (бит 16 в CSR1)
U 1280, DF2A,15 ;13824           MCOM LS[#FF000000] TO WR[0]            ; занесение в WR0 кода FFFFFF
U 1281, 2100,15 ;13825           DEC WR[0]                              ; теперь регистр содержит FFFFFE(H)
U 1282, BE12,15 ;13826           MOV WR[0] TO LS[T9]                    ; адрес находится в LS
                ;13827   LOOP.T22.1:
U 1283, B660,95 ;13828           MOV LS[NXM] TO WR[1]                   ; ожидаемые данные (установленный бит NXM)
U 1284, 1912,35 ;13829           MEM.REQ[WRITE.P] ADRS[T9] DT[WORD]     ; запрос для записи в несуществующее устройство
U 1285, 329E,15 ;13830           WRITE.MEM LS[ONES]                     ; запись  единиц по несуществующему адресу общей шины
                ;13831                                                  ; (должен возникать таймаут)
U 1286, 9D45,75 ;13832           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения CSR1
U 1287, 3022,15 ;13833           MOV MEM.DATA TO WR[0]                  ; содержимое CSR1 находится в рабочем регистре
U 1288, 0869,3C ;13834           JSR [CHECK.RESULT]                     ; проверка, что бит NXM установлен
U 1289, 0928,34 ;13835           JMP [LOOP.T22.1]                       ; цикл при ошибке, если есть разрешение
U 128A, FF82,15 ;13836           INC LS[ERROR.NUMBER]                   ; ошибка 2
                ;13837   LOOP.T22.2:
U 128B, B660,95 ;13838           MOV LS[NXM] TO WR[1]                   ; ожидаемые данные (установленный бит NXM)
U 128C, 9813,B5 ;13839           MEM.REQ[READ.P] ADRS[T9] DT[WORD]      ; запрос для чтения и[з] несуществующего устройства
U 128D, 3022,15 ;13840           MOV MEM.DATA TO WR[0]                  ; чтение единиц по несуществующему адресу общей шины
                ;13841                                                  ; (должен произойти таймаут)
U 128E, 9D45,75 ;13842           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения CSR1
U 128F, 3022,15 ;13843           MOV MEM.DATA TO WR[0]                  ; содержимое CSR1 нах[о]дится в рабочем регистре
U 1290, 0869,3C ;13844           JSR [CHECK.RESULT]                     ; проверка, что бит NXM установлен (при чтении бит NXM
                ;13845                                                  ; устанавливается, если был таймаут)
U 1291, 8928,B4 ;13846           JMP [LOOP.T22.2]                       ; цикл при ошибке, е[с]ть разрешение
                ;13847   END.T22:
```

```
;13848  .PAGE "ТЕСТЫ ДРУГИХ ЛОГИЧЕСКИХ СХЕМ КОНТРОЛЛЕРА ОЗУ"
;13849  .TOC "ТЕСТ 23 - прекращение инструкции при ошибке паритета в WCS (модуль MCT или DAP)"
;13850  ;
;13851  ; ОПИСАНИЕ ТЕСТА:
;13852  ;
;13853  ;       Этот тест проверяет логические схемы, которые вынуждают микрокод MCT перей-
;13854  ;   ти в адрес 100(H), если во время начального ветвления обнаружена ошибка пари-
;13855  ;   тета в микрокоде центрального процессора. Формируется сигнал CPU ATTN с уста-
;13856  ;   новленным в слове управления и состояния битом 9, который запрашивает монитор
;13857  ;   создать ошибку паритета WCS в ячейке, заданной в LS 7. Ячейка LS 7 была за-
;13858  ;   гружена ранее адресом инструкции MEM.REQ, используемой в этом тесте. Вначале
;13859  ;   тест записывает все единицы по адресу 0 буфера трансляции TB. Затем выдается
;13860  ;   инструкция MEM.REQ, содержащая ошибку паритета, для записи всех нулей по ад-
;13861  ;   ресу 0 TB. Ошибка паритета должна вынудить микрокод памяти прекратить началь-
;13862  ;   ное ветвление и перейти к программе инициализации по адресу 100(H). Затем
;13863  ;   производится проверка по адресу 0 TB, чтобы удостовериться, что запись не
;13864  ;   происходила. Монитор, вместо того, чтобы сообщить об ошибке паритета, как об
;13865  ;   обычной ошибке, ее игнорирует.
;13866  ;
;13867  ; ПРЕДПОЛОЖЕНИЯ:
;13868  ;
;13869  ;   Принимается, что все предыдущие тесты прошли успешно.
;13870  ;
;13871  ; ШАГИ ТЕСТА:
;13872  ;
;13873  ;   1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок) и
;13874  ;      очистка в LS номера предыдущей ошибки.
;13875  ;   2) Загрузка маски ошибок с проверяемыми только битами данных 22-1
;13876  ;   3) Загрузка в LS 7 адреса инструкции MEM.REQ для шага 5 ниже, затем выдача
;13877  ;      CPU ATTN при установленном бите 9 для создания ошибки паритета в этом ми-
;13878  ;      крослове.
;13879  ;   4) Запись в TB по адресу 0 всех единиц в битах 22-1 и проверка.
;13880  ;   5) Выдача инструкции MEM.REQ с ошибкой паритета при MF=WRITE.TB и типе дан-
;13881  ;      ных=длинное слово.
;13882  ;   6) Выдача инструкции WRITE.MEM LS с ячейкой LS, содержащей нули. Эта инст-
;13883  ;      рукция запишет единицы в TB только в случае, если логические схемы ошибки
;13884  ;      паритета не вынудят MCT перейти к циклу инициализации.
;13885  ;   7) Чтение из адреса 0 TB с проверкой, что биты 22-1 все еще содержат нули.
;13886  ;   8) Восстановление правильного паритета в микрослове для шага 5 и восстанов-
;13887  ;      ление разрешения контроля паритета.
;13888  ;
;13889  ; ОШИБКИ:
;13890  ;
;13891  ;   ПРИМЕЧАНИЕ: ожидаемыми и полученными данными являются биты данных с 22 по 1
;13892  ;   из TB по адресу 0.
;13893  ;
;13894  ;   ошибка 1 - начальная запись в TB 0 неуспешна.
;13895  ;   ошибка 2 - ошибка паритета не вынудила MCT перейти в цикл инициализации.
;13896  ;
;13897  ; НАЛАДКА:
;13898  ;
;13899  ;       ОШИБКА 1 - Эти логические схемы проверялись предыдущими тестами. Если
;13900  ;   произошла эта ошибка, выполните снова предыдущие тесты.
;13901  ;
;13902  ;       ОШИБКА 2 - Эта ошибка указывает на неисправность логических схем ошибки
```

```
                ;13903  ;   паритета в модуле MCT или неправильный вход из модуля DAP. Центральный про-
                ;13904  ;   цессор необходимо остановить в пошаговом режиме на инструкции MEM.REQ, со-
                ;13905  ;   держащей ошибку паритета. Память должна вернуться в холостой цикл. Если нет,
                ;13906  ;   то проверьте сигнал CS PAR ERR H, поступающий в инвертор. Если этот сигнал
                ;13907  ;   не является высоким, проследите его назад к источнику в модуле DAP централь-
                ;13908  ;   ного процессора.
                ;13909  ;      Если сигнал CS PAR ERR H около инвертора является высоким, проверьте его на
                ;13910  ;   ножке 1 микросхемы И-или-НЕ на низкий уровень. если он правильный, и пошаго-
                ;13911  ;   вый режим MCT можно осуществить, то переведите MCT в пошаговый режим и пере-
                ;13912  ;   ведите по шагам центральный процессор снова к инструкции MEM.REQ с ошибкой
                ;13913  ;   паритета. Выполните один шаг MCT, так что управление окажется на инструкции
                ;13914  ;   START.ADDR.PROM и проверьте высокий уровень сигнала STOP MEM H. Если он не-
                ;13915  ;   правильный, то проверьте оба сигнала CS PERR L и DISP EN L, входящие в мик-
                ;13916  ;   росхему. Если они низкие, то микросхема неисправна.
                ;13917  ;      Если сигнал STOP MEM H имеет высокий уровень, то проверьте на низкий уро-
                ;13918  ;   вень сигнал ADRS 8 L. Если он правильный, проверьте на высокий уровень сигна-
                ;13919  ;   лы разрешения мультиплексоров ветвления BEN X.
                ;13920  ;      Если они правильные, проверьте сигнал разрешения на мультиплексоре, кото-
                ;13921  ;   рый обеспечивает биты адреса с 4 по 7 управляющих ПЗУ. Этот сигнал разрешения
                ;13922  ;   должен быть высоким. если он неправильный, его можно проследить назад.
                ;13923  ;      Наконец, проверьте выходы адреса. Все они должны быть в состоянии высокого
                ;13924  ;   импеданса (не возбуждены) за исключением ADRS 8 L, который должен быть низ-
                ;13925  ;   ким (возбужден).
                ;13926  ;
                ;13927
                ;13928  T.23:
U 1292, B65E,15 ;13929          MOV LS[BEGIN.TEST] TO WR[0]          ; установка в WR0 бита 15 для слова управления и
                ;13930                                               ; состояния
U 1293, 3E80,15 ;13931          MOV WR[0] TO LS[CONTROL.STATUS]      ; установка бита 15 в слове управления и состояния. Бит
                ;13932                                               ; 15 указывает для конс.процессора начало теста
U 1294, 10E0,15 ;13933          MISC [SET.CP.ATTN]                   ; выдача для консольного процессора CPU ATTN
                ;13934  WAIT.T23.0:
U 1295, 8929,54 ;13935          JMP [WAIT.T23.0]                     ; цикл для ожидания ответа конс.процессора
U 1296, 0A1A,9C ;13936          JSR [SETUP]                          ; установка масок, кода модуля и т.д. и очистка CSR1
                ;13937                                               ; MCT
U 1297, 4742,15 ;13938          BIS LS[CPU] TO WR[0]                 ; а также установка бита 1
U 1298, 3E8C,15 ;13939          MOV WR[0] TO LS[MODULE.NUM]          ; запоминание в LS кода модуля, указывающего плату MCT
                ;13940                                               ; или DAP
U 1299, B62A,15 ;13941          MOV LS[#FF000000] TO WR[0]           ; установка старшего байта в качестве маски ошибок
U 129A, C76E,15 ;13942          BIS LS[BIT23] TO WR[0]               ; а также установка бита 23
U 129B, C740,15 ;13943          BIS LS[BIT0] TO WR[0]                ; и бита 0
U 129C, 3E8A,15 ;13944          MOV WR[0] TO LS[ERROR.MASK]          ; теперь биты с 22 по 1 будут проверяться на наличие
                ;13945                                               ; ошибок
U 129D, B658,15 ;13946          MOV LS[#1000] TO WR[0]               ; рабочий регистр содержит 1000
U 129E, C752,15 ;13947          BIS LS[#200] TO WR[0]                ; рабочий регистр содержит 1200
U 129F, 474E,15 ;13948          BIS LS[#80] TO WR[0]                 ; рабочий регистр содержит 1280
U 12A0, C74C,15 ;13949          BIS LS[#40] TO WR[0]                 ; рабочий регистр содержит 12C0
U 12A1, 3E0E,15 ;13950          MOV WR[0] TO LS[T7.SUB]              ; загрузка в LS 7 адреса инструкции MEM.REQ для
                ;13951                                               ; занесения неправильного паритета
U 12A2, B652,15 ;13952          MOV LS[SET.PA.ERR] TO WR[0]          ; установка в рабочем регистре бита 9
U 12A3, 3E80,15 ;13953          MOV WR[0] TO LS[CONTROL.STATUS]      ; установка бита 9 в словеуправления и состояния
                ;13954                                               ; установленный бит 9 указывает монитору, что необходимо
                ;13955                                               ; занести неправильный паритет в слово WCS, заданное в
                ;13956                                               ; LS 7
U 12A4, 10E0,15 ;13957          MISC [SET.CP.ATTN]                   ; выдача для конс. процессора CPU ATTN
```

```
                    ;13958   WAIT.T23.1:
U 12A5,  B92A,54    ;13959           JMP [WAIT.T23.1]                        ; здесь цикл ожидания реакции конс. процессора
                    ;13960   LOOP.T23.1:
U 12A6,  369E,95    ;13961           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 12A7,  989C,F5    ;13962           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в TB по адресу 0
U 12A8,  329E,15    ;13963           WRITE.MEM LS[ONES]                      ; запись в TB 0 всех единиц
U 12A9,  9C9D,F5    ;13964           MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; запрос для чтения в TB по адресу 0
U 12AA,  3022,15    ;13965           MOV MEM.DATA TO WR[0]                   ; чтение содержимого TB 0
U 12AB,  0869,3C    ;13966           JSR [CHECK.RESULT]                      ; проверка, что запись в TB выполнена правильно
U 12AC,  B92A,64    ;13967           JMP [LOOP.T23.1]                        ; цикл при ошибке, если есть разрешение
U 12AD,  FF82,15    ;13968           INC LS[ERROR.NUMBER]                    ; ошибка 2
                    ;13969   LOOP.T23.2:
U 12AE,  369E,95    ;13970           MOV LS[ONES] TO WR[1]                   ; ожидаемые данные (изменений в TB не произошло)
U 12AF,  B92C,04    ;13971           JMP [T23.PAR.ERR]                       ; переход к запросу-памяти с ошибкой паритета в
                    ;13972                                                   ; инструкции
                    ;13973   12C0:
                    ;13974   T23.PAR.ERR:
U 12C0,  989C,F5    ;13975           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для попытки записи в TB
U 12C1,  B29C,15    ;13976           WRITE.MEM LS[ZERO]                      ; попытка записи нулей в TB
U 12C2,  9C9D,F5    ;13977           MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; запрос для чтения TB
U 12C3,  3022,15    ;13978           MOV MEM.DATA TO WR[0]                   ; прием содержимого TB
U 12C4,  0869,3C    ;13979           JSR [CHECK.RESULT]                      ; проверка, что содержимое TB не изменилось
U 12C5,  092A,E4    ;13980           JMP [LOOP.T23.2]                        ; цикл при ошибке, если есть разрешение
U 12C6,  B65B,15    ;13981           MOV LS[#1000] TO WR[0]                  ; рабочий регистр содержит 1000
U 12C7,  C752,15    ;13982           BIS LS[#200] TO WR[0]                   ; рабочий регистр содержит 1200
U 12C8,  474E,15    ;13983           BIS LS[#80] TO WR[0]                    ; рабочий регистр содержит 1280
U 12C9,  C74C,15    ;13984           BIS LS[#40] TO WR[0]                    ; рабочий регистр содержит 12C0
U 12CA,  3E0E,15    ;13985           MOV WR[0] TO LS[T7.SUB]                 ; загрузка в LS 7 адреса инструкции MEM.REQ для
                    ;13986                                                   ; перезаписи ее с правильным паритетом
U 12CB,  B676,15    ;13987           MOV LS[CLR.PA.ERR] TO WR[0]             ; установка в рабочем регистре бита 27
U 12CC,  3E80,15    ;13988           MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 27 в слове управления и состояния
                    ;13989                                                   ; установленный бит 27 указывает монитору, что
                    ;13990                                                   ; необходимо занести правильный паритет в слово WCS,
                    ;13991                                                   ; заданное в LS 7
U 12CD,  10E0,15    ;13992           MISC [SET.CP.ATTN]                      ; выдача для конс. процессора CPU ATTN
                    ;13993   WAIT.T23.2:
U 12CE,  092C,E4    ;13994           JMP [WAIT.T23.2]                        ; цикл для ожидания    ета конс.процессора
                    ;13995   END.T23:
```

```
;13996  .PAGE "ТЕСТ 24 - тест *МАРШ* для битов CSR1 и бита ошибки ERR SUM (модуль МСТ)"
;13997  ;
;13998  ;  ОПИСАНИЕ ТЕСТА:
;13999  ;
;14000  ;        Этот тест проверяет, что каждый бит ошибки в CSR1 (за исключением UBBSY,
;14001  ;     RDS и CRD) может устанавливаться независимо, и что каждый бит сам приводит
;14002  ;     к установке бита суммарной ошибки системы памяти ERROR SUM. кроме того,
;14003  ;     проверяются логические схемы сигнала CLR ERR L, который очищает биты ошибок
;14004  ;     при любой записи в CSR1. Первая часть теста также проверяет, что сигнал
;14005  ;     ADDR PH очищает биты ошибок VALID, TB PAR ERROR и TB MISS (это функции ПМЛ
;14006  ;     CSR1). Этот тест использует те же логические схемы для установки битов оши-
;14007  ;     бок, которые использовались в предыдущих тестах, за исключением того, что
;14008  ;     осуществляется проверка всех битов, чтобы удостовериться, что устанавлива-
;14009  ;     ется только один бит. После того, как каждый бит проверен, выполняется про-
;14010  ;     пуск по отсутствию ошибки системы памяти с целью убедиться, что был уста-
;14011  ;     новлен бит ERROR SUM.
;14012  ;
;14013  ;  ПРЕДПОЛОЖЕНИЯ:
;14014  ;
;14015  ;        Принимается, что все предыдущие тесты выполнены успешно.
;14016  ;
;14017  ;  ШАГИ ТЕСТА:
;14018  ;
;14019  ;        1) Установка в LS номера ошибки и номера модуля (для распечатки ошибок)
;14020  ;           и очистка в LS номера предыдущей ошибки.
;14021  ;        2) Установка маски ошибок для проверки битов от 14 до 23. Очистка битов
;14022  ;           режима (MODE) в PSL для задания режима ядра (KERNAL). Установка в
;14023  ;           CSR1 битов MME и TB PAR DIAG (биты 27 и 29).
;14024  ;        3) Запись в буфер трансляции TB по адресу 0 с установленными битами 25
;14025  ;           и 28. Этим устанавливаются биты BYTE OFFSET и биты PROT принимают
;14026  ;           значение 2.
;14027  ;        4) Выполнение функции памяти TEST.V.RCHK с адресом F00000. Выполнение
;14028  ;           инструкции MOV MEM.DATA TO WR для завершения обращения к памяти.
;14029  ;        5) Чтение CSR1 и проверка, что биты VALID, TB MISS и TB PAR ERR установ-
;14030  ;           лены, другие очищены.
;14031  ;        6) Выполнение функции памяти READ.P по адресу F00000. Выдача инструкции
;14032  ;           MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14033  ;           бит UB ADAP REG SEL установлен, другие очищены.
;14034  ;        7) Выполнение инструкции SKIP.IF(MEM.REF.OK) и проверка, что пропуск не
;14035  ;           происходит. Запись в CSR1 и проверка, что все биты ошибок очищены.
;14036  ;        8) Выполнение функции памяти READ.P по адресу 540000. Выдача инструкции
;14037  ;           MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14038  ;           бит NXM установлен, другие  очищены. затем повторение шага 7.
;14039  ;        9) Выполнение функции памяти READ.P по адресу FFFFFF. Выдача инструкции
;14040  ;           MOV MEM.DATA для завершения обращения к памяти. Проверка, что бит ILL
;14041  ;           UB OPER установлен, другие очищены. Затем повторение шага 7.
;14042  ;       10) Запись в TB по адресу 0 с данными, состоящими из 1 в битах BYTE OFFSET,
;14043  ;           VALID, MOD и 2 в битах PROT (установлены биты 25,26,28 и 31). Другие
;14044  ;           биты нулевые.
;14045  ;       11) Выполнение функции памяти TEST.V.RCHK по адресу 0. Выдача инструкции
;14046  ;           MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14047  ;           бит TB PAR ERR установлен, другие очищены. Затем повторение шага 7.
;14048  ;       12) Очистка в CSR1 бита TB PAR DIAG (бит MME остается установленным).
;14049  ;       13) Выполнение функции памяти TEST.V.WCHK по адресу 1FF. Выдача инструкции
;14050  ;           MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
```

```
;14051  ;          бит ошибки WR ACROSS PG ERR установлен, другие очищены. Затем повто-
;14052  ;          рение шага 7.
;14053  ;       14) Запись в ТВ 0 данных, состоящих из 1 в битах BYTE OFFSET, VALID и 2 в
;14054  ;          битах PROT (установлены биты 25, 28 и 31). Другие биты равны 0.
;14055  ;       15) Выполнение функции памяти TEST.V.WCHK по адресу 0. Выдача инструкции
;14056  ;          MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14057  ;          бит MODIFY REFUSED установлен, другие очищены. Затем повторение шага 7.
;14058  ;       16) Запись в ТВ 0 данных, состоящих из 1 в битах BYTE OFFSET и VALID
;14059  ;          (установлены биты 25 и 31). Другие биты равны 0.
;14060  ;       17) Выполнение функции памяти TEST.V.WCHK по адресу 0. Выдача инструкции
;14061  ;          MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14062  ;          бит ACCESS REFUSED установлен, другие очищены. Затем повторение шага 7.
;14063  ;       18) Запись в ТВ 0 данных, состоящих из 1 в бите BYTE OFFSET и 2 в битах
;14064  ;          PROT (установлены биты 25 и 28). Другие биты равны 0.
;14065  ;       19) Выполнение функции памяти TEST.V.RCHK по адресу 0. Выдача инструкции
;14066  ;          MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка, что
;14067  ;          бит VALID установлен, другие очищены. Затем повторение шага 7.
;14068  ;       20) Запись в ТВ 0 данных, состоящих из 1 в бите VALID и 2 в битах PROT
;14069  ;          (установлены биты 28 и 31). Другие биты равны 0.
;14070  ;       21) Выполнение функции памяти TEST.V.RCHK по адресу 0. Выдача инструк-
;14071  ;          ции MOV MEM.DATA TO WR для завершения обращения к памяти. Проверка,
;14072  ;          что бит TB MISS установлен, другие очищены. Затем повторение шага 7.
;14073  ;
;14074  ;  ОШИБКИ:
;14075  ;
;14076  ;       ПРИМЕЧАНИЕ: ожидаемыми и полученными данными являются биты ошибок CSR1
;14077  ;       (биты 14-23).
;14078  ;
;14079  ;       ошибка  1 - не срабатывают или закорочены между собой ранее проверенные
;14080  ;                   биты CSR.
;14081  ;       ошибка  2 - сигнал ADDR PH не очищает в CSR биты ошибок для виртуального
;14082  ;                   адреса или существует замыкание между битом UB ADAP REG SEL
;14083  ;                   и другими.
;14084  ;       ошибка  3 - по биту UB ADP REG SEL не устанавливается ERROR SUM.
;14085  ;       ошибка  4 - бит UB ADP REG SEL не очищается при записи в CSR1.
;14086  ;       ошибка  5 - при установке NXM устанавливаются другие биты.
;14087  ;       ошибка  6 - по биту NXM не устанавливается ERROR SUM.
;14088  ;       ошибка  7 - бит NXM не очищается при записи в CSR1.
;14089  ;       ошибка  8 - при установке бита ILL UB OPER устанавливаются другие биты.
;14090  ;       ошибка  9 - по биту ILL UB OPER не возбуждается  ERROR SUM.
;14091  ;       ошибка  A - бит ILL UB OPER не очищается при записи в CSR1.
;14092  ;       ошибка  B - при установке бита TB PAR ERR устанавливаются другие биты.
;14093  ;       ошибка  C - по биту TB PAR ERR не возбуждается ERROR SUM.
;14094  ;       ошибка  D - бит TB PAR ERR не очищается при записи в CSR1.
;14095  ;       ошибка  E - при установке бита WR ACROSS PG ERR устанавливаются другие
;14096  ;                   биты.
;14097  ;       ошибка  F - по биту WR ACROSS PG ERR не возбуждается ERROR SUM.
;14098  ;       ошибка 10 - бит WR ACROSS PG ERR не очищается при записи в CSR1.
;14099  ;       ошибка 11 - при установке бита MODIFY REFUSED устанавливаются другие биты.
;14100  ;       ошибка 12 - по биту MODIFY REFUSED не возбуждается ERROR SUM.
;14101  ;       ошибка 13 - бит MODIFY REFUSED не очищается при записи в CSR1.
;14102  ;       ошибка 14 - при установке бита ACCESS REFUSED устанавливаются другие биты.
;14103  ;       ошибка 15 - по биту ACCESS REFUSED не возбуждается ERROR SUM.
;14104  ;       ошибка 16 - бит ACCESS REFUSED не очищается при записи в CSR1.
;14105  ;       ошибка 17 - при установке бита VALID устанавливаются другие биты.
```

```
                ;14106  ;       ошибка 18 - по биту VALID не устанавливается ERROR SUM.
                ;14107  ;       ошибка 19 - бит VALID не очищается при записи в CSR1.
                ;14108  ;       ошибка 1A - при установке бита TB MISS устанавливаются другие биты.
                ;14109  ;       ошибка 1B - по биту TB MISS не устанавливается ERROR SUM.
                ;14110  ;       ошибка 1C - бит TB MISS не очищается при записи в CSR1.
                ;14111  ;
                ;14112  ;  НАЛАДКА:
                ;14113  ;
                ;14114  ;          ОШИБКА 1 - Эта ошибка указывает на проблему, связанную с ранее прове-
                ;14115  ;       рявшимися логическими схемами или на замыкание между битами. Должны быть
                ;14116  ;       установленными только биты 14, 15 и 21. Если все предыдущие тесты прохо-
                ;14117  ;       дят, следует подозревать замыкание между битами или неисправность ПМЛ,
                ;14118  ;       которая их генерирует.
                ;14119  ;
                ;14120  ;          ОШИБКА 2 - Если установлены какие-либо биты, кроме бита 18, следует по-
                ;14121  ;       дозревать неисправность ПМЛ CSR, которая генерирует данный бит, или сиг-
                ;14122  ;       нал ADDR PH L на входе ПМЛ CSR. Если сигнал ADDR PH L не становится низ-
                ;14123  ;       ким, биты 14, 15 или 21 не будут очищаться. Если бит 18 не установлен,
                ;14124  ;       проверьте на замыкание между втим битом и другими.
                ;14125  ;
                ;14126  ;          ОШИБКИ 3,6,9,C,F,12,15,18,1B - Подозревается микросхема ПМЛ CSR, кото-
                ;14127  ;       рая генерирует данный бит ошибки.
                ;14128  ;
                ;14129  ;          ОШИБКИ 4,7,A,D,10,13,16,19,1C - Подозревается ПМЛ CSR, которая генери-
                ;14130  ;       рует данный бит ошибки или вход CLR ERR L.
                ;14131  ;
                ;14132  ;          ОШИБКИ 5,8,B,E,11,14,17,1A - Подозревается замыкание между битами оши-
                ;14133  ;       бок или ПМЛ CSR, которая генерирует бит ошибки.
                ;14134  ;
                ;14135  ;
                ;14136
                ;14137  T.24:
U 12CF, B65E,15 ;14138          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;14139                                                  ; состояния
U 12D0, 3E80,15 ;14140          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;14141                                                  ; 15 указывает для консольного процессора начало теста
U 12D1, 10E0,15 ;14142          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;14143  WAIT.T24.0:
U 12D2, 892D,24 ;14144          JMP [WAIT.T24.0]                        ; цикл для ожидания ответа консольного процессора
U 12D3, 0A1A,AC ;14145          JSR [SETUP.1]                           ; установка масок, номера модуля и др.
U 12D4, 3628,15 ;14146          MOV LS[#FFFF] TO WR[0]                  ; установка в рабочем регистре битов 0-15
U 12D5, C72A,15 ;14147          BIS LS[#FF000000] TO WR[0]              ; а также установка в рабочем регистре битов 24-31
                ;14148                                                  ; теперь очищены только биты 16-23
U 12D6, C55C,15 ;14149          BIC LS[VALID.ERR] TO WR[0]              ; очистка бита 14
U 12D7, 455E,15 ;14150          BIC LS[TB.PAR.ERR] TO WR[0]             ; и бита 15
U 12D8, 3E8A,15 ;14151          MOV WR[0] TO LS[ERROR.MASK]             ; будут проверяться биты 14-23 (биты ошибок CSR1)
U 12D9, 2F80,15 ;14152          CLR WR[0]                               ; очистка рабочего регистра
U 12DA, BFFE,15 ;14153          MOV WR[0] TO LS[PSL.HW]                 ; очистка PSL для установки текущего режима=ядро
                ;14154                                                  ; (KERNEL)
U 12D8, 0A17,DC ;14155          JSR [WRITE.CSR1.MME]                    ; разрешение работы диспетчера памяти
U 12DC, 3672,15 ;14156          MOV LS[BIT25] TO WR[0]                  ; установка в рабочем регистре бита 25 (используется для
                ;14157                                                  ; установки в TB бита BYTE OFFSET)
U 12E0, 4780,15 ;14158          BIS LS[BIT28] TO WR[0]                  ; используется для установки в TB значения 2 для битов
                ;14159                                                  ; PROT
U 12E2, 3E14,15 ;14160          MOV WR[0] TO LS[TB]                     ; вместимое данных в LS
```

```
U 12DF, 6512,15 ;14161          CLR LS[T9]                              ; очистка адреса в LS
U 12E0, 9B12,F5 ;14162          MEM.REQ[WRITE.TB] ADRS[T9] DT[LONG]     ; запрос для записи в ТВ по адресу 0
U 12E1, B214,15 ;14163          WRITE.MEM LS[T10]                       ; установка в ТВ битов BYTE OFFSET и PROT B
U 12E2, B676,15 ;14164          MOV LS[MME] TO WR[0]                    ; установка в рабочем регистре бита 27
U 12E3, C77A,15 ;14165          BIS LS[TB.PAR.DIAG] TO WR[0]            ; а также установка бита 29
U 12E4, BA17,FC ;14166          JSR [WRITE.CSR1]                        ; биты разрешения диспетчера памяти MME и ошибки
                ;14167                                                  ; паритета ТВ (TB PAR ERR)
U 12E5, B65D,15 ;14168          MOV LS[VALID.ERR] TO WR[2]              ; установка в рабочем регистре бита 14
U 12E6, C76B,15 ;14169          BIS LS[TB.MISS] TO WR[2]                ; а также установка бита 21
U 12E7, 475F,15 ;14170          BIS LS[TB.PAR.ERR] TO WR[2]             ; и бита 15. Это ожидаемые данные
U 12E8, B73A,15 ;14171          MOV LS[#F00000] TO WR[0]                ; установка в рабочем регистре адреса регистра адаптера
                ;14172                                                  ; общей шины (UB ADAP REG)
U 12E9, BE12,15 ;14173          MOV WR[0] TO LS[T9]                     ; подготовка LS для обращения к регистру адаптера общей
                ;14174                                                  ; шины
                ;14175  LOOP.T24.1:
U 12EA, A004,95 ;14176          MOV WR[2] TO WR[1]                      ; ожидаемые данные (установленные биты VALID ERR, TB
                ;14177                                                  ; MISS и TB PAR ERR)
U 12EB, 9813,75 ;14178          MEM.REQ[TEST.V.RCHK] ADRS[T9] DT[LONG]  ; запрос для пробного чтения ТВ по адресу 0
U 12EC, 3022,15 ;14179          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 12ED, 9D45,75 ;14180          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 12EE, 3022,15 ;14181          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 12EF, 0869,3C ;14182          JSR [CHECK.RESULT]                      ; проверка, что установлены биты VALID ERR, TB MISS и TB
                ;14183                                                  ; PAR ERR
U 12F0, 092E,A4 ;14184          JMP [LOOP.T24.1]                        ; цикл при ошибке, если есть разрешение
U 12F1, 3643,95 ;14185          MOV LS[#2] TO WR[3]                     ; номер ошибки 2 запоминается в рабочем регистре для
                ;14186                                                  ; восстановления
                ;14187  LOOP.T24.3:
U 12F2, E580,15 ;14188          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 12F3, BE83,95 ;14189          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 2
                ;14190  LOOP.T24.2:
U 12F4, 3664,95 ;14191          MOV LS[ADP.REG] TO WR[1]                ; ожидаемые данные (установленный бит UB ADAP REG SEL)
U 12F5, 1813,F5 ;14192          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения по физическому адресу F00000(H)
U 12F6, 3022,15 ;14193          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 12F7, 9D45,75 ;14194          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 12F8, 3022,15 ;14195          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 12F9, 0869,3C ;14196          JSR [CHECK.RESULT]                      ; проверка, что установлен бит UB ADAP REG SEL
U 12FA, 092F,44 ;14197          JMP [LOOP.T24.2]                        ; цикл при ошибке, если разрешено
U 12FB, FF82,15 ;14198          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 12FC, 5B00,1D ;14199          SKIP.IF[MEM.REF.OK]                     ; проверка наличия суммарной ошибки ERROR SUM (пропуск
                ;14200                                                  ; не должен происходить)
U 12FD, 0930,04 ;14201          JMP [TEST.T24.4]                        ; ошибки нет, переход к следующей части теста
U 12FE, 093A,BC ;14202          JSR [T24.NO.ERRSUM]                     ; сообщение об отсутствии ERROR SUM (ошибка 3)
U 12FF, 092F,24 ;14203          JMP [LOOP.T24.3]                        ; цикл при ошибке, если есть разрешение
                ;14204  TEST.T24.4:
U 1300, 8938,1C ;14205          JSR [CHECK.CLRERR]                      ; проверка, что запись в CSR приводит к очистке битов
                ;14206                                                  ; ошибок, номер ошибки 4
U 1301, 3645,95 ;14207          MOV LS[#4] TO WR[3]                     ; занесение в WR3 числа 4
U 1302, 2047,95 ;14208          INC WR[3]                               ; в WR3 запоминается номер ошибки 5
U 1303, 3738,15 ;14209          MOV LS[#540000] TO WR[0]                ; несуществующий адрес памяти (NXM)
U 1304, BE12,15 ;14210          MOV WR[0] TO LS[T9]                     ; запоминание в LS
                ;14211  LOOP.T24.6:
U 1305, E580,15 ;14212          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 1306, BE83,95 ;14213          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 5
                ;14214  LOOP.T24.5:
U 1307, B660,95 ;14215          MOV LS[NXM] TO WR[1]                    ; ожидаемые данные (установленный бит NXM)
```

```
U 1308,  1813,F5 ;14216          MEM.REQ[READ.P] ADRS[T9] DT[LONG]      ; запрос для чтения по физическому адресу 540000(H)
U 1309,  3022,15 ;14217          MOV MEM.DATA TO WR[0]                  ; выдача инструкции MOV для завершения микроцикла
U 130A,  9D45,75 ;14218          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения CSR1
U 130B,  3022,15 ;14219          MOV MEM.DATA TO WR[0]                  ; прием содержимого CSR1
U 130C,  0869,3C ;14220          JSR [CHECK.RESULT]                     ; проверка, что бит NXM установлен
U 130D,  8930,74 ;14221          JMP [LOOP.T24.5]                       ; цикл при ошибке, если есть разрешение
U 130E,  FF82,15 ;14222          INC LS[ERROR.NUMBER]                   ; ошибка 6
U 130F,  5B00,1D ;14223          SKIP.IF[MEM.REF.OK]                    ; проверка суммарной ошибки (пропуск не должен
                 ;14224                                                 ; произойти)
U 1310,  8931,34 ;14225          JMP [TEST.T24.7]                       ; ошибки нет. Переход к следующей части теста
U 1311,  093A,BC ;14226          JSR [T24.NO.ERRSUM]                    ; сообщение об отсутствии ERROR SUM (ошибка 6)
U 1312,  0930,54 ;14227          JMP [LOOP.T24.6]                       ; цикл при ошибке, если есть разрешение
                 ;14228  TEST.T24.7:
U 1313,  893B,1C ;14229          JSR [CHECK.CLRERR]                     ; проверка, что запись в CSR очищает биты ошибок. Номер
                 ;14230                                                 ; ошибки 7
                 ;14231  TEST.T24.8:
U 1314,  C0C1,95 ;14232          ADD LS[#3(H)] TO WR[3]                 ; номер ошибки 8 запоминается в WR3
U 1315,  DF2A,15 ;14233          MCOM LS[#FF000000] TO WR[0]            ; несуществующее устройство общей шины (адрес FFFFFF)
U 1316,  BE12,15 ;14234          MOV WR[0] TO LS[T9]                    ; запоминание в LS
                 ;14235  LOOP.T24.9:
U 1317,  E580,15 ;14236          CLR LS[CONTROL.STATUS]                 ; нормальная распечатка ошибки
U 1318,  BE83,95 ;14237          MOV WR[3] TO LS[ERROR.NUMBER]          ; ошибка 8
                 ;14238  LOOP.T24.8:
U 1319,  3668,95 ;14239          MOV LS[OP.ERR] TO WR[1]                ; ожидаемые данные (бит ILL UB OPER ERR установлен)
U 131A,  9813,B5 ;14240          MEM.REQ[READ.P] ADRS[T9] DT[WORD]      ; запрос для чтения по физическому адресу FFFFFF(H)
U 1318,  3022,15 ;14241          MOV MEM.DATA TO WR[0]                  ; выдача инструкции MOV для завершения цикла памяти
U 131C,  9D45,75 ;14242          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения CSR1
U 131D,  3022,15 ;14243          MOV MEM.DATA TO WR[0]                  ; выборка содержимого CSR1
U 131E,  0869,3C ;14244          JSR [CHECK.RESULT]                     ; проверка, что бит ILL UB OPER ERR установлен
U 131F,  8931,94 ;14245          JMP [LOOP.T24.8]                       ; цикл при ошибке, если есть разрешение
U 1320,  FF82,15 ;14246          INC LS[ERROR.NUMBER]                   ; ошибка 9
U 1321,  5B00,1D ;14247          SKIP.IF[MEM.REF.OK]                    ; проверка суммарной ошибки (пропуск не должен
                 ;14248                                                 ; произойти)
U 1322,  8932,54 ;14249          JMP [TEST.T24.A]                       ; ошибки нет. Переход к следующей части теста
U 1323,  093A,BC ;14250          JSR [T24.NO.ERRSUM]                    ; сообщение (ошибка 9) об отсутствии ERROR SUM
U 1324,  0931,74 ;14251          JMP [LOOP.T24.9]                       ; цикл при ошибке, если есть разрешение
                 ;14252  TEST.T24.A:
U 1325,  893B,1C ;14253          JSR [CHECK.CLRERR]                     ; проверка, что запись в CSR очищает биты ошибок. Номер
                 ;14254                                                 ; ошибки A
                 ;14255  TEST.T24.B:
U 1326,  C0C1,95 ;14256          ADD LS[#3(H)] TO WR[3]                 ; в WR3 запоминается номер ошибки 8
U 1327,  0A17,DC ;14257          JSR [WRITE.CSR1.MME]                   ; установка бита разрешения диспетчера памяти (MME)
U 1328,  3672,15 ;14258          MOV LS[BIT25] TO WR[0]                 ; бит BYTE OFFSET
U 1329,  4774,15 ;14259          BIS LS[BIT26] TO WR[0]                 ; бит MODIFY
U 132A,  4778,15 ;14260          BIS LS[BIT28] TO WR[0]                 ; бит PROT B
U 132B,  477E,15 ;14261          BIS LS[BIT31] TO WR[0]                 ; бит VALID
U 132C,  BE14,15 ;14262          MOV WR[0] TO LS[T10]                   ; подготовка ячейки LS для записи в буфер трансляции
U 132D,  989C,F5 ;14263          MEM.REQ[WRITE.TB] ADRS[00] DT[LONG]    ; запрос для записи в TB
U 132E,  B214,15 ;14264          WRITE.MEM LS[T10]                      ; установка в TB по адресу 0 битов BYTE OFFSET, MODIFY,
                 ;14265                                                 ; PROT B и VALID
U 132F,  B676,15 ;14266          MOV LS[MME] TO WR[0]                   ; установка в рабочем регистре бита 27
U 1330,  C77A,15 ;14267          BIS LS[TB.PAR.DIAG] TO WR[0]           ; а также установка бита 29
U 1331,  8A17,FC ;14268          JSR [WRITE.CSR1]                       ; установка разрешения диспетчера памяти и ошибки
                 ;14269                                                 ; паритета TB
                 ;14270  LOOP.T24.C:
```

```
U 1332, E580,15 ;14271          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 1333, BE83,95 ;14272          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка B
                ;14273  LOOP.T24.B:
U 1334, 365E,95 ;14274          MOV LS[TB.PAR.ERR] TO WR[1]             ; ожидаемые данные (установленный бит TB PAR ERR)
U 1335, 989D,75 ;14275          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]  ; запрос для пробного чтения TB по адресу 0
U 1336, 3022,15 ;14276          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 1337, 9D45,75 ;14277          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1338, 3022,15 ;14278          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 1339, 0869,3C ;14279          JSR [CHECK.RESULT]                      ; проверка, что бит TB PAR ERR установлен
U 133A, 8933,44 ;14280          JMP [LOOP.T24.B]                        ; цикл при ошибке, если есть разрешение
U 133B, FF82,15 ;14281          INC LS[ERROR.NUMBER]                    ; ошибка C
U 133C, 5B00,1D ;14282          SKIP.IF[MEM.REF.OK]                     ; проверка суммарной ошибки (пропуск не должен
                ;14283                                                  ; произойти)
U 133D, 8934,04 ;14284          JMP [TEST.T24.D]                        ; ошибки нет. Переход к следующей части теста
U 133E, 093A,BC ;14285          JSR [T24.NO.ERRSUM]                     ; сообщение в качестве ошибки B об отсутствии суммарной
                ;14286                                                  ; ошибки
U 133F, 8933,24 ;14287          JMP [LOOP.T24.C]                        ; цикл при ошибке, если есть разрешение
                ;14288  TEST.T24.D:
U 1340, 893B,1C ;14289          JSR [CHECK.CLRERR]                      ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14290                                                  ; ошибки D
                ;14291  TEST.T24.E:
U 1341, C0C1,95 ;14292          ADD LS[#3(H)] TO WR[3]                  ; номер ошибки E запоминается в WR3
U 1342, B626,15 ;14293          MOV LS[#FF] TO WR[0]                    ; в рабочем регистре адрес FF
U 1343, 4750,15 ;14294          BIS LS[#100] TO WR[0]                   ; адрес 1FF (граница страницы)
U 1344, BE12,15 ;14295          MOV WR[0] TO LS[T9]                     ; запоминание в LS
U 1345, 0A17,DC ;14296          JSR [WRITE.CSR1.MME]                    ; разрешение диспетчера памяти (очистка бита TB PAR
                ;14297                                                  ; DIAG)
                ;14298  LOOP.T24.F:
U 1346, E580,15 ;14299          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 1347, BE83,95 ;14300          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка E
                ;14301  LOOP.T24.E:
U 1348, B666,95 ;14302          MOV LS[WR.ACROSS.PG] TO WR[1]           ; ожидаемые данные (установленный бит WR ACROSS PAGE
                ;14303                                                  ; ERR)
U 1349, 1912,F5 ;14304          MEM.REQ[TEST.V.WCHK] ADRS[T9] DT[LONG]  ; запрос для пробного чтения на границе страницы
U 134A, 3022,15 ;14305          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 134B, 9D45,75 ;14306          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 134C, 3022,15 ;14307          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 134D, 0869,3C ;14308          JSR [CHECK.RESULT]                      ; проверка, что бит WR ACROSS PAGE ERR установлен
U 134E, 0934,84 ;14309          JMP [LOOP.T24.E]                        ; цикл при ошибке, если есть разрешение
U 134F, FF82,15 ;14310          INC LS[ERROR.NUMBER]                    ; ошибка F
U 1350, 5B00,1D ;14311          SKIP.IF[MEM.REF.OK]                     ; проверка суммарной ошибки (пропуск не должен
                ;14312                                                  ; произойти)
U 1351, 8935,44 ;14313          JMP [TEST.T24.10]                       ; ошибки нет. Переход к следующей части теста
U 1352, 093A,BC ;14314          JSR [T24.NO.ERRSUM]                     ; сообщение (ошибки нет) об отсутствии ERROR SUM
U 1353, 8934,64 ;14315          JMP [LOOP.T24.F]                        ; цикл при ошибке, если есть разрешение
                ;14316  TEST.T24.10:
U 1354, 093B,CC ;14317          JSR [CHECK.CLRERR.NODIAG]               ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14318                                                  ; ошибки 10
                ;14319  TEST.T24.11:
U 1355, C0C1,95 ;14320          ADD LS[#3(H)] TO WR[3]                  ; в WR3 запоминается номер ошибки 11
U 1356, 3672,15 ;14321          MOV LS[BIT25] TO WR[0]                  ; бит BYTE OFFSET установлен
U 1357, 4778,15 ;14322          BIS LS[BIT28] TO WR[0]                  ; бит PROT B установлен
U 1358, 477E,15 ;14323          BIS LS[BIT31] TO WR[0]                  ; бит VALID установлен
U 1359, BE14,15 ;14324          MOV WR[0] TO LS[T10]                    ; запоминание данных в LS
U 135A, 989C,F5 ;14325          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в TB по адресу 0
```

```
U 1358, B214,15 ;14326          WRITE.MEM LS[T10]                       ; установка в ТВ по адресу 0 битов BYTE OFFSET, PROT B,
                ;14327                                                  ; VALID
                ;14328  LOOP.T24.12:
U 135C, E580,15 ;14329          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 135D, BE83,95 ;14330          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 11
                ;14331  LOOP.T24.11:
U 135E, 366E,95 ;14332          MOV LS[MODIFY.REFUSED] TO WR[1]         ; ожидаемые данные (установлен бит MODIFY REFUSED)
U 135F, 199C,F5 ;14333          MEM.REQ[TEST.V.WCHK] ADRS[#0] DT[LONG]  ; запрос для пробного чтения ТВ по адресу 0
U 1360, 3022,15 ;14334          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 1361, 9D45,75 ;14335          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1362, 3022,15 ;14336          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 1363, 0869,3C ;14337          JSR [CHECK.RESULT]                      ; проверка, что бит MODIFY REFUSED установлен
U 1364, 8935,E4 ;14338          JMP [LOOP.T24.11]                       ; цикл при ошибке, если есть разрешение
U 1365, FF82,15 ;14339          INC LS[ERROR.NUMBER]                    ; ошибка 12
U 1366, 5B00,1D ;14340          SKIP.IF[MEM.REF.OK]                     ; проверка, что была суммарная  ошибка (пропуск не
                ;14341                                                  ; должен произойти)
U 1367, 0936,A4 ;14342          JMP [TEST.T24.13]                       ; ошибки нет. Переход к следующей части теста
U 1368, 093A,BC ;14343          JSR [T24.NO.ERRSUM]                     ; сообщение (ошибка 12) об отсутствии ERROR SUM
U 1369, 0935,C4 ;14344          JMP [LOOP.T24.12]                       ; цикл при ошибке, если есть разрешение
                ;14345  TEST.T24.13:
U 136A, 093B,CC ;14346          JSR [CHECK.CLRERR.NODIAG]               ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14347                                                  ; ошибки 13
                ;14348  TEST.T24.14:
U 136B, C0C1,95 ;14349          ADD LS[#3(H)] TO WR[3]                  ; в WR3 запоминается номер ошибки 14
U 136C, 3672,15 ;14350          MOV LS[BIT25] TO WR[0]                  ; бит BYTE OFFSET установлен
U 136D, 477E,15 ;14351          BIS LS[BIT31] TO WR[0]                  ; бит VALID установлен
U 136E, BE14,15 ;14352          MOV WR[0] TO LS[T10]                    ; занесение данных в LS
U 136F, 989C,F5 ;14353          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в ТВ по адресу 0
U 1370, B214,15 ;14354          WRITE.MEM LS[T10]                       ; установка в ТВ битов BYTE OFPSET и VALID
                ;14355  LOOP.T24.15:
U 1371, E580,15 ;14356          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 1372, BE83,95 ;14357          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 14
                ;14358  LOOP.T24.14:
U 1373, B66C,95 ;14359          MOV LS[ACCESS.REFUSED] TO WR[1]         ; ожидаемые данные (бит ACCESS REFUSED установлен)
U 1374, 199C,F5 ;14360          MEM.REQ[TEST.V.WCHK] ADRS[#0] DT[LONG]  ; пробное чтение ТВ по адресу 0
U 1375, 3022,15 ;14361          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 1376, 9D45,75 ;14362          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1377, 3022,15 ;14363          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 1378, 0869,3C ;14364          JSR [CHECK.RESULT]                      ; проверка, что бит ACCESS REFUSED установлен
U 1379, 8937,34 ;14365          JMP [LOOP.T24.14]                       ; цикл при ошибке, если есть разрешение
U 137A, FF82,15 ;14366          INC LS[ERROR.NUMBER]                    ; ошибка 15
U 137B, 5B00,1D ;14367          SKIP.IF[MEM.REF.OK]                     ; проверка суммарной ошибки (пропуск не должен
                ;14368                                                  ; произойти)
U 137C, 8937,F4 ;14369          JMP [TEST.T24.16]                       ; ошибки нет. Переход к следующей части теста
U 137D, 093A,BC ;14370          JSR [T24.NO.ERRSUM]                     ; сообщение (ошибка 15) об отсутствии ERROR SUM
U 137E, 0937,14 ;14371          JMP [LOOP.T24.15]                       ; цикл при ошибке, если есть разрешение
                ;14372  TEST.T24.16:
U 137F, 093B,CC ;14373          JSR [CHECK.CLRERR.NODIAG]               ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14374                                                  ; ошибки 16
                ;14375  TEST.T24.17:
U 1380, C0C1,95 ;14376          ADD LS[#3(H)] TO WR[3]                  ; в WR3 запоминается номер ошибки 17
U 1381, 3672,15 ;14377          MOV LS[BIT25] TO WR[0]                  ; бит BYTE OFFSET установлен
U 1382, 4778,15 ;14378          BIS LS[BIT28] TO WR[0]                  ; бит PROT B установлен
U 1383, BE14,15 ;14379          MOV WR[0] TO LS[T10]                    ; запоминание данных в LS
U 1384, 989C,F5 ;14380          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в ТВ по адресу 0
```

```
U 1385, B214,15 ;14381          WRITE.MEM LS[T10]                       ; установка в TB битов BYTE OFFSET и PROT B
                ;14382  LOOP.T24.1B:
U 1386, E580,15 ;14383          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибки
U 1387, BE83,95 ;14384          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 17
                ;14385  LOOP.T24.17:
U 1388, B65C,95 ;14386          MOV LS[VALID.ERR] TO WR[1]              ; ожидаемые данные (бит VALID ERR установлен)
U 1389, 989D,75 ;14387          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]  ; пробное чтение TB по адресу 0
U 138A, 3022,15 ;14388          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 138B, 9D45,75 ;14389          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 138C, 3022,15 ;14390          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 138D, 0869,3C ;14391          JSR [CHECK.RESULT]                      ; проверка, что бит VALID ERR установлен
U 138E, 0938,B4 ;14392          JMP [LOOP.T24.17]                       ; цикл при ошибке, если есть разрешение
U 138F, FF82,15 ;14393          INC LS[ERROR.NUMBER]                    ; ошибка 18
U 1390, 5B00,1D ;14394          SKIP.IF[MEM.REF.OK]                     ; проверка суммарной ошибки (пропуск не должен
                ;14395                                                  ; произойти)
U 1391, 8939,44 ;14396          JMP [TEST.T24.19]                       ; ошибки нет. Переход к следующей части теста
U 1392, 093A,BC ;14397          JSR [T24.NO.ERRSUM]                     ; сообщение (ошибка 18) об отсутствии ERR SUM
U 1393, 8938,64 ;14398          JMP [LOOP.T24.18]                       ; цикл при ошибке, если есть разрешение
                ;14399  TEST.T24.19:
U 1394, 093B,CC ;14400          JSR [CHECK.CLRERR.NODIAG]               ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14401                                                  ; ошибки 19
                ;14402  TEST.T24.1A:
U 1395, C0C1,95 ;14403          ADD LS[#3(H)] TO WR[3]                  ; в WR3 запоминается номер ошибки 1A
U 1396, 367B,15 ;14404          MOV LS[BIT28] TO WR[0]                  ; бит PROT B установлен
U 1397, 477E,15 ;14405          BIS LS[BIT31] TO WR[0]                  ; бит VALID установлен
U 1398, BE14,15 ;14406          MOV WR[0] TO LS[T10]                    ; запоминание данных в LS
U 1399, 989C,F5 ;14407          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи в TB по адресу 0
U 139A, B214,15 ;14408          WRITE.MEM LS[T10]                       ; установка в TB битов VALID и PROT B
                ;14409  LOOP.T24.1B:
U 139B, E5B0,15 ;14410          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибок
U 139C, BE83,95 ;14411          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 1A
                ;14412  LOOP.T24.1A:
U 139D, B66A,95 ;14413          MOV LS[TB.MISS] TO WR[1]                ; ожидаемые данные (бит TB MISS установлен)
U 139E, 989D,75 ;14414          MEM.REQ[TEST.V.RCHK] ADRS[#0] DT[LONG]  ; пробное чтение TB по адресу 0
U 139F, 3022,15 ;14415          MOV MEM.DATA TO WR[0]                   ; выдача инструкции MOV для завершения микроцикла
U 13A0, 9D45,75 ;14416          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 13A1, 3022,15 ;14417          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 13A2, 0869,3C ;14418          JSR [CHECK.RESULT]                      ; проверка, что бит TB MISS установлен
U 13A3, 8939,D4 ;14419          JMP [LOOP.T24.1A]                       ; цикл при ошибке, если разрешено
U 13A4, FF82,15 ;14420          INC LS[ERROR.NUMBER]                    ; ошибка 1B
U 13A5, 5B00,1D ;14421          SKIP.IF[MEM.REF.OK]                     ; проверка суммарной ошибки (пропуск не должен
                ;14422                                                  ; произойти)
U 13A6, 093A,94 ;14423          JMP [TEST.T24.1C]                       ; переход к следующему тесту
U 13A7, 093A,BC ;14424          JSR [T24.NO.ERRSUM]                     ; сообщение (ошибка 1B) об отсутствии ERR SUM
U 13A8, 8939,B4 ;14425          JMP [LOOP.T24.1B]                       ; цикл при ошибке, если есть разрешение
                ;14426  TEST.T24.1C:
U 13A9, 093B,CC ;14427          JSR [CHECK.CLRERR.NODIAG]               ; проверка, что запись в CSR очищает биты ошибок. Номер
                ;14428                                                  ; ошибки 1C
U 13AA, 093C,64 ;14429          JMP [T24.END]                           ; конец теста
                ;14430  T24.NO.ERRSUM:
U 13AB, B66E,15 ;14431          MOV LS[NA.EXP.REC] TO WR[0]             ; установка в рабочем регистре бита 23
U 13AC, 3E80,15 ;14432          MOV WR[0] TO LS[CONTROL.STATUS]         ; печать в сообщении об ошибке N/A (не применяются) под
                ;14433                                                  ; ожидаемыми и полученными данными
U 13AD, 2F82,95 ;14434          CLR WR[1]                               ; этим принудительно устанавливается ошибка
U 13AE, 0869,3C ;14435          JSR [CHECK.RESULT]                      ; сообщение об ошибке
```

```
U 13AF, 5800,14 ;14436          RETURN                                  ; возврат при зацикливании на ошибке
U 13B0, D800,16 ;14437          RETURN+1                                ; возврат для нормального продолжения после ошибки
                ;14438  CHECK.CLRERR:
U 13B1, E580,15 ;14439          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибок
U 13B2, FF82,15 ;14440          INC LS[ERROR.NUMBER]                    ; подготовка номера ошибки
                ;14441  LOOP.CLRERR:
U 13B3, B676,15 ;14442          MOV LS[MME] TO WR[0]                    ; установка в рабочем регистре бита 27
U 13B4, C77A,15 ;14443          BIS LS[TB.PAR.DIAG] TO WR[0]            ; а также установка бита 29
U 13B5, 8A17,FC ;14444          JSR [WRITE.CSR1]                        ; установлено разрешение диспетчера памяти и ошибка
                ;14445                                                  ; паритета ТВ. Запись в CSR1 должна очистить биты
                ;14446                                                  ; ошибок
U 13B6, 2F82,95 ;14447          CLR WR[1]                               ; ожидаемые данные
U 13B7, 9D45,75 ;14448          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 13B8, 3022,15 ;14449          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 13B9, 0869,3C ;14450          JSR [CHECK.RESULT]                      ; проверка, что все биты ошибок очищены
U 13BA, 893B,34 ;14451          JMP [LOOP.CLRERR]                       ; цикл при ошибке, если есть разрешение
U 13BB, 5800,14 ;14452          RETURN                                  ; возврат в основную программу
                ;14453  CHECK.CLRERR.NODIAG:
U 13BC, E580,15 ;14454          CLR LS[CONTROL.STATUS]                  ; нормальная распечатка ошибок
U 13BD, FF82,15 ;14455          INC LS[ERROR.NUMBER]                    ; подготовка номера ошибки
                ;14456  LOOP.CLRERR2:
U 13BE, B676,15 ;14457          MOV LS[MME] TO WR[0]                    ; установка в рабочем регистре бита 27
U 13BF, 8A17,FC ;14458          JSR [WRITE.CSR1]                        ; установка разрешения диспетчера памяти. Запись в CSR1
                ;14459                                                  ; должна очистить биты ошибок
U 13C0, 2F82,95 ;14460          CLR WR[1]                               ; ожидаемые данные
U 13C1, 9D45,75 ;14461              REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 13C2, 3022,15 ;14462          MOV MEM.DATA TO WR[0]                   ; прием содержимого CSR1
U 13C3, 0869,3C ;14463          JSR [CHECK.RESULT]                      ; проверка, что все биты ошибок очищены
U 13C4, 093B,E4 ;14464          JMP [LOOP.CLRERR2]                      ; цикл при ошибке, если есть разрешение
U 13C5, 5800,14 ;14465          RETURN                                  ; возврат в основную программу
                ;14466  T24.END:
```

```
;14467  .PAGE "ТЕСТ 25 - проверка записи в память байта и слова (модуль МСТ)"
;14468  ;
;14469  ; ОПИСАНИЕ ТЕСТА:
;14470  ;
;14471  ;          Этот тест проверяет логические схемы, используемые для записи в ос-
;14472  ;       новную память байта или слова данных. Ответственность за размещение но-
;14473  ;       вого байта или слова в правильную позицию в памяти без изменения окру-
;14474  ;       жающих данных несет ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ. Этот тест выполня-
;14475  ;       ет обширную проверку ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ.
;14476  ;          Этот тест записывает байты, содержащие 0, в позиции каждого байта
;14477  ;       длинного слова, содержащего единицы, и проверяет, что очищен только один
;14478  ;       байт. Затем он записывает слова, содержащие 0, в позиции каждого байта
;14479  ;       внутри длинного слова, содержащего единицы, и проверяет, что очищены
;14480  ;       только 2 байта. При записи в позицию 3-го байта также будет проверяться
;14481  ;       следующее длинное слово, чтобы удостовериться, что очищается только
;14482  ;       первый байт. Затем с позиции 0 записывается длинное слово, содержащие
;14483  ;       нули (выравненное длинное слово) с программной задержкой между инструк-
;14484  ;       циями мем.REQ и WRITE.MEM. Таким способом вызывается путь в микрокодах,
;14485  ;       отличающийся от того, какой использовался в тестах модуля памяти, и про-
;14486  ;       веряются другие цепи в ПМЛ УПР.РЕГИСТРАМ ДАННЫХ ПАМЯТИ, используемые
;14487  ;       только для записи выравненных слов, которые не получают CPU DATA REQUEST
;14488  ;       (запрос данных из центрального процессора) после MEM.REQ в промежутке
;14489  ;       около 3 циклов центрального процессора (в зависимости от того, требуется
;14490  ;       ли и когда требуется регенерация). Затем выполняется запись невыравненных
;14491  ;       длинных слов с позиций 1,2 и 3 и проверяются 2 длинных слова.
;14492  ;          Наконец, записывается байт, содержащий нули, в позицию байта 0 длинно-
;14493  ;       го слова, содержащего единицы (как указано выше) с программной задержкой
;14494  ;       между инструкциями MEM.REQ и WRITE.MEM. Таким способом проверяется еще
;14495  ;       один путь в микрокодах.
;14496  ;
;14497  ; ПРЕДПОЛОЖЕНИЯ:
;14498  ;
;14499  ;       Принимается, что все предыдущие тесты прошли успешно.
;14500  ;
;14501  ; ШАГИ ТЕСТА:
;14502  ;
;14503  ;       1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для рас-
;14504  ;          печатки ошибок) и очистка в LS номера предыдущей ошибки.
;14505  ;       2) Загрузка в WR2 значения объема памяти (ячейка LS MM.LASTADDR+1) и
;14506  ;          проверка, что объем не 0.
;14507  ;       3) Запись всех единиц в ячейку памяти 0.
;14508  ;       4) Запись байта, состоящего из нулей, в ячейку 0. Проверка результата
;14509  ;          на очищенный байт 0.
;14510  ;       5) Повторение шагов 3 и 4 в позициях байтов 1,2 и 3.
;14511  ;       6) Повторение шагов от 3 до 5 для слов данных. При записи слова с пози-
;14512  ;          ции байта 3, предварительная запись единицами ячеек 0 и 4 и проверка
;14513  ;          обоих длинных слов после записи слова.
;14514  ;       7) Повторение шага 3, затем выдача MEM.REQ при MF=WRITE.P и DT=длинное
;14515  ;          слово, с использованием в качестве приемника адреса 0.
;14516  ;       8) Задержка на 6 циклов центрального процессора и выдача инструкции
;14517  ;          WRITE MEM.DATA LS с данными, содержащими все 0. Проверка ячейки 0 на
;14518  ;          все нули.
;14519  ;       9) Запись всех единиц в ячейку памяти 0 и 4 (два последовательных длин-
;14520  ;          ных слова).
;14521  ;      10) Запись длинного слова, состоящего из нулей, с позиции байта 1. Счи-
```

```
;14522  ;          тывание длинных слов из ячеек 0 и 4, и проверка правильности данных.
;14523  ;      11) Повторение шагов 9 и 10 с позиций байтов 2 и 3.
;14524  ;      12) Запись всех единиц в ячейку памяти 0.
;14525  ;      13) Запись байта, состоящего из нулей, в ячейку 0 с программной задержкой
;14526  ;          (инструкции NOP) между инструкциями MEM.REQ и WRITE.MEM. Проверка
;14527  ;          результата на очищенный байт 0.
;14528  ;
;14529  ; ОШИБКИ:
;14530  ;
;14531  ;       ПРИМЕЧАНИЕ: данные под OTHER представляют собой проверяемый адрес матрицы
;14532  ;       памяти.
;14533  ;
;14534  ;       ошибка 1 - при записи байта, состоящего из нулей, не выполнена запись
;14535  ;                  одного и только одного байта.
;14536  ;       ошибка 2 - при записи слова, состоящего из 0, не выполнена запись одного
;14537  ;                  и только одного слова в пределах длинного слова с ячейки 0
;14538  ;                  (или только байта 3 при двухцикловой записи).
;14539  ;       ошибка 3 - при двухцикловой записи слова не записан второй байт записы-
;14540  ;                  ваемых данных в позиции байта 0 ячейки 4.
;14541  ;       ошибка 4 - отказ при записи выравненного длинного слова при задержке
;14542  ;                  между MEM.REQ и WRITE.MEM LS.
;14543  ;       ошибка 5 - отказ при записи в ячейку 0 невыравненного длинного слова.
;14544  ;       ошибка 6 - отказ при записи в ячейку 4 невыравненного длинного слова.
;14545  ;       ошибка 7 - при записи байта с задержкой между MEM.REQ и WRITE.MEM LS
;14546  ;                  не произошла запись одного и только одного байта.
;14547  ;
;14548  ; НАЛАДКА:
;14549  ;
;14550  ;       ОШИБКА 1 - Эта ошибка указывает на неисправность в ПМЛ УПР.РЕГИСТРОМ
;14551  ;                  ДАННЫХ ПАМЯТИ или в ее входах. Ожидаемые данные в сообщении
;14552  ;                  об ошибке указывают, который байт тест пытался записать
;14553  ;                  (байт, состоящий из нулей). В пошаговом режиме следует оста-
;14554  ;                  новить центр. процессор на инструкции MEM.REQ С MF=WRITE.P и
;14555  ;                  DT=BYTE. Проверьте выходы ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ
;14556  ;                  низкий уровень выхода, соответствующего байту, для которого
;14557  ;                  тест делал попытку записи. Проверьте оставшиеся 3 выхода на
;14558  ;                  высокий уровень. Если они правильные, может быть неправильный
;14559  ;                  микрокод или имеется короткое замыкание между сигналами LAT
;14560  ;                  OUTPUT BYT L и одним из сигналов разрешения около вентилей
;14561  ;                  "И", которые осуществляют выдачу LAT OUTPUT BYTX L.
;14562  ;                     Если выходы из ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ неправильные,
;14563  ;                  следует проверить входы на высокий уровень на ROT C0 H и CPH/
;14564  ;                  UBL H. Проверьте низкий уровень на 2ND MEM CYC H. Проверьте
;14565  ;                  высокий уровень для A1 L, если происходит запись байтов 0
;14566  ;                  или 1, проверьте на низкий уровень для A1 L, если записывают-
;14567  ;                  ся байты 2 или 3. Если эти входы правильные, тогда вероятно,
;14568  ;                  что ПМЛ неисправна.
;14569  ;                     Другой возможностью является то, что неправильно работает
;14570  ;                  переход по ALIGN LW L. Это вызывает очистку всех байтов вмес-
;14571  ;                  то одного байта. Это можно проверить пошаговым продвижением к
;14572  ;                  той же инструкции MEM.REQ, как и раньше и наблюдением этого
;14573  ;                  входа к мультиплексору ветвления 2 (BEN 2). Он должен быть
;14574  ;                  высоким.
;14575  ;
;14576  ;       ОШИБКА 2 - Если это первая ошибка, то подозревается ПМЛ УПР.РЕГИСТРОМ
```

```
                ;14577  ;                   ДАННЫХ ПАМЯТИ.
                ;14578  ;
                ;14579  ;       ОШИБКА 3 - Если это первая ошибка, то с наибольшей вероятностью можно
                ;14580  ;                  подозревать ПМЛ УПР.РЕГИСТРОМ ДАННЫХ ПАМЯТИ. Одним из входов
                ;14581  ;                  при предыдущих ошибках, является вход 2ND MEM CYC H, который
                ;14582  ;                  при правильной работе должен стать высоким для второго цикла
                ;14583  ;                  памяти. Проверьте при зацикливании на ошибке, что 2ND MEM CYC
                ;14584  ;                  H становиться высоким. Если он правильный, подозревается эта
                ;14585  ;                  ПМЛ.
                ;14586  ;
                ;14587  ;       ОШИБКА 4 ПО 6 - Если это первая ошибка, следует подозревать ПМЛ УПР.РЕ-
                ;14588  ;                  ГИСТРОМ ДАННЫХ ПАМЯТИ.
                ;14589  ;
                ;14590  ;       ОШИБКА 7 - Под подозрением находится ПЗУ микрокодов, так как этот тест
                ;14591  ;                  проходит микрокоды по другому пути.
                ;14592  ;
                ;14593  ;
                ;14594  T.25:
U 13C6, B65E,15 ;14595          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR 0 бита 15 для слова управления и
                ;14596                                                  ; состояния
U 13C7, 3E80,15 ;14597          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;14598                                                  ; 15 указывает для консольного процессора начало теста
U 13C8, 10E0,15 ;14599          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;14600  WAIT.T25.0:
U 13C9, 093C,94 ;14601          JMP [WAIT.T25.0]                        ; цикл для ожидания ответа консольного процессора
U 13CA, 0A1A,9C ;14602          JSR [SETUP]                             ; подготовка маски, кода модуля и т.д. и очистка CSR 1
                ;14603                                                  ; MCT
U 13CB, B670,15 ;14604          MOV LS[OTHER.DATA] TO WR[0]             ; установка бита печати поля "ДРУГИЕ ДАННЫЕ" (OTHER)
U 13CC, 3E80,15 ;14605          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка управляющего слова для печати адреса под
                ;14606                                                  ; OTHER (другие данные) в случае ошибки
U 13CD, 6512,15 ;14607          CLR LS[T9]                              ; адрес для записи в основную память
U 13CE, 6588,15 ;14608          CLR LS[ADDRESS.DATA]                    ; и очистка адреса для печати
U 13CF, B62C,15 ;14609          MOV LS[#FFFFFF00] TO WR[0]              ; ожидаемые данные (байт 0 очищен)
                ;14610  REPEAT.T25.1:
U 13D0, BE14,15 ;14611          MOV WR[0] TO LS[T10]                    ; запоминание ожидаемых данных в LS
U 13D1, 999C,75 ;14612          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; подготовка записи в ячейку 0 основной памяти
U 13D2, 329E,15 ;14613          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово с адреса 0
                ;14614  LOOP.T25.1:
U 13D3, B614,95 ;14615          MOV LS[T10] TO WR[1]                    ; ожидаемые данные (один байт очищен)
U 13D4, 9912,15 ;14616          MEM.REQ[WRITE.P] ADRS[T9] DT[BYTE]      ; запрос для записи байта по адресу 0
U 13D5, B29C,15 ;14617          WRITE.MEM LS[ZERO]                      ; запись нуля в байте, заданном в LS9
U 13D6, 189D,F5 ;14618          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос чтения длинного слова с адреса 0
U 13D7, 3022,15 ;14619          MOV MEM.DATA TO WR[0]                   ; прием длинного слова
U 13D8, 0869,3C ;14620          JSR [CHECK.RESULT]                      ; проверка, что один байт = 0, остальные единицы
U 13D9, B93D,34 ;14621          JMP [LOOP.T25.1]                        ; цикл при ошибке, если есть разрешение
U 13DA, 0944,1C ;14622          JSR [SHIFT.T25.8BITS]                   ; подготовка позиции следующего байта
U 13DB, B644,95 ;14623          MOV LS[#4] TO WR[1]                     ; занесение значения 4 в WR1
                ;14624            CMP WR[1] WITH LS[T9],                ; проверка, испытан ли байт 3
U 13DC, CF12,B5 ;14625          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 13DD, B93D,01 ;14626          JMP.IF[NEQ] TO [REPEAT.T25.1]           ; повторение, пока не будут испытаны байты 0-3
U 13DE, FF82,15 ;14627          INC LS[ERROR.NUMBER]                    ; ошибка 2
U 13DF, 3683,95 ;14628          MOV LS[ERROR.NUMBER] TO WR[3]           ; запоминание номера ошибки
U 13E0, 6512,15 ;14629          CLR LS[T9]                              ; подготовка адреса для записи в основную память
U 13E1, 5F28,15 ;14630          MCOM LS[#FFFF] TO WR[0]                 ; ожидаемые данные (байт 0 и 1 очищены)
                ;14631  REPEAT.T25.2:
```

```
U 13E2,  BE14,15 ;14632          MOV WR[0] TO LS[T10]                    ; запоминание в LS ожидаемых данных
U 13E3,  999C,75 ;14633          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; подготовка записи в ячейку 0 основной памяти
U 13E4,  329E,15 ;14634          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово по адресу 0
                 ;14635    LOOP.T25.2:
U 13E5,  B614,95 ;14636          MOV LS[T10] TO WR[1]                    ; ожидаемые данные (одно слово очищено)
U 13E6,  1912,35 ;14637          MEM.REQ[WRITE.P] ADRS[T9] DT[WORD]      ; запрос для записи слова с адреса 0
U 13E7,  B29C,15 ;14638          WRITE.MEM LS[ZERO]                      ; запись нулей в слово, заданное в LS9
U 13E8,  189D,F5 ;14639          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос чтения длинного слова с адреса 0
U 13E9,  3022,15 ;14640          MOV MEM.DATA TO WR[0]                   ; прием длинного слова
U 13EA,  0869,3C ;14641          JSR [CHECK.RESULT]                      ; проверка, что одно слово=0, другое - единицы
U 13EB,  893E,54 ;14642          JMP [LOOP.T25.2]                        ; цикл при ошибке, если есть разрешение
U 13EC,  0944,1C ;14643          JSR [SHIFT.T25.8BITS]                   ; подготовка позиции следующего байта
U 13ED,  B6C0,95 ;14644          MOV LS[#3(H)] TO WR[1]                  ; занесение значения 3 в WR1
                 ;14645            CMP WR[1] WITH LS[T9],                ; проверка, выполнен ли тест для слова, начинающегося с
                 ;14646                                                  ; позиции байта 2, и установка кодов условий
U 13EE,  CF12,85 ;14647          DT(LONG)&SET.ALU.CC                     ; выполнена проверка одноцикловой записи
U 13EF,  093E,21 ;14648          JMP.IF[NEQ] TO [REPEAT.T25.2]           ; проверка двухцикловой записи
U 13F0,  999C,75 ;14649          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи в ячейку 0 основной памяти
U 13F1,  329E,15 ;14650          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово по адресу 0
U 13F2,  9944,75 ;14651          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]      ; запрос для записи в ячейку 4 основной памяти
U 13F3,  329E,15 ;14652          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово с адресом 4
                 ;14653    LOOP.T25.3:
U 13F4,  BE83,95 ;14654          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 2
U 13F5,  6588,15 ;14655          CLR LS[ADDRESS.DATA]                    ; адрес для печати в случае ошибки
                 ;14656    LOOP.T25.2B:
U 13F6,  5F2A,95 ;14657          MCOM LS[#FF000000] TO WR[1]             ; ожидаемые данные (нули в байте 3)
U 13F7,  1912,35 ;14658          MEM.REQ[WRITE.P] ADRS[T9] DT[WORD]      ; запрос для записи слова по адресу 3
U 13F8,  B29C,15 ;14659          WRITE.MEM LS[ZERO]                      ; запись нулей, начиная с байта 3
U 13F9,  189D,F5 ;14660          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения длинного слова с адреса 0
U 13FA,  3022,15 ;14661          MOV MEM.DATA TO WR[0]                   ; выборка длинного слова
U 13FB,  0869,3C ;14662          JSR [CHECK.RESULT]                      ; проверка, что байт 3=0
U 13FC,  093F,64 ;14663          JMP [LOOP.T25.2B]                       ; цикл при ошибке, если есть разрешение
U 13FD,  FF82,15 ;14664          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 13FE,  3644,15 ;14665          MOV LS[#4] TO WR[0]                     ; занесение значения 4 в WR0
U 13FF,  BE88,15 ;14666          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати в сучае ошибки (адрес 4)
U 1400,  362C,95 ;14667          MOV LS[#FFFFFF00] TO WR[1]              ; ожидаемые данные (байт 0 в ячейке 4 очищен)
U 1401,  1845,F5 ;14668          MEM.REQ[READ.P] ADRS[#4] DT[LONG]       ; запрос для чтения длинного слова с адреса 4
U 1402,  3022,15 ;14669          MOV MEM.DATA TO WR[0]                   ; выборка длинного слова
U 1403,  0869,3C ;14670          JSR [CHECK.RESULT]                      ; проверка, что байт 0 = нули в длинном слове с адреса
                 ;14671                                                  ; 4
U 1404,  893F,44 ;14672          JMP [LOOP.T25.3]                        ; цикл при ошибке, если есть  разрешение
U 1405,  FF82,15 ;14673          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 1406,  6588,15 ;14674          CLR LS[ADDRESS.DATA]                    ; адрес для печати в случае ошибки
U 1407,  999C,75 ;14675          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи длинного слова с адреса 0
U 1408,  329E,15 ;14676          WRITE.MEM LS[ONES]                      ; запись в ячейку 0 длинного слова, содержащего единицы
                 ;14677    LOOP.T25.4:
U 1409,  B69C,95 ;14678          MOV LS[ZERO] TO WR[1]                   ; ожидаемые данные (все нули)
U 140A,  999C,75 ;14679          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи нулей в длинное слово с адреса 0
U 140B,  0A1B,4C ;14680          JSR [NOP.DELAY]                         ; выполнение задержки из 5 циклов
U 140C,  DB00,15 ;14681          NOP                                     ; задержка еще на один цикл
U 140D,  B29C,15 ;14682          WRITE.MEM LS[ZERO]                      ; запись данных после задержки из 6 холостых циклов
U 140E,  189D,F5 ;14683          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения результата
U 140F,  3022,15 ;14684          MOV MEM.DATA TO WR[0]                   ; выборка результата из длинного слова с адреса 0
U 1410,  0869,3C ;14685          JSR [CHECK.RESULT]                      ; проверка на все нули
U 1411,  8940,94 ;14686          JMP [LOOP.T25.4]                        ; цикл при ошибке, если есть разрешение
```

```
U 1412, FF82,15 ;14687          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 1413, 3683,95 ;14688          MOV LS[ERROR.NUMBER] TO WR[3]           ; запоминание в WR3
U 1414, 6512,15 ;14689          CLR LS[T9]                              ; адрес для записи в основную память
U 1415, FF12,15 ;14690          INC LS[T9]                              ; первым адресом будет 1
U 1416, DF2C,15 ;14691          MCOM LS[#FFFFFF00] TO WR[0]             ; первые ожидаемые данные (нули в байтах 1, 2 и 3 адреса
                ;14692                                                  ; 0)
                ;14693  REPEAT.T25.5:
U 1417, BE14,15 ;14694          MOV WR[0] TO LS[T10]                    ; ожидаемые данные для ячейки 0
U 1418, F816,15 ;14695          MCOM WR[0] TO LS[T11]                   ; ожидаемые данные для ячейки 4 (инверсия ожидаемых
                ;14696                                                  ; данных для ячейки 0)
U 1419, 999C,75 ;14697          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи в ячейку 0 основной памяти
U 141A, 329E,15 ;14698          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово с адреса 0
U 141B, 9944,75 ;14699          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]      ; запрос для записи в ячейку 4 основной памяти
U 141C, 329E,15 ;14700          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово с адреса 4
                ;14701  LOOP.T25.6:
U 141D, BE83,95 ;14702          MOV WR[3] TO LS[ERROR.NUMBER]           ; ошибка 5
U 141E, 6588,15 ;14703          CLR LS[ADDRESS.DATA]                    ; адрес для печати в случае ошибки
                ;14704  LOOP.T25.5:
U 141F, B614,95 ;14705          MOV LS[T10] TO WR[1]                    ; ожидаемые данные
U 1420, 9912,75 ;14706          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи длинного слова
U 1421, B29C,15 ;14707          WRITE.MEM LS[ZERO]                      ; запись нулей, начиная с позиции байта, заданной в LS
                ;14708                                                  ; 9
U 1422, 189D,F5 ;14709          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения длинного слова с адреса 0
U 1423, 3022,15 ;14710          MOV MEM.DATA TO WR[0]                   ; прием длинного слова
U 1424, 0869,3C ;14711          JSR [CHECK.RESULT]                      ; проверка правильности результата
U 1425, 0941,F4 ;14712          JMP [LOOP.T25.5]                        ; цикл при ошибке, если есть разрешение
U 1426, FF82,15 ;14713          INC LS[ERROR.NUMBER]                    ; ошибка 6
U 1427, 3644,15 ;14714          MOV LS[#4] TO WR[0]                     ; занесение значения 4 в WR0
U 1428, BE88,15 ;14715          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати в случае ошибки
U 1429, 3616,95 ;14716          MOV LS[T11] TO WR[1]                    ; ожидаемые данные для адреса 4
U 142A, 1845,F5 ;14717          MEM.REQ[READ.P] ADRS[#4] DT[LONG]       ; запрос для чтения длинного слова с адреса 4
U 142B, 3022,15 ;14718          MOV MEM.DATA TO WR[0]                   ; прием длинного слова
U 142C, 0869,3C ;14719          JSR [CHECK.RESULT]                      ; проверка правильности результата
U 142D, 8941,D4 ;14720          JMP [LOOP.T25.6]                        ; цикл при ошибке, если есть разрешение
U 142E, B62A,15 ;14721          MOV LS[#FF000000] TO WR[0]              ; установка в WR0 битов 23-31
U 142F, ED14,15 ;14722          BIS WR[0] TO LS[T10]                    ; установка старшего байта ожидаемых данных. Во время
                ;14723                                                  ; сдвига единицы будут перемещаться в более низкий байт
U 1430, 0944,1C ;14724          JSR [SHIFT.T25.8BITS]                   ; подготовка для следующей позиции байта
U 1431, B644,95 ;14725          MOV LS[#4] TO WR[1]                     ; занесение значения 4 в WR1
                ;14726            CMP WR[1] WITH LS[T9],                ; проверка, выполнен ли тест для позиции байта 3 и
                ;14727                                                  ; установка кодов условий
U 1432, CF12,B5 ;14728          DT(LONG)%SET.ALU.CC                     ; повторение, если не выполнено
U 1433, 8941,71 ;14729          JMP.IF[NEQ] TO [REPEAT.T25.5]           ;
U 1434, FF82,15 ;14730          INC LS[ERROR.NUMBER]                    ; ошибка 7
U 1435, 6588,15 ;14731          CLR LS[ADDRESS.DATA]                    ; очистка адреса для печати
U 1436, 999C,75 ;14732          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи в ячейку 0 основной памяти
U 1437, 329E,15 ;14733          WRITE.MEM LS[ONES]                      ; запись единиц в длинное слово с адреса 0
                ;14734  LOOP.T25.7:
U 1438, 362C,95 ;14735          MOV LS[#FFFFFF00] TO WR[1]              ; ожидаемые данные (очищенный байт 0)
U 1439, 999C,15 ;14736          MEM.REQ[WRITE.P] ADRS[#0] DT[BYTE]      ; запрос для записи байта по адресу 0
U 143A, 0A1B,4C ;14737          JSR [NOP.DELAY]                         ; выполнение задержки на 5 циклов
U 143B, B29C,15 ;14738          WRITE.MEM LS[ZERO]                      ; запись 0 в байт 0
U 143C, 189D,F5 ;14739          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения длинного слова с адреса 0
U 143D, 3022,15 ;14740          MOV MEM.DATA TO WR[0]                   ; прием длинного слова
U 143E, 0869,3C ;14741          JSR [CHECK.RESULT]                      ; проверка, что байт 0 содержит нули, остальные байты
```

```
                 ;14742                                                  ; содержат единицы
U 143F, 0943,84  ;14743          JMP [LOOP.T25.7]                        ; цикл при ошибке, если есть разрешение
U 1440, 8944,84  ;14744          JMP [END.T25]                           ; иначе конец теста
                 ;14745    SHIFT.T25.8BITS:
U 1441, 3614,15  ;14746          MOV LS[T10] TO WR[0]                    ; выборка последних ожидаемых данных
U 1442, FF12,15  ;14747          INC LS[T9]                              ; позиция следующего байта
U 1443, 3646,95  ;14748          MOV LS[#8] TO WR[1]                     ; счетчик сдвигов
                 ;14749    SHIFT.T25.LOOP:
U 1444, A3C0,15  ;14750          ROL WR[0]                               ; сдвиг ожидаемых данных
                 ;14751            DEC WR[1],                            ; уменьшение счетчика
U 1445, A102,85  ;14752          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 1446, 8944,41  ;14753          JMP.IF[NEQ] TO [SHIFT.T25.LOOP]         ; повторение, пока не будут завершены все 8 сдвигов
U 1447, 5B00,14  ;14754          RETURN                                  ; конец сдвига
                 ;14755    END.T25:
```

```
;14756  .PAGE "ТЕСТ 26 - проверка разрешения работы диспетчера памяти (модуль МСТ)"
;14757  ;
;14758  ;  ОПИСАНИЕ ТЕСТА:
;14759  ;
;14760  ;     Этот тест провеяет ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. и ее вход L MME H. Сигнал MME
;14761  ;  (разрешение диспетчера памяти) поступает из CSR1 и уже проверялся около CSR.
;14762  ;  Если MME очищен, ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. возбуждает при обращениях к памяти
;14763  ;  сигнал ADDR PH до тех пор, пока не производится запись в буфер трансляции (TB).
;14764  ;  Этот тест поддерживает данный бит установленным, пока в TB записывается вирту-
;14765  ;  альный адрес. Инверсные данные записываются в две ячейки памяти, одна из ко-
;14766  ;  торых задается при помощи TB, а другая нет. Выполняется фукция READ.V.NOCHK,
;14767  ;  которая должна выполнить обращение к ячейке, указываемой с помощью TB. Затем
;14768  ;  производится запись в CSR1, которая очищает MME, и повторяется READ.V.NOCHK.
;14769  ;  На этот раз ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. должна возбудить ADDR PH L и должно прои-
;14770  ;  зойти обращение к физической ячейке.
;14771  ;        Последняя часть этого теста проверяет, что даже если диспетчер памяти
;14772  ;  запрещен, запись или чтение буфера трансляции продолжает работать (ADDR PH L
;14773  ;  не возбуждается, если происходит обращение к буферу трансляции).
;14774  ;
;14775  ;        ПРЕДПОЛОЖЕНИЯ:
;14776  ;
;14777  ;     Принимается, что все предыдущие тесты прошли успешно.
;14778  ;
;14779  ;        ШАГИ ТЕСТА:
;14780  ;
;14781  ;        1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для рас-
;14782  ;           печатки ошибок) и очистка в LS номера предыдущей ошибки.
;14783  ;        2) Загрузка CSR для установки бита MME.
;14784  ;        3) Загрузка TB данными, содержащими единицы в битах VALID, BYTE OFFSET и
;14785  ;           PA 09 (биты 0 и 31 в посылаемых данных) по адресу TB 0. Это приводит
;14786  ;           к трансляции в физический адрес 200 (H)(вторая страница памяти).
;14787  ;        4) Запись данных, состоящих из всех нулей, по физическому адресу 0.
;14788  ;           Запись данных, состоящих из всех единиц, по физическому адресу 200(H).
;14789  ;        5) Выполнение MEM.REQ с MF=READ.V.NOCHK, используя в качестве адреса 0
;14790  ;           (транслируется в физический адрес 200(H)). Проверка результата на все
;14791  ;           единицы.
;14792  ;        6) Очистка CSR1 для запрета диспетчера памяти.
;14793  ;        7) Повторение шага 5, за исключением проверки результата на все нули
;14794  ;           (должно выполняться обращение по физическому адресу 0).
;14795  ;        8) Изменение маски ошибок для проверки битов от 1 до 22. Поддерживая
;14796  ;           запрет диспетчера памяти, выполнение записи и чтения TB по адресу 0
;14797  ;           сначала всех единиц, затем всех нулей. Этим проверяется, что TB DATA
;14798  ;           EN L запрещает ADDR PH.
;14799  ;
;14800  ;        ОШИБКИ:
;14801  ;
;14802  ;        ошибка 1 - отказ чтения по виртуальному адресу посредством TB. Должен
;14803  ;                   был считываться физический адрес 200. Ожидаемыми и получены-
;14804  ;                   ми данными являются данные из матрицы памяти.
;14805  ;        ошибка 2 - чтение по виртуальному адресу при очищенном MME не перешло
;14806  ;                   в чтение по физическому адресу.
;14807  ;        ошибка 3 - обращение к TB не вызвало установки высокого уровня ADDR PH L
;14808  ;                   при адресных данных, содержащих все единицы. Ожидаемыми и по-
;14809  ;                   лученными данными является содержимое TB по адресу TB 0.
;14810  ;        ошибка 4 - обращение к TB не вызвало установки высокого уровня ADDR PH L
```

```
                    ;14811  ;                при адресных данных, состоящих из всех нулей. Ожидаемыми и по-
                    ;14812  ;                лученными данными является содержимое ТВ по адресу ТВ 0.
                    ;14813  ;
                    ;14814  ;  НАЛАДКА:
                    ;14815  ;
                    ;14816  ;        ОШИБКА 1 - Эта ошибка указывает на неисправность в ранее проверенных ло-
                    ;14817  ;      гических схемах. Следует снова выполнить предыдущие тесты.
                    ;14818  ;
                    ;14819  ;        ОШИБКА 2 - Эта ошибка указывает на неправильную работу ПМЛ РАЗНЫЕ ФУНК-
                    ;14820  ;      ЦИИ УПР. или ее входа L MME H. Вход L MME H можно проверить после того, как
                    ;14821  ;      центральный процессор в пошаговом режиме остановлен на инструкции MEM.REQ
                    ;14822  ;      после очистки CSR. Около ПМЛ РАЗНЫЕ ФУНКЦИИ УПР. Этот сигнал должен быть
                    ;14823  ;      низким. Если он правильный, то вероятно, что неисправна ПМЛ (другие вхо-
                    ;14824  ;      ды уже проверялись).
                    ;14825  ;
                    ;14826  ;        ОШИБКА 3 И 4 - Эти ошибки указывают на неправильную работу ПМЛ РАЗНЫЕ
                    ;14827  ;      ФУНКЦИИ УПР. или входа ТВ DATA EN L. Следует проверить ТВ DATA EN L во
                    ;14828  ;      время зацикливания на ошибке. Он должен становиться низким на то время
                    ;14829  ;      когда происходит обращение к ТВ. Если он правильный, то вероятно, что не-
                    ;14830  ;      исправна ПМЛ РАЗНЫЕ ФУНКЦИИ УПР.
                    ,14831   T.26:
U 1448, B65E,15 ;14832           MOV LS[BEGIN.TEST] TO WR[0]                ; установка в WR0 бита 15 для слова управления и
                    ;14833                                                  ; состояния
U 1449, 3E80,15 ;14834           MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                    ;14835                                                  ; 15 указывает для консольного процессора начало теста
U 144A, 10E0,15 ;14836           MISC [SET.CP.ATTN]                         ; выдача для консольного процессора CPU ATTN
                    ;14837   WAIT.T26.0:
U 144B, 8944,B4 ;14838           JMP [WAIT.T26.0]                           ; ожидание ответа консольного процессора
U 144C, 0A1A,AC ;14839           JSR [SETUP.1]                              ; установка масок, кода модуля и т.д.
U 144D, 0A17,DC ;14840           JSR [WRITE.CSR1.MME]                       ; разрешение диспетчера памяти
U 144E, B640,15 ;14841           MOV LS[BIT0] TO WR[0]                      ; установка бита 0 (входит в качестве PA 09 в ТВ)
U 144F, 477E,15 ;14842           BIS LS[BIT31] TO WR[0]                     ; установка бита 31 (входит в качестве бита VALID в ТВ)
U 1450, 4772,15 ;14843           BIS LS[BIT25] TO WR[0]                     ; установка бита 25 (входит в ТВ в качестве бита BYTE
                    ;14844                                                  ; OFFSET)
U 1451, BE14,15 ;14845           MOV WR[0] TO LS[T10]                       ;
U 1452, 989C,F5 ;14846           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]        ; запрос для записи по адресу 0 в ТВ
U 1453, B214,15 ;14847           WRITE.MEM LS[T10]                          ; установка в ТВ битов VALID, BYTE OFFSET, PA 09
U 1454, 999C,75 ;14848           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]         ; запрос для записи по адресу 0
U 1455, B29C,15 ;14849           WRITE.MEM LS[ZERO]                         ; запись данных, содержащих все 0, по адресу 0
U 1456, 1952,75 ;14850           MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]       ; запрос для записи по адресу 200
U 1457, 329E,15 ;14851           WRITE.MEM LS[ONES]                         ; запись данных, состоящих из всех 1, по адресу 200
                    ;14852   LOOP.T26.1:
U 1458, 369E,95 ;14853           MOV LS[ONES] TO WR[1]                      ; ожидаемые данные
U 1459, 9A9C,75 ;14854           MEM.REQ[READ.V.NOCHK] ADRS[#0] DT[LONG]    ; запрос для чтения по адресу 0
U 145A, 3022,15 ;14855           MOV MEM.DATA TO WR[0]                      ; пересылка данных из физической ячейки
U 145B, 0869,3C ;14856           JSR [CHECK.RESULT]                         ; проверка на все единицы
U 145C, 0945,B4 ;14857           JMP [LOOP.T26.1]                           ; цикл при ошибке, если есть разрешение
U 145D, FF82,15 ;14858           INC LS[ERROR.NUMBER]                       ; ошибка 2
U 145E, 0A17,EC ;14859           JSR [CLEAR.CSR1]                           ; запрет диспетчера памяти
                    ;14860   LOOP.T26.2:
U 145F, 2F82,95 ;14861           CLR WR[1]                                  ; ожидаемые данные
U 1460, 9A9C,75 ;14862           MEM.REQ[READ.V.NOCHK] ADRS[#0] DT[LONG]    ; запрос для чтения по физическому адресу 0
U 1461, 3022,15 ;14863           MOV MEM.DATA TO WR[0]                      ; пересылка данных из физической ячейки 0 (очистка MME
                    ;14864                                                  ; вызывает установку ADDR PH)
U 1462, 0869,3C ;14865           JSR [CHECK.RESULT]                         ; проверка данных на все нули
```

```
U 1463, 8945,F4 ;14866          JMP [LOOP.T26.2]                        ; цикл при ошибке, если есть разрешение
U 1464, FF82,15 ;14867          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 1465, B62A,15 ;14868          MOV LS[#FF000000] TO WR[0]              ; установка старшего байта для маски ошибок
U 1466, C76E,15 ;14869          BIS LS[BIT23] TO WR[0]                  ; установка бита 23
U 1467, C740,15 ;14870          BIS LS[BIT0] TO WR[0]                   ; и установка бита 0. теперь биты от 1 до 22 будут
                ;14871                                                  ; проверяться на отсутствие ошибки
U 1468, 3E8A,15 ;14872          MOV WR[0] TO LS[ERROR.MASK]             ; запоминание маски ошибок в LS
                ;14873  LOOP.T26.3:
U 1469, 369E,95 ;14874          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 146A, 989C,F5 ;14875          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи по адресу ТВ 0
U 146B, 329E,15 ;14876          WRITE.MEM LS[ONES]                      ; запись в ТВ всех единиц
U 146C, 9C9D,F5 ;14877          MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; запрос для чтения из ТВ
U 146D, 3022,15 ;14878          MOV MEM.DATA TO WR[0]                   ; пересылка результата
U 146E, 0869,3C ;14879          JSR [CHECK.RESULT]                      ; проверка результата
U 146F, 8946,94 ;14880          JMP [LOOP.T26.3]                        ; цикл при ошибке, если есть разрешение
U 1470, FF82,15 ;14881          INC LS[ERROR.NUMBER]                    ; ошибка 4
                ;14882  LOOP.T26.4:
U 1471, B69C,95 ;14883          MOV LS[ZERO] TO WR[1]                   ; ожидаемые данные
U 1472, 989C,F5 ;14884          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи по адресу ТВ 0
U 1473, B29C,15 ;14885          WRITE.MEM LS[ZERO]                      ; запись в ТВ всех нулей
U 1474, 9C9D,F5 ;14886          MEM.REQ[READ.TB] ADRS[#0] DT[LONG]      ; запрос для чтения из ТВ
U 1475, 3022,15 ;14887          MOV MEM.DATA TO WR[0]                   ; прием результата
U 1476, 0869,3C ;14888          JSR [CHECK.RESULT]                      ; проверка результата
U 1477, 8947,14 ;14889          JMP [LOOP.T26.4]                        ; цикл при ошибке, если есть разрешение
                ;14890  END.T26:
```

```
;14891  .PAGE "ТЕСТ 27 - проверка логических схем предвыборки (модуль MCT или модуль DAP)"
;14892  ;
;14893  ; ОПИСАНИЕ ТЕСТА:
;14894  ;
;14895  ;          Этот тест проверяет логические схемы предвыборки в модуле MCT. В па-
;14896  ;       мять, начиная с адреса 0, записывается 128 длинных слов (одна страница) с
;14897  ;       данными, равными собственному адресу в старшем байте (уникальные данные).
;14898  ;       По адресу 200 (129-е длинное слово) записываются чередующиеся единицы и
;14899  ;       нули. Этот тест генерирует MEM.REQ с MF=READ.V.RCHK.IFILL. Предварительно
;14900  ;       очищается бит MME в CSR1 (диспетчер памяти запрещен), так что использует-
;14901  ;       ся физический адрес (этим предотвращается необходимость подготовки буфера
;14902  ;       трансляции). Функция READ.V.RCHK.IFILL осуществляет считывание ячейки па-
;14903  ;       мяти, заданной с MEM.REQ, и посылает ее содержимое в центральный процессор
;14904  ;       для загрузки в регистр предвыборки инструкции. ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕД-
;14905  ;       ВЫБОРКИ в модуле MCT должна быть инкрементирована к концу данного цикла
;14906  ;       MEM.REQ и должна теперь указывать на следующее длинное слово в памяти. Те-
;14907  ;       перь генерируется инструкция DECODE с функцией MOV IB.DATA TO OS и регистр
;14908  ;       считывается для выборки первого набора данных (0, соответствующий адресу
;14909  ;       памяти 0). Это декодирование выполняется при PC=3 (в LS10 было загружено
;14910  ;       значение 3) и должен инициироваться новый запрос памяти, использующий инк-
;14911  ;       рементированный адрес из ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫБОРКИ. Снова выполя-
;14912  ;       ется MOV IB.DATA TO OS. На этот раз в OS должны появиться следующие адрес-
;14913  ;       ные данные (единица). Этим проверяются логические схемы сигналов LOAD IB
;14914  ;       и разрешения для РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫБОРКИ. Остальная часть теста
;14915  ;       считывает остаток страницы с целью проверки ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫ-
;14916  ;       БОРКИ. Наконец, выполняется еще одна инструкция MOV IB.DATA TO OS для про-
;14917  ;       верки, что генерируется сигнал PAGE BOUNDARY. Микрокод виртуального адре-
;14918  ;       са инкрементирует VAR и считывает код по адресу 200(H).
;14919  ;
;14920  ; ПРЕДПОЛОЖЕНИЯ:
;14921  ;
;14922  ;       Принимается, что все предыдущие тесты прошли успешно. Этот тест исполь-
;14923  ;       зует логические схемы модуля DAP, которые могут быть еще не проверены.
;14924  ;       Программа диагностирования модуля DAP не может проверять логические схе-
;14925  ;       мы сигнала IB PREFETCH до тех пор, пока не была проверена память. Микро-
;14926  ;       диагностика модуля DAP, которая выполняется после проверки памяти, снова
;14927  ;       проверяет эти логические схемы с позиций модуля DAP. Если известно, что
;14928  ;       используется исправный модуль DAP, то отказ в этом тесте является резуль-
;14929  ;       татом неисправности модуля MCT. Раздел наладки предполагает, что модуль
;14930  ;       DAP исправен.
;14931  ;
;14932  ; ШАГИ ТЕСТА:
;14933  ;
;14934  ;       1) Занесение в LS маски ошибки, номера ошибки и номера модуля (для рас-
;14935  ;          печатки ошибок), очистка в LS номера предыдущей ошибки и очистка в
;14936  ;          CSR1 бита MME.
;14937  ;       2) Запись 0 в память по адресу 0. Запись 1 в байте 3 по физическому адре-
;14938  ;          су 4 (данные 1000000(H)). Запись 2 в байте 3 по физическому адресу 8
;14939  ;          (данные 2000000(H)). Продолжение до тех пор, пока не будет записана
;14940  ;          вся страница памяти (128 длинных слов). Затем запись кода чередую-
;14941  ;          щихся 1 и 0 по адресу 200.
;14942  ;       3) Загрузка в OS всех единиц и загрузка значения 3 в LS 10. Выдача
;14943  ;          MEM.REQ с MF=READ.V.RCHK.IFILL, с использованием LS10 в качестве ис-
;14944  ;          точника адреса.
;14945  ;       4) Выдача инструкции MOV IB.DATA TO OS. Этим байт 3 регистра предыбор-
```

```
;14946  ;          ки пересылается в регистр OS. Чтение регистра OS для проверки на 0.
;14947  ;       5) Загрузка LS10 значением 203 (H) и повторение шагов 3 и 4. Результат
;14948  ;          должен быть чередующиеся 1 и 0 (55(H)).
;14949  ;       6) Повторение шагов 3 и 4, но без проверки результата. Инструкция MOV
;14950  ;          IB.DATA TO OS в шаге 4 должна была загрузить IB следующим длинным
;14951  ;          словом данных (адрес 4). Увеличение LS10 (макро PC) на 2, так что
;14952  ;          теперь PC=6.
;14953  ;       7) Выдача двух инструкций MOV IB.DATA TO OS. Первая устанавливает PC=3,
;14954  ;          вторая считывает третий байт IB и выполняет следующую предвыборку.
;14955  ;       8) Проверка OS на правильность данных. Увеличение макро PC на 2 и пов-
;14956  ;          торение шагов 7 и 8 до тех пор, пока не произойдет обращение к пос-
;14957  ;          леднему длинному слову страницы (длинное слово по адресу 1FC).
;14958  ;       9) Выдача еще двух инструкций DECODE (MOV IB.DATA TO OS) и проверка OS
;14959  ;          на чередующиеся 1 и 0. Этим проверяется логика страницы в последова-
;14960  ;          тельности микрокодов.
;14961  ;
;14962  ; ОШИБКИ:
;14963  ;
;14964  ;       ПРИМЕЧАНИЕ: ожидаемыми и полученными данными является содержимое ре-
;14965  ;       гистра OS, которое загружается из регистра IB (8 битов).
;14966  ;
;14967  ;       ошибка 1 - функция READ.V.RCHK.IFILL не выполнила правильной загрузки
;14968  ;                  IB или в модуле DAP не выполнена инструкция MOV IB.DATA TO OS
;14969  ;                  при данных, содержащих нули.
;14970  ;       ошибка 2 - функция READ.V.RCHK.IFILL не выполнила правильной загрузки IB
;14971  ;                  или в модуле DAP не выполнена инструкция MOV IB.DATA TO OS
;14972  ;                  при данных из чередующихся 1 и 0.
;14973  ;       ошибка 3 - инструкция DECODE при MOV IB.DATA TO OS не выполнила предвы-
;14974  ;                  борки следующей ячейки длинного слова.
;14975  ;       ошибка 4 - неправильно инкрементируется ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫБОРКИ.
;14976  ;       ошибка 5 - неправильно работает логическая схема сигнала PAGE BOUNDARY.
;14977  ;
;14978  ; НАЛАДКА:
;14979  ;
;14980  ;       ПРИМЕЧАНИЕ: если контроллер ОЗУ "зависает" (вызывая таймаут централь-
;14981  ;       ного процессора на инструкции MOV IB.DATA TO OS), затруднение вызвано мо-
;14982  ;       дулем DAP. Этот модуль несет ответственность за возбуждение сигнала DATA
;14983  ;       RCVD во время инструкции DECODE. Память "зависает" в цикле проверки сня-
;14984  ;       тия сигнала CPU GRANT, если сигнал DATA RCVD не возбужден модулем DAP.
;14985  ;
;14986  ;          ОШИБКА 1 И 2 - Эта ошибка указывает на отказ логических схем предвыбор-
;14987  ;       ки либо в модуле MCT, либо в модуле DAP. В этой процедуре наладки пред-
;14988  ;       полагается, что модуль DAP работоспособен.
;14989  ;          При зацикливании на ошибке проверьте сигнал LOAD IB H, выходящий из
;14990  ;       модуля контроллера ОЗУ. Если этот сигнал во время цикла ошибки не со-
;14991  ;       держит импульса высокого уровня, проверьте сигнал I LD IB L(C36), вхо-
;14992  ;       дящий в вентиль "И" с инверсными входами. Этот сигнал поступает непо-
;14993  ;       средственно из управляющего ЗУ контроллера ОЗУ. Убедитесь также, что
;14994  ;       во время цикла ошибки сигнал ENABLE ERROR H становится низким.
;14995  ;          Если LOAD IB H выходит из модуля MCT, то вероятно, что отказ имеется
;14996  ;       в модуле DAP.
;14997  ;
;14998  ;          ОШИБКА 3 - Эта ошибка указывает на отказ логических схем предвыборки
;14999  ;       либо в модуле MCT, либо в модуле DAP. В этой процедуре наладки предпола-
;15000  ;       гается, что модуль DAP работоспособен. При зацикливании на ошибке про-
```

```
                ;15001  ;       верьте сигнал LOAD IB H, поступающий в ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫБОРКИ.
                ;15002  ;       Во время цикла ошибки этот сигнал должен образовать импульсы высокого
                ;15003  ;       уровня. Если он не становится высоким, проследите его назад.
                ;15004  ;          Если на LOAD IB H имеются импульсы, проверьте сигнал OP PREF ADR L.
                ;15005  ;       Он должен быть низким для большей части цикла памяти. Если нет, проверьте
                ;15006  ;       его около ПМЛ УПР.ПРЕДВЫБОРКОЙ. Если там неправильный, проверте вход к
                ;15007  ;       этой ПМЛ на низкий уровень CSR 19 H в то время, когда CONT FUNC LAT L
                ;15008  ;       является высоким, а CPU GRANT L низким. Если эти входы правильные, вере-
                ;15009  ;       ятно, что неисправна ПМЛ.
                ;15010  ;          Если сигнал OP PREF ADR L становится низким, то следует убедиться,
                ;15011  ;       что OP ARY ADR L становится высоким в то же самое время, когда OP PREF
                ;15012  ;       ADR L является низким. Если это так, то отказ, вероятно, имеется в моду-
                ;15013  ;       ле DAP.
                ;15014  ;
                ;15015  ;          ОШИБКА 4 - Подозрительна сама ПМЛ РЕГ/СЧЕТЧ.АДРЕСА ПРЕДВЫБОРКИ, или
                ;15016  ;       отсутствует связь на одном из входов LVA 0X H.
                ;15017  ;
                ;15018  ;          ОШИБКА 5 - При зацикливании на ошибке проверьте сигнал PG BND PREF H.
                ;15019  ;       Он должен содержать импульсы высокого уровня. Если да, то следует про-
                ;15020  ;       верить его на входе к ПМЛ УПР.ПРЕДВЫБОРКОЙ.
                ;15021  ;          Если сигнал PG BND PREF H около ПМЛ УПР.ПРЕДВЫБОРКОЙ правильный, то
                ;15022  ;       вероятно, что неисправна ПМЛ.
                ;15023  ;
                ;15024   T.27:
U 1478, B65E,15 ;15025           MOV LS[BEGIN.TEST] TO WR[0]            ; установка в WR0 бита 15 для слова управления и
                ;15026                                                  ; состояния
U 1479, 3E80,15 ;15027           MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                ;15028                                                  ; 15 указывает для консольного процессора начало теста
U 147A, 10E0,15 ;15029           MISC [SET.CP.ATTN]                     ; выдача для консольного процессора CPU ATTN
                ;15030   WAIT.T27.0:
U 147B, 8947,84 ;15031           JMP [WAIT.T27.0]                       ; цикл для ожидания ответа консольного процессора
U 147C, 0A1A,9C ;15032           JSR [SETUP]                            ; установка маски, кода модуля и т.д. и очистка CSR1
                ;15033                                                  ; MCT
U 147D, 4742,15 ;15034           BIS LS[CPU] TO WR[0]                   ; установка также бита для модуля центрального
                ;15035                                                  ; процессора
U 147E, 3E8C,15 ;15036           MOV WR[0] TO LS[MODULE.NUM]            ; запоминание в LS кода модуля для индикации платы MCT
                ;15037                                                  ; или DAP (центрального процесссора)
U 147F, B62C,15 ;15038           MOV LS[#FFFFFF00] TO WR[0]             ; биты 0-7=0, биты 8-31=единицы
U 1480, 3E8A,15 ;15039           MOV WR[0] TO LS[ERROR.MASK]            ; занесение маски ошибок для проверки 8 битов
U 1481, 3653,15 ;15040           MOV LS[#200] TO WR[2]                  ; WR2 содержит десятичное 128X4
U 1482, B671,95 ;15041           MOV LS[BIT24] TO WR[3]                 ; занесение в WR3 1000000(H) для инкрементирования
                ;15042                                                  ; тестовых данных
U 1483, 6512,15 ;15043           CLR LS[T9]                             ; адрес для записи в основную память
U 1484, 6514,15 ;15044           CLR LS[T10]                            ; данные для записи по адресу в LS 9
                ;15045   BACK.T27:
U 1485, 9912,75 ;15046           MEM.REQ[WRITE.P] ADRS[T9] DT[ ]        ; запрос для записи в память
U 1486, B214,15 ;15047           WRITE.MEM LS[T10]                      ; данных из LS T10 (данные=адрес в байте 3 длинного
                ;15048                                                  ; слова)
U 1487, 3644,15 ;15049           MOV LS[#4] TO WR[0]                    ; занесение в WR0 значения 4
U 1488, 6C12,15 ;15050           ADD WR[0] TO LS[T9]                    ; следующий адрес
U 1489, 6C15,95 ;15051           ADD WR[3] TO LS[T10]                   ; прибавление к данным 1000000 (увеличение байта 3)
                ;15052             CMP LS[T9] WITH WR[2],               ; проверка, что записано 128 длинных слов (одна
U 148A, 4E13,35 ;15053           DT(LONG)&SET.ALU.CC                    ; страница) и установка кодов условий
U 148B, 0948,51 ;15054           JMP.IF[NEQ] TO [BACK.T27]              ; повторение, пока не будет записана полная страница
U 148C, B69A,95 ;15055           MOV LS[#55555555] TO WR[1]             ; занесение в WR1 чередующихся 1 и 0
```

```
U 148D, DF2A,15 ;15056          MCOM LS[#FF000000] TO WR[0]                    ; занесение в WR0 00FFFFFF
U 148E, AF40,95 ;15057          BIC WR[0] TO WR[1]                             ; тестовые данные содержат 55000000
U 148F, 3E14,95 ;15058          MOV WR[1] TO LS[T10]                           ; занесение в LS данных для записи
U 1490, 9912,75 ;15059          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]             ; запрос для записи по адресу памяти 200
U 1491, B214,15 ;15060          WRITE.MEM LS[T10]                              ; данных и из LS T10 (чередующиеся 1 и 0)
U 1492, B69E,15 ;15061          MOV LS[ONES] TO WR[0]                          ; занесение всех единиц в WR0
U 1493, 3EF8,15 ;15062          MOV WR[0] TO LS[OS]                            ; загрузка регистра OS всеми единицами
                ;15063  LOOP.T27.1:
U 1494, 36C0,15 ;15064          MOV LS[#3(H)] TO WR[0]                         ; занесение 3 в рабочий регистр
U 1495, 3E20,15 ;15065          MOV WR[0] TO LS[PC]                            ; загрузка ячейки LS 10 (PC)
U 1496, 1B21,75 ;15066          MEM.REQ[READ.V.RCHK.IFILL] ADRS[PC] DT[LONG] ; загрузка регистра предвыборки из памяти
U 1497, 2F82,95 ;15067          CLR WR[1]                                      ; ожидаемые данные
U 1498, 0A1B,4C ;15068          JSR [NOP.DELAY]                                ; выполнение задержки из 5 циклов для обеспечения
                ;15069                                                         ; окончания заполнения IB
U 1499, 8401,4E ;15070          MOV IB.DATA TO OS                              ; загрузка текущих данных из байта 3 регистра
                ;15071                                                         ; предвыборки в регистр OS
U 149A, DB00,15 ;15072          NOP                                            ; пропуск по IB VALID должен пропустить эту инструкцию
U 149B, B6F8,15 ;15073          MOV LS[OS] TO WR[0]                            ; выборка результата в WR0
U 149C, 0869,3C ;15074          JSR [CHECK.RESULT]                             ; проверка OS на правильность данных
U 149D, 0949,44 ;15075          JMP [LOOP.T27.1]                               ; цикл при ошибке, если есть разрешение
U 149E, FF82,15 ;15076          INC LS[ERROR.NUMBER]                           ; ошибка 2
                ;15077  LOOP.T27.2:
U 149F, B652,15 ;15078          MOV LS[#200] TO WR[0]                          ; занесение 200 в рабочий регистр
U 14A0, C0C0,15 ;15079          ADD LS[#3(H)] TO WR[0]                         ; теперь рабочий регистр содержит 203
U 14A1, 3E20,15 ;15080          MOV WR[0] TO LS[PC]                            ; подготовка PC
U 14A2, 1B21,75 ;15081          MEM.REQ[READ.V.RCHK.IFILL] ADRS[PC] DT[LONG] ; загрузка регистра предвыборки из памяти
U 14A3, B69A,95 ;15082          MOV LS[#55555555] TO WR[1]                     ; ожидаемые данные
U 14A4, 0A1B,4C ;15083          JSR [NOP.DELAY]                                ; выполнение задержки на 5 циклов для завершения
                ;15084                                                         ; заполнения  буфера
U 14A5, 8401,4E ;15085          MOV IB.DATA TO OS                              ; загрузка текущих данных из байта 3 регистра
                ;15086                                                         ; предвыборки в регистр OS
U 14A6, DB00,15 ;15087          NOP                                            ; пропуск по IB VALID должен пропустить эту инструкцию
U 14A7, B6F8,15 ;15088          MOV LS[OS] TO WR[0]                            ; выборка результата в WR0
U 14A8, 0869,3C ;15089          JSR [CHECK.RESULT]                             ; проверка регистра OS на правильность данных
U 14A9, 8949,F4 ;15090          JMP [LOOP.T27.2]                               ; цикл при ошибке, если есть разрешение
U 14AA, FF82,15 ;15091          INC LS[ERROR.NUMBER]                           ; ошибка 3
U 14AB, 3641,15 ;15092          MOV LS[#1] TO WR[2]                            ; начальная установка WR2 на ожидаемый результат
U 14AC, 36C1,95 ;15093          MOV LS[#3(H)] TO WR[3]                         ; WR3 содержит 3
                ;15094  LOOP.T27.3.4:
U 14AD, 3E21,95 ;15095          MOV WR[3] TO LS[PC]                            ; подготовка PC
U 14AE, 1B21,75 ;15096          MEM.REQ[READ.V.RCHK.IFILL] ADRS[PC] DT[LONG] ; занесение начального значения регистра
                ;15097                                                         ; предвыборки и установка в модуле DAP сигнала PC EQUALS
                ;15098                                                         ; 3
U 14AF, 0A1B,4C ;15099          JSR [NOP.DELAY]                                ; выполнение задержки из 5 циклов, чтобы разрешить
                ;15100                                                         ; завершение заполнения IB
U 14B0, 8401,4E ;15101          MOV IB.DATA TO OS                              ; выполнение предвыборки и загрузка текущих данных из
                ;15102                                                         ; байта 3 регистра предвыборки в регистр OS
U 14B1, DB00,15 ;15103          NOP                                            ; пропуск по IB VALID должен пропустить эту инструкцию
U 14B2, 0A1B,4C ;15104          JSR [NOP.DELAY]                                ; выполнение задержки на 5 циклов, чтобы разрешить
                ;15105                                                         ; завершение предвыборки
U 14B3, 3642,15 ;15106          MOV LS[#2] TO WR[0]                            ; занесение 2 в WR0
U 14B4, EC20,15 ;15107          ADD WR[0] TO LS[PC]                            ; увеличение PC на 2
                ;15108  REPEAT.T27.3.4:
U 14B5, A004,95 ;15109          MOV WR[2] TO WR[1]                             ; ожидаемые данные
U 14B6, 8401,4E ;15110          MOV IB.DATA TO OS                              ; установка сигнала PC EQUALS 3 и чтение байта
```

```
U 14B7, DB00,15 ;15111          NOP                                     ; пропуск по IB VALID должен пропустить эту инструкцию
U 14B8, 8401,4E ;15112          MOV IB.DATA TO OS                       ; выполнение предвыборки и загрузка текущих данных из
                ;15113                                                  ; байта 3 регистра предвыборки в регистр OS
U 14B9, DB00,15 ;15114          NOP                                     ; пропуск по IB VALID должен пропустить эту инструкцию
U 14BA, B6F8,15 ;15115          MOV LS[OS] TO WR[0]                     ; выборка результата в WR0
U 14BB, 0869,3C ;15116          JSR [CHECK.RESULT]                      ; проверка правильности данных в регисре OS
U 14BC, 094A,D4 ;15117          JMP [LOOP.T27.3.4]                      ; цикл при ошибке, если есть разрешение
U 14BD, 3644,15 ;15118          MOV LS[#4] TO WR[0]                     ; занесение 4 в рабочий регистр
U 14BE, BE82,15 ;15119          MOV WR[0] TO LS[ERROR.NUMBER]           ; этот цикл теперь будет указывать ошибку 4
U 14BF, 3642,15 ;15120          MOV LS[#2] TO WR[0]                     ; подготовка к прибавлению 2
U 14C0, EC20,15 ;15121          ADD WR[0] TO LS[PC]                     ; увеличение макро PC на 2
U 14C1, C045,95 ;15122          ADD LS[#4] TO WR[3]                     ; увеличение старого значения PC (для зацикливания на
                ;15123                                                  ; ошибке)
U 14C2, 2045,15 ;15124          INC WR[2]                               ; увеличение ожидаемого результата
                ;15125            CMP WR[2] WITH LS[#80],               ; проверка, следует ли граница страницы и установка
                ;15126                                                  ; кодов условий
U 14C3, CF4F,35 ;15127          DT(LONG)&SET.ALU.CC                     ; повторение, пока не будет проверена одна страница
U 14C4, 094B,51 ;15128          JMP.IF[NEQ] TO [REPEAT.T27.3.4]         ;
U 14C5, FF82,15 ;15129          INC LS[ERROR.NUMBER]                    ; ошибка 5
                ;15130  LOOP.T27.5:
U 14C6, B69A,95 ;15131          MOV LS[#55555555] TO WR[1]              ; ожидаемые данные
U 14C7, 8401,4E ;15132          MOV IB.DATA TO OS                       ; установка PC=3 для следующей инструкции DECODE
U 14C8, DB00,15 ;15133          NOP                                     ; пропуск по IB VALID должен пропустить эту инструкцию
U 14C9, 8401,4E ;15134          MOV IB.DATA TO OS                       ; выполнение предвыборки и загрузка текущих данных из
                ;15135                                                  ; байта 3 регистра предвыборки в регистр OS
U 14CA, DB00,15 ;15136          NOP                                     ; пропуск по IB VALID должен пропустить эту инструкцию
U 14CB, B6F8,15 ;15137          MOV LS[OS] TO WR[0]                     ; выбор результата в WR0
U 14CC, 0869,3C ;15138          JSR [CHECK.RESULT]                      ; проверка на правильность данных в регистре OS
U 14CD, 894C,F4 ;15139          JMP [ERRORLOOP.T27.5]                   ; цикл при ошибке, если есть разрешение
U 14CE, 094D,A4 ;15140          JMP [END.T27]                           ; конец
                ;15141  ERRORLOOP.T27.5:
U 14CF, B652,15 ;15142          MOV LS[#200] TO WR[0]                   ; занесение 200 в WR0
U 14D0, 2100,15 ;15143          DEC WR[0]                               ; WR0 содержит 1FF
U 14D1, 3E20,15 ;15144          MOV WR[0] TO LS[PC]                     ; старое значение PC для цикла при ошибке 5
U 14D2, 1B21,75 ;15145          MEM.REQ[READ.V.RCHK.IFILL] ADRS[PC] DT[LONG] ; заполнение IB старыми данными и установки PC=3
U 14D3, 0A1B,4C ;15146          JSR [NOP.DELAY]                         ; выполнение задержки на 5 циклов для разрешения
                ;15147                                                  ; окончания заполнения IB
U 14D4, 8401,4E ;15148          MOV IB.DATA TO OS                       ; выполнение предвыборки при границе страницы
U 14D5, DB00,15 ;15149          NOP                                     ; пропуск по IB VALID должен пропустить эту инструкцию
U 14D6, B652,15 ;15150          MOV LS[#200] TO WR[0]                   ; занесение 200 в WR0
U 14D7, C042,15 ;15151          ADD LS[#2] TO WR[0]                     ; WR0 содержит 202
U 14D8, 3E20,15 ;15152          MOV WR[0] TO LS[PC]                     ; подготовка макро PC для инструкции MOV IB.DATA
U 14D9, 894C,64 ;15153          JMP [LOOP.T27.5]                        ; цикл при ошибке
                ;15154  END.T27:
                ;15155
```

```
;15156  .PAGE "ТЕСТ 28 - прекращение заполнения IB при неисправимой ошибке (модуль MCT)"
;15157  ;
;15158  ; ОПИСАНИЕ ТЕСТА:
;15159  ;
;15160  ;       Этот тест проверяет, что неисправимая ошибка в ячейке памяти, к которой
;15161  ;    происходит обращение в операции заполнения IB, вызывает прекращение процеду-
;15162  ;    ры заполнения IB. В ячейку памяти 4 записываются все нули с неисправимой ошиб-
;15163  ;    кой, закодированной в контрольных битах. Ошибка записывается с использовани-
;15164  ;    ем процедуры MAINT.ECC.DATA. В ячейку 0 записываются данные, состоящие из
;15165  ;    всех единиц и правильных контрольных битов. Этот тест выдает MEM.REQ с MF=
;15166  ;    READ.V.RCHK.IFILL и DT=длинное слово. Используемым адресом является адрес 0
;15167  ;    (бит MME в CSR1 очищен ранее, так что происходит обращение по физическому
;15168  ;    адресу). IB проверяется на все единицы. Затем снова выдается MEM.REQ с MF=
;15169  ;    READ.V.RCHK.IFILL и адресом 4 в качестве приемника. Поскольку в адресе 4 со-
;15170  ;    держится неисправимая ошибка, аппаратура должна предотвратить загрузку IB
;15171  ;    путем запрета LOAD IB H. Эту функцию выполняют совместно STALL MBSY L и
;15172  ;    ERROR L. Будет проверяться CSR1, чтобы удостовериться, что установлен только
;15173  ;    бит RDS. Затем проверяется CSR0 на правильные контрольные биты для двойной
;15174  ;    ошибки (это, главным образом, проверка микрокода).
;15175  ;
;15176  ; ПРЕДПОЛОЖЕНИЯ:
;15177  ;
;15178  ;    Принимается, что все предыдущие тетсы прошли успешно.
;15179  ;
;15180  ; ШАГИ ТЕСТА:
;15181  ;
;15182  ;    1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для распечат-
;15183  ;       ки ошибок) и очистка в LS номера предыдущей ошибки. Установка маски оши-
;15184  ;       бок на проверку младших битов.
;15185  ;    2) Загрузка ячейки памяти 0 всеми единицами. Затем загрузка в CSR1 контроль-
;15186  ;       ных битов 1111101(B) (перед записью в CSR1 они должны инвертироваться)
;15187  ;       для создания ситуации двойной ошибки при данных, состоящих из всех нулей.
;15188  ;       Очистка остальных битов CSR.
;15189  ;    3) Загрузка промежуточной ячейки LS значением 4, а также установка бита 0
;15190  ;       (он используется для ветвления в правильную процедуру для записи ошибки в
;15191  ;       память).
;15192  ;    4) Выдача MEM.REQ с MF=MAINT.ECC.DATA и DT=длинное слово. Использование в
;15193  ;       качестве адреса промежуточной ячейки LS. Затем выдача WRITE.MEM LS с дан-
;15194  ;       ными, состоящими из нулей.
;15195  ;    5) Очистка LS10 (используемой в качестве адреса для MOV IB.DATA TO OS) и
;15196  ;       выдача MEM.REQ с MF=READ.V.RCHK.IFILL. Использование ячейки в LS, содер-
;15197  ;       жащей 0, в качестве адреса. Выдача MOV IB.DATA TO OS и проверка регистра
;15198  ;       OS на все единицы.
;15199  ;    6) Повторение шага 5 с использованием ячейки LS, содержащей 4, в качестве
;15200  ;       адреса. Регистр OS должен все еще содержать единицы.
;15201  ;    7) Замена маски ошибок для проверки битов CSR (14-23 и 25-31) и проверка,
;15202  ;       что установлен только бит 31 (RDS).
;15203  ;    8) Замена маски ошибок для проверки только битов 6-0 и чтение CSR0. Проверка
;15204  ;       правильности битов сквт.
;15205  ;
;15206  ; ОШИБКИ:
;15207  ;
;15208  ;    ошибка 1 - не выполнена функция IB FILL. Ожидаемыми и полученными данными
;15209  ;               является содержимое регистра OS, которое было загружено из регис-
;15210  ;               тра IB (8 битов).
```

```
                ;15211  ;    ошибка 2 - сигнал LOAD IB не запрещен неисправимой ошибкой памяти.
                ;15212  ;    ошибка 3 - очищен бит RDS или установлены другие биты при неисправимой ошиб-
                ;15213  ;               ке памяти. Ожидаемыми и полученными данными являются биты CSR1
                ;15214  ;               14-23 и 25-31.
                ;15215  ;    ошибка 4 - контрольные биты в CSR0 неправильные для двойной ошибки, которая
                ;15216  ;               произошла. Ожидаемыми и полученными данными являются биты 0-6 CSR0
                ;15217  ;
                ;15218  ; НАЛАДКА:
                ;15219  ;
                ;15220  ;       ОШИБКА 1 - Следует снова выполнить предыдущие тесты.
                ;15221  ;
                ;15222  ;       ОШИБКА 2 - Эта ошибка указывает на отказ в логических схемах сигнала LOAD
                ;15223  ;    IB в модуле MCT. При зацикливании на ошибке следует проверить входы I LD IB L
                ;15224  ;    и ENABLE ERROR H вентиля "И" с инверсными входами, который генерирует LOAD
                ;15225  ;    IB H. Сигнал ENABLE ERROR H должен быть высоким всегда, когда I LD IB L явля-
                ;15226  ;    ется низким, препятствуя этим возбуждению сигнала LOAD IB H. Если сигнал не-
                ;15227  ;    правильный, проследите его назад до ERROR L и STALL MBSY L. Оба эти сигнала
                ;15228  ;    должны быть низкими, когда I LD IB H низкий.
                ;15229  ;
                ;15230  ;       ОШИБКА 3 - Эта ошибка указывает на отказ в ранее проверенных логических
                ;15231  ;    схемах. Следует снова выполнить предыдущие тесты.
                ;15232  ;
                ;15233  ;       ОШИБКА 4 - Эта ошибка с наибольшей вероятностью указывает на неисправность
                ;15234  ;    в микрокодах. Проверьте, что в цикле READ.V.RCHK.IFILL.D.E имеется сигнал
                ;15235  ;    CSR CBSYN CLK.
                ;15236  ;
                ;15237   T.28:
U 14DA, B65E,15 ;15238           MOV LS[BEGIN.TEST] TO WR[0]            ; установка в WR[0] бита 15 для слова управления и состояния
U 14DB, 3E80,15 ;15239           MOV WR[0] TO LS[CONTROL.STATUS]        ; установка бита 15 в слове управления и состояния. Бит
                ;15240                                                  ; 15 указывает для конс.процессора начало теста
U 14DC, 10E0,15 ;15241           MISC [SET.CP.ATTN]                     ; выдача для конс.процессора CPU ATTN
                ;15242   WAIT.T28.0:
U 14DD, B94D,D4 ;15243           JMP [WAIT.T28.0]                       ; цикл для ожидания ответа конс.процессора
U 14DE, 0A1A,9C ;15244           JSR [SETUP]                            ; установка маски, кода модуля и т.д. и очистка CSR1
                ;15245                                                  ; MCT
U 14DF, B62D,95 ;15246           MOV LS[#FFFFFF00] TO WR[3]              ; в рабочем регистре младший байт содержит нули
U 14E0, 3E8B,95 ;15247           MOV WR[3] TO LS[ERROR.MASK]             ; при ошибках проверяется только младший байт
U 14E1, B62F,15 ;15248           MOV LS[CSR1.MASK] TO WR[2]              ; маска ошибки для ошибки 3. Не должны проверяться биты
                ;15249                                                  ; 0-13 или бит 24
U 14E2, 999C,75 ;15250           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]     ; запрос для записи единиц в памяти по физическому
                ;15251                                                  ; адресу 0
U 14E3, 329E,15 ;15252           WRITE.MEM LS[ONES]                     ; запись всех единиц в ячейке 0 основной памяти
U 14E4, 370E,15 ;15253           MOV LS[CKBTS.1111101] TO WR[0]         ; выборка в WR0 контрольных разрядов для двойной ошибки
U 14E5, A0C0,15 ;15254           COM WR[0]                              ; инвертирование контрольных битов для правильного
                ;15255                                                  ; формата в аппаратуре
U 14E6, 452C,15 ;15256           BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех байтов, кроме младшего
U 14E7, BA17,FC ;15257           JSR [WRITE.CSR1]                       ; занесение в CSR1 контрольных битов для двойной ошибки,
                ;15258                                                  ; очистка всех других битов CSR1
U 14E8, 3644,15 ;15259           MOV LS[#4] TO WR[0]                    ; занесение значения 4 в WR0
U 14E9, C740,15 ;15260           BIS LS[BIT0] TO WR[0]                  ; а также занесение LVA 00 для ветвления
U 14EA, BE12,15 ;15261           MOV WR[0] TO LS[T9]                    ; запоминание адреса в LS 9
U 14EB, 9D12,F5 ;15262           MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической процедуры
U 14EC, B29C,15 ;15263           WRITE.MEM LS[ZERO]                     ; запись нулей в ячейку 4 с двойной ошибкой,
                ;15264                                                  ; закодированной в контрольных битах
                ;15265   LOOP.T28.1:
```

```
U 14ED, 369E,95 ;15266          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 14EE, E520,15 ;15267          CLR LS[PC]                              ; байт в регистре IB для загрузки в OS
U 14EF, 6512,15 ;15268          CLR LS[T9]                              ; адрес для обращения к памяти
U 14F0, 9B13,75 ;15269          MEM.REQ[READ.V.RCHK.IFILL] ADRS[T9] DT[LONG] ; запрос для загрузки IB из адреса 0 основной
                ;15270                                                  ; памяти (содержит единицы)
U 14F1, B401,4E ;15271          MOV IB.DATA TO OS                       ; занесение IB в регистр OS
U 14F2, DB00,15 ;15272          NOP                                     ; инструкция MOV IB.DATA выполнит пропуск при IB.VALID
U 14F3, B6F8,15 ;15273          MOV LS[OS] TO WR[0]                     ; выборка результата в WR0 для проверки
U 14F4, 0869,3C ;15274          JSR [CHECK.RESULT]                      ; проверка единиц
U 14F5, 894E,D4 ;15275          JMP [LOOP.T28.1]                        ; цикл при ошибке, если есть разрешение
                ;15276  LOOP.T28.3.4:
U 14F6, 3642,15 ;15277          MOV LS[#2] TO WR[0]                     ; занесение 2 в рабочий регистр
U 14F7, BE82,15 ;15278          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 2
U 14F8, 3E8B,95 ;15279          MOV WR[3] TO LS[ERROR.MASK]             ; проверка только 8 младших битов (установка на случай
                ;15280                                                  ; зацикливания при ошибках 3 или 4)
                ;15281  LOOP.T28.2:
U 14F9, 369E,95 ;15282          MOV LS[ONES] TO WR[1]                   ; ожидаемые данные
U 14FA, E520,15 ;15283          CLR LS[PC]                              ; байт в регистре IB для загрузки в OS
U 14FB, 3644,15 ;15284          MOV LS[#4] TO WR[0]                     ; адрес для обращения к памяти
U 14FC, BE12,15 ;15285          MOV WR[0] TO LS[T9]                     ; занесение в LS
U 14FD, 9B13,75 ;15286          MEM.REQ[READ.V.RCHK.IFILL] ADRS[T9] DT[LONG] ; запрос для загрузки IB из адреса 4 основной
                ;15287                                                  ; памяти (содержит нули с ошибкой RDS)
U 14FE, B401,4E ;15288          MOV IB.DATA TO OS                       ; занесение IB в регистр OS
U 14FF, DB00,15 ;15289          NOP                                     ; если заполнение IB не прекращено, произойдет пропуск
                ;15290                                                  ; по IB VALID
U 1500, B6F8,15 ;15291          MOV LS[OS] TO WR[0]                     ; выборка результата в WR0 для проверки
U 1501, 0869,3C ;15292          JSR [CHECK.RESULT]                      ; проверка единиц (загрузка IB при прекращенном
                ;15293                                                  ; заполнении)
U 1502, 894F,94 ;15294          JMP [LOOP.T28.2]                        ; цикл при ошибке, если есть разрешение
U 1503, FF82,15 ;15295          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 1504, B67E,95 ;15296          MOV LS[RDS] TO WR[1]                    ; ожидаемые данные (бит RDS установлен, остальные
                ;15297                                                  ; очищенные)
U 1505, BE8B,15 ;15298          MOV WR[2] TO LS[ERROR.MASK]             ; проверка битов ошибок и управляющих битов CSR1 (биты
                ;15299                                                  ; 14-23 и 25-31)
U 1506, 9D45,75 ;15300          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1507, 3022,15 ;15301          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR1
U 1508, 0869,3C ;15302          JSR [CHECK.RESULT]                      ; проверка на RDS установленный, осальные очищенные
U 1509, 894F,64 ;15303          JMP [LOOP.T28.3.4]                      ; цикл при ошибке, если есть разрешение
U 150A, FF82,15 ;15304          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 150B, A006,15 ;15305          MOV WR[3] TO WR[0]                      ; WR0 содержит FFFFF00
U 150C, 474E,15 ;15306          BIS LS[BIT7] TO WR[0]                   ; установка в WR0 бита 7
U 150D, 3E8A,15 ;15307          MOV WR[0] TO LS[ERROR.MASK]             ; маска ошибок для проверки только битов 6-0
U 150E, B70E,95 ;15308          MOV LS[CKBTS.1111101] TO WR[1]          ; выборка в WR1 контрольных битов с двойной ошибкой
U 150F, A0C2,95 ;15309          COM WR[1]                               ; инвертирование контрольных битов для приведения в
                ;15310                                                  ; соответствие с аппаратурой (ожидаемые данные)
U 1510, 9D9D,75 ;15311          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ; запрос для чтения CSR0
U 1511, 3022,15 ;15312          MOV MEM.DATA TO WR[0]                   ; чтение CSR0 для проверки правильности контрольных
                ;15313                                                  ; битов
U 1512, 0869,3C ;15314          JSR [CHECK.RESULT]                      ; проверка результата
U 1513, 894F,64 ;15315          JMP [LOOP.T28.3.4]                      ; цикл при ошибке, если есть разрешение
                ;15316  END.T28:
                ;15317
```

```
;15318  .PAGE "TEST 29 - тест очистки CSR и инициализации по CINIT, UBS DCLO (модуль MCT или WCS)"
;15319  ;
;15320  ; ОПИСАНИЕ ТЕСТА:
;15321  ;
;15322  ;       Этот тест проверяет логические схемы, используемые для очистки управля-
;15323  ;       ющих битов CSR1 (битов 25-29) и вызова процедуры инициализации памяти
;15324  ;       PWRFL. Для вызова ее используются два сигнала, генерируемые в модуле
;15325  ;       WCS: CINIT L и UBS DCLO L. Если они возбуждены, то проходят через два бу-
;15326  ;       фера (защелки) и порождают L INIT DLY H на два цикла позднее. Этот сиг-
;15327  ;       нал генерирует CLR CSR L и используется в качестве входа к ПМЛ СИГН.КОНТ.
;15328  ;       ПИТАНИЯ И ИНИЦИАЛИЗАЦИИ. Эта ПМЛ несет ответственность за возбуждение
;15329  ;       сигнала PWRFL L, который переводит микрокод памяти на программу инициа-
;15330  ;       лизации по включению питания, когда либо возбужден сигнал ALLOW REF, ли-
;15331  ;       бо имеется переход в пустой цикл.
;15332  ;         Эта программа записывает все единицы в биты CSR1 25-29 и выполняет
;15333  ;       MEM.REQ с MF=WRITE.TB. Затем выполняется WRITE.MEM данными из череду-
;15334  ;       ющихся 1 и 0. Затем выдается MEM REQ с MF=READ.TB. Для консольного про-
;15335  ;       цессора выдается CPU ATTN с целью генерации CINIT, и управление снова
;15336  ;       возвращается центральному процессору. Выполняется задержка из 5 нерабо-
;15337  ;       чих циклов и затем выполняется MOV MEM.DATA TO WRI0]. Полученные данные
;15338  ;       должны отличаться от тех, которые посылались, так как прежде, чем была
;15339  ;       выдана инструкция MOV MEM.DATA, произошло пропадание питания. Пропадание
;15340  ;       питания удаляет с шины MC данные, которые запрашивались инструкцией
;15341  ;       MEM.REQ[READ.TB]. Наконец, CSR1 проверяется на нули в битах 29-25. CINIT
;15342  ;       очищается до того, как происходит чтение CSR. Этот тест повторяется с
;15343  ;       возбужденным сигналом UBS DCLO.
;15344  ;
;15345  ; ПРЕДПОЛОЖЕНИЯ:
;15346  ;
;15347  ;       Принимается, что все предыдущие тесты прошли успешно.
;15348  ;
;15349  ; ШАГИ ТЕСТА:
;15350  ;       1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для распе-
;15351  ;          чатки ошибок) и очистка в LS номера предыдущей ошибки.
;15352  ;       2) Запись кода из чередующихся 1 и 0 в ячейку 0 буфера трансляции.
;15353  ;       3) Запись в CSR1 единиц в битах 29-29. Затем выдача MEM.REQ с MF=READ.TB.
;15354  ;          выдача CPU ATTN с запросом консольному процессору для возбуждения
;15355  ;          CINIT.
;15356  ;       4) Выполнение 5 нерабочих циклов после возвращения управления центрально-
;15357  ;          му процессору в качестве задержки, необходимой для получения установи-
;15358  ;          вшегося состояния шины MC. Затем выдача MOV MEM.DATA TO WRI0] и про-
;15359  ;          верка, что результат не содержит чередующихся 1 и 0.
;15360  ;       5) Выдача CPU ATTN для снятия CINIT, изменение маски ошибок и чтение
;15361  ;          CSR1. Проверка, что биты 25-29 очищены.
;15362  ;       6) Повторение шагов от 2 до 5 при возбужденном UBS DCLO.
;15363  ;
;15364  ; ОШИБКИ:
;15365  ;
;15366  ;       ошибка 1 - в MCT не выполняется процедура PWRFL при возбужденном CINIT.
;15367  ;       ошибка 2 - биты 29-25 CSR1 не очищаются при возбужденном CINIT.
;15368  ;       ошибка 3 - в MCT не выполняется процедура PWRFL при возбужденном UBS DCLO.
;15369  ;       ошибка 4 - биты 29-25 CSR1 не очищаются при возбужденном UBS DCLO.
;15370  ;
;15371  ; НАЛАДКА:
;15372  ;
```

```
                    ;15373  ;        ОШИБКА 1 - Эта ошибка указывает на отказ в ПМЛ СИГН.КОНТР.ПИТАНИЯ И
                    ;15374  ;      ИНИЦИАЛИЗАЦИИ, ее входах или в логических схемах, которые генерируют сиг-
                    ;15375  ;      нал INIT DLY H из CINIT L. Центральный процессор следует остановить в по-
                    ;15376  ;      шаговом режиме на инструкции, предшествующей MOV MEM.DATA TO WR[0], и
                    ;15377  ;      проверить на высокий уровень сигнал L INIT DLY H на входе ПМЛ СИГН.КОНТР.
                    ;15378  ;      ПИТАНИЯ И ИНИЦИАЛИЗАЦИИ. Если он неправильный, проследите его назад через
                    ;15379  ;      защелки и вентиль "ИЛИ" с инверсными входами до CINIT. Если в этой точке
                    ;15380  ;      CINIT является высоким, отказ может быть в модуле WCS.
                    ;15381  ;         Если сигнал L INIT DLY H около ПМЛ.КОНТР.ПИТАНИЯ И ИНИЦИАЛИЗАЦИИ являет-
                    ;15382  ;      ся высоким, проверьте на высокий уровень ALLOW REF H. Если он правильный,
                    ;15383  ;      выход PWRFL L должен быть низким. Если нет - ПМЛ неисправна.
                    ;15384  ;         Если PWRFL L является низким, проверьте, что он доходит до логической
                    ;15385  ;      схемы, которая формирует STOP MEM H. Если правильно, проверьте его на
                    ;15386  ;      входе к логической схеме, формирующей MUX SEL L. Логические схемы после
                    ;15387  ;      этой точки проверены ранее.
                    ;15388  ;
                    ;15389  ;         ОШИБКА 2 - Если это первая ошибка, она указывает на отказ инвертора,
                    ;15390  ;      который формирует CLR CSR L, или буфера, который выводит биты 29-25 CSR1.
                    ;15391  ;      Если сигнал неправильный около этого буфера, то следуя процедуре, изложе-
                    ;15392  ;      нной выше, проследите CLR CSR L назад. Если CLR CSR L является низким,то
                    ;15393  ;      может быть неисправным сам буфер.
                    ;15394  ;
                    ;15395  ;         ОШИБКА 3 - Пройдите по шагам, как при ошибке 1, и проверьте на низкий
                    ;15396  ;      уровень сигнал UBS DCLO L. Проследите его до выхода из вентиля "ИЛИ" с
                    ;15397  ;      инверсными входами. Логические схемы после этой точки проверялись ранее.
                    ;15398  ;
                    ;15399  ;         ОШИБКА 4 - См. ошибку 2.
                    ;15400   T.29:
U 1514, B65E,15 ;15401           MOV LS[BEGIN.TEST] TO WR[0]          ; установка в WR[0] бита 15 для слова управления и
                    ;15402                                            ; состояния
U 1515, 3E80,15 ;15403           MOV WR[0] TO LS[CONTROL.STATUS]      ; установка бита 15 в слове управления и состояния. Бит
                    ;15404                                            ; 15 указывает для консольного процессора начало теста
U 1516, 10E0,15 ;15405           MISC [SET.CP.ATTN]                   ; выдача для консольного процессора CPU ATTN
                    ;15406   WAIT.T29.0:
U 1517, 0951,74 ;15407           JMP [WAIT.T29.0]                     ; цикл для ожидания ответа конс.процессора
U 1518, 0A1A,9C ;15408           JSR [SETUP]                          ; установка маски, кода модуля и т.д. и очистка CSR1 MCT
U 1519, C740,15 ;15409           BIS LS[WCS] TO WR[0]                 ; а также установка бита для WCS
U 151A, 3EBC,15 ;15410           MOV WR[0] TO LS[MODULE.NUM]          ; запоминание кода модуля в LS для индикации платы MCT
                    ;15411                                            ; или WCS
U 151B, B640,15 ;15412           MOV LS[#1] TO WR[0]                  ; занесение 1 в рабочий регистр
U 151C, BEFC,15 ;15413           MOV WR[0] TO LS[SIZE]                ; установка РЕГ. SIZE на тип данных=слово
U 151D, 989C,F5 ;15414           MEM.REQ[WRITE.TB] ADRS[ZERO] DT[LONG] ; запрос для записи в ТВ по адресу 0
U 151E, B29A,15 ;15415           WRITE.MEM LS[#55555555]               ; запись в ТВ чередующихся 1 и 0
U 151F, B69E,15 ;15416           MOV LS[ONES] TO WR[0]                ; занесение в WR0 данных для CSR
U 1520, BA17,FC ;15417           JSR [WRITE.CSR1]                     ; вызов программ для записи 1 в битах 25-29 CSR
                    ;15418   LOOP.T29.1.2:
U 1521, B640,15 ;15419           MOV LS[#1] TO WR[0]                  ; занесение 1 в рабочий регистр
U 1522, BE82,15 ;15420           MOV WR[0] TO LS[ERROR.NUMBER]        ; установка в LS номера ошибки
U 1523, 3660,15 ;15421           MOV LS[INTERUPT.EN] TO WR[0]         ; установка в WR0 бита 16
U 1524, 3E80,15 ;15422           MOV WR[0] TO LS[CONTROL.STATUS]      ; запрос на выключение CINIT (и любых других прерываний,
                    ;15423                                            ; если установлены)
U 1525, 10E0,15 ;15424           MISC [SET.CP.ATTN]                   ; выдача для консольного процессора CPU ATTN
                    ;15425   WAIT.T29.1:
U 1526, 8952,64 ;15426           JMP [WAIT.T29.1]                     ; цикл для ожидания ответа консольного процессора
U 1527, 9C9D,F5 ;15427           MEM.REQ[READ.TB] ADRS[ZERO] DT[LONG] ; запрос для чтения ТВ по адресу 0
```

```
U 1528, 3660,15 ;15428          MOV LS[INTERUPT.EN] TO WR[0]            ; установка бита 16 в WR0
U 1529, 4778,15 ;15429          BIS LS[CINIT] TO WR[0]                  ; установка бита для выдачи из консольного процессора
                ;15430                                                  ; CINIT
U 152A, C76E,15 ;15431          BIS LS[NA.EXP.REC] TO WR[0]             ; не печатать данных под EXP (ожидаемые) и REC
                ;15432                                                  ; (полученные)
U 152B, 3E80,15 ;15433          MOV WR[0] TO LS[CONTROL.STATUS]         ; подготовка слова управления и состояния
U 152C, 10E0,15 ;15434          MISC [SET.CP.ATTN]                      ; выдача для конс.процессора CPU ATTN
                ;15435  WAIT.T29.2:
U 152D, 0952,D4 ;15436          JMP [WAIT.T29.2]                        ; цикл для ожидания ответа конс.процессора
U 152E, 0A1B,4C ;15437          JSR [NOP.DELAY]                         ; выполнение задержки на 5 циклов, чтобы разрешить
                ;15438                                                  ; установиться состоянию шины MC
U 152F, 2F82,95 ;15439          CLR WR[1]                               ; занесение неправильных ожидаемых данных
U 1530, 3022,15 ;15440          MOV MEM.DATA TO WR[0]                   ; выборка данных после инициализации
                ;15441          CMP LS[#55555555] WITH WR[0],           ; проверка, что результат не
                ;15442                                                  ; совпадает с данными, переданными в TB
U 1531, CE9A,55 ;15443          DT(SIZE)&SET.ALU.CC                     ; сигнал CINIT должен был это предотвратить. Установка
                ;15444                                                  ; кодов условий по младшему слову
U 1532, 0953,51 ;15445          JMP.IF[NEQ] TO [TEST.T29.2]             ; переход, если не ошибка
U 1533, 0869,3C ;15446          JSR [CHECK.RESULT]                      ; иначе печать сообщения об ошибке
U 1534, 0952,14 ;15447          JMP [LOOP.T29.1.2]                      ; цикл при ошибке, если есть разрешение
                ;15448  TEST.T29.2:
U 1535, 3660,15 ;15449          MOV LS[INTERUPT.EN] TO WR[0]            ; установка в WR0 бита 16
U 1536, 3E80,15 ;15450          MOV WR[0] TO LS[CONTROL.STATUS]         ; запрос на выключение CINIT (и любых других
                ;15451                                                  ; прерываний, если установлены)
U 1537, 10E0,15 ;15452          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;15453  WAIT.T29.3:
U 1538, 8953,84 ;15454          JMP [WAIT.T29.3]                        ; цикл для ожидания ответа конс.процессора
U 1539, FF82,15 ;15455          INC LS[ERROR.NUMBER]                    ; ошибка 2
U 153A, DF2A,15 ;15456          MCOM LS[#FF000000] TO WR[0]             ; занесение в WR0 значения 00FFFFFF
U 153B, 477E,15 ;15457          BIS LS[BIT31] TO WR[0]                  ; установка бита 31
U 153C, C77C,15 ;15458          BIS LS[BIT30] TO WR[0]                  ; установка бита 30
U 153D, C770,15 ;15459          BIS LS[BIT24] TO WR[0]                  ; установка бита 24
U 153E, 3E8A,15 ;15460          MOV WR[0] TO LS[ERROR.MASK]             ; проверка битов 29-24
U 153F, 2F82,95 ;15461          CLR WR[1]                               ; ожидаемые данные
U 1540, 9D45,75 ;15462          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос на чтение CSR
U 1541, 3022,15 ;15463          MOV MEM.DATA TO WR[0]                   ; выборка данных CSR1
U 1542, 0869,3C ;15464          JSR [CHECK.RESULT]                      ; проверка, что биты 29-24 очищены
U 1543, 0952,14 ;15465          JMP [LOOP.T29.1.2]                      ; цикл при ошибке, если есть разрешение
U 1544, E58A,15 ;15466          CLR LS[ERROR.MASK]                      ; проверка всех битов
U 1545, 989C,F5 ;15467          MEM.REQ[WRITE.TB] ADRS[ZERO] DT[LONG]   ; запрос для записи в TB по адресу 0
U 1546, B29A,15 ;15468          WRITE.MEM LS[#55555555]                 ; запись в TB чередующихся 1 и 0
U 1547, B49E,15 ;15469          MOV LS[ONES] TO WR[0]                   ; подготовка в WR0 данных для CSR
U 1548, BA17,FC ;15470          JSR [WRITE.CSR1]                        ; вызов программы для записи единиц в битах 25-29 CSR1
                ;15471  LOOP.T29.3.4:
U 1549, 36C0,15 ;15472          MOV LS[#3(H)] TO WR[0]                  ; занесение в WR0 значения 3
U 154A, BEB2,15 ;15473          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 3
U 154B, 3660,15 ;15474          MOV LS[INTERUPT.EN] TO WR[0]            ; установка в WR0 бита 16
U 154C, 3E80,15 ;15475          MOV WR[0] TO LS[CONTROL.STATUS]         ; запрос на выключение CINIT (и любых других прерываний,
                ;15476                                                  ; если установлены)
U 154D, 10E0,15 ;15477          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;15478  WAIT.T29.4:
U 154E, 0954,E4 ;15479          JMP [WAIT.T29.4]                        ; цикл для ожидания ответа конс.процессора
U 154F, 9C9D,F5 ;15480          MEM.REQ[READ.TB] ADRS[ZERO] DT[LONG]    ; запрос на чтение TB по адресу 0
U 1550, 3660,15 ;15481          MOV LS[INTERUPT.EN] TO WR[0]            ; установка в WR0 бита 16
U 1551, C77A,15 ;15482          BIS LS[UBS.DCLO] TO WR[0]                ; запрос на выдачу из консоли DCLO
```

```
U 1552, C76E,15 ;15483          BIS LS[NA.EXP.REC] TO WR[0]             ; данные под EXP (ожидаемые) и REC (полученные) не
                ;15484                                                  ; печатаются
U 1553, 3E80,15 ;15485          MOV WR[0] TO LS[CONTROL.STATUS]         ; подготовка слова управления и состояния
U 1554, 10E0,15 ;15486          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;15487  WAIT.T29.5:
U 1555, 0955,54 ;15488          JMP [WAIT.T29.5]                        ; цикл для ожидания ответа конс.процессора
U 1556, 0A1B,4C ;15489          JSR [NOP.DELAY]                         ; выполнение задержки на 5 циклов, что позволяет
                ;15490                                                  ; установиться состоянию шины MC
U 1557, 2F82,95 ;15491          CLR WR[1]                               ; занесение ложных ожидаемых данных
U 1558, 3022,15 ;15492          MOV MEM.DATA TO WR[0]                   ; выборка данных после инициализации
                ;15493          CMP LS[#55555555] WITH WR[0],           ; проверка, что результат не
                ;15494                                                  ; совпадает с данными переданными в ТВ. Сигнал DCLO
                ;15495                                                  ; должен был это предотвратить
U 1559, CE9A,35 ;15496          DI(LONG)&SET.ALU.CC                     ; установка кодов условий
U 155A, 8955,D1 ;15497          JMP.IF[NEQ] TO [TEST.T29.4]             ; переход, если не ошибка
U 155B, 0869,3C ;15498          JSR [CHECK.RESULT]                      ; иначе печать сообщения об ошибке
U 155C, 8954,94 ;15499          JMP [LOOP.T29.3.4]                      ; цикл при ошибке, если есть разрешение
                ;15500  TEST.T29.4:
U 155D, 3660,15 ;15501          MOV LS[INTERUPT.EN] TO WR[0]            ; установка в WR0 бита 16
U 155E, 3E80,15 ;15502          MOV WR[0] TO LS[CONTROL.STATUS]         ; запрос на выключение DCLO (и любых других прерываний,
                ;15503                                                  ; если установлены)
U 155F, 10E0,15 ;15504          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;15505  WAIT.T29.6:
U 1560, 0956,04 ;15506          JMP [WAIT.T29.6]                        ; цикл для ожидания ответа конс.процессора
U 1561, FF82,15 ;15507          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 1562, DF2A,15 ;15508          MCOM LS[#FF000000] TO WR[0]             ; занесение в WR0 значения 00FFFFFF
U 1563, 477E,15 ;15509          BIS LS[BIT31] TO WR[0]                  ; установка бита 31
U 1564, C77C,15 ;15510          BIS LS[BIT30] TO WR[0]                  ; установка бита 30
U 1565, C770,15 ;15511          BIS LS[BIT24] TO WR[0]                  ; установка бита 24
U 1566, 3E8A,15 ;15512          MOV WR[0] TO LS[ERROR.MASK]             ; проверка битов 29-24
U 1567, 2F82,95 ;15513          CLR WR[1]                               ; ожидаемые данные
U 1568, 9D45,75 ;15514          MEM.REQ[READ.CSR] ADRS[CSR1] DTI[LONG]  ; запрос на чтение CSR
U 1569, 3022,15 ;15515          MOV MEM.DATA TO WR[0]                   ; выборка данных CSR1
U 156A, 0869,3C ;15516          JSR [CHECK.RESULT]                      ; проверка, что биты 29-24 очищены
U 156B, 8954,94 ;15517          JMP [LOOP.T29.3.4]                      ; цикл при ошибке, если есть разрешение
                ;15518  END.T29:
                ;15519
```

```
;15520  .PAGE "ТЕСТЫ ПРОВЕРКИ МИКРОПРОГРАММ"
;15521  .TOC  "ТЕСТ 2A - проверка микропрограмм чтения из модуля памяти (модуль MCT)"
;15522  ;
;15523  ; ОПИСАНИЕ ТЕСТА:
;15524  ;
;15525  ;       Этот тест предназначен для обнаружения неисправностей в управляющих ПЗУ
;15526  ;   проверкой всех путей в микрокодах. Этот тест проверяет микропрограммы READ
;15527  ;   ARRAY при задержке ответа CPU DR (запрос данных из центрального процессора)
;15528  ;   и при 2 циклах чтения. Ни одна из этих микропрограмм ранее не использовалась.
;15529  ;       Микропрограмма с задержанным CPU DR проверяется путем выполнения READ.P с
;15530  ;   нерабочими циклами после MEM.REQ для задержки CPU DATA REQ.Это вынуждает мик-
;15531  ;   рокод в цикле CPU RD V CONT C7 выбрать путь CPU DR NOT*ERROR SUM NOT. При этом
;15532  ;   также в следующем цикле выбирается ветвление CPU DR NOT*ERR NOT.
;15533  ;       Микропрограмма двух циклов чтения проверяется путем выполнения чтения длин-
;15534  ;   ного слова по физическому адресу 2. При этом происходит выборка последних 2
;15535  ;   байтов длинного слова 0 и первых двух байтов по адресу 4.
;15536  ;
;15537  ; ПРЕДПОЛОЖЕНИЯ:
;15538  ;
;15539  ;   Принимается, что все предыдущие тесты выполнены успешно, и что вся аппаратура
;15540  ;   (кроме ПЗУ управляющей памяти MCT) работает правильно.
;15541  ;
;15542  ; ШАГИ ТЕСТА:
;15543  ;
;15544  ;     1)Занесение в LS маски ошибок, номера ошибки и номера модуля и очистка в LS
;15545  ;       номера предыдущей ошибки. также очистка CSR1, чтобы не было задано никакого
;15546  ;       диагностического режима.
;15547  ;     2)Запись данных FFFF0000 по адресу 0 и 0000FFFF по адресу 4.
;15548  ;     3)Выполнение MEM, REQ[READ.P] по адресу 0 при DT=длинное слово. Выполнение хо-
;15549  ;       лостых циклов для задержки CPU DR, затем выдача инструкции MOV MEM.DATA TO
;15550  ;       WR[0] и проверка данных. Повторение при адресе 4.
;15551  ;     4)Выполнение MEM.REQ[READ.P] по адресу 2 при DT=длинное слово. Проверка резуль-
;15552  ;       тата на все единицы.
;15553  ;
;15554  ; ОШИБКИ:
;15555  ;
;15556  ;   ПРИМЕЧАНИЕ: ожидаемыми и полученными данными являются данные модуля памяти.
;15557  ;
;15558  ;  ошибка 1 - неправильно работает микропрограмма чтения из модуля памяти (READ
;15559  ;             ARRAY) при адресе 0 и задержанном CPU DR (данные FFFF0000).
;15560  ;  ошибка 2 - неправильно работает микропрограмма чтения из модуля памяти (READ
;15561  ;             ARRAY) при адресе 4 и задержанном CPU DR (данные 0000FFFF).
;15562  ;  ошибка 3 - неправильно работает микропрограмма чтения двумя циклами из модуля
;15563  ;             памяти (TWO CYCLE READ ARRAY)при адресе 2.
;15564  ;
;15565  ; НАЛАДКА:
;15566  ;
;15567  ;       ОШИБКА 1,2 и 3 - аппаратура, связанная с этими операциями чтения, уже испы-
;15568  ;   тывалась ранее. Если происходит ошибка, можно подозревать микросхемы ПЗУ
;15569  ;   управляющей памяти. Лучшим методом обнаружения ошибки является пошаговая
;15570  ;   работа MCT и проверка, что выбраны правильные пути. Если это так, то можно
;15571  ;   проверять ожидаемый результат на каждом шаге.
;15572  ;--
;15573
;15574   T.2A:
```

```
U 156C, B65E,15 ;15575          MOV LS[BEGIN.TEST] TO WR[0]          ; установка в WR0 бита 15 для слова управления и
                ;15576                                               ; состояния
U 156D, 3EB0,15 ;15577          MOV WR[0] TO LS[CONTROL.STATUS]       ; установка бита 15 в слове упр. и состояния. Бит 15
                ;15578                                               ; указывает для консольного процессора начало теста
U 156E, 10E0,15 ;15579          MISC [SET.CP.ATTN]                   ; выдача для консольного процессора CPU ATTN
                ;15580  WAIT.T2A.0:
U 156F, 0956,F4 ;15581          JMP [WAIT.T2A.0]                     ; цикл для ожидания ответа консольного процессора
U 1570, 0A1A,9C ;15582          JSR [SETUP]                          ; установка маски, кода модуля и т.д. и очистка CSR1
U 1571, 5F28,15 ;15583          MCOM LS[#FFFF] TO WR[0]              ; занесение в рабочий регистр кода FFFF0000
U 1572, BE12,15 ;15584          MOV WR[0] TO LS[T9]                  ; запоминание данных в LS
U 1573, 9990,75 ;15585          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]   ; запрос для записи по адресу 0
U 1574, B212,15 ;15586          WRITE.MEM LS[T9]                     ; запись в память FFFF0000
U 1575, 9944,75 ;15587          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]   ; запрос для записи по адресу 4
U 1576, B228,15 ;15588          WRITE.MEM LS[#FFFF]                  ; запись в память 0000FFFF
                ;15589  LOOP.T2A.1:
U 1577, 189D,F5 ;15590          MEM.REQ[READ.P] ADRS[#0] DT[LONG]    ; запрос для чтения по адресу 0
U 1578, 0A1B,4C ;15591          JSR [NOP.DELAY]                      ; выполнение задержки на 5 циклов для задержки CPU
                ;15592                                               ; DR, что позволяет выбрать новый путь в микрокодах
U 1579, DF28,95 ;15593          MCOM LS[#FFFF] TO WR[1]              ; ожидаемые данные (FFFF0000)
U 157A, 3022,15 ;15594          MOV MEM.DATA TO WR[0]                ; выборка результата из памяти
U 157B, 0869,3C ;15595          JSR [CHECK.RESULT]                   ; проверка наличия ошибок
U 157C, 0957,74 ;15596          JMP [LOOP.T2A.1]                     ; цикл при ошибке, если есть разрешение
U 157D, FF82,15 ;15597          INC LS[ERROR.NUMBER]                 ; ошибка 2
                ;15598  LOOP.T2A.2:
U 157E, 1845,F5 ;15599          MEM.REQ[READ.P] ADRS[#4] DT[LONG]    ; запрос для чтения по адресу 4
U 157F, 0A1B,4C ;15600          JSR [NOP.DELAY]                      ; выполнение 5 циклов задержки для задержки CPU DR, что
                ;15601                                               ; позволяет выбрать новый путь в микрокодах
U 1580, B628,95 ;15602          MOV LS[#FFFF] TO WR[1]               ; ожидаемые данные
U 1581, 3022,15 ;15603          MOV MEM.DATA TO WR[0]                ; выборка результата из памяти
U 1582, 0869,3C ;15604          JSR [CHECK.RESULT]                   ; проверка на ошибки
U 1583, 0957,E4 ;15605          JMP [LOOP.T2A.2]                     ; цикл при ошибке, если есть разрешение
U 1584, FF82,15 ;15606          INC LS[ERROR.NUMBER]                 ; ошибка 3
                ;15607  LOOP.T2A.3:
U 1585, 1843,F5 ;15608          MEM.REQ[READ.P] ADRS[#2] DT[LONG]    ; запрос для 2 циклов чтения, начинающихся адресом 2
U 1586, 369E,95 ;15609          MOV LS[ONES] TO WR[1]                ; ожидаемые данные
U 1587, 3022,15 ;15610          MOV MEM.DATA TO WR[0]                ; выборка результата
U 1588, 0869,3C ;15611          JSR [CHECK.RESULT]                   ; проверка на ошибки
U 1589, 8958,54 ;15612          JMP [LOOP.T2A.3]                     ; цикл при ошибке, если есть разрешение
                ;15613  END.T2A:
```

```
;15614  .PAGE "тест 2B - тест микропрограммы чтения из памяти при CRD/RDS (модуль MCT)"
;15615  ;
;15616  ; ОПИСАНИЕ ТЕСТА:
;15617  ;
;15618  ;   Этот тест предназначен для обнаружения неисправностей в ПЗУ управляющей па-
;15619  ; мяти методом проверки всех путей в микрокодах. Этот тест проверяет путь, кото-
;15620  ; рый выбирается, если в памяти обнаружена ошибка в одиночном бите или двойная.
;15621  ; Этот тест записывает в памяти ошибку в одиночном бите или двойную ошибку с ис-
;15622  ; пользованием программы MAINT.ECC.DATA. Записываются контрольные биты для всех
;15623  ; 0 с данными 10000(H) или 30000(H), что дает в результате одиночную или двойную ошиб-
;15624  ; ку соответственно. Затем выполняется чтение этой ячейки. Для ошибки в одиночном
;15625  ; бите данные должны корректироваться до всех 0 и полученные биты синдрома долж-
;15626  ; ны стробироваться в CSR0. В CSR1 должен быть установлен бит CRD (исправимая
;15627  ; ошибка данных), а все другие биты ошибок должны быть очищены. При двойной ошиб-
;15628  ; ке данные не должны измениться, в CSR0 должны быть контрольные биты, а в CSR1
;15629  ; должен быть установлен бит RDS (двойная ошибка) и остальные биты ошибок очище-
;15630  ; ны. Этот тест выполняется дважды, один раз без задержки между MEM.REQ[READ.P]
;15631  ; и MOV MEM.DATA TO WR[0], а другой раз с холостыми циклами между этими двумя инст-
;15632  ; рукциями. Этим будут проверяться разные пути, выбираемые в зависимости от то-
;15633  ; го, имеется CPU DR или задержан. Также проверяется чтение двумя циклами с оди-
;15634  ; ночной или двойной ошибкой (но без задержки, так как в отношении CPU DR исполь-
;15635  ; DR используется тот же путь, как и при чтении в один цикл).
;15636  ;
;15637  ; ПРЕДПОЛОЖЕНИЯ:
;15638  ;
;15639  ;   Принимается, что все предыдущие тесты выполнены успешно и что вся аппаратура
;15640  ; (кроме ПЗУ управляющей памяти MCT) работает правильно.
;15641  ;
;15642  ; ШАГИ ТЕСТА:
;15643  ;
;15644  ;    1)Занесение в LS маски ошибок, номера ошибки и номера модуля (для распечатки
;15645  ;      ошибок) и очистка в LS номера предыдущей ошибки.
;15646  ;    2)Запись всех единиц по адресу 4.
;15647  ;    3)Загрузка в CSR1 контрольных битов 0111100(B) (перед записью в CSR1 они
;15648  ;      должны инвертироваться). Эти контрольные биты соответствуют данным из всех
;15649  ;      0. Очистка других битов CSR1.
;15650  ;    4)Выдача MEM.REQ с MF=MAINT.ECC.DATA и DT=длинное слово. Адрес 1 (LVA00 уста-
;15651  ;      новлен) используется для записи по адресу 0 (LVA00 используется в качестве
;15652  ;      условия ветвления в микропрограмму MAINT.ECC.DATA). Затем выдается WRITE.
;15653  ;      MEM LS с данными 10000(H) (единица в бите 0 третьего байта).
;15654  ;    5)Чтение ячейки 0 с использованием микопрограммы READ.P. Проверка данных на
;15655  ;      все нули (откорректированные данные).
;15656  ;    6)Замена маски ошибок и проверка CSR1 на бит CRD установленный, остальные
;15657  ;      очищенные.
;15658  ;    7)Замена маски ошибок и проверка CSR0 на правильные биты синдрома.
;15659  ;    8)Повторение шагов от 5 до 7 с задержкой из холостых циклов между MEM.REQ и
;15660  ;      MOV MEM.DATA для задержки CPU DR. Этим будет выбран другой путь в микроко-
;15661  ;      дах.
;15662  ;    9)Чтение ячейки 2 с использованием микропрограммы READ.P (чтение двумя цик-
;15663  ;      лами). Проверка данных на FFFF0000 (откорректированные данные). Затем пов-
;15664  ;      торение шагов 6 и 7.
;15665  ;   10)Повторение вышеперечисленных шагов от 3 до 8, за исключением записи по ад-
;15666  ;      ресу данных 30000(H) (двойная ошибка) и проверки на отсутствие коррекции
;15667  ;      данных, установленный бит RDS и контрольные биты (вместо синдрома) в CSR0.
;15668  ;   11)Чтение ячейки 2 с использованием микропрограммы READ.P (чтение двумя цикла-
```

```
;15669  ;     ми). Проверка данных на 3(H) (без коррекции). Затем проверка CSR1 на уста-
;15670  ;     новленный бит RDS и на наличие в CSR0 контрольных битов.
;15671  ;
;15672  ; ОШИБКИ:
;15673  ;
;15674  ;   ошибка 1 - не откорректирована правильно одиночная ошибка. Ожидаемыми и
;15675  ;              полученными данными является содержимое модуля памяти.
;15676  ;   ошибка 2 - одиночная ошибка не установила CRD или установила другие би-
;15677  ;              ты в CSR1. Ожидаемыми и полученными данными являются биты CSR1 14-23
;15678  ;              и 25-31.
;15679  ;   ошибка 3 - одиночная ошибка не установила в CSR0 правильных битов синдро-
;15680  ;              ма. Ожидаеммыми данными являются биты CSR0 0-6.
;15681  ;   ошибка 4,5 или 6 - то же, что и ошибки 1,2 или 3, за исключением задержанно-
;15682  ;              го CPU DR.
;15683  ;   ошибка 7,8 или 9 - то же, что и ошибки 1,2 или 3, за исключеием чтения двумя
;15684  ;              циклами по адресу 2.
;15685  ;   ошибка A - модифицированы данные в памяти при двойной ошибке. Ожидаемыми и по-
;15686  ;              лученными данными является содержимое модуля памяти.
;15687  ;   ошибка B - двойная ошибка не установила в CSR1 бита RDS или установила другие
;15688  ;              биты.Ожидаемыми и полученными данными являются биты CSR1 14-23 и
;15689  ;              25-31.
;15690  ;   ошибка C - двойная ошибка не установила правильных контрольных битов в CSR0.
;15691  ;              Ожидаемыми и полученными даными являются биты CSR0 0-6.
;15692  ;   ошибки D,E или F - то же, что и ошибки A,B или C, за исключением задержанного
;15693  ;              CPU DR.
;15694  ;   ошибки 10,11 или 12 - то же, что и ошибки A,B или C, за исключением чтения
;15695  ;              двумя циклами по адресу 2.
;15696  ;
;15697  ; НАЛАДКА:
;15698  ;
;15699  ;   ПРИМЕЧАНИЕ: следующие ошибки с наибольшей вероятностью указывают на отказ в
;15700  ;   микропрограмме. Аппаратура, используемая микрокодами, ранее проверялась.
;15701  ;   Для обнаружения отказа необходимо проверить последовательности, указанные
;15702  ;   ниже. Если последовательность правильная, можно подозревать неправильный бит
;15703  ;   в одном из ПЗУ управляющей памяти и необходимо проверить каждое состояие
;15704  ;   на правильные сигналы разрешения.
;15705  ;
;15706  ;   ОШИБКИ ОТ 1 ДО 3 - Указывают ошибку в пути микрокодов, проходящем из CPU RD
;15707  ;   V CONT C7 через C8A, через ERR C9 K SERR C10.
;15708  ;
;15709  ;   ОШИБКИ ОТ 4 ДО 6 - Указывают ошибку в пути микрокодов, проходящем из CPU RD V
;15710  ;   CONT C7 через C8A, через ERR C9B K SERR C10.
;15711  ;
;15712  ;   ОШИБКИ ОТ 7 ДО 9 - Указывают ошибку в пути микрокодов, проходящем из ARY CYC
;15713  ;   C4 через 2 ARY CYC C5, через 2 ARY CYC RD C6 K 2 ARY CYC RD C7 и затем по пу-
;15714  ;   ти для одиночной ошибки.
;15715  ;
;15716  ;   ОШИБКИ ОТ A ДО C - Указывают ошибку в пути микрокодов, проходящем из CPU RD V
;15717  ;   C7 через C8A K ERR C9 и затем по пути SERR NOT (отсутствие одиночной ошиб-
;15718  ;   ки).
;15719  ;
;15720  ;   ОШИБКИ ОТ D ДО F - То же, что и при ошибках от 4 до 6, за исключением того,
;15721  ;   что выбирается путь SERR NOT вместо перехода к SERR C10.
;15722  ;
;15723  ;   ОШИБКИ ОТ 10 ДО 12 - Указывают ошибку в пути микрокодов, проходящем из ARY
```

```
                ;15724  ;    CYC C4 через 2 ARY CYC C5, через 2 ARY CYC RD C6, через 2 ARY RD C7 и затем
                ;15725  ;    по пути отсутствия одиночной ошибки.
                ;15726
                ;15727  T.2B:
U 158A, B65E,15 ;15728          MOV LS[BEGIN.TEST] TO WR[0]                     ; установка в WR0 бита 15 для слова управления и
                ;15729                                                          ; состояния
U 158B, 3E80,15 ;15730          MOV WR[0] TO LS[CONTROL.STATUS]                 ; установка бита 15 в слове управления и состояния. Бит
                ;15731                                                          ; 15 указывает для консольного процессора начало тест
U 158C, 10E0,15 ;15732          MISC [SET.CP.ATTN]                              ; выдача для консольного процессора CPU ATTN
                ;15733  WAIT.T2B.0:
U 158D, 0958,D4 ;15734          JMP [WAIT.T2B.0]                                ; цикл для ожидания ответа консольного процессора
U 158E, 0A1A,9C ;15735          JSR [SETUP]                                     ; установка маски, кода модуля и т.д. и очистка CSR 1
                ;15736                                                          ; MCT
U 158F, B713,95 ;15737          MOV LS[#FFFFFF80]  TO WR[3]                     ; установка битов 7-31
U 1590, B62F,15 ;15738          MOV LS[CSR1.MASK] TO WR[2]                      ; маска ошибки для ошибки 2. Не проверяются биты 0-13 и
                ;15739                                                          ; бит 24
U 1591, B648,95 ;15740          MOV LS[#10] TO WR[1]                            ; установка бита 4
U 1592, 474A,95 ;15741          BIS LS[#20] TO WR[1]                            ; и бита 5
U 1593, 474C,95 ;15742          BIS LS[#40] TO WR[1]                            ; и бита 6(1110000(B))
U 1594, A0C2,95 ;15743          COM WR[1]                                       ; ожидаемый синдром
U 1595, BE10,95 ;15744          MOV WR[1] TO LS[TB]                             ; запоминание в промежуточной ячейке LS
U 1596, 9944,75 ;15745          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]              ; запрос для записи по адресу 4
U 1597, 329E,15 ;15746          WRITE.MEM LS[ONES]                              ; запись в память всех единиц
U 1598, B700,15 ;15747          MOV LS[CKBTS.01111100] TO WR[0]                 ; выборка в WR0 контрольных разрядов для нулевых данных
U 1599, A0C0,15 ;15748          COM WR[0]                                       ; инвертирование контрольных разрядов для согласования с
                ;15749                                                          ; аппаратурой
U 159A, 452C,15 ;15750          BIC LS[#FFFFFF00] TO WR[0]                      ; очистка всех байтов, кроме младшего
U 159B, 8A17,FC ;15751          JSR [WRITE.CSR1]                                ; занесение в CSR1 контрольных разрядов для нулевых
                ;15752                                                          ; данных. очистка в CSR всех других битов
U 159C, 1D40,F5 ;15753          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG]       ; запрос для использования диагностической программы
                ;15754                                                          ; LVA 00 установлен для ветвления
U 159D, B260,15 ;15755          WRITE.MEM LS[#10000]                            ; запись 10000 в ячейку 0 с контрольными битами для
                ;15756                                                          ; данных, состоящих из нулей
                ;15757  LOOP.T2B.2.3:
U 159E, B640,15 ;15758          MOV LS[#1] TO WR[0]                             ; занесение 1 в рабочий регистр
U 15AF, BE82,15 ;15759          MOV WR[0] TO LS[ERROR.NUMBER]                   ; занесение в LS номера ошибки
U 15A0, E58A,15 ;15760          CLR LS[ERROR.MASK]                              ; проверка всех битов
                ;15761  LOOP.T2B.1:
U 15A1, 2F82,95 ;15762          CLR WR[1]                                       ; ожидаемые данные
U 15A2, 189D,F5 ;15763          MEM.REQ[READ.P] ADRS[#0] DT[LONG]               ; запрос для чтения по адресу памяти 0
U 15A3, 3022,15 ;15764          MOV MEM.DATA TO WR[0]                           ; выборка результата
U 15A4, 0869,3C ;15765          JSR [CHECK.RESULT]                              ; проверка данных на 0
U 15A5, 895A,14 ;15766          JMP [LOOP.T2B.1]                                ; зацикливание при ошибке, если есть разрешение
U 15A6, 095E,9C ;15767          JSR [CHECK.CSRS.SINGLE]                         ; проверка CSR1 и CSR0 (ошибки 2 и 3)
U 15A7, 8959,E4 ;15768          JMP [LOOP.T2B.2.3]                              ; зацикливание при ошибке, если есть разрешение
                ;15769  LOOP.T2B.5.6:
U 15A8, 3644,15 ;15770          MOV LS[#4] TO WR[0]                             ; значение 4 в рабочем регистре
U 15A9, BE82,15 ;15771          MOV WR[0] TO LS[ERROR.NUMBER]                   ; значение в LS номера ошибки (ошибка 4)
U 15AA, E58A,15 ;15772          CLR LS[ERROR.MASK]                              ; проверка всех битов
                ;15773  LOOP.T2B.4:
U 15AB, 2F82,95 ;15774          CLR WR[1]                                       ; ожидаемые данные
U 15AC, 189D,F5 ;15775          MEM.REQ[READ.P] ADRS[#0] DT[LONG]               ; запрос для чтения адреса памяти 0
U 15AD, 0A1B,4C ;15776          JSR [NOP.DELAY]                                 ; выполнение задержки на 5 циклов для задержания сигнала
                ;15777                                                          ; CPU DR
U 15AE, DB00,15 ;15778          NOP                                             ; задержка еще на один цикл
```

```
U 15AF, 3022,15 ;15779          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15B0, 0869,3C ;15780          JSR [CHECK.RESULT]                      ; проверка нулевых данных
U 15B1, 895A,B4 ;15781          JMP [LOOP.T2B.4]                        ; зацикливание при ошибке, если есть разрешение
U 15B2, 095E,9C ;15782          JSR [CHECK.CSRS.SINGLE]                 ; проверка CSR1 и CSR0 (ошибки 5 и 6)
U 15B3, 895A,B4 ;15783          JMP [LOOP.T2B.5.6]                      ; зацикливание при ошибке, если есть разрешение
                ;15784  LOOP.T2B.8.9:
U 15B4, B646,15 ;15785          MOV LS[#8] TO WR[0]                     ; значение 8 в рабочем регистре
U 15B5, 2100,15 ;15786          DEC WR[0]                               ; ошибка 7
U 15B6, BE82,15 ;15787          MOV WR[0] TO LS[ERROR.NUMBER]           ; запоминание номера ошибки в LS
U 15B7, E58A,15 ;15788          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;15789  LOOP.T2B.7:
U 15B8, DF28,95 ;15790          MCOM LS[#FFFF] TO WR[1]                 ; ожидаемые данные (FFFF0000)
U 15B9, 1843,F5 ;15791          MEM.REQ[READ.P] ADRS[#2] DT[LONG]       ; запрос для чтения из памяти по адресу 2 (чтение двумя
                ;15792                                                  ; циклами)
U 15BA, 3022,15 ;15793          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15BB, 0869,3C ;15794          JSR [CHECK.RESULT]                      ; проверка, что данные FFFF0000
U 15BC, 095B,B4 ;15795          JMP [LOOP.T2B.7]                        ; зацикливание при ошибке, если есть разрешение
U 15BD, 095E,9C ;15796          JSR [CHECK.CSRS.SINGLE]                 ; проверка CSR1 и CSR0 (ошибки 8 и 9)
U 15BE, 095B,44 ;15797          JMP [LOOP.T2B.8.9]                      ; зацикливание при ошибке, если есть разрешение
U 15BF, B700,15 ;15798          MOV LS[CKBTS.01111100] TO WR[0]         ; выборка контрольных битов для нулевых данных в WR0
U 15C0, A0C0,15 ;15799          COM WR[0]                               ; инвертирование контрольных битов для согласования с
                ;15800                                                  ; аппаратурой
U 15C1, 452C,15 ;15801          BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех байтов, кроме младшего
U 15C2, 8A17,FC ;15802          JSR [WRITE.CSR1]                        ; занесение контрольных битов для нулевых данных в
                ;15803                                                  ; CSR1, очистка в CSR всех других битов
U 15C3, 1D40,F5 ;15804          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической прогрммы.
                ;15805                                                  ; LVA00 установлен для ветвления
U 15C4, B334,15 ;15806          WRITE.MEM LS[#30000]                    ; запись значения 30000(H) в ячейку 0 с контрольными
                ;15807                                                  ; битами для данных из всех нулей
                ;15808  LOOP.T2B.B.C:
U 15C5, B646,15 ;15809          MOV LS[#8] TO WR[0]                     ; значение 8 в рабочем регистре
U 15C6, C042,15 ;15810          ADD LS[#2] TO WR[0]                     ; ошибка A
U 15C7, BE82,15 ;15811          MOV WR[0] TO LS[ERROR.NUMBER]           ; запоминание номера ошибки в LS
U 15C8, E58A,15 ;15812          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;15813  LOOP.T2B.A:
U 15C9, B734,95 ;15814          MOV LS[#30000] TO WR[1]                 ; ожидаемые данные
U 15CA, 189D,F5 ;15815          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения памяти адресу 0
U 15CB, 3022,15 ;15816          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15CC, 0869,3C ;15817          JSR [CHECK.RESULT]                      ; проверка, что данные не модифицированы
U 15CD, 095C,94 ;15818          JMP [LOOP.T2B.A]                        ; зацикливание при ошибке, если есть разрешение
U 15CE, 095F,8C ;15819          JSR [CHECK.CSRS.DOUBLE]                 ; проверка CSR1 и CSR0 (ошибки B и C)
U 15CF, 095C,54 ;15820          JMP [LOOP.T2B.B.C]                      ; зацикливание при ошибке, если есть разрешение
                ;15821  LOOP.T2B.E.F:
U 15D0, B646,15 ;15822          MOV LS[#8] TO WR[0]                     ; значение 8 в рабочем регистре
U 15D1, C044,15 ;15823          ADD LS[#4] TO WR[0]                     ; рабочий регистр содержит c
U 15D2, 2040,15 ;15824          INC WR[0]                               ; ошибка D
U 15D3, BE82,15 ;15825          MOV WR[0] TO LS[ERROR.NUMBER]           ; запоминание номера ошибки в LS
U 15D4, E58A,15 ;15826          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;15827  LOOP.T2B.D:
U 15D5, B734,95 ;15828          MOV LS[#30000] TO WR[1]                 ; ожидаемые данные (не модифицированные)
U 15D6, 189D,F5 ;15829          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос из памяти по адресу 0
U 15D7, 0A1B,4C ;15830          JSR [NOP.DELAY]                         ; выполнение задержки на 5 циклов для задержания CPU DR
U 15D8, DB00,15 ;15831          NOP                                     ; задержка еще на один цикл
U 15D9, 3022,15 ;15832          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15DA, 0869,3C ;15833          JSR [CHECK.RESULT]                      ; проверка, что данные равны 30000(H)
```

```
U 15DB, 895D,54 ;15834          JMP [LOOP.T2B.D]                        ; зацикливание при ошибке, если есть разрешение
U 15DC, 095F,8C ;15835          JSR [CHECK.CSRS.DOUBLE]                 ; проверка CSR1 и CSR0 (ошибки E и F)
U 15DD, 895D,04 ;15836          JMP [LOOP.T2B.E.F]                      ; зацикливание при ошибке, если есть разрешение
                ;15837  LOOP.T2B.11.12:
U 15DE, 3648,15 ;15838          MOV LS[#10] TO WR[0]                    ; ошибка 10
U 15DF, BE82,15 ;15839          MOV WR[0] TO LS[ERROR.NUMBER]           ; занесение в LS номера ошибки
U 15E0, E58A,15 ;15840          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;15841  LOOP.T2B.10:
U 15E1, B6C0,95 ;15842          MOV LS[#3(H)] TO WR[1]                  ; рабочий регистр содержит 3(H)
U 15E2, 1843,F5 ;15843          MEM.REQ[READ.P] ADRS[#2] DT[LONG]       ; запрос для чтения памяти по адресу 2 (чтение двумя
                ;15844                                                  ; циклами)
U 15E3, 3022,15 ;15845          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15E4, 0869,3C ;15846          JSR [CHECK.RESULT]                      ; проверка, что данные не модифицированы
U 15E5, 095E,14 ;15847          JMP [LOOP.T2B.10]                       ; зацикливание при ошибке, если есть разрешение
U 15E6, 095F,8C ;15848          JSR [CHECK.CSRS.DOUBLE]                 ; проверка CSR1 и CSR0 (ошибки 11 и 12)
U 15E7, 095D,E4 ;15849          JMP [LOOP.T2B.11.12]                    ; зацикливание при ошибке, если есть разрешение
U 15E8, 8960,84 ;15850          JMP [END.T2B]                           ; конец
                ;15851  CHECK.CSRS.SINGLE:
U 15E9, FF82,15 ;15852          INC LS[ERROR.NUMBER]                    ; следующая ошибка
U 15EA, BE8B,15 ;15853          MOV WR[2] TO LS[ERROR.MASK]             ; маска ошибки для проверки CSR1
U 15EB, 367C,95 ;15854          MOV LS[CRD] TO WR[1]                    ; ожидаемые данные (бит CRD установлен, остальные
                ;15855                                                  ; очищены)
U 15EC, 9D45,75 ;15856          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; чтение CSR1
U 15ED, 3022,15 ;15857          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15EE, 0869,3C ;15858          JSR [CHECK.RESULT]                      ; проверка, что установлен только бит CRD
U 15EF, 5B00,14 ;15859          RETURN                                  ; зацикливание при ошибке, если есть разрешение
U 15F0, FF82,15 ;15860          INC LS[ERROR.NUMBER]                    ; следующая ошибка
U 15F1, 3E8B,95 ;15861          MOV WR[3] TO LS[ERROR.MASK]             ; маска ошибки для проверки CSR0
U 15F2, 3610,95 ;15862          MOV LS[T8] TO WR[1]                     ; ожидаемый синдром (1110000(B))
U 15F3, 9D9D,75 ;15863          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ; чтение CSR0
U 15F4, 3022,15 ;15864          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15F5, 0869,3C ;15865          JSR [CHECK.RESULT]                      ; проверка, что синдром правильный
U 15F6, 5B00,14 ;15866          RETURN                                  ; зацикливание при ошибке, если есть разрешение
U 15F7, DB00,16 ;15867          RETURN+1                                ; возврат при отсутствии зацикливания по ошибке
                ;15868  CHECK.CSRS.DOUBLE:
U 15F8, FF82,15 ;15869          INC LS[ERROR.NUMBER]                    ; следующая ошибка
U 15F9, BE8B,15 ;15870          MOV WR[2] TO LS[ERROR.MASK]             ; маска ошибки для проверки CSR1
U 15FA, B67E,95 ;15871          MOV LS[RDS] TO WR[1]                    ; ожидаемые данные (бит RDS установлен, остальные
                ;15872                                                  ; очищены)
U 15FB, 9D45,75 ;15873          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; чтение CSR1
U 15FC, 3022,15 ;15874          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 15FD, 0869,3C ;15875          JSR [CHECK.RESULT]                      ; проверка, что установлен только бит RDS
U 15FE, 5B00,14 ;15876          RETURN                                  ; зацикливание при ошибке, если есть разрешение
U 15FF, FF82,15 ;15877          INC LS[ERROR.NUMBER]                    ; следующая ошибка
U 1600, 3E8B,95 ;15878          MOV WR[3] TO LS[ERROR.MASK]             ; маска ошибки для проверки CSR0
U 1601, 3700,95 ;15879          MOV LS[CKBTS.0111100] TO WR[1]          ; инверсные ожидаемые данные
U 1602, A0C2,95 ;15880          COM WR[1]                               ; ожидаемые данные (контрольные биты)
U 1603, 9D9D,75 ;15881          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ; чтение CSR0
U 1604, 3022,15 ;15882          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 1605, 0869,3C ;15883          JSR [CHECK.RESULT]                      ; проверка правильности контрольных битов
U 1606, 5B00,14 ;15884          RETURN                                  ; зацикливание при ошибке, если есть разрешение
U 1607, DB00,16 ;15885          RETURN+1                                ; возврат при отсутствии зацикливания по ошибке
                ;15886  END.T2B:
```

```
;15887  .PAGE "ТЕСТ 2C - тесты диагностической проверки и запрета коррекции (модуль MCT)"
;15888  ;
;15889  ;
;15890  ; ОПИСАНИЕ ТЕСТА:
;15891  ;
;15892  ;       Этот тест предназначен для обнаружения неисправностей в ПЗУ управляющей
;15893  ;    памяти методом проверки всех путей в микрокодах. Тест проверяет пути,
;15894  ;    выбираемые, когда разрешены один или оба режима DIAG CHK (диагностическая
;15895  ;    проверка) и ECC DIS (запрет коррекции ошибок) путем установки битов в CSR 1.
;15896  ;    Если возбужден режим DIAG CHK, в логических схемах коррекции ошибок исполь-
;15897  ;    зуются контрольные биты из битов 6-0 CSR1 вместо контрольных битов, считан-
;15898  ;    ных из памяти. Этот тест проверяет, что записанные в CSR1 контрольные биты
;15899  ;    для одиночной ошибки изменяют данные, считанные из правильной ячейки памяти
;15900  ;    (благодаря тому, что "КОРРЕКТИРУЕТСЯ" одиночная ошибка). Он также проверяет,
;15901  ;    что двойная ошибка устанавливает бит RDS без изменения данных, и что конт-
;15902  ;    рольные биты в CSR1, которые являются правильными для данных в памяти, ниче-
;15903  ;    го не изменяют и не вызывают индикации ошибки.
;15904  ;       Если возбужден ECC DIS, контрольные биты из памяти запоминаются в CSR 0.
;15905  ;    Если происходит одиночная или двойная ошибка, в CSR1 устанавливается бит
;15906  ;    RDS. При одиночной ошибке коррекция не выполняется. Этот тест проверяет
;15907  ;    контрольные биты в CSR0, бит CRD в CSR1 и данные для сучаев отсутствия
;15908  ;    ошибки и одиночной ошибки.
;15909  ;       Если установлены оба бита ECC DIS и DIAG CHK, в логических схемах коррекции
;15910  ;    ошибки используются контрольные биты из битов 6-0 CSR1, как и при DIAG CHK
;15911  ;    выше. Разница в том, что эти контрольные биты будут стробироваться в CSR0, а
;15912  ;    коррекция не произойдет. Если произойдет одиночная ошибка, в CSR1 будет ус-
;15913  ;    тановлен бит CRD. Этот тест проверяет запись в CSR0 контрольных битов из CSR1
;15914  ;    и бит CRD в CSR1.
;15915  ;
;15916  ; ПРЕДПОЛОЖЕНИЯ:
;15917  ;
;15918  ;    Принимается, что все предыдущие тесты выполнены успешно, и что вся аппара-
;15919  ;    тура (кроме ПЗУ управляющей памяти MCT) работает правильно.
;15920  ;
;15921  ; ШАГИ ТЕСТА:
;15922  ;
;15923  ;    1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для распе-
;15924  ;       чатки ошибок) и очистка в LS номера предыдущей ошибки. Также, очистка
;15925  ;       CSR1, чтобы не были разрешены никакие диагностические режимы.
;15926  ;    2) Запись 1 в ячейку памяти 0 и чтение для проверки данных.
;15927  ;    3) Запись в CSR1 инверсии контрольных битов для данных, состоящих из всех
;15928  ;       нулей (действительные контрольные биты равны 0111100) и установка бита
;15929  ;       режима диагностической проверки.
;15930  ;    4) Чтение ячейки памяти 0, проверка откорректированных данных на все нули.
;15931  ;       Этим проверяется путь микропрограммы, который ответвляется при диагности-
;15932  ;       ческой проверке.
;15933  ;    5) Запись в CSR1 контрольных битов для данных, равных 1. Также поддерживает-
;15934  ;       ся установленным бит диагностической проверки.
;15935  ;    6) Чтение ячейки памяти 0 с данными, равными 1. Также проверяется CSR1, что
;15936  ;       биты ошибок не установлены. Этим проверяется путь в микропрограмме, на
;15937  ;       который она ответвляется при диагностической проверке, если нет ошибки
;15938  ;       данных.
;15939  ;    7) Запись данных, равных 2, в ячейку памяти 0. Чтение при установленном режи-
;15940  ;       ме диагностической проверки и при CSR1, содержащем контрольные биты для
;15941  ;       данных, равных 1. Данные должны быть возвращены без коррекции (ошибка в
```

```
;15942  ;       двух битах).
;15943  ;    8) Чтение CSR1 и проверка, что бит RDS установлен. Этим проверяется путь
;15944  ;       в микропрограмме, который ответвляется по биту диагностической проверки
;15945  ;       и снова ответвляется по SERR (одиночная ошибка).
;15946  ;    9) Запись данных, состоящих из нулей, в ячейку памяти 0. Установка в CSR1
;15947  ;       бита режима ECC DISABLE. Чтение ячейки 0 и проверка, что данные содержат
;15948  ;       нули.
;15949  ;   10) Чтение CSR0 для получения инверсного значения контрольных битов данных,
;15950  ;       состоящих из нулей. Действительные контрольные биты составляют 0111100.
;15951  ;       Этим проверяется ветвление по ECC DIS в микропрограмме READ ARRAY.
;15952  ;   11) Чтение CSR1 и проверка, что все биты ошибок очищены.
;15953  ;   12) Запись в память данных, состоящих из нулей с контрольными битами для данных,
;15954  ;       равных 1, с использованием программы ECC MAINT.
;15955  ;   13) Чтение ячейки 0 с установленным в CSR1 битом ECC DISABLE. Проверка, что
;15956  ;       коррекция не происходит (считанные данные содержат все нули).
;15957  ;   14) Чтение CSR0 для получения инвертированных контрольных битов для данных,
;15958  ;       равных 1. Действительные контрольные биты должны быть 1100100 (B).
;15959  ;   15) Чтение CSR1 и проверка, что установлен бит ошибки CRD. Этим проверяется
;15960  ;       ветвление по ERR в микропрограмме ECC DISABLE.
;15961  ;   16) Запись в ячейку памяти 0 данных, равных 1. Запись CSR1 контрольными бита-
;15962  ;       ми для данных, равных 0, и установка обоих битов ECC DIS и DIAG CHK.
;15963  ;   17) Чтение ячейки памяти 0 и проверка, что данные не откорректированы (равны 1).
;15964  ;   18) Проверка, что CSR0 содержит инвертированные контрольные биты для данных,
;15965  ;       равных 0 (действительными контрольными битами являются 0111100 (B)).
;15966  ;       Этим проверяется ветвление, когда оба сигнала ECC DIS и DIAG CHK установ-
;15967  ;       лены.
;15968  ;   19) Чтение CSR1 и проверка, что бит CRD установлен. Этим проверяется ветвле-
;15969  ;       ние по ERR в микропрограмме.
;15970  ;   20) Запись в CSR1 инвертированных контрольных битов для данных, равных 1, и
;15971  ;       установка битов ECC DIS и DIAG CHK.
;15972  ;   21) Чтение ячейки памяти 0, затем чтение CSR 0 и проверка, что контрольные
;15973  ;       биты занесены для данных, равных 1 (действительными контрольными битами
;15974  ;       являются 1100100 (B)).
;15975  ;   22) Чтение CSR1 и проверка, что никакие биты ошибок не установлены. Этим про-
;15976  ;       веряется ветвление в микропрограмме по ERR NOT.
;15977  ;
;15978  ; ОШИБКИ:
;15979  ;
;15980  ;   ПРИМЕЧАНИЕ: данные под "ОЖИДАЕМЫЕ" (EXP) и "ПОЛУЧЕННЫЕ" (REC) в сообщении об
;15981  ;   ошибке представляют собой данные памяти, если маска ошибок содержит все нули.
;15982  ;   Данные представляют собой содержимое CSR1, если маска ошибок равна
;15983  ;   07003FFF(H). Данные представляют собой содержимое CSR0, если маска ошибок
;15984  ;   равна FFFFFF80(H).
;15985  ;
;15986  ;   ошибка 1 - отказ при записи 1 в память, чтении 1 из памяти.
;15987  ;   ошибка 2 - отказ микропрограммы режима диагностической проверки в использо-
;15988  ;              вании контрольных битов из CSR1 и выполнении коррекции данных.
;15989  ;   ошибка 3 - микропрограмма режима диагностической проверки изменила считан-
;15990  ;              ные данные, когда CSR1 содержал правильные контрольные биты.
;15991  ;   ошибка 4 - микропрограмма режима диагностической проверки установила в CSR1
;15992  ;              биты ошибок, когда не должны были произойти ошибки.
;15993  ;   ошибка 5 - микропрограмма диагностической проверки изменила считанные данные
;15994  ;              при двойной ошибке.
;15995  ;   ошибка 6 - микропрограмма диагностической проверки не установила в CSR1 бита
;15996  ;              RDS при двойной ошибке или установила в CSR1 биты ошибок.
```

```
;15997  ;    ошибка 7 - отказ режима запрета коррекции ошибок.
;15998  ;               данные в памяти изменены.
;15999  ;    ошибка 8 - в режиме запрета коррекции ошибок не загружены в CSR0 контрольные
;16000  ;               биты памяти.
;16001  ;    ошибка 9 - режим запрета коррекции ошибок установил в CSR1 индикацию ошибок
;16002  ;               при несуществующей ошибке.
;16003  ;    ошибка A - режим запрета коррекции ошибок разрешил коррекцию одиночной ошиб-
;16004  ;               ки.
;16005  ;    ошибка B - в режиме запрета коррекции ошибок не загружены в CSR0 контрольные
;16006  ;               биты памяти.
;16007  ;    ошибка C - в режиме запрета коррекции ошибок не установлена ошибка CRD при
;16008  ;               обнаружении одиночной ошибки.
;16009  ;    ошибка D - в режиме запрета коррекции ошибок и диагностической проверки изме-
;16010  ;               нены данные в памяти при одиночной ошибке.
;16011  ;    ошибка E - в режиме запрета коррекции ошибок и диагностической проверки не
;16012  ;               произошла правильная загрузка CSR0 из CSR1.
;16013  ;    ошибка F - в режиме запрета коррекции ошибок и диагностической проверки в
;16014  ;               CSR1 не установлен бит ошибки при одиночной ошибке.
;16015  ;    ошибка 10- в режиме запрета коррекции ошибок и диагностической проверки не
;16016  ;               произошла правильная загрузка CSR0 из CSR1 (в данных не содер-
;16017  ;               жится одиночной ошибки).
;16018  ;    ошибка 11- в режиме запрета коррекции ошибок и диагностической проверки в
;16019  ;               CSR1 установлена индикация ошибки при отсутствии ошибки.
;16020  ;
;16021  ; НАЛАДКА:
;16022  ;
;16023  ;    Следующие ошибки с наибольшей вероятностью указывают на отказ в потоке мик-
;16024  ;    рокодов. Аппаратура, используемая микрокодами, ранее проверялась. Чтобы ло-
;16025  ;    кализовать отказ, необходимо проверить последовательности, указанные ниже. Если
;16026  ;    последовательность правильная, подозревается дефектный бит в ПЗУ управляю-
;16027  ;    щей памяти и необходимо проверить каждое состояние на правильность разрешаю-
;16028  ;    щих сигналов.
;16029  ;
;16030  ;    ОШИБКА 1 - Эта ошибка указывает на отказ в обычной записи или чтении из па-
;16031  ;               мяти. Если отказ имеется, следует выполнить предыдущие тесты.
;16032  ;    ОШИБКА 2 - Эта ошибка указывает на отказ в ветви DIAG CHECK или ветвлениях
;16033  ;               по ошибкам данных в микропрограмме READ ARRAY. Правильной после-
;16034  ;               довательностью является из C6 через RD DIAG C7, через RD DIAG C7A, через
;16035  ;               RD DIAG C8 к слову, которое проверяет наличие ошибки, через
;16036  ;               RD DIAG C10 к SERR C10.
;16037  ;    ОШИБКА 3 - При этой ошибке последовательность микрокодов такая же, как и при
;16038  ;               ошибке 2 выше, но вместо прохождения через RD DIAG C10, микропро-
;16039  ;               грамма проходит через RD DIAG C11 и продолжается по ранее прове-
;16040  ;               ренному пути микрокодов.
;16041  ;    ОШИБКА 4 - То же, что и при ошибке 3. При этой ошибке проверялись данные
;16042  ;               CSR1.
;16043  ;    ОШИБКА 5 - При этой ошибке последовательность микропрограммы такая же, как и
;16044  ;               при ошибке 2 выше, но вместо прохождения через SERR 10, происхо-
;16045  ;               дит ветвление по отсутствию одиночной ошибки.
;16046  ;    ОШИБКА 6 - То же, что и при ошибке 5 выше, но проверяется содержимое CSR1.
;16047  ;    ОШИБКА 7 - Эта ошибка указывает на отказ в ветви при запрете коррекции оши-
;16048  ;               бок микропрограммы READ ARRAY. Правильным является путь от C6
;16049  ;               через RD ECC DIS C7, через RD ECC DIS CBA К RD DIAG C11.
;16050  ;    ОШИБКА 8 - То же, что и при ошибке 7.
;16051  ;    ОШИБКА 9 - То же, что и при ошибке 7 (подозревается ветвление из RD ECC DIS
```

```
                    ;16052  ;              C7. Здесь должен выбираться путь по отсутствию ошибки).
                    ;16053  ;   ОШИБКА A - Эта ошибка указывает на отказ в ветви при запрете коррекции оши-
                    ;16054  ;              бок в микропрограмме READ ARRAY. Правильным является путь от
                    ;16055  ;              C6 через RD ECC DIS C7, через RD ECC DIS C8, через RD ECC DIS C8A
                    ;16056  ;              к RD DIAG C11.
                    ;16057  ;   ОШИБКА B - То же, что и при ошибке A (подозревается ветвление из RD ECC DIS
                    ;16058  ;              C7. Должен выбираться путь по ERR).
                    ;16059  ;   ОШИБКА C - То же, что и при ошибке A.
                    ;16060  ;   ОШИБКА D - Эта ошибка указывает на отказ в ветви потока микрокодов при за-
                    ;16061  ;              прете коррекции ошибок и при диагностической проверке микропрог-
                    ;16062  ;              раммы READ ARRAY. Правильным является путь от C6 через RD DIAG
                    ;16063  ;              ECC DIS C7 через RD DIAG ECC DIS C8, через слово, которое осу-
                    ;16064  ;              ществляет ветвление по ERR К RD ECC DIS C8, через RD ECC DIS
                    ;16065  ;              C8A к RD DIAG C11.
                    ;16066  ;   ОШИБКА E - То же, что и при ошибке D (подозревается ветвление по ECC DIS и
                    ;16067  ;              DIAG CHK из C6).
                    ;16068  ;   ОШИБКА F - То же, что и при ошибке D (подозревается ветвление по ERR. Микро-
                    ;16069  ;              программа должна выбрать путь ERR).
                    ;16070  ;   ОШИБКА 10- Эта ошибка указывает на отказ в ветви потока микрокодов при запрете
                    ;16071  ;              корекции ошибок и при диагностической проверке микропрограммы
                    ;16072  ;              READ ARRAY. Правильным является путь от C6 через RD DIAG ECC DIS
                    ;16073  ;              C7 через RD DIAG  ECC DIS C8 через слово, которое осуществляет
                    ;16074  ;              ветвление по ERR на RD ECC DIS C8A и к RD DIAG C11.
                    ;16075  ;   ОШИБКА 11- То же, что и при ошибке 10 (подозревается ветвление по ERR. Микро-
                    ,16076  ;              программа должна выбрать путь по отсутствию ошибки).
                    ;16077   T.2C:
U 160B, B65E,15 ;16078           MOV LS[BEGIN.TEST] TO WR[0]                ; установка в WR0 бита 15 для слова управления и
                    ;16079                                                  ; состояния
U 1609, 3EB0,15 ;16080           MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                    ;16081                                                  ; 15 указывает для консольного процессора начало теста
U 160A, 10E0,15 ;16082           MISC [SET.CP.ATTN]                         ; выдача для консольного процессора CPU ATTN
                    ;16083   WAIT.T2C.0:
U 160B, 8960,B4 ;16084           JMP [WAIT.T2C.0]                           ; цикл для ожидания ответа консольного процессора
U 160C, 0A1A,9C ;16085           JSR [SETUP]                                ; установка маски, кода модуля и т.д. и очистка CSR1
                    ;16086                                                  ; МСТ
U 160D, B713,95 ;16087           MOV LS[#FFFFFFB0] TO WR[3]                 ; установка битов 7-31
U 160E, B62F,15 ;16088           MOV LS[CSR1.MASK] TO WR[2]                 ; маска ошибки для CSR1
U 160F, C775,15 ;16089           BIS LS[DIAG.CHK] TO WR[2]                  ; запрет проверки бита DIAG.CHK
U 1610, C773,15 ;16090           BIS LS[ECC.DIS] TO WR[2]                   ; запрет проверки бита ECC.DIS. Теперь не будут
                    ;16091                                                  ; проверяться биты 0-13 и 24-26
                    ;16092   LOOP.T2C.1:
U 1611, 999C,75 ;16093           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]         ; запрос для записи по адресу памяти 0
U 1612, 3240,15 ;16094           WRITE.MEM LS[#1]                           ; запись в память данных=1
U 1613, 189D,F5 ;16095           MEM.REQ[READ.P] ADRS[#0] DT[LONG]          ; запрос для чтения памяти по адресу 0
U 1614, 3022,15 ;16096           MOV MEM.DATA TO WR[0]                      ; чтение из адреса 0
U 1615, 3640,95 ;16097           MOV LS[#1] TO WR[1]                        ; ожидание результата
U 1616, 0869,3C ;16098           JSR [CHECK.RESULT]                         ; проверка, что по адресу памяти 0 содержится 1
U 1617, 0961,14 ;16099           JMP [LOOP.T2C.1]                           ; цикл при ошибке, если есть разрешение
U 1618, FF82,15 ;16100           INC LS[ERROR.NUMBER]                       ; ошибка 2
U 1619, B700,15 ;16101           MOV LS[CKBTS.0111100] TO WR[0]             ; выборка в WR0 контрольных битов для данных = 0
U 161A, A0C0,15 ;16102           COM WR[0]                                  ; инвертирование контрольных битов для согласования с
                    ;16103                                                  ; аппаратурой
U 161B, 452C,15 ;16104           BIC LS[#FFFFFF00] TO WR[0]                 ; очистка всех байтов, кроме младшего
U 161C, 4774,15 ;16105           BIS LS[DIAG.CHK] TO WR[0]                  ; установка бита DIAG CHK (диагностическая проверка)
U 161D, BA17,FC ;16106           JSR [WRITE.CSR1]                           ; запись в CSR1 контрольных битов и DIAG CHK
```

```
                    ;16107   LOOP.T2C.2:
U 161E, 189D,F5     ;16108           MEM.REQ[READ.P] ADRS[#0] DT[LONG]      ; запрос для чтения памяти по адресу 0
U 161F, 3022,15     ;16109           MOV MEM.DATA TO WR[0]                  ; чтение из адреса 0
U 1620, 2F82,95     ;16110           CLR WR[1]                              ; ожидаемый результат (откорректированные данные)
U 1621, 0869,3C     ;16111           JSR [CHECK.RESULT]                     ; проверка, что адрес памяти 0 содержит 0
U 1622, 0961,E4     ;16112           JMP [LOOP.T2C.2]                       ; цикл при ошибке, если есть разрешение
U 1623, 3704,15     ;16113           MOV LS[CKBTS.1100100] TO WR[0]         ; выборка в WR0 контрольных битов для данных = 1
U 1624, A0C0,15     ;16114           COM WR[0]                              ; инвертирование контрольных битов для согласования с
                    ;16115                                                  ; аппаратурой
U 1625, 452C,15     ;16116           BIC LS[#FFFFFF00] TO WR[0]             ; очистка всех битов, кроме младшего
U 1626, 4774,15     ;16117           BIS LS[DIAG.CHK] TO WR[0]              ; установка бита DIAG CHK
U 1627, 8A17,FC     ;16118           JSR [WRITE.CSR1]                       ; запись в CSR1 контрольных битов и DIAG CHK
                    ;16119   LOOP.T2C.4:
U 1628, 36C0,15     ;16120           MOV LS[#3(H)] TO WR[0]                 ; рабочий регистр содержит 3
U 1629, BE82,15     ;16121           MOV WR[0] TO LS[ERROR.NUMBER]          ; ошибка 3
U 162A, E58A,15     ;16122           CLR LS[ERROR.MASK]                     ; проверка всех битов
                    ;16123   LOOP.T2C.3:
U 162B, 189D,F5     ;16124           MEM.REQ[READ.P] ADRS[#0] DT[LONG]      ; запрос для чтения из адреса памяти 0
U 162C, 3022,15     ;16125           MOV MEM.DATA TO WR[0]                  ; чтение из адреса 0
U 162D, 3640,95     ;16126           MOV LS[#1] TO WR[1]                    ; ожидаемый результат (неоткорректированные данные)
U 162E, 0869,3C     ;16127           JSR [CHECK.RESULT]                     ; проверка, что адрес памяти содержит 1
U 162F, 0962,B4     ;16128           JMP [LOOP.T2C.3]                       ; цикл при ошибке, если есть разрешение
U 1630, FF82,15     ;16129           INC LS[ERROR.NUMBER]                   ; ошибка 4
U 1631, BE88,15     ;16130           MOV WR[2] TO LS[ERROR.MASK]            ; маска для проверки битов ошибок CSR1
U 1632, 2F82,95     ;16131           CLR WR[1]                              ; ожидаемые данные (все биты ошибок очищены)
U 1633, 9D45,75     ;16132           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения CSR1
U 1634, 3022,15     ;16133           MOV MEM.DATA TO WR[0]                  ; чтение CSR1
U 1635, 0869,3C     ;16134           JSR [CHECK.RESULT]                     ; проверка правильности содержимого CSR1
U 1636, 0962,B4     ;16135           JMP [LOOP.T2C.4]                       ; цикл при ошибке, если есть разрешение
U 1637, FF82,15     ;16136           INC LS[ERROR.NUMBER]                   ; ошибка 5
U 1638, 999C,75     ;16137           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]     ; запрос для записи по адресу памяти 0
U 1639, B242,15     ;16138           WRITE.MEM LS[#2]                       ; запись в память значения 2 (будут неправильными биты 0
                    ;16139                                                  ; и 1, поэтому ошибка RDS)
U 163A, 3682,15     ;16140           MOV LS[ERROR.NUMBER] TO WR[0]          ; занесение номера ошибки в рабочий регистр
U 163B, 3E10,15     ;16141           MOV WR[0] TO LS[T8]                    ; запоминание в промежуточной ячейке LS
                    ;16142   LOOP.T2C.6:
U 163C, B610,15     ;16143           MOV LS[T8] TO WR[0]                    ; номер ошибки
U 163D, BE82,15     ;16144           MOV WR[0] TO LS[ERROR.NUMBER]          ; восстановление номера ошибки в LS
U 163E, E58A,15     ;16145           CLR LS[ERROR.MASK]                     ; снова проверка всех битов
                    ;16146   LOOP.T2C.5:
U 163F, 189D,F5     ;16147           MEM.REQ[READ.P] ADRS[#0] DT[LONG]      ; запрос для чтения из адреса памяти 0
U 1640, 3022,15     ;16148           MOV MEM.DATA TO WR[0]                  ; чтение из адреса 0
U 1641, B642,95     ;16149           MOV LS[#2] TO WR[1]                    ; ожидаемый результат (неоткорректированные данные)
U 1642, 0869,3C     ;16150           JSR [CHECK.RESULT]                     ; проверка, что адрес памяти 0 содержит 2
U 1643, 0963,F4     ;16151           JMP [LOOP.T2C.5]                       ; цикл при ошибке, если есть разрешение
U 1644, FF82,15     ;16152           INC LS[ERROR.NUMBER]                   ; ошибка 6
U 1645, BE88,15     ;16153           MOV WR[2] TO LS[ERROR.MASK]            ; маска для проверки битов ошибок CSR1
U 1646, B67E,95     ;16154           MOV LS[RDS] TO WR[1]                   ; ожидаемые данные (RDS установлен, остальные очищены)
U 1647, 9D45,75     ;16155           MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; запрос для чтения
U 1648, 3022,15     ;16156           MOV MEM.DATA TO WR[0]                  ; чтение CSR1
U 1649, 0869,3C     ;16157           JSR [CHECK.RESULT]                     ; проверка правильности содержимого CSR1
U 164A, 0963,C4     ;16158           JMP [LOOP.T2C.6]                       ; цикл при ошибке, если разрешено
U 164B, FF82,15     ;16159           INC LS[ERROR.NUMBER]                   ; ошибка 7
U 164C, 0A17,EC     ;16160           JSR [CLEAR.CSR1]                       ; очистка CSR
U 164D, 999C,75     ;16161           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]     ; запрос для записи в ячейку 0
```

```
U 164E, B29C,15 ;16162          WRITE.MEM LS[ZERO]                          ; запись в память всех нулей
U 164F, 3672,15 ;16163          MOV LS[ECC.DIS] TO WR[0]                    ; установка бита ECC DIS (запрет коррекци ошибок)
U 1650, BA17,FC ;16164          JSR [WRITE.CSR1]                            ; запись данных в CSR1
U 1651, 3682,15 ;16165          MOV LS[ERROR.NUMBER] TO WR[0]               ; занесение в рабочий регистр номера ошибки
U 1652, 3E10,15 ;16166          MOV WR[0] TO LS[T8]                         ; запоминание в промежуточной ячейке LS
                ;16167  LOOP.T2C.8.9:
U 1653, B610,15 ;16168          MOV LS[T8] TO WR[0]                         ; номер ошибки
U 1654, BE82,15 ;16169          MOV WR[0] TO LS[ERROR.NUMBER]               ; номер ошибки, восстановленный из LS
U 1655, E58A,15 ;16170          CLR LS[ERROR.MASK]                          ; проверка всех битов
                ;16171  LOOP.T2C.7:
U 1656, 189D,F5 ;16172          MEM.REQ[READ.P] ADRS[#0] DT[LONG]           ; запрос для чтения из адреса памяти 0
U 1657, 3022,15 ;16173          MOV MEM.DATA TO WR[0]                       ; выборка результата
U 1658, 2F82,95 ;16174          CLR WR[1]                                   ; ожидаемые данные
U 1659, 0869,3C ;16175          JSR [CHECK.RESULT]                          ; проверка на все нули
U 165A, 0965,64 ;16176          JMP [LOOP.T2C.7]                            ; цикл при ошибке, если есть разрешение
U 165B, FF82,15 ;16177          INC LS[ERROR.NUMBER]                        ; ошибка 8
U 165C, 3E8B,95 ;16178          MOV WR[3] TO LS[ERROR.MASK]                 ; проверка только битов 0-6
U 165D, 9D9D,75 ;16179          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]       ; запрос для чтения CSR0
U 165E, 3022,15 ;16180          MOV MEM.DATA TO WR[0]                       ; чтение CSR0
U 165F, 3700,95 ;16181          MOV LS[CKBTS.0111100] TO WR[1]              ; инверсия ожидаемых контрольных битов
U 1660, A0C2,95 ;16182          COM WR[1]                                   ; ожидаемые данные
U 1661, 0869,3C ;16183          JSR [CHECK.RESULT]                          ; проверка, что получены правильные контрольные биты
U 1662, 0965,34 ;16184          JMP [LOOP.T2C.8.9]                          ; цикл при ошибке, если есть разрешение
U 1663, FF82,15 ;16185          INC LS[ERROR.NUMBER]                        ; ошибка 9
U 1664, BE8B,15 ;16186          MOV WR[2] TO LS[ERROR.MASK]                 ; маска для проверки битов ошибок CSR1
U 1665, 9D45,75 ;16187          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; запрос для чтения CSR1
U 1666, 3022,15 ;16188          MOV MEM.DATA TO WR[0]                       ; чтение CSR1
U 1667, 2F82,95 ;16189          CLR WR[1]                                   ; ожидаемые данные (все биты ошибок очищены)
U 1668, 0869,3C ;16190          JSR [CHECK.RESULT]                          ; проверка, что данные CSR1 правильные
U 1669, 0965,34 ;16191          JMP [LOOP.T2C.8.9]                          ; цикл при ошибке, если есть разрешение
U 166A, FF82,15 ;16192          INC LS[ERROR.NUMBER]                        ; ошибка A
U 166B, 3704,15 ;16193          MOV LS[CKBTS.1100100] TO WR[0]              ; выборка в WR0 контрольных битов для данных = 1
U 166C, A0C0,15 ;16194          COM WR[0]                                   ; инвертирование контрольных битов для согласования с
                ;16195                                                      ; аппаратурой
U 166D, 452C,15 ;16196          BIC LS[#FFFFFF00] TO WR[0]                  ; очистка всех байтов, кроме младшего
U 166E, BA17,FC ;16197          JSR [WRITE.CSR1]                            ; занесение в CSR1 контрольных битов для данных = 0
                ;16198                                                      ; очистка в CSR1 всех других битов
U 166F, 1D40,F5 ;16199          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической
                ;16200                                                      ; программы. LVA00 установлен для ветвления
U 1670, B29C,15 ;16201          WRITE.MEM LS[ZERO]                          ; запись всех нулей в ячейку 0 с контрольными битами для
                ;16202                                                      ; данных = 1
U 1671, 3672,15 ;16203          MOV LS[ECC.DIS] TO WR[0]                    ; установка бита ECC DIS (запрет коррекции ошибок)
U 1672, BA17,FC ;16204          JSR [WRITE.CSR1]                            ; запись данных в CSR1
U 1673, 3682,15 ;16205          MOV LS[ERROR.NUMBER] TO WR[0]               ; занесение номера ошибки в рабочий регистр
U 1674, 3E10,15 ;16206          MOV WR[0] TO LS[T8]                         ; запоминание в промежуточной ячейке LS
                ;16207  LOOP.T2C.B.C:
U 1675, B610,15 ;16208          MOV LS[T8] TO WR[0]                         ; выборка номера ошибки
U 1676, BE82,15 ;16209          MOV WR[0] TO LS[ERROR.NUMBER]               ; восстановление номера ошибки в LS
U 1677, E58A,15 ;16210          CLR LS[ERROR.MASK]                          ; проверка всех битов
                ;16211  LOOP.T2C.A:
U 1678, 189D,F5 ;16212          MEM.REQ[READ.P] ADRS[#0] DT[LONG]           ; запрос для чтения из адреса памяти 0
U 1679, 3022,15 ;16213          MOV MEM.DATA TO WR[0]                       ; чтение адреса 0
U 167A, 2F82,95 ;16214          CLR WR[1]                                   ; ожидаемые данные
U 167B, 0869,3C ;16215          JSR [CHECK.RESULT]                          ; проверка, что коррекция не произошла
U 167C, 0967,84 ;16216          JMP [LOOP.T2C.A]                            ; цикл при ошибке, если есть разрешение
```

```
U 167D, FF82,15 ;16217          INC LS[ERROR.NUMBER]                    ; ошибка B
U 167E, 3E8B,95 ;16218          MOV WR[3] TO LS[ERROR.MASK]             ; проверка только битов 0-6
U 167F, 9D9D,75 ;16219          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ; запрос для чтения CSR0
U 1680, 3022,15 ;16220          MOV MEM.DATA TO WR[0]                   ; чтение CSR0
U 1681, B704,95 ;16221          MOV LS[CKBTS.1100100] TO WR[1]          ; инвертирование ожидаемых контрольных битов
U 1682, A0C2,95 ;16222          COM WR[1]                               ; ожидаемые данные
U 1683, 0869,3C ;16223          JSR [CHECK.RESULT]                      ; проверка, что полученные контрольные биты правильные
U 1684, 8967,54 ;16224          JMP [LOOP.T2C.B.C]                      ; цикл при ошибке, если есть разрешение
U 1685, FF82,15 ;16225          INC LS[ERROR.NUMBER]                    ; ошибка C
U 1686, BE8B,15 ;16226          MOV WR[2] TO LS[ERROR.MASK]             ; маска для проверки битов ошибок CSR1
U 1687, 9D45,75 ;16227          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 1688, 3022,15 ;16228          MOV MEM.DATA TO WR[0]                   ; чтение CSR1
U 1689, 367C,95 ;16229          MOV LS[CRD] TO WR[1]                    ; ожидаемые данные (установленный бит CRD)
U 168A, 0869,3C ;16230          JSR [CHECK.RESULT]                      ; проверка, что данные CSR1 правильные
U 168B, 8967,54 ;16231          JMP [LOOP.T2C.B.C]                      ; цикл при ошибке, если есть разрешение
U 168C, FF82,15 ;16232          INC LS[ERROR.NUMBER]                    ; ошибка D
U 168D, 0A17,EC ;16233          JSR [CLEAR.CSR1]                        ; очистка CSR1
U 168E, 999C,75 ;16234          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи по адресу памяти 0
U 168F, 3240,15 ;16235          WRITE.MEM LS[#1]                        ; запись в память значения 1
U 1690, B700,15 ;16236          MOV LS[CKBTS.0111100] TO WR[0]          ; выборка в WR0 контрольных битов для данных = 0
U 1691, A0C0,15 ;16237          COM WR[0]                               ; инвертирование контрольных битов для согласования с
                ;16238                                                  ; аппаратурой
U 1692, 452C,15 ;16239          BIC LS[#FFFFFF00] TO WR[0]              ; очистка всех байтов, кроме младшего
U 1693, 4772,15 ;16240          BIS LS[ECC.DIS] TO WR[0]                ; установка бита ECC DIS
U 1694, 4774,15 ;16241          BIS LS[DIAG.CHK] TO WR[0]               ; а также установка бита режима DIAG CHK
U 1695, 8A17,FC ;16242          JSR [WRITE CSR1]                        ; запись данных в CSR1
U 1696, 3682,15 ;16243          MOV LS[ERROR.NUMBER] TO WR[0]           ; занесение номера ошибки в рабочий регистр
U 1697, 3E10,15 ;16244          MOV WR[0] TO LS[T8]                     ; запоминание в промежуточной ячейке LS
                ;16245  LOOP.T2C.E.F:
U 1698, B610,15 ;16246          MOV LS[T8] TO WR[0]                     ; номер ошибки
U 1699, BE82,15 ;16247          MOV WR[0] TO LS[ERROR.NUMBER]           ; номер ошибки восстановлен в LS
U 169A, E58A,15 ;16248          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;16249  LOOP.T2C.D:
U 169B, 189D,F5 ;16250          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения ячейки 0
U 169C, 3022,15 ;16251          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 169D, 3640,95 ;16252          MOV LS[#1] TO WR[1]                     ; ожидаемые данные
U 169E, 0869,3C ;16253          JSR [CHECK.RESULT]                      ; проверка, что данные не откорректированны
U 169F, 8969,B4 ;16254          JMP [LOOP.T2C.D]                        ; цикл при ошибке, если есть разрешение
U 16A0, FF82,15 ;16255          INC LS[ERROR.NUMBER]                    ; ошибка E
U 16A1, 3E8B,95 ;16256          MOV WR[3] TO LS[ERROR.MASK]             ; проверка только битов 0-6
U 16A2, 9D9D,75 ;16257          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ; запрос для чтения CSR0
U 16A3, 3022,15 ;16258          MOV MEM.DATA TO WR[0]                   ; чтение CSR0
U 16A4, 3700,95 ;16259          MOV LS[CKBTS.0111100] TO WR[1]          ; инвертирование ожидаемых контрольных битов
U 16A5, A0C2,95 ;16260          COM WR[1]                               ; ожидаемые данные
U 16A6, 0869,3C ;16261          JSR [CHECK.RESULT]                      ; проверка, что получены правильные контрольные биты
U 16A7, 8969,84 ;16262          JMP [LOOP.T2C.E.F]                      ; цикл при ошибке, если есть разрешение
U 16A8, FF82,15 ;16263          INC LS[ERROR.NUMBER]                    ; ошибка F
U 16A9, BE8B,15 ;16264          MOV WR[2] TO LS[ERROR.MASK]             ; маска для проверки битов ошибок CSR1
U 16AA, 9D45,75 ;16265          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]   ; запрос для чтения CSR1
U 16AB, 3022,15 ;16266          MOV MEM.DATA TO WR[0]                   ; чтение CSR1
U 16AC, 367C,95 ;16267          MOV LS[CRD] TO WR[1]                    ; ожидаемые данные (установленный бит CRD)
U 16AD, 0869,3C ;16268          JSR [CHECK.RESULT]                      ; проверка, что данные в CSR1 правильные
U 16AE, 8969,B4 ;16269          JMP [LOOP.T2C.E.F]                      ; цикл при ошибке, если есть разрешение
U 16AF, FF82,15 ;16270          INC LS[ERROR.NUMBER]                    ; ошибка 10
U 16B0, 3704,15 ;16271          MOV LS[CKBTS.1100100] TO WR[0]          ; выборка в WR0 контрольных битов для данных = 1
```

```
U 16B1, A0C0,15 ;16272          COM WR[0]                                   ; инвертирование контрольных битов для согласования с
                ;16273                                                      ; аппаратурой
U 16B2, 452C,15 ;16274          BIC LS[#FFFFFF00] TO WR[0]                  ; очистка всех байтов, кроме младшего
U 16B3, 4772,15 ;16275          BIS LS[ECC.DIS] TO WR[0]                    ; установка бита ECC DIS
U 16B4, 4774,15 ;16276          BIS LS[DIAG.CHK] TO WR[0]                   ; а также установка бита режима DIAG CHK
U 16B5, 8A17,FC ;16277          JSR [WRITE.CSR1]                            ; запись данных в CSR1
U 16B6, 3682,15 ;16278          MOV LS[ERROR.NUMBER] TO WR[0]               ; занесение в рабочий регистр номера ошибки
U 16B7, 3E10,15 ;16279          MOV WR[0] TO LS[T8]                         ; запоминание в промежуточной ячейке LS
                ;16280  LOOP.T2C.11:
U 16B8, B610,15 ;16281          MOV LS[T8] TO WR[0]                         ; номер ошибки
U 16B9, BE82,15 ;16282          MOV WR[0] TO LS[ERROR.NUMBER]               ; восстановление номера ошибки в LS
U 16BA, 3E8B,95 ;16283          MOV WR[3] TO LS[ERROR.MASK]                 ; проверка только битов 0-6
                ;16284  LOOP.T2C.10:
U 16BB, 189D,F5 ;16285          MEM.REQ[READ.P] ADRS[#0] DT[LONG]           ; запрос для чтения ячейки 0
U 16BC, 3022,15 ;16286          MOV MEM.DATA TO WR[0]                       ; выборка результата без его проверки
U 16BD, 9D9D,75 ;16287          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]       ; запрос для чтения CSR0
U 16BE, 3022,15 ;16288          MOV MEM.DATA TO WR[0]                       ; чтение CSR0
U 16BF, B704,95 ;16289          MOV LS[CKBTS.1100100] TO WR[1]              ; инвертирование ожидаемых контрольных битов
U 16C0, A0C2,95 ;16290          COM WR[1]                                   ; ожидаемые данные
U 16C1, 0869,3C ;16291          JSR [CHECK.RESULT]                          ; проверка, что получены правильные контрольные биты
U 16C2, 096B,B4 ;16292          JMP [LOOP.T2C.10]                           ; цикл при ошибке, если есть разрешение
U 16C3, FF82,15 ;16293          INC LS[ERROR.NUMBER]                        ; ошибка 11
U 16C4, BE8B,15 ;16294          MOV WR[2] TO LS[ERROR.MASK]                 ; маска для проверки битов ошибок CSR1
U 16C5, 9D45,75 ;16295          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]       ; запрос для чтения CSR1
U 16C6, 3022,15 ;16296          MOV MEM.DATA TO WR[0]                       ; чтение CSR1
U 16C7, 2F82,95 ;16297          CLR WR[1]                                   ; ожидаемые данные
U 16C8, 0869,3C ;16298          JSR [CHECK.RESULT]                          ; проверка, что данные CSR1 правильные
U 16C9, 096B,84 ;16299          JMP [LOOP.T2C.11]                           ; цикл при ошибке, если есть разрешение
                ;16300  END.T2C:
```

```
;16301  .PAGE  "ТЕСТ 2D - тесты функции READ.V при отсутствии активности общей шины (модуль MCT)"
;16302  ;
;16303  ; ОПИСАНИЕ ТЕСТА:
;16304  ;
;16305  ;       Этот тест предназначен для обнаружения неисправностей в ПЗУ управля-
;16306  ;   ющей памяти методом проверки всех путей в микрокодах. Тест проверяет пу-
;16307  ;   ти, выбираемые, когда выполняются функции READ.V.NOCHK, READ.V.RCHK,READ.V.
;16308  ;   WCHK и READ.V.WCHK.LOCK при отсутствии какой-либо активности общей шины.
;16309  ;      Аппаратура, связанная с ПЗУ защиты ранее проверялась. Этот тест просто
;16310  ;   проверяет, что функция чтения по виртуальному адресу может правильно выпол-
;16311  ;   няться с любой точки входа, описанной выше. Этот тест записывает 1 в буфер
;16312  ;   трансляции (TB) по адресу 0, так что чтение по виртуальному адресу 0 будет
;16313  ;   в действительности происходить как чтение по физическому адресу 200. Код
;16314  ;   чередующихся данных записывается по физическому адресу 0, поэтому, если чтение
;16315  ;   будет происходить оттуда, будет обнаружена ошибка. Используемыми тестовыми
;16316  ;   данными являются все единицы и все нули.
;16317  ;
;16318  ; ПРЕДПОЛОЖЕНИЯ:
;16319  ;
;16320  ;       Принимается, что все предыдущие тесты прошли успешно и что вся аппара-
;16321  ;   тура (кроме ПЗУ управляющей памяти MCT) работает правильно.
;16322  ;
;16323  ; ШАГИ ТЕСТА:
;16324  ;
;16325  ;       1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для рас-
;16326  ;          печатки ошибок) и очистка в LS номера предыдущей ошибки. Кроме того,
;16327  ;          установка в CSR1 бита MME, что разрешает работу диспетчера памяти.
;16328  ;       2) Запись 1 в адресной части буфера трансляции по адресу 0.
;16329  ;       3) Запись чередующегося кода (55555555(H)) в ячейку памяти 0, используя
;16330  ;          WRITE.P и чтение при помощи READ.P с целью проверки данных.
;16331  ;       4) Запись всех нулей в физическую ячейку 200 с использованием WRITE.P.
;16332  ;          Чтение при помощи READ.V.NOCHK по виртуальному адресу 0 и проверка
;16333  ;          результата.
;16334  ;       5) Повторение шага 4 с данными, состоящими из всех единиц.
;16335  ;       6) Повторение шагов 4 и 5 с функциями READ.V.RCHK,READ.V.WCHK и READ.V
;16336  ;          WCHK.LOCKED.
;16337  ;
;16338  ; ОШИБКИ:
;16339  ;
;16340  ;       ошибка 1 - отказ подготовительной записи/чтения по адресу 0.
;16341  ;       ошибка 2 - функция READ.V.NOCHK не выполнила правильного чтения по вир-
;16342  ;                  туальному адресу 0 (физический адрес 200). Данные - все нули.
;16343  ;       ошибка 3 - функция READ.V.NOCHK не выполнила правильного чтения по вир-
;16344  ;                  туальному адресу 0 (физический адрес 200). Данные - все единицы.
;16345  ;       ошибка 4 - функция READ.V.RCHK не выполнила правильного чтения по вир-
;16346  ;                  туальному адресу 0 (физический адрес 200). Данные - все нули.
;16347  ;       ошибка 5 - функция READ.V.RCHK не выполнила правильного чтения по вир-
;16348  ;                  туальному адресу 0 (физичецкий адрес 200). Данные - все единицы.
;16349  ;       ошибка 6 - READ.V.WCHK не выполнила правильного чтения по виртуальному
;16350  ;                  адресу 0 (физический адрес 200). Данные - все нули.
;16351  ;       ошибка 7 - функция READ.V.WCHK не выполнила праильно чтения по виртуаль-
;16352  ;                  ному адресу 0 (физический адрес 200). Даные - все единицы.
;16353  ;       ошибка 8 - функция READ.V.WCHK.LOCKED не выполнила правильного чтения
;16354  ;                  по виртуальному адресу 0 (физический адрес 200). Данные - нули.
;16355  ;       ошибка 9 - функция READ.V.WCHK.LOCKED не выпонила правильного чтения
```

```
                ;16356  ;                   по виртуальному адресу 0 (физический адрес 200). Данные -
                ;16357  ;                   все единицы.
                ;16358  ;
                ;16359  ;       НАЛАДКА:
                ;16360  ;
                ;16361  ;        ПРИМЕЧАНИЕ: следующие ошибки с наибольшей вероятностью указывают на от-
                ;16362  ;    каз в потоке микрокодов. Аппаратура, используемая микрокодами, ранее проверена.
                ;16363  ;    чтобы локализовать неисправность, необходимо проверить последовательности, ука-
                ;16364  ;    занные ниже. Если последовательность правильная, подозревается дефектный
                ;16365  ;    бит в одном из ПЗУ управляюшей памяти и необходимо проверить каждое состоя-
                ;16366  ;    ние на правильность разрешающих сигналов.
                ;16367  ;
                ;16368  ;       ОШИБКА 1 - Эта ошибка указывает на принципиальную неисправность памяти.
                ;16369  ;    Для уточнения неисправности выполните снова предыдущие тесты.
                ;16370  ;
                ;16371  ;       ОШИБКА 2 - Эта ошибка указывает на отказ в микрокодах микропрограммы
                ;16372  ;    READ ARRAY. Правильной является последовательность из C1 через CPU.RD.V.NOCHK
                ;16373  ;    к ARY CYC C4. Остальная часть микропрограммы идентична микропрограмме READ.P,
                ;16374  ;    проверенной ранее.
                ;16375  ;
                ;16376  ;       ОШИБКА 3 - То же, что и ошибка 2, при данных, состоящих из всех единиц.
                ;16377  ;
                ;16378  ;       ОШИБКА 4 - эта ошибка указывает на отказ в потоке микрокодов микропрог-
                ;16379  ;    раммы READ.ARRAY. Правильной является последовательность из C1 через CPU.
                ;16380  ;    RD.V.RDCHK к ARY.CYC.C4. Остальная часть микропрограммы идентична с ранее
                ;16381  ;    проверенной микропрограммой READ.P.
                ;16382  ;
                ;16383  ;       ОШИБКА 5 - То же, что и ошибка 4, при данных, состоящих из всех единиц.
                ;16384  ;
                ;16385  ;       ОШИБКА 6 - Эта ошибка указывает на отказ в потоке микрокодов микропрог-
                ;16386  ;    раммы READ.ARRAY. Правильной является последовательность от C1 через CPU.
                ;16387  ;    RD.V.WRCHK к ARY.CYC.C4. Остальная часть микропрограммы идентична с ранее
                ;16388  ;    проверенной микропрограммой READ.P.
                ;16389  ;
                ;16390  ;       ОШИБКА 7 - То же, что и ошибка 6 при данных состоящих из всех единиц.
                ;16391  ;
                ;16392  ;       ОШИБКА 8 - Эта ошибка указывает на отказ в потоке микрокодов микро-
                ;16393  ;    программы READ.ARRAY. Правильной является последовательность от C1 к READ.
                ;16394  ;    V.WCHK.LOCK через READ.V.WCHK.LOCK.C4 к ARY CYC C4. Остальная часть микро-
                ;16395  ;    программы идентична с ранее проверенной микропрограммой READ.P.
                ;16396  ;
                ;16397  ;       ОШИБКА 9 - То же, что и ошибка 8 при данных, состоящих из всех единиц.
                ;16398  ;
                ;16399
                ;16400  T.2D:
U 16CA, B65E,15 ;16401          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;16402                                                  ; состояния
U 16CB, 3E80,15 ;16403          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;16404                                                  ; 15 указывает для консольного процессора начало теста
U 16CC, 10E0,15 ;16405          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;16406  WAIT.T2D.0:
U 16CD, B96C,D4 ;16407          JMP [WAIT.T2D.0]                        ; цикл для ожидания ответа консольного процессора
U 16CE, 0A1A,AC ;16408          JSR [SETUP.1]                           ; установка маски, кода модуля и т.д.
U 16CF, 0A17,DC ;16409          JSR [WRITE.CSR1.MME]                    ; установка в CSR1 бита MME, очистка других битов
U 16D0, 989C,F5 ;16410          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи по адресу TB 0
```

```
U 16D1, 3240,15 ;16411          WRITE.MEM LS[#1]                        ; запись 1 в адресную часть ТВ
                ;16412  LOOP.T2D.1:
U 16D2, 999C,75 ;16413          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]       ; запрос для записи в память по адресу 0
U 16D3, B29A,15 ;16414          WRITE.MEM LS[#55555555]                  ; запись чередующегося кода по адресу 0
U 16D4, B69A,95 ;16415          MOV LS[#55555555] TO WR[1]               ; ожидаемые данные
U 16D5, 189D,F5 ;16416          MEM.REQ[READ.P] ADRS[#0] DT[LONG]        ; запрос для чтения памяти
U 16D6, 3022,15 ;16417          MOV MEM.DATA TO WR[0]                    ; чтение данных по адресу 0
U 16D7, 0869,3C ;16418          JSR [CHECK.RESULT]                       ; проверка наличия ошибки
U 16D8, 096D,24 ;16419          JMP [LOOP.T2D.1]                         ; цикл при ошибке, если есть разрешение
U 16D9, FF82,15 ;16420          INC LS[ERROR.NUMBER]                     ; ошибка 2
U 16DA, 1952,75 ;16421          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]     ; запрос для записи в память по адресу 200
U 16DB, B29C,15 ;16422          WRITE.MEM LS[ZERO]                       ; запись данных, состоящих из нулей, по адресу 200
                ;16423  LOOP.T2D.2:
U 16DC, 2F82,95 ;16424          CLR WR[1]                                ; ожидаемые данные
U 16DD, 9A9C,75 ;16425          MEM.REQ[READ.V.NOCHK] ADRS[#0] DT[LONG]  ; запрос для чтения по виртуальному адресу 0
U 16DE, 3022,15 ;16426          MOV MEM.DATA TO WR[0]                    ; чтение из виртуального адреса 0 (физ.200)
U 16DF, 0869,3C ;16427          JSR [CHECK.RESULT]                       ; проверка, что данные правильные
U 16E0, 896D,C4 ;16428          JMP [LOOP.T2D.2]                         ; цикл при ошибке, если есть разрешение
U 16E1, FF82,15 ;16429          INC LS[ERROR.NUMBER]                     ; ошибка 3
U 16E2, 1952,75 ;16430          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]     ; запрос для записи в память по адресу 200
U 16E3, 329E,15 ;16431          WRITE.MEM LS[ONES]                       ; запись данных, состоящих из всех единиц, по адресу
                ;16432                                                   ; 200
                ;16433  LOOP.T2D.3:
U 16E4, 369E,95 ;16434          MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 16E5, 9A9C,75 ;16435          MEM.REQ[READ.V.NOCHK] ADRS[#0] DT[LONG]  ; запрос для чтения по виртуальному адресу 0
U 16E6, 3022,15 ;16436          MOV MEM.DATA TO WR[0]                    ; чтение из вирт.адреса 0 (физич.200)
U 16E7, 0869,3C ;16437          JSR [CHECK.RESULT]                       ; проверка, что данные правильные
U 16E8, 096E,44 ;16438          JMP [LOOP.T2D.3]                         ; цикл при ошибке, если есть разрешение
U 16E9, FF82,15 ;16439          INC LS[ERROR.NUMBER]                     ; ошибка 4
U 16EA, 1952,75 ;16440          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]     ; запрос для записи в память по адресу 200
U 16EB, B29C,15 ;16441          WRITE.MEM LS[ZERO]                       ; запись данных из всех 0 по адресу 200
                ;16442  LOOP.T2D.4:
U 16EC, 2F82,95 ;16443          CLR WR[1]                                ; ожидаемые данные
U 16ED, 1A9D,75 ;16444          MEM.REQ[READ.V.RCHK] ADRS[#0] DT[LONG]   ; запрос для чтения по виртуальному адресу 0
U 16EE, 3022,15 ;16445          MOV MEM.DATA TO WR[0]                    ; чтение из вирт.адреса 0 (физич. 200)
U 16EF, 0869,3C ;16446          JSR [CHECK.RESULT]                       ; проверка, что данные правильные
U 16F0, 896E,C4 ;16447          JMP [LOOP.T2D.4]                         ; цикл при ошибке, если есть разрешение
U 16F1, FF82,15 ;16448          INC LS[ERROR.NUMBER]                     ; ошибка 5
U 16F2, 1952,75 ;16449          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]     ; запрос для записи по адресу памяти 200
U 16F3, 329E,15 ;16450          WRITE.MEM LS[ONES]                       ; запись данных, состоящих из всех 1, по адр.200
                ;16451  LOOP.T2D.5:
U 16F4, 369E,95 ;16452          MOV LS[ONES] TO WR[1]                    ; ожидаемые данные
U 16F5, 1A9D,75 ;16453          MEM.REQ[READ.V.RCHK] ADRS[#0] DT[LONG]   ; запрос для чтения по виртуальному адресу 0
U 16F6, 3022,15 ;16454          MOV MEM.DATA TO WR[0]                    ; чтение из вирт.адреса 0 (физич.200)
U 16F7, 0869,3C ;16455          JSR [CHECK.RESULT]                       ; проверка, что данные правильные
U 16F8, 896F,44 ;16456          JMP [LOOP.T2D.5]                         ; цикл при ошибке, если есть разрешение
U 16F9, FF82,15 ;16457          INC LS[ERROR.NUMBER]                     ; ошибка 6
U 16FA, 1952,75 ;16458          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]     ; запрос для записи по адресу памяти 200
U 16FB, B29C,15 ;16459          WRITE.MEM LS[ZERO]                       ; запись данных из всех 0 по адресу 200
                ;16460  LOOP.T2D.6:
U 16FC, 2F82,95 ;16461          CLR WR[1]                                ; ожидаемые данные
U 16FD, 1A9C,F5 ;16462          MEM.REQ[READ.V.WCHK] ADRS[#0] DT[LONG]   ; запрос для чтения по виртуальному адресу 0
U 16FE, 3022,15 ;16463          MOV MEM.DATA TO WR[0]                    ; чтение из вирт.адреса 0 (физич.200)
U 16FF, 0869,3C ;16464          JSR [CHECK.RESULT]                       ; проверка, что данные правильные
U 1700, 096F,C4 ;16465          JMP [LOOP.T2D.6]                         ; цикл при ошибке, если есть разрешение
```

```
U 1701, FF82,15 ;16466          INC LS[ERROR.NUMBER]                          ; ошибка 7
U 1702, 1952,75 ;16467          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]          ; запрос для записи по адресу памяти 200
U 1703, 329E,15 ;16468          WRITE.MEM LS[ONES]                            ; запись данных из всех 1 по адресу 200
                ;16469  LOOP.T2D.7:
U 1704, 369E,95 ;16470          MOV LS[ONES] TO WR[1]                         ; ожидаемые данные
U 1705, 1A9C,F5 ;16471          MEM.REQ[READ.V.WCHK] ADRS[#0] DT[LONG]        ; запрос для чтения по виртуальному адресу 0
U 1706, 3022,15 ;16472          MOV MEM.DATA TO WR[0]                         ; чтение по вирт.адресу 0 (физич.200)
U 1707, 0869,3C ;16473          JSR [CHECK.RESULT]                            ; проверка, что данные правильные
U 1708, 0970,44 ;16474          JMP [LOOP.T2D.7]                              ; цикл при ошибке, если есть разрешение
U 1709, FF82,15 ;16475          INC LS[ERROR.NUMBER]                          ; ошибка 8
U 170A, 1952,75 ;16476          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]          ; запрос для записи по адресу памяти 200
U 170B, B29C,15 ;16477          WRITE.MEM LS[ZERO]                            ; запись данных из всех 0 по адресу 200
                ;16478  LOOP.T2D.8:
U 170C, 2F82,95 ;16479          CLR WR[1]                                     ; ожидаемые данные.
U 170D, 9E9C,F5 ;16480          MEM.REQ[READ.V.WCHK.LOCKED] ADRS[#0] DT[LONG]; запрос для чтения по виртуальному адресу 0
U 170E, 3022,15 ;16481          MOV MEM.DATA TO WR[0]                         ; чтение по вирт.адресу 0 (физич.200)
U 170F, 0869,3C ;16482          JSR [CHECK.RESULT]                            ; проверка, что данные правильные
U 1710, B970,C4 ;16483          JMP [LOOP.T2D.8]                              ; цикл при ошибке, если есть разрешение
U 1711, FF82,15 ;16484          INC LS[ERROR.NUMBER]                          ; ошибка 9
U 1712, 1952,75 ;16485          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]          ; запрос для записи памяти по адресу 200
U 1713, 329E,15 ;16486          WRITE.MEM LS[ONES]                            ; запись данных из всех 1 по адресу 200
                ;16487  LOOP.T2D.9:
U 1714, 369E,95 ;16488          MOV LS[ONES] TO WR[1]                         ; ожидаемые данные
U 1715, 9E9C,F5 ;16489          MEM.REQ[READ.V.WCHK.LOCKED] ADRS[#0] DT[LONG]; запрос для чтения по виртуальному адресу 0
U 1716, 3022,15 ;16490          MOV MEM.DATA TO WR[0]                         ; чтение по вирт.адресу 0 (физич.200)
U 1717, 0869,3C ;16491          JSR [CHECK.RESULT]                            ; проверка, что данные правильные
U 1718, B971,44 ;16492          JMP [LOOP.T2D.9]                              ; цикл при ошибке, если есть разрешеие
                ;16493  END.T2D:
```

```
;16494  .PAGE "ТЕСТ 2Е - микропрограмма READ.V.RCHK.IFILL при ошибке (модуль МСТ)"
;16495  ;
;16496  ; ОПИСАНИЕ ТЕСТА:
;16497  ;
;16498  ;       Этот тест предназначен для обнаружения неисправностей в ПЗУ управляющей
;16499  ;    памяти методом проверки всех путей в микрокодах. Тест проверяет путь, который
;16500  ;    выбирается, если происходит одиночная ошибка или суммарная ошибка (ERR SUM)
;16501  ;    при заполнении регистра предвыборки инструкций (PFR). Этот тест записывает
;16502  ;    в память одиночную ошибку с использованием программы MAINT.ECC.DATA.
;16503  ;    Контрольные биты для данных из всех 0 записываются вместе с данными = 1 для
;16504  ;    получения в ячейке 0 одиночной ошибки. Затем по адресу 0 выполняется
;16505  ;    READ.V.RCHK.IFILL. Выполняется инструкция MOV IB.DATA TO OS при PC=0 и OS
;16506  ;    проверяется на правильность данных (все нули). CSR1 проверяется на установ-
;16507  ;    ленный бит CRD, а CSR0 проверяется на правильные биты синдрома. Затем в па-
;16508  ;    мять записываются все единицы с правильными контрольными битами. В CSR1 ус-
;16509  ;    танавливается бит TB PAR DIAG и тест повторяется. Вызванная ошибка ERR SUM
;16510  ;    должна предотвратить заполнение регистра предвыборки инструкций (PFR) и снова
;16511  ;    должны считываться все нули. Проверяется CSR1, чтобы удостовериться, что про-
;16512  ;    изошла ошибка паритета буфера трансляции (TB PAR ERR).
;16513  ;
;16514  ; ПРЕДПОЛОЖЕНИЯ:
;16515  ;
;16516  ;    Принимается, что все предыдущие тесты выполнены успешно и что вся аппаратура
;16517  ;    (кроме ПЗУ управляющей памяти МСТ) работает правильно.
;16518  ;
;16519  ; ШАГИ ТЕСТА:
;16520  ;
;16521  ;    1) Установка в LS маски ошибок, номера ошибки и номера модуля (для распечат-
;16522  ;       ки ошибок) и очистка в LS номера предыдущей ошибки.
;16523  ;    2) Загрузка в CSR1 контрольных битов 0111100(в) (перед записью в CSR1 они
;16524  ;       должны инвертироваться). Это контрольные биты для данных из всех 0. Очис-
;16525  ;       тка других битов CSR1.
;16526  ;    3) Выдача MEM.REQ с MF=MAINT.ECC.DATA и DT=длинное слово. Использование ад-
;16527  ;       реса 1 для записи по адресу 0 (LVA00 используется в качестве условия вет-
;16528  ;       вления к микропрограмме MAINT.ECC.DATA). Затем выдача WRITE.MEM LS с дан-
;16529  ;       ными = 1.
;16530  ;    4) Запись 1 в позицию бита VALID буфера трансляции по адресу 0, так что вир-
;16531  ;       туальный адрес 0 соответствуют физическому адресу 0.
;16532  ;    5) Установка в CSR1 бита MME и выполнение READ.V.RCHK.IBFILL по адресу 0.
;16533  ;       Затем выполнение MOV IB.DATA TO OS для выборки результата из IB.
;16534  ;    6) Проверка OS на все нули (откорректированные данные). Затем проверка CSR1
;16535  ;       на установленный бит CRD и CSR0 на правильные биты синдрома.
;16536  ;    7) Запись всех единиц по адресу 0. Повторение шага 5 с установленным в CSR1,
;16537  ;       кроме того, битом TB PAR DIAG. Проверка, что регистр OS не изменился (при
;16538  ;       ошибке памяти ERR SUM данные не модифицированы) и проверка CSR1 на уста-
;16539  ;       новленный бит TB PAR ERR.
;16540  ;
;16541  ; ОШИБКИ:
;16542  ;
;16543  ;    ПРИМЕЧАНИЕ: ожидаемыми и полученными данными являются следующие: данные IB
;16544  ;                при маске ошибки FFFFFF00, CSR1 при маске ошибки 29003FFF и CSR0
;16545  ;                при маске ошибки FFFFFFB0.
;16546  ;
;16547  ;    ошибка 1 - ошибка в одиночном бите не откорректирована должным образом.
;16548  ;    ошибка 2 - в CSR1 не установлен бит CRD или установлены другие биты.
```

```
                    ;16549  ;   ошибка 3 - биты синдрома в CSR0 неправильные.
                    ;16550  ;   ошибка 4 - заполнение буфера предвыборки не прекращено при ошибке памяти (ERR SUM).
                    ;16551  ;   ошибка 5 - в CSR1 не установлен бит TB PAR ERR или установлены другие биты.
                    ;16552  ;
                    ;16553  ; НАЛАДКА:
                    ;16554  ;
                    ;16555  ;   ПРИМЕЧАНИЕ: следующие ошибки с наибольшей вероятностью указывают на отказ в
                    ;16556  ;   потоке микрокодов. Аппаратура, используемая микрокодами, ранее проверялась.
                    ;16557  ;   Для локализации неисправности необходимо проверить последовательности, ука-
                    ;16558  ;   занные ниже. Если последовательность правильная, подозревается дефектный бит
                    ;16559  ;   в одном из ПЗУ управляющей памяти и необходимо проверить каждое состояние
                    ;16560  ;   на правильность разрешающих сигналов.
                    ;16561  ;
                    ;16562  ;      ОШИБКА 1 - эта ошибка указывает на отказ в потоке микрокодов, связанном с
                    ;16563  ;   ветвью SERR микропрограммы READ.V.RCHK.IFILL. Правильной является следующая
                    ;16564  ;   последовательность микроинструкций от CPU.ISTREAM.REQ.C6. Выбирается ветвь
                    ;16565  ;   не ERRSUM и не CPUG. Отсюда должен выбираться путь ERR и READ.V.RCHK.IFILL.D.E.
                    ;16566  ;   Далее должен выбираться путь SERR через READ.V.RCHK.IFILL.D.E.C1, через
                    ;16567  ;   READ.V.RCHK.IFILL.D.E.C2, через READ.V.RCHK.IFILL.D.E.C3, через READ.V.RCHK.
                    ;16568  ;   IFILL.C8 к IDLE.
                    ;16569  ;
                    ;16570  ;      ОШИБКА 2 и 3 - то же, что и при ошибке 1. Можно подозревать отказ в пути SERR.
                    ;16571  ;
                    ;16572  ;      ОШИБКА 4 - эта ошибка указывает на отказ в пути ERRSUM. Из CPU.ISTREAM.REQ.C6
                    ;16573  ;   должно выбираться ветвление на путь ERRSUM и не CPUG.
                    ;16574  ;
                    ;16575  ;      ОШИБКА 5 - подозревается отказ в READ.V.RCHK.IFILL.C5, так как именно это
                    ;16576  ;   микрослово стробирует запись битов ошибок в CSR1 при ERRSUM.
                    ;16577  ;
                    ;16578  ;-----
                    ;16579  ;
                    ;16580   T.2E:
U 1719, B65E,15     ;16581           MOV LS[BEGIN.TEST] TO WR[0]                ; установка в WR0 бита 15 для слова управления и
                    ;16582                                                      ; состояния
U 171A, 3EB0,15     ;16583           MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                    ;16584                                                      ; 15 указывает для конс.процессора начало теста
U 171B, 10E0,15     ;16585           MISC [SET.CP.ATTN]                         ; выдача для конс.процессора CPU ATTN
                    ;16586   WAIT.T2E.0:
U 171C, 0971,C4     ;16587           JMP [WAIT.T2E.0]                           ; цикл для ожидания ответа конс.процессора
U 171D, 0A1A,9C     ;16588           JSR [SETUP]                                ; установка маски, кода модуля и т.д. и очистка CSR1
                    ;16589                                                      ; MCT
U 171E, B713,95     ;16590           MOV LS[#FFFFFFB0] TO WR[3]                  ; установка битов 7-31
U 171F, B62F,15     ;16591           MOV LS[CSR1.MASK] TO WR[2]                  ; маска ошибки для ошибки 2. Не проверяются биты 0-13 и
                    ;16592                                                      ; и бит 24
U 1720, 4777,15     ;16593           BIS LS[MME] TO WR[2]                       ; не проверяется бит MME
U 1721, 477B,15     ;16594           BIS LS[TB.PAR.DIAG] TO WR[2]                ; и бит TB PAR DIAG
U 1722, 999C,75     ;16595           MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]          ; запрос для записи по адресу памяти 0
U 1723, 329E,15     ;16596           WRITE.MEM LS[ONES]                         ; запись всех единиц по адресу 0
U 1724, 989D,75     ;16597           MEM.REQ[READ.V.RCHK.IFILL] ADRS[#0] DT[LONG] ; запрос для заполнения IB из адреса памяти 0
U 1725, 367E,15     ;16598           MOV LS[BIT31] TO WR[0]                      ; установка в рабочем регистре бита 31
U 1726, C77A,15     ;16599           BIS LS[BIT29] TO WR[0]                      ; бита 29
U 1727, 4772,15     ;16600           BIS LS[BIT25] TO WR[0]                      ; и бита 25
U 1728, BE12,15     ;16601           MOV WR[0] TO LS[T9]                         ; запоминание в промежуточной ячейке LS
U 1729, 989C,F5     ;16602           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]         ; запрос для записи в TB по адресу 0
U 172A, B212,15     ;16603           WRITE.MEM LS[T9]                           ; установка битов VALID, BYTE OFFSET и PROT C и
```

```
                 ;16604                                                ; отображение в TB виртуального адреса 0 в физический
                 ;16605                                                ; адрес 0
U 172B, B700,15  ;16606          MOV LS[CKBTS.0111100] TO WR[0]        ; выборка в WR0 контрольных битов для данных=0
U 172C, A0C0,15  ;16607          COM WR[0]                             ; инвертирование контрольных битов для согласования с
                 ;16608                                                ; аппаратурой
U 172D, 452C,15  ;16609          BIC LS[#FFFFFF00] TO WR[0]            ; очистка всех байтов, кроме младшего
U 172E, 8A17,FC  ;16610          JSR [WRITE.CSR1]                      ; занесение в CSR1 контрольных битов для данных=0,
                 ;16611                                                ; очистка в CSR1 всех других битов
U 172F, 1D40,F5  ;16612          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической
                 ;16613                                                ; программы. LVA00 установлен для ветвления
U 1730, 3240,15  ;16614          WRITE.MEM LS[#1]                      ; запись в ячейку 0 значения 1 с контрольными битами для
                 ;16615                                                ; данных из всех 0
U 1731, 0A17,DC  ;16616          JSR [WRITE.CSR1.MME]                  ; запись в CSR1 бита MME
                 ;16617   LOOP.T2E.2.3:
U 1732, B640,15  ;16618          MOV LS[#1] TO WR[0]                   ; рабочий регистр содержит 1
U 1733, BEB2,15  ;16619          MOV WR[0] TO LS[ERROR.NUMBER]         ; запись в LS номера ошибки
U 1734, B62C,15  ;16620          MOV LS[#FFFFFF00] TO WR[0]            ; установка маски ошибки
U 1735, 3E8A,15  ;16621          MOV WR[0] TO LS[ERROR.MASK]           ; маска ошибки для проверки только младшего байта
                 ;16622   LOOP.T2E.1:
U 1736, 2FB2,95  ;16623          CLR WR[1]                             ; ожидаемые данные
U 1737, E520,15  ;16624          CLR LS[PC]                            ; байт регистра IB, подлежащий загрузке в OS
U 1738, 9B9D,75  ;16625          MEM.REQ[READ.V.RCHK.IFILL] ADRS[#0] DT[LONG] ; запрос для заполнения IB из адреса памяти 0
U 1739, 8401,4E  ;16626          MOV IB.DATA TO OS                     ; выборка результата
U 173A, DB00,15  ;16627          NOP                                   ; эта микроинструкция должна пропускаться при пропуске
                 ;16628                                                ; по IB.VALID
U 173B, B6F8,15  ;16629          MOV LS[OS] TO WR[0]                   ; выборка результата из регистра IB
U 173C, 0869,3C  ;16630          JSR [CHECK.RESULT]                    ; проверка, что данные=0
U 173D, 8973,64  ;16631          JMP [LOOP.T2E.1]                      ; цикл при ошибке, если есть разрешение
U 173E, FF82,15  ;16632          INC LS[ERROR.NUMBER]                  ; ошибка 2
U 173F, BE8B,15  ;16633          MOV WR[2] TO LS[ERROR.MASK]           ; маска ошибки для проверки CSR1
U 1740, 367C,95  ;16634          MOV LS[CRD] TO WR[1]                  ; ожидаемые данные (бит CRD установлен, остальные
                 ;16635                                                ; очищены)
U 1741, 9D45,75  ;16636          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG] ; чтение CSR1
U 1742, 3022,15  ;16637          MOV MEM.DATA TO WR[0]                 ; выборка результата
U 1743, 0869,3C  ;16638          JSR [CHECK.RESULT]                    ; проверка, что установлен только CRD
U 1744, 0973,24  ;16639          JMP [LOOP.T2E.2.3]                    ; цикл при ошибке, если есть разрешение
U 1745, FF82,15  ;16640          INC LS[ERROR.NUMBER]                  ; ошибка 3
U 1746, 3E8B,95  ;16641          MOV WR[3] TO LS[ERROR.MASK]           ; маска ошибки для проверки CSR0
U 1747, 3646,95  ;16642          MOV LS[#8] TO WR[1]                   ; установка бита 3
U 1748, C748,95  ;16643          BIS LS[#10] TO WR[1]                  ; бита 4
U 1749, 474C,95  ;16644          BIS LS[#40] TO WR[1]                  ; и бита 6 (1011000(B))
U 174A, A0C2,95  ;16645          COM WR[1]                             ; ожидаемый синдром
U 174B, 9D9D,75  ;16646          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG] ; чтение CSR0
U 174C, 3022,15  ;16647          MOV MEM.DATA TO WR[0]                 ; выборка результата
U 174D, 0869,3C  ;16648          JSR [CHECK.RESULT]                    ; проверка, что биты синдрома правильные
U 174E, 0973,24  ;16649          JMP [LOOP.T2E.2.3]                    ; цикл при ошибке, если есть разрешение
U 174F, 999C,75  ;16650          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]    ; запрос для записи по адресу памяти 0
U 1750, 329E,15  ;16651          WRITE.MEM LS[ONES]                    ; запись в адрес 0 всех 1
U 1751, B676,15  ;16652          MOV LS[MME] TO WR[0]                  ; запись в рабочий регистр бита MME
U 1752, C77A,15  ;16653          BIS LS[TB.PAR.DIAG] TO WR[0]          ; также установка бита TB PAR DIAG
U 1753, 8A17,FC  ;16654          JSR [WRITE.CSR1]                      ; запись в CSR1
U 1754, B62D,95  ;16655          MOV LS[#FFFFFF00] TO WR[3]            ; установка в WR3 маски ошибок для проверки младшего
                 ;16656                                                ; байта
                 ;16657   LOOP.T2E.5:
U 1755, 3644,15  ;16658          MOV LS[#4] TO WR[0]                   ; рабочий регистр содержит 4
```

```
U 1756,  BEB2,15 ;16659          MOV WR[0] TO LS[ERROR.NUMBER]          ; запоминание номера ошибки в LS
U 1757,  3E8B,95 ;16660          MOV WR[3] TO LS[ERROR.MASK]            ; запоминание маски ошибок для проверки младшего байта
                 ;16661   LOOP.T2E.4:
U 1758,  2F82,95 ;16662          CLR WR[1]                              ; ожидаемые данные
U 1759,  E520,15 ;16663          CLR LS[PC]                             ; байт регистра IB, подлежащий загрузке в OS
U 175A,  989D,75 ;16664          MEM.REQ[READ.V.RCHK.IFILL] ADRS[#0] DT[LONG]; запрос для заполнения IB из адреса памяти 0
U 175B,  8401,4E ;16665          MOV IB.DATA TO OS                      ; выборка результата
U 175C,  D800,15 ;16666          NOP                                    ; пропуск по IB.VALID должен пропустить эту инструкцию
U 175D,  B6F8,15 ;16667          MOV LS[OS] TO WR[0]                    ; выборка результата из регистра IB
U 175E,  0869,3C ;16668          JSR [CHECK.RESULT]                     ; проверка, что данные=0
U 175F,  0975,84 ;16669          JMP [LOOP.T2E.4]                       ; цикл при ошибке, если есть разрешение
U 1760,  FF82,15 ;16670          INC LS[ERROR.NUMBER]                   ; ошибка 5
U 1761,  BE8B,15 ;16671          MOV WR[2] TO LS[ERROR.MASK]            ; маска ошибки для проверки CSR1
U 1762,  365E,95 ;16672          MOV LS[TB.PAR.ERR] TO WR[1]            ; ожидаемые данные.(установлен бит TB PAR ERR, остальные
                 ;16673                                                 ; очищены)
U 1763,  9D45,75 ;16674          MEM.REQ[READ.CSR] ADRS[CSR1] DT[LONG]  ; чтение CSR1
U 1764,  3022,15 ;16675          MOV MEM.DATA TO WR[0]                  ; выборка результата
U 1765,  0869,3C ;16676          JSR [CHECK.RESULT]                     ; проверка, что установлен только TB PAR ERR
U 1766,  8975,54 ;16677          JMP [LOOP.T2E.5]                       ; цикл при ошибке, если есть разрешение
                 ;16678   END.T2E:
```

```
;16679  .PAGE "ТЕСТ 2F - тест WRITE.V.WCHK и прекращения записи (модуль МСТ)"
;16680  ;
;16681  ;
;16682  ; ОПИСАНИЕ ТЕСТА:
;16683  ;
;16684  ;       Этот тест предназначен для обнаружения отказов в ПЗУ управляющей памяти
;16685  ;    методом проверки всех путей в микрокодах. Тест проверяет путь, выбирае-
;16686  ;    мый, когда задается функция WRITE.V.WCHK для записи выравненного и невырав-
;16687  ;    ненного длинного слова. Также, проверяется путь прекращения записи при ошибке
;16688  ;    памяти (ERRSUM). Буфер трансляции ТВ записывается так, чтобы виртуальный адрес 0
;16689  ;    отображался в физический адрес 200, и в CSR1 устанавливается бит MME. Выполня-
;16690  ;    ется запись функцией WRITE.P всех единиц по адресу 200, за которой следует вы-
;16691  ;    равненая запись по виртуальному адресу 0 с данными, состоящими из всех 0. Дан-
;16692  ;    ные проверяются посредством микропрограммы READ.P. Далее проверяется невырав-
;16693  ;    ненная запись путем записи единиц по виртуальному адресу 2 и проверкой слова
;16694  ;    нулей и слова единиц по физическому адресу 200. Наконец, делается попытка запи-
;16695  ;    си длинного слова нулей по виртуальному адресу 0 при установленном в CSR1 би-
;16696  ;    те TB PAR DIAG для возбуждения ошибки ERR.SUM. Проверяется ячейка 200, чтобы
;16697  ;    удостовериться, что запись была прекращена.
;16698  ;
;16699  ; ПРЕДПОЛОЖЕНИЯ:
;16700  ;
;16701  ;    Принимается, что все предыдущие тесты прошли успешно и что вся аппаратура (кро-
;16702  ;    ме ПЗУ управляющей памяти) работает правильно.
;16703  ;
;16704  ; ШАГИ ТЕСТА:
;16705  ;
;16706  ;    1)Занесение в LS маски ошибок, номера ошибки и номера модуля (для распечатки
;16707  ;      ошибок) и очистка в LS номера предыдущей ошибки.
;16708  ;    2)Очистка CSR1. Запись всех единиц по физическому адресу 200 и запись нулей
;16709  ;      по физическому адресу 204.
;16710  ;    3)Запись по адресу 0 в ТВ установленных битов VALID,PROT C,MODIFY, BYTE
;16711  ;      OFFSET и PA 09, поэтому виртуальный адрес 0 будет отображаться в физ.адрес 200
;16712  ;    4)Установка в CSR1 бита MME.
;16713  ;    5)Выполнение WRITE.V.WCHK с данными из всех 0 по виртуальному адресу 0 и чте-
;16714  ;      ние результата функцией READ.P по адресу 200. Проверка на все 0.
;16715  ;    6)Выполнение WRITE.V.WCHK с данными из всех 1 по виртуальному адресу 2 (невы-
;16716  ;      равенная запись по физическому адресу 202). Чтение результата функцией READ.P
;16717  ;      по адресу 200 и проверка на FFFF0000(H).
;16718  ;    7)Установка в CSR1 бита TB PAR DIAG и выполнение WRITE.V.WCHK с данными из
;16719  ;      всех нулей по виртуальному адресу 0. Чтение по физическому адресу 200 и про-
;16720  ;      верка, что данные не были модифицированы (ERR SUM должна была прервать за-
;16721  ;      пись).
;16722  ;
;16723  ; ОШИБКИ:
;16724  ;
;16725  ;    ошибка 1 - функция WRITE.V.WCHK не выполнила записи по виртуальному адресу 0.
;16726  ;               (физический адрес 200).
;16727  ;    ошибка 2 - функция WRITE.V.WCHK не выполнила записи невыравненного длинного
;16728  ;               слова по виртуальному адресу 2 (физический адрес 202).
;16729  ;    ошибка 3 - функция WRITE.V.WCHK не прекратила записи при ошибке ERR SUM (ошибка
;16730  ;               паритета ТВ).
;16731  ;
;16732  ; НАЛАДКА:
;16733  ;
```

```
                ;16734  ;   Следующие ошибки с наибольшей вероятностью указывают на отказ в потоке микро-
                ;16735  ;   кодов. Аппаратура, используемая микрокодами, ранее испытывалась. Для лакализациия
                ;16736  ;   неисправности необходимо проверить последовательности, указанные ниже. Если по-
                ;16737  ;   следовательность правильная, можно подозревать дефектный бит в одном из ПЗУ
                ;16738  ;   управляющей памяти и следует проверить каждое состояние на правильность раз-
                ;16739  ;   решающих сигналов.
                ;16740  ;
                ;16741  ;       ОШИБКА 1 - Эта ошибка указывает на отказ в потоке микрокодов, связанных с
                ;16742  ;   WRITE.V. WCHK. Правильной является последовательность инструкций из WRITE.V.WCHK
                ;16743  ;   через WR ARY WCHK C4, через WR ARY WCHK C5, через WR ARY ALIGN LW C6 к WR ARY
                ;16744  ;   ALIGN LW C7. Состояние после WR ARY ALIGN LW C7 проверялось ранее.
                ;16745  ;
                ;16746  ;       ОШИБКА 2 - Эта ошибка вероятно указывает на отказ в ветвлении по ALIGN LW из WR
                ;16747  ;   ARY WCHK C5. Последовательность такая же, как и выше, за исключением того, что
                ;16748  ;   после WR ARY WCHK C5 микропрограмма должна перейти к WR ARY C6 и затем к WR
                ;16749  ;   ARY C7. Состояния, следующие за этим, ранее проверялись.
                ;16750  ;
                ;16751  ;       ОШИБКА 3 - Эта ошибка указывает на отказ в ветвлении по ERRSUM из WR ARY ALIGN
                ;16752  ;   LW C6 или за ним. Микрокоды те же, как и для ошибки 1, до WR ARY ALIGN LW C7. От-
                ;16753  ;   сюда микропрограмма должна перейти к ABORT WR CYC C1, через ABORT WR WAIT к
                ;16754  ;   UB PREP и IDLE.
                ;16755  ;--
                ;16756
                ;16757   T.2F:
U 1767, B65E,15 ;16758           MOV LS[BEGIN.TEST] TO WR[0]                ; установка бита 15 в WR0 для слова управления и
                ;16759                                                      ; состояния
U 1768, 3EB0,15 ;16760           MOV WR[0] TO LS[CONTROL.STATUS]             ; установка бита 15 в слове управления и состояния. Бит
                ;16761                                                      ; 15 указывает для конс.процессора начало теста
  1769, 10E0,15 ;16762           MISC [SET.CP.ATTN]                         ; выдача для конс.процесора CPU ATTN
                ;16763   WAIT.T2F.0:
U 176A, 8976,A4 ;16764           JMP [WAIT.T2F.0]                           ; цикл для ожидания ответа конс.процессора
U 176B, 0A1A,9C ;16765           JSR [SETUP]                                ; установка маски,кода модуля и т.д. и очистка CSR1
U 176C, 1952,75 ;16766           MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]       ; запрос для записи по адресу памяти 200
U 176D, 329E,15 ;16767           WRITE.MEM LS[ONES]                         ; запись по адресу 200 всех 1
U 176E, B652,15 ;16768           MOV LS[#200] TO WR[0]                      ; рабочий регистр содержит 200
U 176F, 4744,15 ;16769           BIS LS[#4] TO WR[0]                        ; рабочий регистр содержит 204
U 1770, BE12,15 ;16770           MOV WR[0] TO LS[T9]                        ; запоминание адреса в промежуточной ячейке LS
U 1771, 9912,75 ;16771           MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]         ; запрос для записи в память по адресу 204
U 1772, B29C,15 ;16772           WRITE.MEM LS[ZERO]                         ; запись всех нулей по адресу 204
U 1773, 367E,15 ;16773           MOV LS[BIT31] TO WR[0]                     ; установка в рабочем регистре бита 31
U 1774, C77A,15 ;16774           BIS LS[BIT29] TO WR[0]                     ; и бита 29
U 1775, 4774,15 ;16775           BIS LS[BIT26] TO WR[0]                     ; и бита 26
U 1776, 4772,15 ;16776           BIS LS[BIT25] TO WR[0]                     ; и бита 25
U 1777, C740,15 ;16777           BIS LS[BIT0] TO WR[0]                      ; и бита 0
U 1778, BE12,15 ;16778           MOV WR[0] TO LS[T9]                        ; запоминание в промежуточной ячейке LS
U 1779, 989C,F5 ;16779           MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]        ; запрос для записи по адресу TB 0
U 177A, B212,F5 ;16780           WRITE.MEM LS[T9]                           ; установка в TB битов VALID,PROT C,MODIFY и BYTE OFFSET
                ;16781                                                      ; и отображение виртуального адреса 0 в физ.адрес 200
U 177B, 0A17,DC ;16782           JSR [WRITE.CSR1.MME]                       ; запись в CSR1 бита MME
                ;16783   LOOP.T2F.1:
U 177C, 2F82,95 ;16784           CLR WR[1]                                  ; ожидаемые данные
U 177D, 989C,F5 ;16785           MEM.REQ[WRITE.V.WCHK] ADRS[#0] DT[LONG];   запрос для записи по виртуальному адресу 0 (физич.адрес
                ;16786                                                      ; 200)
U 177E, B29C,15 ;16787           WRITE.MEM LS[ZERO]                         ; запись нулей по физ.адресу 200
U 177F, 9853,F5 ;16788           MEM.REQ[READ.P] ADRS[#200] DT[LONG]        ; запрос для чтения по адресу памяти 200
```

```
U 1780,  3022,15 ;16789           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 1781,  0869,3C ;16790           JSR [CHECK.RESULT]                       ; проверка, что данные=0
U 1782,  0977,C4 ;16791           JMP [LOOP.T2F.1]                         ; цикл при ошибке, если есть разрешение
U 1783,  FF82,15 ;16792           INC LS[ERROR.NUMBER]                     ; ошибка 2
                 ;16793   LOOP.T2F.2:
U 1784,  DF28,95 ;16794           MCOM LS[#FFFF] TO WR[1]                  ; ожидаемые данные (FFFF0000)
U 1785,  9B42,F5 ;16795           MEM.REQ[WRITE.V.WCHK] ADRS[#2] DT[LONG]; запрос для записи по виртуальному адресу 2 (физ.адрес
                 ;16796                                                    ; 202)
U 1786,  329E,15 ;16797           WRITE.MEM LS[ONES]                       ; запись единиц по физич.адресу 202
U 1787,  9853,F5 ;16798           MEM.REQ[READ.P] ADRS[#200] DT[LONG]      ; запрос для чтения по адресу памяти 200
U 1788,  3022,15 ;16799           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 1789,  0869,3C ;16800           JSR [CHECK.RESULT]                       ; проверка,что данные=FFFF0000
U 178A,  8978,44 ;16801           JMP [LOOP.T2F.2]                         ; цикл при ошибке, если есть разрешение
U 178B,  FF82,15 ;16802           INC LS[ERROR.NUMBER]                     ; ошибка 3
U 178C,  B676,15 ;16803           MOV LS[MME] TO WR[0]                     ; установка в рабочем регистре бита MME
U 178D,  C77A,15 ;16804           BIS LS[TB.PAR.DIAG] TO WR[0]             ; также установка бита TB PAR DIAG
U 178E,  8A17,FC ;16805           JSR [WRITE.CSR1]                         ; запись в CSR1
                 ;16806   LOOP.T2F.3:
U 178F,  DF28,95 ;16807           MCOM LS[#FFFF] TO WR[1]                  ; ожидаемые данные
U 1790,  9B9C,F5 ;16808           MEM.REQ[WRITE.V.WCHK] ADRS[#0] DT[LONG]; запрос для записи по виртуальному адресу 0 (физ.адрес
                 ;16809                                                    ; 200)
U 1791,  B29C,15 ;16810           WRITE.MEM LS[ZERO]                       ; попытка записать 0 по физ.адресу 200
U 1792,  9853,F5 ;16811           MEM.REQ[READ.P] ADRS[#200] DT[LONG]      ; запрос для чтения из адреса 200
U 1793,  3022,15 ;16812           MOV MEM.DATA TO WR[0]                    ; выборка результата
U 1794,  0869,3C ;16813           JSR [CHECK.RESULT]                       ; проверка, что данные=FFFF0000 (не изменились)
U 1795,  0978,F4 ;16814           JMP [LOOP.T2F.3]                         ; цикл при ошибке, если есть разрешение
                 ;16815   END.T2F:
                 ;16816
```

```
;16817  .PAGE "ТЕСТ 30 - микропрограмма записи при запрете коррекции или ошибке (модуль MCT)"
;16818  ;
;16819  ; ОПИСАНИЕ ТЕСТА:
;16820  ;
;16821  ;       Этот тест предназначен для обнаружения неисправности в ПЗУ управляющей
;16822  ;    памяти методом проверки всех путей в микрокодах. Тест проверяет путь,
;16823  ;    выбираемый, если выдается функция WRITE.P При установленном бите ECC DIS
;16824  ;    (запрет коррекции ошибок). Вторая часть этого теста проверяет путь в микро-
;16825  ;    кодах для случая, если ошибка в одиночном бите или двойная ошибка возникает
;16826  ;    в части чтения операции записи.
;16827  ;       Если перед записью установлен бит ECC DIS, микрокод записывает в CSR0 конт-
;16828  ;    рольные биты, которые генерируются для записываемых данных. В этом заключает-
;16829  ;    ся единственное отличие от записи при очищенном ECC DIS. Это проверяется пу-
;16830  ;    тем записи в память данных из всех 0 и данных = 1 при установленном бите ECC
;16831  ;    DIS и проверкой CSR0 на правильность контрольных битов, а также проверкой
;16832  ;    ячейки памяти на правильность данных. Этот тест выполняется дважды, один
;16833  ;    раз при WRITE.MEM, следующем непосредственно за MEM.REQ и один раз с задерж-
;16834  ;    кой из 5 нерабочих циклов между этими двумя частями операции записи. Это по-
;16835  ;    зволяет проверить два отличающихся пути.
;16836  ;       Вторая часть этого теста проверяет путь, когда ошибка данных возникает во
;16837  ;    время записи невыравненного длинного слова (выравненное длиное слово записы-
;16838  ;    вает полностью новые контрольные биты и в этом случае не имеет значения,
;16839  ;    какими были старые).
;16840  ;
;16841  ; ПРЕДПОЛОЖЕНИЯ:
;16842  ;
;16843  ;    Принимается, что все предыдущие тесты прошли успешно, и что вся аппаратура
;16844  ;    (кроме ПЗУ управляющей памяти MCT) работает правильно.
;16845  ;
;16846  ; ШАГИ ТЕСТА:
;16847  ;
;16848  ;    1) Занесение в LS маски ошибок, номера ошибки и номера модуля (для распечатки
;16849  ;       ошибок) и очистка в LS номера предыдущей ошибки.
;16850  ;    2) Установка в CSR1 бита ECC DIS. Запись данных из всех 0 по адресу памяти 0.
;16851  ;       Чтение CSR0 и проверка контрольных битов для данных = 0(0111100).
;16852  ;    3) Чтение из адреса 0 и проверка на все 0.
;16853  ;    4) Запись в память данных = 1 по адресу 0. Чтение CSR0 и проверка контрольнх
;16854  ;       битов для данных = 1 (1100100).
;16855  ;    5) Чтение из адреса 0 и проверка, что данные = 1.
;16856  ;    6) Повторение шагов от 2 до 5 с задержкой из 5 холостых циклов между MEM.REQ
;16857  ;       и WRITE.MEM при записи данных. Этим проверяются разные пути в микрокодах,
;16858  ;       поскольку сигнал CPU DR задержан.
;16859  ;    7) Запись 1 в ячейку памяти 0 с контрольными битами для данных 0, при исполь-
;16860  ;       зовании программы MAINT.ECC.DATA. Запись 10000 по адресу 4 с контрольны-
;16861  ;       ми битами для всех 0. Этим вносится одиночная ошибка в обе ячейки для
;16862  ;       длинного слова.
;16863  ;    8) Запись невыравненного длинного слова из всех 1 в ячейку 2.
;16864  ;    9) Чтение ячейки 0 и проверка, что данные = FFFF0000.
;16865  ;   10) Чтение ячейки 4 и проверка, что данные = 0000FFFF.
;16866  ;   11) Повторение шагов от 7 до 10 с задержкой из 5 холостых циклов между MEM.REQ
;16867  ;       и WRITE.MEM во время записи данных. Этим проверяется другой путь через
;16868  ;       микрокоды, так как сигнал CPU DR задержан.
;16869  ;   12) Запись значения 3 в ячейку памяти 0 с контрольными битами для данных=0 при
;16870  ;       использовании программы MAINT.ECC.DATA. Запись всех 0 по адресу 4 с пра-
;16871  ;       вильными контрольными битами. Этим по адресу 0 вносится двойная ошибка.
```

```
;16872  ;  13) Запись невыравненного длинного слова из всех 1 в ячейку 2.
;16873  ;  14) Чтение ячейки 0 и проверка, что данные =3 (не изменены, так как запись
;16874  ;      прекращена).
;16875  ;  15) Чтение ячейки 4 и проверка данных на все 0.
;16876  ;  16) Повторение шагов от 12 до 14 с задержкой из 5 холостых циклов между
;16877  ;      MEM.REQ и WRITE.MEM во время записи данных. Этим проверяется другой путь
;16878  ;      в микрокодах, так как сигнал CPU DR задержан. На этот раз проверяется
;16879  ;      только первая ячейка, так как путь отличается только в первом цикле двух-
;16880  ;      цикловой записи.
;16881  ;  17) Запись значения 3 в ячейку памяти 4 с контрольными битами для данных=0,
;16882  ;      при использовании программы MAINT.ECC.DATA. Запись всех 0 в ячейку памяти-
;16883  ;      0 с правильными контрольными битами. Этим по адресу 4 вносится двойная
;16884  ;      ошибка.
;16885  ;  18) Запись в ячеику 2 невыравненного длинного слова из всех 1.
;16886  ;  19) Чтение ячейки 0 и проверка, что данные = FFFF0000 (первый цикл будет за-
;16887  ;      вершен, так, как ошибка возникает на втором цикле).
;16888  ;  20) Чтение ячейки 4 и проверка, что данные = 3 (запись прекращена).
;16889  ;
;16890  ; ОШИБКИ:
;16891  ;
;16892  ;   ПРИМЕЧАНИЕ: ожидаемые и полученные данные представляют собой содержимое модуля
;16893  ;   памяти при маске ошибок=0 и данные из CSR0 при маске ошибок FFFFFF80.
;16894  ;
;16895  ;   ошибка 1 - функция WRITE.P при установленном бите ECC DIS не загрузила в CSR0
;16896  ;              правильных контрольных битов (0111100).
;16897  ;   ошибка 2 - функция WRITE.P при установленном бите ECC DIS не выполнила пра-
;16898  ;              вильной записи в память данных из всех 0.
;16899  ;   ошибка 3 - функция WRITE.P при установленном бите ECC DIS не загрузила в CSR0
;16900  ;              правильных контрольных битов (1100100).
;16901  ;   ошибка 4 - функция WRITE.P при установленном бите ECC DIS не выполнила пра-
;16902  ;              вильной записи в памяти данных=1.
;16903  ;   ошибка от 5 до 8 - то же, что и ошибки от 1 до 4, за исключением того, что
;16904  ;              выбирается другой путь.
;16905  ;   ошибка 9 - функция WRITE.P для невыравненного длинного слова не выполнила
;16906  ;              правильной записи по адресу 0, когда адрес 0 содержал одиночную
;16907  ;              ошибку.
;16908  ;   ошибка A - функция WRITE.P для невыравненного длинного слова не выполнила
;16909  ;              правильной эписи по адресу 4, когда адреса 0 и 4 содержали одиноч-
;16910  ;              ные ошибки.
;16911  ;   ошибка B и C - то же, что и ошибки 9 и A выше, за исключением того, что CPU DR
;16912  ;              задержан.
;16913  ;   ошибка D - функция WRITE.P для невыравненного длинного слова не прекратила
;16914  ;              записи на первом цикле записи при двойной ошибке.
;16915  ;   ошибка E - функция WRITE.P для невыравненного длинного слова не прекратила
;16916  ;              записи на втором цикле, когда двойная ошибка возникла на первом
;16917  ;              цикле.
;16918  ;   ошибка F - то же, что и ошибка D, за исключением задержанного CPU DR.
;16919  ;   ошибка 10- функция WRITE.P для невыравненного длинного слова не выполнила
;16920  ;              правильной записи в первом цикле, когда двойная ошибка содержа-
;16921  ;              лась во втором цикле.
;16922  ;   ошибка 11- функция WRITE.P для невыравненного длинного слова не прекратила
;16923  ;              записи во втором цикле, когда двойная ошибка содержалась во вто-
;16924  ;              ром цикле.
;16925  ;
;16926  ; НАЛАДКА:
```

```
;16927  ;
;16928  ;   ПРИМЕЧАНИЕ: следующие ошибки с найбольшей вероятностью указывают на отказ
;16929  ;   в потоках микрокодов. Аппаратура, используемя микрокодами, ранее проверялась.
;16930  ;   Чтобы локализовать неисправность, необходимо проверить последовательности, ука-
;16931  ;   занные ниже. Если последовательности правильные, можно подозревать дефектый
;16932  ;   бит в одном из ПЗУ управляющей памяти и следует проверить каждое состояние на
;16933  ;   правильные разрешающие сигналы.
;16934  ;
;16935  ;      ОШИБКА 1 - Эта ошибка указывает на отказ в микропрограмме, связанный с фун-
;16936  ;   кцией WRITE.P при установленном бите ECC DIS. Путь до WR ARY ALIGN LW C8 ис-
;16937  ;   пользовался ранее. Отсюда в следующем слове должно произойти ветвление по
;16938  ;   ECC DIS к слову, которое возбуждает CSR CB+SYN CLK. Отсюда микропрограмма
;16939  ;   должна перейти к WR ARY ALIGN LW C11.
;16940  ;
;16941  ;      ОШИБКА 2 - То же, что и при ошибке 1. Можно подозревать, что неправильно
;16942  ;   работает что-либо в цикле, который возбуждает CSR CB+SYN CLK.
;16943  ;
;16944  ;      ОШИБКА 3 - То же, что и при ошибке 1, но с другими данными.
;16945  ;
;16946  ;      ОШИБКА 4 - То же, что и при ошибке 2, но с другими данными.
;16947  ;
;16948  ;      ОШИБКА 5 - Здесь микропрограмма выбирает другой путь, чем при ошибках выше,
;16949  ;   так как в цикле WR ARY ALIGN LW C7 не возбужден CPU DR. Микропрограмма до WR
;16950  ;   ARY C10 использовалась ранее. Отсюда должно быть выбрано ветвление по ECC
;16951  ;   DIS к WR ARY C11 ECC DIS. Тогда микропрограмма должна перейти к WR ARY C12.
;16952  ;
;16953  ;      ОШИБКА 6 - то же, что и при ошибке 5. Можно подозревать, что неправильно
;16954  ;   работает что-то в цикле WR ARY C11 ECC DIS.
;16955  ;
;16956  ;      ОШИБКА 7 - То же, что и при ошибке 5, но с другими данными.
;16957  ;
;16958  ;      ОШИБКА 8 - То же, что и при ошибке 6, но с другими данными.
;16959  ;
;16960  ;      ОШИБКА 9 - Здесь микропрограмма должна выбрать путь при ERR и CPU DR из
;16961  ;   цикла WR ARY C7. Слово, к которому происходит переход, должно выбирать путь
;16962  ;   при SER к WR ARY WSDE C8A, через WR ARY WSDE C9 к WR ARY C8. Остальная часть
;16963  ;   первого цикла записи уже проверялась.
;16964  ;
;16965  ;      ОШИБКА A - Здесь микропрограмма должна выбрать путь при ERR из WR ARY C20
;16966  ;   через WR ARY WDE C21, через WR ARY WSDE C22, через WR ARY WSDE C23 к WR ARY
;16967  ;   C21. Остальная часть второго цикла записи ранее проверялась.
;16968  ;
;16969  ;      ОШИБКА B - То же, что и при ошибке 9, за исключением того, что выбранным яв-
;16970  ;   ляется ветвление по ERR и не CPU DR к WR ARY WDE C8 и к WR ARY WSDE C8A. От-
;16971  ;   сюда также, как и при ошибке 9.
;16972  ;
;16973  ;      ОШИБКА C - То же, что и при ошибке A.
;16974  ;
;16975  ;      ОШИБКА D - Здесь микропрограмма должна выбрать путь из WR ARY C7 при ERR и
;16976  ;   CPU DR. Следующее слово должно выбрать путь при UNC ERR к ABORT WR WUDE,
;16977  ;   через ABORT WR.WUDE C1, через ABORT WR WUDE C2, через OCTA WRITE C 17A, через
;16978  ;   OCTA WRITE EXIT к IDLE.
;16979  ;
;16980  ;      ОШИБКА E - То же, что и при ошибке D. Не должно произойти второго цикла за-
;16981  ;   писи.
```

```
                ;16982  ;
                ;16983  ;       ОШИБКА F - То же, что и при ошибке D, за исключением того, что должен быть вы-
                ;16984  ;    бран путь по ERR и не CPU DR к WR ARY WDE C8 и к ABORT WR.WUDE.
                ;16985  ;
                ;16986  ;       ОШИБКА 10 - Здесь микропрограмма должна выбрать путь при не ERR и CPU DR из
                ;16987  ;    цикла WR ARY C7 и продолжаться, как обычно.
                ;16988  ;
                ;16989  ;       ОШИБКА 11 - Здесь из WR ARY WDE C21 должен выбираться путь по не SER к инст-
                ;16990  ;    рукции, с которой имеется переход к ABORT WR WUDE C1. Остальной путь такой
                ;16991  ;    же, как и при ошибке D после ABORT WR WUDE C1.
                ;16992  ;
                ;16993  T.30:
U 1796, B65E,15 ;16994          MOV LS[BEGIN.TEST] TO WR[0]             ; установка в WR0 бита 15 для слова управления и
                ;16995                                                  ; состояния
U 1797, 3E80,15 ;16996          MOV WR[0] TO LS[CONTROL.STATUS]         ; установка бита 15 в слове управления и состояния. Бит
                ;16997                                                  ; 15 указывает для консольного процессора начало теста
U 1798, 10E0,15 ;16998          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;16999  WAIT.T30.0:
U 1799, 8979,94 ;17000          JMP [WAIT.T30.0]                        ; цикл для ожидания ответа консольного процессора
U 179A, 0A1A,AC ;17001          JSR [SETUP.1]                           ; установка маски, кода модуля и т.д.
U 179B, 3713,15 ;17002          MOV LS[#FFFFFF80] TO WR[2]              ; установлены биты 7-31. Маска ошибки для проверки битов
                ;17003                                                  ; 0-6
U 179C, B641,95 ;17004          MOV LS[#1] TO WR[3]                     ; рабочий регистр содержит 1 для номера ошибки
U 179D, 3672,15 ;17005          MOV LS[ECC.DIS] TO WR[0]                ; установка в рабочем регистре ECC DIS
U 179E, 8A17,FC ;17006          JSR [WRITE.CSR1]                        ; запись бита ECC DIS в CSR 1
                ;17007  LOOP.T30.2:
U 179F, BE83,95 ;17008          MOV WR[3] TO LS[ERROR.NUMBER]           ; запоминание номера ошибки в LS
U 17A0, BE88,15 ;17009          MOV WR[2] TO LS[ERROR.MASK]             ; проверка только битов 0-6
                ;17010  LOOP.T30.1:
U 17A1, 3700,95 ;17011          MOV LS[CKBTS.0111100] TO WR[1]          ; инвертирование ожидаемых данных
U 17A2, A0C2,95 ;17012          COM WR[1]                               ; ожидаемые контрольные биты
U 17A3, 999C,75 ;17013          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи по адресу памяти 0
U 17A4, B29C,15 ;17014          WRITE.MEM LS[ZERO]                      ; запись всех 0 по адресу 0
U 17A5, 9D9D,75 ;17015          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ;
U 17A6, 3022,15 ;17016          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR0
U 17A7, 0869,3C ;17017          JSR [CHECK.RESULT]                      ; проверка правильности контрольных битов
U 17A8, 097A,14 ;17018          JMP [LOOP.T30.1]                        ; цикл при ошибке, если есть разрешение
U 17A9, FF82,15 ;17019          INC LS[ERROR.NUMBER]                    ; ошибка 2
U 17AA, 2F82,95 ;17020          CLR WR[1]                               ; ожидаемые данные
U 17AB, E58A,15 ;17021          CLR LS[ERROR.MASK]                      ; проверка всех битов
U 17AC, 189D,F5 ;17022          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения по адресу 0
U 17AD, 3022,15 ;17023          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 17AE, 0869,3C ;17024          JSR [CHECK.RESULT]                      ; проверка, что данные=0
U 17AF, 8979,F4 ;17025          JMP [LOOP.T30.2]                        ; цикл при ошибке, если есть разрешение
U 17B0, FF82,15 ;17026          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 17B1, 3683,95 ;17027          MOV LS[ERROR.NUMBER] TO WR[3]           ; запоминание номера ошибки в WR3
                ;17028  LOOP.T30.4:
U 17B2, BE83,95 ;17029          MOV WR[3] TO LS[ERROR.NUMBER]           ; занесение номера ошибки в LS
U 17B3, BE88,15 ;17030          MOV WR[2] TO LS[ERROR.MASK]             ; проверка только битов 0-6
                ;17031  LOOP.T30.3:
U 17B4, B704,95 ;17032          MOV LS[CKBTS.1100100] TO WR[1]          ; инвертирование ожидаемых данных
U 17B5, A0C2,95 ;17033          COM WR[1]                               ; ожидаемые контрольные биты
U 17B6, 999C,75 ;17034          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи по адресу памяти 0
U 17B7, 3240,15 ;17035          WRITE.MEM LS[#1]                        ; запись единицы по адресу 0
U 17B8, 9D9D,75 ;17036          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ;
```

```
U 17B9, 3022,15 ;17037          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR0
U 17BA, 0869,3C ;17038          JSR [CHECK.RESULT]                      ; проверка правильности контрольных битов
U 17BB, 897B,44 ;17039          JMP [LOOP.T30.3]                        ; цикл при ошибке, если есть разрешение
U 17BC, FF82,15 ;17040          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 17BD, 3640,95 ;17041          MOV LS[#1] TO WR[1]                     ; ожидаемые данные
U 17BE, E58A,15 ;17042          CLR LS[ERROR.MASK]                      ; проверка всех битов
U 17BF, 189D,F5 ;17043          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения из адреса 0
U 17C0, 3022,15 ;17044          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 17C1, 0869,3C ;17045          JSR [CHECK.RESULT]                      ; проверка, что данные=1
U 17C2, 897B,24 ;17046          JMP [LOOP.T30.4]                        ; цикл при ошибке, если есть разрешение
U 17C3, FF82,15 ;17047          INC LS[ERROR.NUMBER]                    ; ошибка 5
U 17C4, 3683,95 ;17048          MOV LS[ERROR.NUMBER] TO WR[3]           ; запоминание номера ошибки в WR3
                ;17049  LOOP.T30.6:
U 17C5, BE83,95 ;17050          MOV WR[3] TO LS[ERROR.NUMBER]           ; занесение номера ошибки в LS
U 17C6, BE8B,15 ;17051          MOV WR[2] TO LS[ERROR.MASK]             ; проверка только битов 0-6
                ;17052  LOOP.T30.5:
U 17C7, 3700,95 ;17053          MOV LS[CKBTS.01111100] TO WR[1]         ; инвертирование ожидаемых данных
U 17C8, A0C2,95 ;17054          COM WR[1]                               ; ожидаемые контрольные биты
U 17C9, 999C,75 ;17055          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи по адресу памяти 0
U 17CA, 0A1B,4C ;17056          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания выдачи
                ;17057                                                  ; сигнала CPU DR
U 17CB, B29C,15 ;17058          WRITE.MEM LS[ZERO]                      ; запись всех 0 по адресу 0
U 17CC, 9D9D,75 ;17059          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ;
U 17CD, 3022,15 ;17060          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR0
U 17CE, 0869,3C ;17061          JSR [CHECK.RESULT]                      ; проверка, что контрольные биты правильные
U 17CF, 097C,74 ;17062          JMP [LOOP.T30.5]                        ; цикл при ошибке, если есть разрешение
U 17D0, FF82,15 ;17063          INC LS[ERROR.NUMBER]                    ; ошибка 6
U 17D1, 2F82,95 ;17064          CLR WR[1]                               ; ожидаемые данные
U 17D2, E58A,15 ;17065          CLR LS[ERROR.MASK]                      ; проверка всех битов
U 17D3, 189D,F5 ;17066          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения из адреса 0
U 17D4, 3022,15 ;17067          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 17D5, 0869,3C ;17068          JSR [CHECK.RESULT]                      ; проверка, что данные =0
U 17D6, 897C,54 ;17069          JMP [LOOP.T30.6]                        ; цикл при ошибке, если есть разрешение
U 17D7, FF82,15 ;17070          INC LS[ERROR.NUMBER]                    ; ошибка 7
U 17D8, 3683,95 ;17071          MOV LS[ERROR.NUMBER] TO WR[3]           ; запоминание номера ошибки в WR3
                ;17072  LOOP.T30.8:
U 17D9, BE83,95 ;17073          MOV WR[3] TO LS[ERROR.NUMBER]           ; занесение номера ошибки в LS
U 17DA, BE8B,15 ;17074          MOV WR[2] TO LS[ERROR.MASK]             ; проверка только битов 6-0
                ;17075  LOOP.T30.7:
U 17DB, B704,95 ;17076          MOV LS[CKBTS.1100100] TO WR[1]          ; инвертирование ожидаемых данных
U 17DC, A0C2,95 ;17077          COM WR[1]                               ; ожидаемые контрольные биты
U 17DD, 999C,75 ;17078          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]      ; запрос для записи в память по адресу 0
U 17DE, 0A1B,4C ;17079          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания
                ;17080                                                  ; возбуждения сигнала CPU DR
U 17DF, 3240,15 ;17081          WRITE.MEM LS[#1]                        ; запись единицы по адресу 0
U 17E0, 9D9D,75 ;17082          MEM.REQ[READ.CSR] ADRS[CSR0] DT[LONG]   ;
U 17E1, 3022,15 ;17083          MOV MEM.DATA TO WR[0]                   ; выборка содержимого CSR0
U 17E2, 0869,3C ;17084          JSR [CHECK.RESULT]                      ; проверка, что контрольные биты правильные
U 17E3, 897D,B4 ;17085          JMP [LOOP.T30.7]                        ; цикл при ошибке, если есть разрешение
U 17E4, FF82,15 ;17086          INC LS[ERROR.NUMBER]                    ; ошибка 8
U 17E5, 3640,95 ;17087          MOV LS[#1] TO WR[1]                     ; ожидаемые данные
U 17E6, E58A,15 ;17088          CLR LS[ERROR.MASK]                      ; проверка всех битов
U 17E7, 189D,F5 ;17089          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения из адреса 0
U 17E8, 3022,15 ;17090          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 17E9, 0869,3C ;17091          JSR [CHECK.RESULT]                      ; проверка, что данные=1
```

```
U 17EA, 097D,94 ;17092          JMP [LOOP.T30.B]                                ; цикл при ошибке, если есть разрешение
U 17EB, B700,15 ;17093          MOV LS[CKBTS.01111100] TO WR[0]                 ; выборка в WR0 контрольных битов для данных=0
U 17EC, A0C0,15 ;17094          COM WR[0]                                       ; инвертирование контрольных битов для согласования с
                ;17095                                                          ; аппаратурой
U 17ED, 452C,15 ;17096          BIC LS[#FFFFFF00] TO WR[0]                      ; очистка всех битов, кроме младшего
U 17EE, 8A17,FC ;17097          JSR [WRITE.CSR1]                                ; запись в CSR1 контрольных битов для данных=0 очистка
                ;17098                                                          ; всех других битов CSR1
U 17EF, 1D40,F5 ;17099          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической
                ;17100                                                          ; программы. LVA00 установлен для ветвления
U 17F0, 3240,15 ;17101          WRITE.MEM LS[#1]                                ; запись 1 в ячейку 0 с контрольными битами для данных
                ;17102                                                          ; из всех 0
U 17F1, 3644,15 ;17103          MOV LS[#4] TO WR[0]                             ; рабочий регистр содержит 4
U 17F2, 2040,15 ;17104          INC WR[0]                                       ; рабочий регистр содержит 5
U 17F3, BE12,15 ;17105          MOV WR[0] TO LS[T9]                             ; запоминание адреса в LS (4 плюс установленный бит 0)
U 17F4, 9D12,F5 ;17106          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы
                ;17107                                                          ; LVA00 установлен для ветвления
U 17F5, B260,15 ;17108          WRITE.MEM LS[#10000]                            ; запись в ячейку 4 значения 10000 с контрольными битами
                ;17109                                                          ; для данных из всех 0
U 17F6, FF82,15 ;17110          INC LS[ERROR.NUMBER]                            ; ошибка 9
U 17F7, 3683,95 ;17111          MOV LS[ERROR.NUMBER] TO WR[3]                   ; запоминание номера ошибки в рабочем регистре
U 17F8, E58A,15 ;17112          CLR LS[ERROR.MASK]                              ; проверка всех битов
                ;17113  LOOP.T30.A:
U 17F9, BE83,95 ;17114          MOV WR[3] TO LS[ERROR.NUMBER]                   ; установка номера ошибки
                ;17115  LOOP.T30.9:
U 17FA, DF28,95 ;17116          MCOM LS[#FFFF] TO WR[1]                         ; ожидаемые данные (FFFF0000(H))
U 17FB, 9942,75 ;17117          MEM.REQ[WRITE.P] ADRS[#2] DT[LONG]              ; запрос для записи невыравненного длинного слова
U 17FC, 329E,15 ;17118          WRITE.MEM LS[ONES]                              ; запись единиц в невыравненное длинное слово
U 17FD, 189D,F5 ;17119          MEM.REQ[READ.P] ADRS[#0] DT[LONG]               ; запрос для чтения из адреса 0
U 17FE, 3022,15 ;17120          MOV MEM.DATA TO WR[0]                           ; выборка результата
U 17FF, 0869,3C ;17121          JSR [CHECK.RESULT]                              ; проверка на отсутствие ошибок
U 1800, 897F,A4 ;17122          JMP [LOOP.T30.9]                                ; цикл при ошибке, если есть разрешение
U 1801, FF82,15 ;17123          INC LS[ERROR.NUMBER]                            ; ошибка A
U 1802, B628,95 ;17124          MOV LS[#FFFF] TO WR[1]                          ; ожидаемые данные (0000FFFF(H))
U 1803, 1845,F5 ;17125          MEM.REQ[READ.P] ADRS[#4] DT[LONG]               ; запрос для чтения второго длинного слова
U 1804, 3022,15 ;17126          MOV MEM.DATA TO WR[0]                           ; чтение из адреса 4
U 1805, 0869,3C ;17127          JSR [CHECK.RESULT]                              ; проверка правильности данных
U 1806, 897F,94 ;17128          JMP [LOOP.T30.A]                                ; цикл при ошибке, если есть разрешение
U 1807, B700,15 ;17129          MOV LS[CKBTS.01111100] TO WR[0]                 ; выборка в WR0 контрольных битов для данных=0
U 1808, A0C0,15 ;17130          COM WR[0]                                       ; инвертирование контрольных битов для согласования с
                ;17131                                                          ; аппаратурой
U 1809, 452C,15 ;17132          BIC LS[#FFFFFF00] TO WR[0]                      ; очистка всех битов, кроме младшего
U 180A, 8A17,FC ;17133          JSR [WRITE.CSR1]                                ; запись в CSR1 контрольных битов для данных=0 очистка
                ;17134                                                          ; всех других битов CSR1
U 180B, 1D40,F5 ;17135          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической программы.
                ;17136                                                          ; LVA00 установлен для ветвления
U 180C, 3240,15 ;17137          WRITE.MEM LS[#1]                                ; запись единицы в ячейку 0 с контрольными битами для
                ;17138                                                          ; данных из всех 0
U 180D, 9D12,F5 ;17139          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы.
                ;17140                                                          ; LVA00 установлен для ветвления
U 180E, B260,15 ;17141          WRITE.MEM LS[#10000]                            ; запись 10000 в ячейку 4 с контрольными битами для
                ;17142                                                          ; данных из всех 0
U 180F, FF82,15 ;17143          INC LS[ERROR.NUMBER]                            ; ошибка B
U 1810, 3683,95 ;17144          MOV LS[ERROR.NUMBER] TO WR[3]                   ; запоминание номера ошибки в рабочем регистре
U 1811, 9944,75 ;17145          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]              ; запрос для записи второго длинного слова
U 1812, B29C,15 ;17146          WRITE.MEM LS[ZERO]                              ; запись нулей по адресу 4
```

```
U 1813, E5BA,15 ;17147          CLR LS[ERROR.MASK]                      ; проверка всех битов
                ;17148  LOOP.T30.C:
U 1814, BE83,95 ;17149          MOV WR[3] TO LS[ERROR.NUMBER]           ; установка номера ошибки
                ;17150  LOOP.T30.B:
U 1815, DF28,95 ;17151          MCOM LS[#FFFF] TO WR[1]                 ; ожидаемые данные (FFFF0000(H))
U 1816, 9942,75 ;17152          MEM.REQ[WRITE.P] ADRS[#2] DT[LONG]      ; запрос для записи невыравненного длинного слова
U 1817, 0A1B,4C ;17153          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания CPU DR
U 1818, 329E,15 ;17154          WRITE.MEM LS[ONES]                      ; запись единиц в невыравненное длинное слово
U 1819, 189D,F5 ;17155          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения по адресу 0
U 181A, 3022,15 ;17156          MOV MEM.DATA TO WR[0]                   ; выдача результата
U 181B, 0869,3C ;17157          JSR [CHECK.RESULT]                      ; проверка наличия ошибок
U 181C, 0981,54 ;17158          JMP [LOOP.T30.B]                        ; цикл при ошибке, если есть разрешение
U 181D, FF82,15 ;17159          INC LS[ERROR.NUMBER]                    ; ошибка C
U 181E, B628,95 ;17160          MOV LS[#FFFF] TO WR[1]                  ; ожидаемые данные (0000FFFF(H))
U 181F, 1845,F5 ;17161          MEM.REQ[READ.P] ADRS[#4] DT[LONG]       ; запрос для чтения второго длинного слова
U 1820, 3022,15 ;17162          MOV MEM.DATA TO WR[0]                   ; чтение из адреса 4
U 1821, 0869,3C ;17163          JSR [CHECK.RESULT]                      ; проверка правильности данных
U 1822, B981,44 ;17164          JMP [LOOP.T30.C]                        ; цикл при ошибке, если есть разрешение
U 1823, FF82,15 ;17165          INC LS[ERROR.NUMBER]                    ; ошибка D
U 1824, 3683,95 ;17166          MOV LS[ERROR.NUMBER] TO WR[3]           ; сохранение номера ошибки в рабочем регистре
U 1825, 1D40,F5 ;17167          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической программы.
                ;17168                                                  ; LVA00 установлен для ветвления
U 1826, B2C0,15 ;17169          WRITE.MEM LS[#3(H)]                     ; запись значения 3 в ячейку 0 с контрольными битами для
                ;17170                                                  ; данных из всех 0 (двойная ошибка)
U 1827, 9944,75 ;17171          MEM.REQ[WRITE.P] ADRS[#4] DT[LONG]      ; запрос для записи по адресу 4
U 1828, B29C,15 ;17172          WRITE.MEM LS[ZERO]                      ; запись нулей по адресу 4
                ;17173  LOOP.T30.E:
U 1829, BE83,95 ;17174          MOV WR[3] TO LS[ERROR.NUMBER]           ; восстановление номера ошибки
                ;17175  LOOP.T30.D:
U 182A, B6C0,95 ;17176          MOV LS[#3(H)] TO WR[1]                  ; ожидаемые данные (запись прекращена)
U 182B, 9942,75 ;17177          MEM.REQ[WRITE.P] ADRS[#2] DT[LONG]      ; запрос для записи невыравненных данных
U 182C, 329E,15 ;17178          WRITE.MEM LS[ONES]                      ; попытка записи единиц (двойная ошибка по адресу 0
                ;17179                                                  ; должна предотвратить запись)
U 182D, 189D,F5 ;17180          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения из адреса 0
U 182E, 3022,15 ;17181          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 182F, 0869,3C ;17182          JSR [CHECK.RESULT]                      ; проверка наличия ошибок
U 1830, 0982,A4 ;17183          JMP [LOOP.T30.D]                        ; цикл при ошибке, если есть разрешение
U 1831, FF82,15 ;17184          INC LS[ERROR.NUMBER]                    ; ошибка E
U 1832, 2F82,95 ;17185          CLR WR[1]                               ; ожидаемые данные
U 1833, 1845,F5 ;17186          MEM.REQ[READ.P] ADRS[#4] DT[LONG]       ; запрос для чтения из адреса 4
U 1834, 3022,15 ;17187          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 1835, 0869,3C ;17188          JSR [CHECK.RESULT]                      ; проверка наличия ошибок
U 1836, 0982,94 ;17189          JMP [LOOP.T30.E]                        ; цикл при ошибке, если есть разрешение
U 1837, FF82,15 ;17190          INC LS[ERROR.NUMBER]                    ; ошибка F
U 1838, 1D40,F5 ;17191          MEM.REQ[MAINT.ECC.DATA] ADRS[#1] DT[LONG] ; запрос для использования диагностической программы.
                ;17192                                                  ; LVA00 установлен для ветвления
U 1839, B2C0,15 ;17193          WRITE.MEM LS[#3(H)]                     ; запись значения 3 в ячейку 0 с контрольными битами для
                ;17194                                                  ; данных из всех 0 (двойная ошибка)
                ;17195  LOOP.T30.F:
U 183A, B6C0,95 ;17196          MOV LS[#3(H)] TO WR[1]                  ; ожидаемые данные (запись прекращена)
U 183B, 9942,75 ;17197          MEM.REQ[WRITE.P] ADRS[#2] DT[LONG]      ; запрос для записи невыравненных данных
U 183C, 0A1B,4C ;17198          JSR [NOP.DELAY]                         ; выполнение задержки из 5 циклов для задержания CPU DR
U 183D, 329E,15 ;17199          WRITE.MEM LS[ONES]                      ; попытка записи единиц (двойная ошибка по адресу 0
                ;17200                                                  ; должна предотвратить запись)
U 183E, 189D,F5 ;17201          MEM.REQ[READ.P] ADRS[#0] DT[LONG]       ; запрос для чтения по адресу 0
```

```
U 183F, 3022,15 ;17202          MOV MEM.DATA TO WR[0]                       ; выборка результата
U 1840, 0869,3C ;17203          JSR [CHECK.RESULT]                          ; проверка наличия ошибок
U 1841, B983,A4 ;17204          JMP [LOOP.T30.F]                            ; цикл при ошибке, если есть разрешение
U 1842, FF82,15 ;17205          INC LS[ERROR.NUMBER]                        ; ошибка 10
U 1843, 3683,95 ;17206          MOV LS[ERROR.NUMBER] TO WR[3]               ; сохранение номера ошибки в рабочем регистре
U 1844, 9D12,F5 ;17207          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы
                ;17208                                                      ; LVA00 установлен для ветвления
U 1845, B2C0,15 ;17209          WRITE.MEM LS[#3(H)]                         ; запись значения 3 в ячейку 4 с контрольными битами для
                ;17210                                                      ; данных из всех 0 (двойная ошибка)
U 1846, 999C,75 ;17211          MEM.REQ[WRITE.P] ADRS[#0] DT[LONG]          ; запрос для записи по адресу 0
U 1847, B29C,15 ;17212          WRITE.MEM LS[ZERO]                          ; запись нулей по адресу 0
                ;17213  LOOP.T30.11:
U 1848, BE83,95 ;17214          MOV WR[3] TO LS[ERROR.NUMBER]               ; восстановление номера ошибки
                ;17215  LOOP.T30.10:
U 1849, DF28,95 ;17216          MCOM LS[#FFFF] TO WR[1]                     ; ожидаемые данные (первая половина двухцикловой записи
                ;17217                                                      ; будет завершена)
U 184A, 9942,75 ;17218          MEM.REQ[WRITE.P] ADRS[#2] DT[LONG]          ; запрос для записи невыравненных данных
U 184B, 329E,15 ;17219          WRITE.MEM LS[ONES]                          ; запись единицы по адресу 2
U 184C, 189D,F5 ;17220          MEM.REQ[READ.P] ADRS[#0] DT[LONG]           ; запрос для чтения по адресу 0
U 184D, 3022,15 ;17221          MOV MEM.DATA TO WR[0]                       ; выборка результата
U 184E, 0869,3C ;17222          JSR [CHECK.RESULT]                          ; проверка наличия ошибок
U 184F, 0984,94 ;17223          JMP [LOOP.T30.10]                           ; цикл при ошибке, если есть разрешение
U 1850, FF82,15 ;17224          INC LS[ERROR.NUMBER]                        ; ошибка 11
U 1851, B6C0,95 ;17225          MOV LS[#3(H)] TO WR[1]                      ; ожидаемые данные (второй цикл должен быть прекращен)
U 1852, 1845,F5 ;17226          MEM.REQ[READ.P] ADRS[#4] DT[LONG]           ; запрос для чтения адреса 4
U 1853, 3022,15 ;17227          MOV MEM.DATA TO WR[0]                       ; выборка результата
U 1854, 0869,3C ;17228          JSR [CHECK.RESULT]                          ; проверка наличия ошибок
U 1855, 8984,84 ;17229          JMP [LOOP.T30.11]                           ; цикл при ошибке, если есть разрешение
                ;17230  END.T30:
```

```
;17231  .PAGE "ТЕСТ 31 - микропрограмма записи восьмикратных слов (модуль MCT)"
;17232  ;
;17233  ; ОПИСАНИЕ ТЕСТА:
;17234  ;
;17235  ;       Этот тест предназначен для обнаружения неисправностей в ПЗУ управляющей
;17236  ;   памяти методом проверки всех путей в микрокодах. Тест проверяет путь, вы-
;17237  ;   бираемый, когда выдается функция OCTA.WRITE.P или OCTA.WRITE.V.WCHK. Этот
;17238  ;   тест также проверяет путь, выбираемый, если пересечена граница страницы.
;17239  ;   Первая часть теста просто записывает фон из единиц в первых четырех длинных
;17240  ;   словах памяти. Затем выполняется OCTA.WRITE.P с эталонными данными в этих
;17241  ;   ячейках, равными соответственно 1, 2, 4, и 8. Данные проверяются микропрог-
;17242  ;   раммой READ.P. Этим проверяется большая часть микропрограммы OCTA WRITE.
;17243  ;   Этот тест повторяется с задержкой из 5 нерабочих циклов между MEM.REQ и
;17244  ;   каждой функцией WRITE.MEM для задержания CPU DR. Этим проверяется ветвление
;17245  ;   на путь при не CPU DR.
;17246  ;       Следующая часть проверяет последовательность OCTA.WR.V.WCHK, которая просто
;17247  ;   начинается точкой входа на один цикл после входа OCTA.WRITE.P. Затем также
;17248  ;   проверяется прекращение записи путем установки ошибки паритета TB.
;17249  ;       Наконец, проверяется ветвь PG BOUND, так же как и ветвление по общей ошибке
;17250  ;   (ERR SUM), которая возникает после пересечения границы страницы.
;17251  ;
;17252  ; ПРЕДПОЛОЖЕНИЯ:
;17253  ;
;17254  ;   Принимается, что все предыдущие тесты прошли успешно, и что вся аппаратура
;17255  ;   (кроме ПЗУ управляющей памяти MCT) работает правильно.
;17256  ;
;17257  ; ШАГИ ТЕСТА:
;17258  ;
;17259  ;     1) Установка в LS маски ошибок, номера ошибки и номера модуля (для распечат-
;17260  ;        ки ошибок) и очистка номера предыдущей ошибки.
;17261  ;     2) Запись фона из единиц в первых четырех длинных словах памяти.
;17262  ;     3) Запись данных, равных 1, 2, 4 и 8 в первых четырех длинных словах памяти,
;17263  ;        с использованием микропрограммы OCTA.WRITE.P.
;17264  ;     4) Проверка, что эти четыре длинных слова записаны правильно.
;17265  ;     5) Повторение шагов от 2 до 4 с задержками между инструкциями MEM.REQ и
;17266  ;        WRITE.MEM для вынуждения ветвления на путь не CPU DR в микропрограммах.
;17267  ;     6) Подготовка буфера трансляции (TB) для отображения адреса 0 в физический
;17268  ;        адрес 200. Установка в CSR1 бита MME. Запись фона единиц в 4 длинных сло-
;17269  ;        ва, начиная с адреса 200.
;17270  ;     7) Выполнение OCTA.WR.V.WCHK по виртуальчому адресу 0. Проверка результата
;17271  ;        по физическим адресам от 200 до 20F. Этим проверяется начальное ветвление
;17272  ;        микропрограммы записи восьмикратных слов по виртуальному адресу.
;17273  ;     8) Установка в CSR1 битов MME и TB PAR DIAG. Запись фона единиц по адресам
;17274  ;        200-20F.
;17275  ;     9) Выполнение OCTA.WR.V.WCHK по виртуальному адресу 0. Проверка, что запись
;17276  ;        прекращена из-за ошибки паритета TB, методом контроля, что адреса 200-20F
;17277  ;        не модифицированны.
;17278  ;    10) Установка в CSR1 бита MME. Запись по адресу 1 буфера трансляции для ото-
;17279  ;        бражения виртуального адреса 0 в физический 200 и очистка бита VALID для
;17280  ;        генерации суммарной ошибки памяти (ERR SUM). Запись по адресу 0 TB для
;17281  ;        отображения виртуального адреса 0 в физический 0 без ошибки.
;17282  ;    11) Запись всех 1 в четыре длинных слова, начинающиеся с физического адреса
;17283  ;        1FC.
;17284  ;    12) Выполнение OCTA.WR.V.WCHK, начиная с виртуального адреса 1FC. Вторая за-
;17285  ;        пись будет пересекать границу страницы для проверки ветвления по PAGE
```

```
;17286  ;       BOUND и должна быть прекращена из-за ошибки ERR SUM.
;17287  ;   13) Чтение по адресу 1FC(H) и проверка на 1(H). Чтение оставшихся слов и про-
;17288  ;       верка на отсутствие изменений (запись прекращена).
;17289  ;   14) Повторение шагов 11 и 12, начиная с адреса 1F8. Третья запись будет пере-
;17290  ;       секать границу страницы.
;17291  ;   15) Проверка адресов 1F8 и 1FC на 1 и 2 соответственно. Проверка адресов 200
;17292  ;       и 204  на все единицы (не модифицированные данные).
;17293  ;   16) Установка по адресу 1 буфера трансляции бита VALID для исключения ошибки.
;17294  ;   17) Повторение шагов 11 и 12. На этот раз прекращение записи не должно произ-
;17295  ;       ойти и все четыре длинных слова должны содержать правильные данные.
;17296  ;   18) Повторение шага 17 с задержкой между записью первого и второго слова. Этим
;17297  ;       проверяется путь не CPU DR в микропрограмме, выполняемой при границе
;17298  ;       страницы.
;17299  ;
;17300  ; ОШИБКИ:
;17301  ;
;17302  ;   ПРИМЕЧАНИЕ: данными, печатаемыми под OTHER (другие) в сообщении об ошибке,
;17303  ;               является адрес неправильно записанной ячейки.
;17304  ;
;17305  ;   ошибка 1 - функция OCTA.WRITE.P не выполнила правильной записи 4 длинных
;17306  ;              слов (CPU DR не задержан).
;17307  ;   ошибка 2 - функция OCTA.WRITE.P не выполнила правильной записи 4 длинных
;17308  ;              слов (CPU DR задержан).
;17309  ;   ошибка 3 - не выполняется OCTA.WR.V.WCHK. Можно подозревать начальное ветв-
;17310  ;              ление к точке входа.
;17311  ;   ошибка 4 - функция OCTA.WR.V.WCHK не прекратила выполнения при ошибке
;17312  ;              (ошибка паритета TB).
;17313  ;   ошибка 5 - функция OCTA.WR.V.WCHK не записала первого длинного слова.
;17314  ;   ошибка 6 - функция OCTA.WR.V.WCHK не прекратила выполнения в ветви PAGE
;17315  ;              BOUND при ошибке буфера трансляции (ошибка отсутствия истинности).
;17316  ;   ошибка 7 - функция OCTA.WR.V.WCHK не записала первых двух длинных слов.
;17317  ;   ошибка 8 - функция OCTA.WR.V.WCHK не выбрала пути PAGE BOUND с ошибкой TB
;17318  ;              (ошибка отсутствия истинности) при записи в память третьего длин-
;17319  ;              ного слова.
;17320  ;   ошибка 9 - функция OCTA.WR.V.WCHK не записала правильно 4 длинных слова при
;17321  ;              переходе через границу страницы, хотя не возникала ошибка ERR SUM.
;17322  ;   ошибка A - функция OCTA.WR.V.WCHK не записала правильно 4 длинных слова при
;17323  ;              переходе через границу, хотя не возникала ошибка ERR SUM (задер-
;17324  ;              жанный сигнал CPU DR).
;17325  ;
;17326  ; НАЛАДКА:
;17327  ;
;17328  ;   ПРИМЕЧАНИЕ: следующие ошибки с наибольшей вероятностью указывают на отказ в
;17329  ;               потоке микрокодов. Аппаратура, используемая микрокодами, ранее
;17330  ;               проверялась. Чтобы локализавать неисправность, необходимо проверить
;17331  ;               последовательность, указанную ниже. Если последовательность пра-
;17332  ;               вильная, подозревается дефектный бит в одном из ПЗУ управляющей
;17333  ;               памяти и необходимо проверить каждое состояние на правильность
;17334  ;               разрешающих сигналов.
;17335  ;
;17336  ;   ОШИБКА 1 - Это первый случай, когда проверяется запись восьмикратных слов,
;17337  ;              повтаму каждая инструкция в микропрограмме может быть причиной
;17338  ;              отказа. Путем, который необходимо проследить, является:
;17339  ;              WRITE.PH.OCTA через OCTA WRITE V WCHK к OCTA WRITE C4 и последо-
;17340  ;              вательное продолжение до OCTA WRITE C19. Следующими выполняются
```

```
                ;17341  ;              от OCTA WRITE CB до OCTA WRITE C16. Далее выбирается ветвление по
                ;17342  ;              IC0 через OCTA WRITE C17A к OCTA WRITE EXIT и к IDLE. Заметим,
                ;17343  ;              что при прохождении микропрограммы выбираются ветвления по не ERR
                ;17344  ;              SUM, не PG BOUND, L CPU DR и P2.
                ;17345  ;
                ;17346  ;   ОШИБКА 2 - То же что и при ошибке 1, за исключением того, что в микропрограмме
                ;17347  ;              будет выбран путь при не CPU DR.
                ;17348  ;
                ;17349  ;   ОШИБКА 3 - Эта ошибка с наибольшей вероятностью указывает на отказ при на-
                ;17350  ;              чальном ветвлении. Первый адрес, в который происходит переход
                ;17351  ;              при начальном ветвлении, должен быть OCTA WRITE V WCHK вместо
                ;17352  ;              WRITE PH.OCTA. Остальная часть микропрограммы такая же, как и при
                ;17353  ;              ошибке 1.
                ;17354  ;   ОШИБКА 4 - Эта ошибка указывает на отказ в ветвлении при ERR SUM из OCTA
                ;17355  ;              WRITE C6 или в стробировании суммарной ошибки в цикле OCTA WRT C5.
                ;17356  ;              Микропрограмма должна пройти из OCTA WRITE V WCHK через OCTA
                ;17357  ;              WRITE C4, через OCTA WRITE C5 к OCTA WRITE C6 и вдоль пути при
                ;17358  ;              ERR SUM к ABORT WRITE CYC C1.
                ;17359  ;   ОШИБКА 5 - Эта ошибка маловероятна. Она указывает на отказ в первой части
                ;17360  ;              микропрограммы OCTA.WR.V.WCHK, которая была проверена раньше.
                ;17361  ;              Единственным отличением является то, что при следующей записи
                ;17362  ;              будет пересечена граница страницы. Если эта ошибка появилась,
                ;17363  ;              следует снова выполнить предыдущие тесты.
                ;17364  ;   ОШИБКА 6 - Эта ошибка указывает на отказ в ветвлении при PAGE BOUND из
                ;17365  ;              OCTA WRITE C12 или в ветвлении при ERRSUM в конце пути PAGE
                ;17366  ;              BOUND. Из OCTA WRITE C12 микрокод должен выбрать путь к OCTA
                ;17367  ;              WRITE PG BOUND C14 и к циклу, который стробирует ERR SUM, и к
                ;17368  ;              циклу, который проверяет ERRSUM и CPU DR. Должен быть выбран путь
                ;17369  ;              при ERRSUM к циклу, который переходит к ABORT WRITE CYC C1.
                ;17370  ;
                ;17371  ;   ОШИБКА 7 - То же, что и при ошибке 5.
                ;17372  ;
                ;17373  ;   ОШИБКА 8 - Эта ошибка указывает на отказ в ветвлении по границе страницы
                ;17374  ;              из OCTA WRITE C18. Микрокод должен выбрать путь PG BOUND из OCTA
                ;17375  ;              WRITE C18 и перейти к циклу, который переходит к OCTA WRITE C4.
                ;17376  ;              Остальная часть пути ранее проверялась.
                ;17377  ;   ОШИБКА 9 - Эта ошибка указывает на отказ в ветвлении по не ERRSUM в пути
                ;17378  ;              PG BOUND из OCTA WRITE C12. В конце этого пути микрослово, кото-
                ;17379  ;              рое содержит ветвления по ERRSUM и CPU DR, должно выбрать путь
                ;17380  ;              CPU DR и не ERRSUM к OCTA WRITE C14.
                ;17381  ;   ОШИБКА A - Эта ошибка указывает на отказ в ветвлении по CPU DR пути PG BOUND
                ;17382  ;              из OCTA WRITE C12. В конце этого пути микрослово, которое содер-
                ;17383  ;              жит ветвления по ERRSUM и CPU DR, должно выбрать путь при не CPU
                ;17384  ;              DR и не ERRSUM к OCTA WRITE WAIT CPU DR C2.
                ;17385  ;
                ;17386
                ;17387
                ;17388  T.31:
U 1856, B65E,15 ;17389          MOV LSI[BEGIN.TEST] TO WRI[0]           ; установка в WR0 бита 15 для слова управления и
                ;17390                                                  ; состояния
U 1857, 3E80,15 ;17391          MOV WRI[0] TO LSI[CONTROL.STATUS]       ; установка бита 15 в слове управления и состояния. Бит
                ;17392                                                  ; 15 указывает для консольного процессора начало теста
U 1858, 10E0,15 ;17393          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN
                ;17394  WAIT.T31.0:
U 1859, 8985,94 ;17395          JMP [WAIT.T31.0]                        ; цикл для ожидания ответа консольного процессора
```

```
U 185A, 0A1A,9C ;17396          JSR [SETUP]                                    ; установка маски, кода модуля и т.д. и очистка CSR1
                ;17397                                                         ; МСТ
U 185B, B645,15 ;17398          MOV LS[#4] TO WR[2]                            ; значение инкремента адреса
U 185C, 3649,95 ;17399          MOV LS[#10] TO WR[3]                           ; выборка в рабочий регистр значения 10
U 185D, 0995,2C ;17400          JSR [BACK.T31]                                 ; запись всех единиц в первые 4 длинных слова
U 185E, B670,15 ;17401          MOV LS[OTHER.DATA] TO WR[0]                    ; установка бита, что в сообщении об ошибке будет печать
                ;17402                                                         ; под OTHER (другие данные)
U 185F, 3E80,15 ;17403          MOV WR[0] TO LS[CONTROL.STATUS]                ; занесение в слово управления
                ;17404  LOOP.T31.1:
U 1860, 9C9C,75 ;17405          MEM.REQ[OCTA.WRITE.P] ADRS[#0] DT[LONG];       запрос для записи в адреса памяти 0-F(H)
U 1861, 3240,15 ;17406          WRITE.MEM LS[#1]                               ; запись 1 в длинное слово по адресу 0
U 1862, B242,15 ;17407          WRITE.MEM LS[#2]                               ; запись 2 в длинное слово по адресу 4
U 1863, B244,15 ;17408          WRITE.MEM LS[#4]                               ; запись 4 в длинное слово по адресу 8
U 1864, 3246,15 ;17409          WRITE.MEM LS[#8]                               ; запись 8 в длинное слово по адресу C
U 1865, 6588,15 ;17410          CLR LS[ADDRESS.DATA]                           ; очистка адреса для печати
U 1866, E5F8,15 ;17411          CLR LS[OS]                                     ; очистка индекса
U 1867, 6512,15 ;17412          CLR LS[T9]                                     ; начало чтения с адреса 0
                ;17413  READ.T31.1:
U 1868, 36F2,95 ;17414          MOV LS[SHIFT.OS(3-0)] TO WR[1]                 ; выборка ожидаемых данных
U 1869, 1813,F5 ;17415          MEM.REQ[READ.P] ADRS[T9] DT[LONG]              ; запрос для чтения памяти
U 186A, 3022,15 ;17416          MOV MEM.DATA TO WR[0]                          ; выборка результата
U 186B, 0869,3C ;17417          JSR [CHECK.RESULT]                             ; проверка правильности данных
U 186C, 8986,04 ;17418          JMP [LOOP.T31.1]                               ; цикл при ошибке, если есть разрешение
U 186D, 7FF8,15 ;17419          INC LS[OS]                                     ; увеличение индекса
U 186E, EC13,15 ;17420          ADD WR[2] TO LS[T9]                            ; прибавление 4 к адресу памяти
U 186F, EC89,15 ;17421          ADD WR[2] TO LS[ADDRESS.DATA]                  ; прибавление 4 к адресу для печати
                ;17422            CMP WR[3] WITH LS[T9],                       ; проверка окончания (WR3 содержит 10(H))
U 1870, 4F13,B5 ;17423          DT(LONG)&SET.ALU.CC                            ; установка кодов условий
U 1871, 0986,81 ;17424          JMP.IF[NEQ] TO [READ.T31.1]                    ; повторение, пока не будет выполнено
U 1872, FF82,15 ;17425          INC LS[ERROR.NUMBER]                           ; ошибка 2
U 1873, 0995,2C ;17426          JSR [BACK.T31]                                 ; запись всех 1 в первые четыре ячейки
                ;17427  LOOP.T31.2:
U 1874, 9C9C,75 ;17428          MEM.REQ[OCTA.WRITE.P] ADRS[#0] DT[LONG];       запрос для записи в адреса памяти 0-F(H)
U 1875, E5F8,15 ;17429          CLR LS[OS]                                     ; очистка индекса
U 1876, DB00,15 ;17430          NOP                                            ; задержка CPU DR
U 1877, DB00,15 ;17431          NOP                                            ; задержка CPU DR
                ;17432  WRITE.T31.2:
U 1878, DB00,15 ;17433          NOP                                            ; задержка CPU DR
U 1879, DB00,15 ;17434          NOP                                            ; задержка CPU DR
U 187A, 32F2,15 ;17435          WRITE.MEM LS[SHIFT.OS(3-0)]                    ; запись длинного слова данных
U 187B, 7FF8,15 ;17436          INC LS[OS]                                     ; увеличение индекса
                ;17437            CMP WR[3] WITH LS[OS],                       ; проверка окончания (WR3 содержит 10(H))
U 187C, CFF9,B5 ;17438          DT(LONG)&SET.ALU.CC                            ; установка кодов условий
U 187D, 8987,81 ;17439          JMP.IF[NEQ] TO [WRITE.T31.2]                   ; повторение, пока не будет выполнено
U 187E, 6588,15 ;17440          CLR LS[ADDRESS.DATA]                           ; очистка адреса для печати
U 187F, E5F8,15 ;17441          CLR LS[OS]                                     ; очистка индекса
U 1880, 6512,15 ;17442          CLR LS[T9]                                     ; начало чтения с адреса 0
                ;17443  READ.T31.2:
U 1881, 36F2,95 ;17444          MOV LS[SHIFT.OS(3-0)] TO WR[1]                 ; выборка ожидаемых данных
U 1882, 1813,F5 ;17445          MEM.REQ[READ.P] ADRS[T9] DT[LONG]              ; запрос для чтения из памяти
U 1883, 3022,15 ;17446          MOV MEM.DATA TO WR[0]                          ; выборка результата
U 1884, 0869,3C ;17447          JSR [CHECK.RESULT]                             ; проверка, что данные правильные
U 1885, 8987,44 ;17448          JMP [LOOP.T31.2]                               ; цикл при ошибке, если есть разрешение
U 1886, 7FF8,15 ;17449          INC LS[OS]                                     ; увеличение индекса
U 1887, EC13,15 ;17450          ADD WR[2] TO LS[T9]                            ; прибавление 4 к адресу памяти
```

```
U 1888, EC89,15 ;17451          ADD WR[2] TO LS[ADDRESS.DATA]          ; прибавление 4 к адресу для печати
                ;17452            CMP WR[3] WITH LS[T9],               ; проверка окончания (WR3 содержит 10 (H))
U 1889, 4F13,B5 ;17453          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 188A, 8988,11 ;17454          JMP.IF[NEQ] TO [READ.T31.2]            ; повторение, пока не будет выполнено
U 188B, FFB2,15 ;17455          INC LS[ERROR.NUMBER]                   ; ошибка 3
U 188C, 367E,15 ;17456          MOV LS[BIT31] TO WR[0]                 ; установка в рабочем регистре бита 31
U 188D, C77A,15 ;17457          BIS LS[BIT29] TO WR[0]                 ; и бита 29
U 188E, 4774,15 ;17458          BIS LS[BIT26] TO WR[0]                 ; и бита 26
U 188F, 4772,15 ;17459          BIS LS[BIT25] TO WR[0]                 ; и бита 25
U 1890, C740,15 ;17460          BIS LS[BIT0] TO WR[0]                  ; и бита 0
U 1891, 3E10,15 ;17461          MOV WR[0] TO LS[TB]                    ; запоминание в промежуточной ячейке LS
U 1892, 989C,F5 ;17462          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]    ; запрос для записи по адресу TB 0
U 1893, 3210,15 ;17463          WRITE.MEM LS[TB]                       ; установка в буфере трансляции битов VALID, PROT C,
                ;17464                                                 ; MODIFY и BYTE OFFSET и отображение виртуального адреса
                ;17465                                                 ; 0 в физический адрес 200
U 1894, 0A17,DC ;17466          JSR [WRITE.CSR1.MME]                   ; запись в CSR1
U 1895, B653,95 ;17467          MOV LS[#200] TO WR[3]                  ; рабочий регистр содержит 200
U 1896, BE13,95 ;17468          MOV WR[3] TO LS[T9]                    ; запоминание в промежуточной ячейке LS
U 1897, 4749,95 ;17469          BIS LS[#10] TO WR[3]                   ; WR3 = 210 (H) (адрес после окончания записи)
U 1898, 8995,3C ;17470          JSR [BACK.T31.LOOP]                    ; запись всех 1 в 4 длинных слова, начиная с физического
                ;17471                                                 ; адреса 200
                ;17472  LOOP.T31.3:
U 1899, 1F9C,F5 ;17473          MEM.REQ[OCTA.WR.V.WCHK] ADRS[#0] DT[LONG]; запрос для записи по адресам памяти 200-20F(H)
U 189A, 3240,15 ;17474          WRITE.MEM LS[#1]                       ; запись 1 в длинное слово по адресу 200
U 189B, B242,15 ;17475          WRITE.MEM LS[#2]                       ; запись 2 в длинное слово по адресу 204
U 189C, B244,15 ;17476          WRITE.MEM LS[#4]                       ; запись 4 в длинное слово по адресу 208
U 189D, 3246,15 ;17477          WRITE.MEM LS[#8]                       ; запись 8 в длинное слово по адресу 20C
U 189E, E5F8,15 ;17478          CLR LS[OS]                             ; очистка индекса
U 189F, B652,15 ;17479          MOV LS[#200] TO WR[0]                  ; рабочий регистр содержит 200
U 18A0, BE88,15 ;17480          MOV WR[0] TO LS[ADDRESS.DATA]          ; адрес для печати (начиная с 200)
U 18A1, BE12,15 ;17481          MOV WR[0] TO LS[T9]                    ; начало чтения с адреса 200
                ;17482  READ.T31.3:
U 18A2, 36F2,95 ;17483          MOV LS[SHIFT.OS(3-0)] TO WR[1]         ; выборка ожидаемых данных
U 18A3, 1813,F5 ;17484          MEM.REQ[READ.P] ADRS[T9] DT[LONG]      ; запрос для чтения из памяти
U 18A4, 3022,15 ;17485          MOV MEM.DATA TO WR[0]                  ; выборка результата
U 18A5, 0869,3C ;17486          JSR [CHECK.RESULT]                     ; проверка, что данные правильные
U 18A6, 8989,94 ;17487          JMP [LOOP.T31.3]                       ; цикл при ошибке, если есть разрешение
U 18A7, 7FF8,15 ;17488          INC LS[OS]                             ; увеличение индекса
U 18A8, EC13,15 ;17489          ADD WR[2] TO LS[T9]                    ; прибавление 4 к адресу памяти
U 18A9, EC89,15 ;17490          ADD WR[2] TO LS[ADDRESS.DATA]          ; прибавление 4 к адресу для печати
                ;17491            CMP WR[3] WITH LS[T9],               ; проверка окончания (WR3 содержит 210 (H))
U 18AA, 4F13,B5 ;17492          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 18AB, 098A,21 ;17493          JMP.IF[NEQ] TO [READ.T31.3]            ; повторение, пока не будет выполнено
U 18AC, FFB2,15 ;17494          INC LS[ERROR.NUMBER]                   ; ошибка 4
U 18AD, B676,15 ;17495          MOV LS[MME] TO WR[0]                   ; занесение в рабочий регистр бита MME
U 18AE, C77A,15 ;17496          BIS LS[TB.PAR.DIAG] TO WR[0]           ; установка бита TB PAR DIAG
U 18AF, 8A17,FC ;17497          JSR [WRITE.CSR1]                       ; запись в CSR1
U 18B0, B652,15 ;17498          MOV LS[#200] TO WR[0]                  ; рабочий регистр содержит адрес 200
U 18B1, BE12,15 ;17499          MOV WR[0] TO LS[T9]                    ; запоминание адреса в LS
U 18B2, 8995,3C ;17500          JSR [BACK.T31.LOOP]                    ; запись всех единиц в 4 длинных слова, начиная с
                ;17501                                                 ; физического адреса 200
                ;17502  LOOP.T31.4:
U 18B3, 1F9C,F5 ;17503          MEM.REQ[OCTA.WR.V.WCHK] ADRS[#0] DT[LONG]; запрос для записи в адреса памяти 200-20F(H)
U 18B4, 3240,15 ;17504          WRITE.MEM LS[#1]                       ; запись 1 в длинное слово по адресу 200
U 18B5, B242,15 ;17505          WRITE.MEM LS[#2]                       ; запись 2 в длинное слово по адресу 204
```

```
U 18B6, B244,15 ;17506          WRITE.MEM LS[#4]                        ; запись 4 в длинное слово по адресу 208
U 18B7, 3246,15 ;17507          WRITE.MEM LS[#8]                        ; запись 8 в длинное слово по адресу 20C
U 18B8, B652,15 ;17508          MOV LS[#200] TO WR[0]                   ; рабочий регистр содержит 200
U 18B9, BE88,15 ;17509          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати (начиная с 200)
U 18BA, BE12,15 ;17510          MOV WR[0] TO LS[T9]                     ; начало чтения с адреса 200
                ;17511  READ.T31.4:
U 18BB, 369E,95 ;17512          MOV LS[ONES] TO WR[1]                   ; выборка ожидаемых данных
U 18BC, 1813,F5 ;17513          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 18BD, 3022,15 ;17514          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 18BE, 0869,3C ;17515          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
U 18BF, 0988,34 ;17516          JMP [LOOP.T31.4]                        ; цикл при ошибке, если есть разрешение
U 18C0, EC13,15 ;17517          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 18C1, EC89,15 ;17518          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати
                ;17519            CMP WR[3] WITH LS[T9],                ; проверка окончания (WR3 содержит 210 (H))
U 18C2, 4F13,B5 ;17520          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 18C3, B98B,B1 ;17521          JMP.IF[NEQ] TO [READ.T31.4]             ; повторение, пока не будет выполнено
U 18C4, 0A17,DC ;17522          JSR [WRITE.CSR1.MME]                    ; запись в CSR1 бита MME
U 18C5, B610,15 ;17523          MOV LS[TB] TO WR[0]                     ; выборка данных для буфера трансляции
U 18C6, C57E,15 ;17524          BIC LS[BIT31] TO WR[0]                  ; очистка бита VALID
U 18C7, 3E10,15 ;17525          MOV WR[0] TO LS[TB]                     ; занесение обратно в LS для записи в TB
U 18C8, 1852,F5 ;17526          MEM.REQ[WRITE.TB] ADRS[BIT9] DT[LONG]   ; запрос для записи в TB по адресу 1
U 18C9, 3210,15 ;17527          WRITE.MEM LS[TB]                        ; установка по адресу 1 в TB битов PROT C, MODIFY и BYTE
                ;17528                                                  ; OFFSET и отображение виртуального адреса 0 в
                ;17529                                                  ; физический адрес 200
U 18CA, 477E,15 ;17530          BIS LS[BIT31] TO WR[0]                  ; установка бита VALID
U 18CB, 4540,15 ;17531          BIC LS[BIT0] TO WR[0]                   ; очистка бита 0 адреса (отображение виртуального
                ;17532                                                  ; адреса 0 в физический адрес 0)
U 18CC, 3E10,15 ;17533          MOV WR[0] TO LS[TB]                     ; запоминание данных в LS для записи в TB
U 18CD, 989C,F5 ;17534          MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]     ; запрос для записи по адресу 0 TB
U 18CE, 3210,15 ;17535          WRITE.MEM LS[TB]                        ; установка по адресу 0 в TB битов VALID, PROT C, MODIFY
                ;17536                                                  ; и BYTE OFFSET и отображения виртуального адреса 0 в
                ;17537                                                  ; физический адрес 0
U 18CF, B652,15 ;17538          MOV LS[#200] TO WR[0]                   ; рабочий регистр содержит 200
U 18D0, 4144,15 ;17539          SUB LS[#4] FROM WR[0]                   ; в рабочем регистре содержится адрес 1FC (H)
U 18D1, BE12,15 ;17540          MOV WR[0] TO LS[T9]                     ; запоминание адреса в LS 9
U 18D2, BE14,15 ;17541          MOV WR[0] TO LS[T10]                    ; а также запоминание в LS 10
U 18D3, 4145,95 ;17542          SUB LS[#4] FROM WR[3]                   ; подготовка конечного адреса 20C
U 18D4, 8995,3C ;17543          JSR [BACK.T31.LOOP]                     ; запись всех 1 в 4 длинных слова, начиная с физического
                ;17544                                                  ; адреса 1FC (H)
                ;17545  LOOP.T31.5.6:
U 18D5, 3644,15 ;17546          MOV LS[#4] TO WR[0]                     ; занесение в рабочий регистр значения 4
U 18D6, 2040,15 ;17547          INC WR[0]                               ; рабочий регистр содержит 5
U 18D7, BE82,15 ;17548          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 5
U 18D8, 1F14,F5 ;17549          MEM.REQ[OCTA.WR.V.WCHK] ADRS[T10] DT[LONG]; запрос для записи по адресам памяти 1FC-208
U 18D9, 3240,15 ;17550          WRITE.MEM LS[#1]                        ; запись 1 в длинное слово с адреса 1FC
U 18DA, B242,15 ;17551          WRITE.MEM LS[#2]                        ; запись 2 в длинное слово с адреса 200
U 18DB, B244,15 ;17552          WRITE.MEM LS[#4]                        ; запись 4 в длинное слово с адреса 204
U 18DC, 3246,15 ;17553          WRITE.MEM LS[#8]                        ; запись 8 в длинное слово с адреса 208
U 18DD, 3614,15 ;17554          MOV LS[T10] TO WR[0]                    ; рабочий регистр содержит 1FC (H)
U 18DE, BE88,15 ;17555          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати (начиная с 1FC (H)
U 18DF, BE12,15 ;17556          MOV WR[0] TO LS[T9]                     ; начало чтения с адреса 1FC (H)
U 18E0, 3640,95 ;17557          MOV LS[#1] TO WR[1]                     ; выборка ожидаемых данных
U 18E1, 1813,F5 ;17558          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 18E2, 3022,15 ;17559          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 18E3, 0869,3C ;17560          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
```

```
U 18E4, 098D,54 ;17561          JMP [LOOP.T31.5.6]                      ; цикл при ошибке, если есть разрешение
U 18E5, FF82,15 ;17562          INC LS[ERROR.NUMBER]                    ; ошибка 6
U 18E6, EC13,15 ;17563          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 18E7, EC89,15 ;17564          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати
                ;17565  READ.T31.6:
U 18E8, 369E,95 ;17566          MOV LS[ONES] TO WR[1]                   ; выборка ожидаемых данных (не модифицированы)
U 18E9, 1813,F5 ;17567          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 18EA, 3022,15 ;17568          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 18EB, 0869,3C ;17569          JSR [CHECK.RESULT]                      ; проверка, что данные не модифицированы (ошибка памяти
                ;17570                                                  ; должна была прервать запись)
U 18EC, 098D,54 ;17571          JMP [LOOP.T31.5.6]                      ; цикл при ошибке, если есть разрешение
U 18ED, EC13,15 ;17572          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 18EE, EC89,15 ;17573          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати
                ;17574          CMP WR[3] WITH LS[T9],                  ; проверка окончания (WR3 содержит 20C (H))
U 18EF, 4F13,85 ;17575          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 18F0, 898E,81 ;17576          JMP.IF[NEQ] TO [READ.T31.6]             ; повторение, пока не будет выполнено
U 18F1, 3614,15 ;17577          MOV LS[T10] TO WR[0]                    ; рабочий регистр содержит адрес 1FC(H)
U 18F2, 4144,15 ;17578          SUB LS[#4] FROM WR[0]                   ; рабочий регистр содержит адрес 1F8(H)
U 18F3, BE12,15 ;17579          MOV WR[0] TO LS[T9]                     ; запоминание адреса в LS 9
U 18F4, BE14,15 ;17580          MOV WR[0] TO LS[T10]                    ; а также запоминание в LS 10
U 18F5, 4145,95 ;17581          SUB LS[#4] FROM WR[3]                   ; подготовка конечного адреса 208
U 18F6, 8995,3C ;17582          JSR [BACK.T31.LOOP]                     ; запись всех 1 в 4 длинных слова, начиная с физического
                ;17583                                                  ; адреса 1F8(H)
                ;17584  LOOP.T31.7.8:
U 18F7, B646,15 ;17585          MOV LS[#8] TO WR[0]                     ; занесение в рабочий регистр значения 8
U 18F8, 2100,15 ;17586          DEC WR[0]                               ; рабочий регистр содержит 7
U 18F9, BE82,15 ;17587          MOV WR[0] TO LS[ERROR.NUMBER]           ; ошибка 7
U 18FA, 1F14,F5 ;17588          MEM.REQ[OCTA.WR.V.WCHK] ADRS[T10] DT[LONG] ; запрос для записи по адресам памяти 1F8-207
U 18FB, 3240,15 ;17589          WRITE.MEM LS[#1]                        ; запись 1 в длинное слово по адресу 1F8
U 18FC, B242,15 ;17590          WRITE.MEM LS[#2]                        ; запись 2 в длинное слово по адресу 1FC
U 18FD, B244,15 ;17591          WRITE.MEM LS[#4]                        ; запись 4 в длинное слово по адресу 200
U 18FE, 3246,15 ;17592          WRITE.MEM LS[#8]                        ; запись 8 в длинное слово по адресу 204
U 18FF, 3614,15 ;17593          MOV LS[T10] TO WR[0]                    ; рабочий регистр содержит 1F8(H)
U 1900, BE88,15 ;17594          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати (начиная с 1F8(H))
U 1901, BE12,15 ;17595          MOV WR[0] TO LS[T9]                     ; начало чтения с адреса 1F8(H)
U 1902, 3640,95 ;17596          MOV LS[#1] TO WR[1]                     ; выборка ожидаемых данных
U 1903, 1813,F5 ;17597          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 1904, 3022,15 ;17598          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 1905, 0869,3C ;17599          JSR [CHECK.RESULT]                      ; проверка правильности данных
U 1906, 098F,74 ;17600          JMP [LOOP.T31.7.8]                      ; цикл при ошибке, если есть разрешение
U 1907, EC13,15 ;17601          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 1908, EC89,15 ;17602          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для для печати
U 1909, B642,95 ;17603          MOV LS[#2] TO WR[1]                     ; выборка ожидаемых данных
U 190A, 1813,F5 ;17604          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 190B, 3022,15 ;17605          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 190C, 0869,3C ;17606          JSR [CHECK.RESULT]                      ; проверка правильности данных
U 190D, 098F,74 ;17607          JMP [LOOP.T31.7.8]                      ; цикл при ошибке, если разрешено
U 190E, FF82,15 ;17608          INC LS[ERROR.NUMBER]                    ; ошибка 8
U 190F, EC13,15 ;17609          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 1910, EC89,15 ;17610          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати
                ;17611  READ.T31.8:
U 1911, 369E,95 ;17612          MOV LS[ONES] TO WR[1]                   ; выборка ожидаемых данных (не модифицированы)
U 1912, 1813,F5 ;17613          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; запрос для чтения из памяти
U 1913, 3022,15 ;17614          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 1914, 0869,3C ;17615          JSR [CHECK.RESULT]                      ; проверка, что данные не модифицированы (ошибка памяти
```

```
                ;17616                                          ; должна была прервать запись)
U 1915, 098F,74 ;17617          JMP [LOOP.T31.7.B]              ; цикл при ошибке, если есть разрешение
U 1916, EC13,15 ;17618          ADD WR[2] TO LS[T9]              ; прибавление 4 к адресу памяти
U 1917, EC89,15 ;17619          ADD WR[2] TO LS[ADDRESS.DATA]    ; прибавление 4 к адресу для печати
                ;17620          CMP WR[3] WITH LS[T9],           ; проверка, что выполнено (WR3 содержит 20B (H))
U 1918, 4F13,B5 ;17621          DT(LONG)&SET.ALU.CC              ; установка кодов условий
U 1919, 0991,11 ;17622          JMP.IF[NEQ] TO [READ.T31.B]      ; повторение, пока не будет выполнено
U 191A, FF82,15 ;17623          INC LS[ERROR.NUMBER]             ; ошибка 9
U 191B, B610,15 ;17624          MOV LS[TB] TO WR[0]              ; выборка данных для буфера трансляции
U 191C, C740,15 ;17625          BIS LS[BIT0] TO WR[0]            ; установкка бита 0 адреса (отображение виртуального
                ;17626                                          ; адреса 0 в физический 200)
U 191D, 3E10,15 ;17627          MOV WR[0] TO LS[TB]              ; запоминание в LS данных для записи в TB
U 191E, 1852,F5 ;17628          MEM.REQ[WRITE.TB] ADRS[BIT9] DT[LONG]  ; запрос для записи по адресу 1 в TB
U 191F, 3210,15 ;17629          WRITE.MEM LS[TB]                 ; установка по адресу 1 в TB битов VALID, PROT C, MODIFY
                ;17630                                          ; и BYTE OFFSET и отображение виртуального адреса 0 в
                ;17631                                          ; физический адрес 200
U 1920, B652,15 ;17632          MOV LS[#200] TO WR[0]            ; рабочий регистр содержит адрес 200
U 1921, 4144,15 ;17633          SUB LS[#4] FROM WR[0]            ; в рабочем регистре адрес 1FC(H)
U 1922, BE12,15 ;17634          MOV WR[0] TO LS[T9]              ; запоминание адреса в LS 9
U 1923, BE14,15 ;17635          MOV WR[0] TO LS[T10]             ; а также запоминание в LS 10
U 1924, C045,95 ;17636          ADD LS[#4] TO WR[3]              ; установка конечного адреса 204
U 1925, 8995,3C ;17637          JSR [BACK.T31.LOOP]              ; запись всех единиц в 4 длинных слова, начиная с
                ;17638                                          ; физического адреса 1FC(H)
                ;17639  LOOP.T31.9:
U 1926, 1F14,F5 ;17640          MEM.REQ[OCTA.WR.V.WCHK] ADRS[T10] DT[LONG] ; запрос для записи по адресам памяти 1FC-20B
U 1927, 3240,15 ;17641          WRITE.MEM LS[#1]                 ; запись 1 в длинное слово по адресу 1FC
U 1928, B242,15 ;17642          WRITE.MEM LS[#2]                 ; запись 2 в длинное слово по адресу 200
U 1929, B244,15 ;17643          WRITE.MEM LS[#4]                 ; запись 4 в длинное слово по адресу 204
U 192A, 3246,15 ;17644          WRITE.MEM LS[#8]                 ; запись 8 в длинное слово по адресу 20B
U 192B, 3614,15 ;17645          MOV LS[T10] TO WR[0]             ; рабочий регистр содержит 1FC
U 192C, BE88,15 ;17646          MOV WR[0] TO LS[ADDRESS.DATA]    ; адрес для печати (начиная с 1FC(H))
U 192D, BE12,15 ;17647          MOV WR[0] TO LS[T9]              ; начало чтения с адреса 1FC(H)
U 192E, E5F8,15 ;17648          CLR LS[OS]                       ; очистка индекса
                ;17649  READ.T31.9:
U 192F, 36F2,95 ;17650          MOV LS[SHIFT.OS(3-0)] TO WR[1]   ; выборка ожидаемых данных
U 1930, 1B13,F5 ;17651          MEM.REQ[READ.P] ADRS[T9] DT[LONG]  ; запрос для чтения из памяти
U 1931, 3022,15 ;17652          MOV MEM.DATA TO WR[0]            ; выборка результата
U 1932, 0869,3C ;17653          JSR [CHECK.RESULT]               ; проверка, что данные правильные
U 1933, 8992,64 ;17654          JMP [LOOP.T31.9]                 ; цикл при ошибке, если есть разрешение
U 1934, 7FF8,15 ;17655          INC LS[OS]                       ; увеличение индекса
U 1935, EC13,15 ;17656          ADD WR[2] TO LS[T9]              ; прибавление 4 к адресу памяти
U 1936, EC89,15 ;17657          ADD WR[2] TO LS[ADDRESS.DATA]    ; прибавление 4 к адресу для печати
                ;17658          CMP WR[3] WITH LS[T9],           ; проверка, что выполнено (WR3 содержит 204 (H))
U 1937, 4F13,B5 ;17659          DT(LONG)&SET.ALU.CC              ; установка кодов условий
U 1938, 8992,F1 ;17660          JMP.IF[NEQ] TO [READ.T31.9]      ; повторение, пока не будет выполнено
U 1939, FF82,15 ;17661          INC LS[ERROR.NUMBER]             ; ошибка A
U 193A, 3614,15 ;17662          MOV LS[T10] TO WR[0]             ; рабочий регистр содержит адрес 1FC(H)
U 193B, BE12,15 ;17663          MOV WR[0] TO LS[T9]              ; запоминание адреса в LS
U 193C, 8995,3C ;17664          JSR [BACK.T31.LOOP]              ; запись всех 1 в 4 длинных слова, начиная с физического
                ;17665                                          ; адреса 1FC(H)
                ;17666  LOOP.T31.A:
U 193D, 1F14,F5 ;17667          MEM.REQ[OCTA.WR.V.WCHK] ADRS[T10] DT[LONG] ; запрос для записи в адреса памяти 1FC-20B
U 193E, 3240,15 ;17668          WRITE.MEM LS[#1]                 ; запись 1 в длинное слово по адресу 1FC
U 193F, 0A1B,4C ;17669          JSR [NOP.DELAY]                  ; выполнение задержки из 5 циклов для задержки CPU DR
U 1940, B242,15 ;17670          WRITE.MEM LS[#2]                 ; запись 2 в длинное слово по адресу 200
```

```
U 1941, B244,15 ;17671          WRITE.MEM LS[#4]                        ; запись 4 в длинное слово по адресу 204
U 1942, 3246,15 ;17672          WRITE.MEM LS[#8]                        ; запись 8 в длинное слово по адресу 208
U 1943, 3614,15 ;17673          MOV LS[T10] TO WR[0]                    ; рабочий регистр содержит 1FC(H)
U 1944, BE88,15 ;17674          MOV WR[0] TO LS[ADDRESS.DATA]           ; адрес для печати (начиная с 1FC(H))
U 1945, BE12,15 ;17675          MOV WR[0] TO LS[T9]                     ; начало чтения с адреса 1FC(H)
U 1946, E5F8,15 ;17676          CLR LS[OS]                              ; очистка индекса
                ;17677  READ.T31.A:
U 1947, 36F2,95 ;17678          MOV LS[SHIFT.OS(3-0)] TO WR[1]          ; выборка ожидаемых данных
U 1948, 1813,F5 ;17679          MEM.REQ[READ.P] ADRS[T9] DT[LONG]       ; подготовка для чтения из памяти
U 1949, 3022,15 ;17680          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 194A, 0869,3C ;17681          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
U 194B, 8993,D4 ;17682          JMP [LOOP.T31.A]                        ; цикл при ошибке, если есть разрешение
U 194C, 7FF8,15 ;17683          INC LS[OS]                              ; увеличение индекса
U 194D, EC13,15 ;17684          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
U 194E, EC89,15 ;17685          ADD WR[2] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати
                ;17686          CMP WR[3] WITH LS[T9],                  ; проверка, что выполнено (WR3 содержит 204(H))
U 194F, 4F13,B5 ;17687          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 1950, 0994,71 ;17688          JMP.IF[NEQ] TO [READ.T31.A]             ; повторение, пока не будет выполнено
U 1951, 0995,94 ;17689          JMP [END.T31]                           ; конец теста
                ;17690   ; Подпрограмма для записи фона единиц в 4 длинных слова памяти
                ;17691  BACK.T31:
U 1952, 6512,15 ;17692          CLR LS[T9]                              ; начало с адреса 0
                ;17693  BACK.T31.LOOP:
U 1953, 9912,75 ;17694          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи 4 длинных слов
U 1954, 329E,15 ;17695          WRITE.MEM LS[ONES]                      ; запись в память данных, состоящих из всех 1
U 1955, EC13,15 ;17696          ADD WR[2] TO LS[T9]                     ; прибавление 4 к адресу памяти
                ;17697          CMP WR[3] WITH LS[T9],                  ; проверка, что выполнено
U 1956, 4F13,B5 ;17698          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 1957, 0995,31 ;17699          JMP.IF[NEQ] TO [BACK.T31.LOOP]          ; повторение, пока не бубет выполнено
U 1958, 5B00,14 ;17700          RETURN                                  ; конец
                ;17701  END.T31:
```

```
;17702  .PAGE "ТЕСТ 32 - микропрограммы чтения четырехкратных (QUAD) и восьмикратных (OCTA) слов (модуль MCT)"
;17703  ;
;17704  ;
;17705  ; ОПИСАНИЕ ТЕСТА:
;17706  ;
;17707  ;          Этот тест предназначен для обнаружния неиправностей в ПЗУ управляющей
;17708  ;       памяти методом проверки всех путей в микрокодах. Тест проверяет путь,
;17709  ;       выбираемый, если задается функция QUAD.READ.V.RCHK или OCTA READ. Этот
;17710  ;       тест проверит выравненное и невыравненное чтение, а также чтение из
;17711  ;       ячеек с одиночной или двойной ошибкой.
;17712  ;          Первая часть теста только записывает тестовые данные из всех единиц, всех
;17713  ;       нулей и всех единиц в 3 ячейки памяти (от 200 до 20B). Выполняется QUAD.
;17714  ;       READ.V.RCHK и результат проверяется. Этот тест повторяется с задержкой
;17715  ;       из 5 нерабочих циклов между MEM.REQ и каждой инструкцией MOV MEM.DATA для
;17716  ;       задержания CPU DR. Этим проверяется путь при ветвлении, когда нет CPU DR.
;17717  ;       Вторая часть теста проверяет путь, когда ошибка системы памяти происходит
;17718  ;       на первом цикле. Ошибка системы памяти вызывается путем установки в CSR1
;17719  ;       бита TB PAR DIAG до выполнения теста. Первый цикл должен завершиться, но
;17720  ;       второй цикл должен быть прерван.
;17721  ;          Третья часть этого теста проверяет функцию QUAD.READ.V.RCHK при отсутст-
;17722  ;       вии выравнивания на границе. Считываются адреса с 202 по 209. Это повто-
;17723  ;       ряется при установке в CSR1 бита TB PAR DIAG, вызывающего ошибку. В этом
;17724  ;       случае, до того, как микропрограмма прерывается, будет считан адрес 200.
;17725  ;       Далее проверяется микопрограмма OCTA READ, вначале при использовании вы-
;17726  ;       равненного чтения. Этот тест повторяется с задержкой между инструкциями
;17727  ;       MOV MEM.DATA для задержания CPU DR. Затем проверяется невыравненное чте-
;17728  ;       ние, начиная с ареса 202 (виртуальный адрес 2).
;17729  ;          Остальная часть теста проверяет ветвления ERR и ERRSUM в различных точ-
;17730  ;       ках вдоль пути мкропрограммы. Последний тест проверяет OCTA.READ.P.
;17731  ;
;17732  ; ПРЕДПОЛОЖЕНИЯ:
;17733  ;
;17734  ;       Считается, что все предыдущие тесты прошли успешно и что вся аппаратура
;17735  ;       (кроме ПЗУ управляющей памяти MCT) работает правильно.
;17736  ;
;17737  ; ШАГИ ТЕСТА:
;17738  ;
;17739  ;       1) Установка в LS маски ошибки, номера ошибки и номера модуля (для рас-
;17740  ;          печатки ошибки) и очистка в LS номера предыдущей ошибки.
;17741  ;       2) Подготовка буфера трансляции (TB) для отображения виртуального адреса
;17742  ;          0 в физический адрес 200. Установка в CSR1 бита MME. Запись фона из
;17743  ;          всех единиц, всех нулей, всех единиц, всех нулей, начиная с адреса 200.
;17744  ;       3) Выполнение QUAD.READ.V.RCHK и проверка правильности считанных данных.
;17745  ;       4) Повторение шагов 2 и 3 с задержкой между инструкциями MEM.REQ и MOV
;17746  ;          MEM.DATA TO LS для вызова ветвления в микропрограмме на путь, когда
;17747  ;          нет CPU DR.
;17748  ;       5) Установка в CSR1 обоих битов: MME и TB PAR DIAG. Выполнение QUAD.READ.
;17749  ;          V.RCHK и проверка, что первый цикл чтения завершается, и что второй
;17750  ;          цикл чтения прерывается из-за ошибки системы памяти (ERR SUM).
;17751  ;       6) Установка в CSR1 только бита MME. Выполнение QUAD.READ.V.RCHK по вир-
;17752  ;          туальному адресу 2 (невыравненное чтение) и проверка, что оба длинных
;17753  ;          слова считываются правильно.
;17754  ;       7) Повторение шага 6 с задержанным CPU DR.
;17755  ;       8) Установка в CSR1 обоих битов MME и TB PAR DIAG и повторение шага 6.
;17756  ;          Проверка, что адрес 200 (первый считываемый адрес при невыравненном
```

```
;17757  ;           чтении) считывается правильно, и что следующие циклы чтения прекраще-
;17758  ;           ны из-за ошибки системы памяти (ERR SUM).
;17759  ;        9) Установка в CSR1 бита MME. Выполнение OCTA.RD.V.RCHK, начиная с адреса
;17760  ;           200. Проверка всех четырех длинных слов на правильность данных.
;17761  ;       10) Повторение шага 9 с задержкой между инструкциями MOV MEM.DATA для вы-
;17762  ;           бора в микрокодах пути, когда нет CPU DR.
;17763  ;       11) Повторение шага 9, начиная с адреса 202 (невыравненное чтение). Этим
;17764  ;           проверяется путь в микрокодах, когда есть 2 MEM CYC И нет IC0.
;17765  ;       12) Повторение шага 11 с задержанным CPU DR.
;17766  ;       13) Запись ошибки в одиночном бите в пяти длинных словах, начиная с адреса
;17767  ;           200..
;17768  ;       14) Выполнение OCTA.READ.V.RCHK при невыравненном адресе и проверка, что
;17769  ;           ошибка в каждом длинном слове исправлена, и что чтение правильное.
;17770  ;       15) Установка двойной ошибки по адресу 200. Выполнение QUAD.RD.V.RCHK по
;17771  ;           адресу 202 и проверка, что чтение прекращено.
;17772  ;       16) Запись правильных данных по адресу 200 и установка двойной ошибки по
;17773  ;           адресу 204. Выполнение QUAD.RD.V.RCHK и проверка, что чтение прекра-
;17774  ;           щено на втором длинном слове.
;17775  ;       17) Запись правильных данных по адресу 204, установка двойной ошибки по
;17776  ;           адресу 208. Выполнение OCTA.RD.V.RCHK и проверка, что чтение прервано
;17777  ;           на третьем длинном слове.
;17778  ;       18) Подготовка TB по адресу 0 для отображения виртуального адреса 0 в физи-
;17779  ;           ческий 0. Подготовка TB по адресу 1 для отображения виртуального адреса
;17780  ;           0 в физический 200 и очистка бита VALID. Это вызовет ошибку системы па-
;17781  ;           мяти (ERR SUM) при использовании TB 1.
;17782  ;       19) Запись правильных данных по адресу 1FC. Выполнение QUAD.RD.V.RCHK по
;17783  ;           адресу 1FE и проверка, что чтение прекращается на втором длинном сло-
;17784  ;           ве.
;17785  ;       20) Запись правильных данных по адресу 1F8. выполнение OCTA.RD.V.RCHK по
;17786  ;           адресу 1FA и проверка, что чтение прекращено на третьем длинном слове.
;17787  ;       21) Запись по адресу 200 четырех длинных слов с правильными данными. Вы-
;17788  ;           полнение OCTA.READ.P и проверка, что получены правильные данные. Этим
;17789  ;           проверяется точка входа для чтения по физическому адресу.
;17790  ;
;17791  ; ОШИБКИ:
;17792  ;
;17793  ;       ошибка 1 - функция QUAD.READ.V.RCHK не выполнила правильного чтения 2
;17794  ;                  длинных слов (CPU DR не задержан).
;17795  ;       ошибка 2 - функция QUAD.READ.V.RCHK не выполнила правильного чтения 2
;17796  ;                  длинных слов (CPU DR задержан).
;17797  ;       ошибка 3 - функция QUAD.READ.V.RCHK не выполнила правильно первого цик-
;17798  ;                  ла чтения (200) или не прекратила второго цикла чтения (204).
;17799  ;                  Адрес под OTHER указывает, который цикл выполняется непра-
;17800  ;                  вильно.
;17801  ;       ошибка 4 - функция QUAD.READ.V.RCHK не выполнила невыравненного чтения.
;17802  ;       ошибка 5 - функция QUAD.READ.V.RCHK не выполнила невыравненного чтения
;17803  ;                  при задержанном CPU DR.
;17804  ;       ошибка 6 - функция QUAD.READ.V.RCHK не прекратила второго цикла чтения
;17805  ;                  (204) при невыравненном адресе с ошибкой паритета буфера
;17806  ;                  трансляции (TB PAR ERR).
;17807  ;       ошибка 7 - не выполняется OCTA.RD.V.RCHK.
;17808  ;       ошибка 8 - не выполняется OCTA.RD.V.RCHK при задержанном CPU DR.
;17809  ;       ошибка 9 - не выполняется OCTA.RD.V.RCHK для невыравненного чтения.
;17810  ;       ошибка A - не выполняется OCTA.RD.V.RCHK для невыравненного чтения при
;17811  ;                  задержанном CPU DR.
```

```
;17812  ;       ошибка B - не выполняется OCTA.RD.V.RCHK для невыравненного чтения с
;17813  ;                  одиночной ошибкой в каждом длинном слове.
;17814  ;       ошибка C - функция QUAD.RD.V.RCHK не прекратила невыравненного чтения
;17815  ;                  при неисправимой ошибке в первом длинном слове.
;17816  ;       ошибка D - функция QUAD.RD.V.RCHK не выполнила правильного чтения пер-
;17817  ;                  вого длинного слова (202) или не прекратила второго цикла
;17818  ;                  чтения (206) при невыравненном чтении с неисправимой ошибкой
;17819  ;                  во втором длинном слове.
;17820  ;       ошибка E - функция OCTA.RD.V.RCHK не выполнила правильного чтения пер-
;17821  ;                  вого или второго длинного слова (202 или 206) или не прек-
;17822  ;                  ратила третьего цикла чтения (20A) при невыравненном чтении
;17823  ;                  с неисправимой ошибкой в третьем длинном слове.
;17824  ;       ошибка F - функция QUAD.RD.V.RCHK не выполнила правильного чтения пер-
;17825  ;                  вого длинного слова (1FE) или не прекратила второго цикла
;17826  ;                  чтения (202) при невыравненном чтении с ошибкой системы памя-
;17827  ;                  ти (TB не истинный) на втором длинном слове.
;17828  ;       ошибка 10- функция OCTA.RD.V.RCHK не выполнила правильного чтения перво-
;17829  ;                  го или второго длиного слова (1FA или 1FE) или не прекратила
;17830  ;                  третьего цикла чтения (202) при невыравненном чтении с ошиб-
;17831  ;                  кой системы памяти (TB не истинный) на третьем длинном слове.
;17832  ;       ошибка 11- функция OCTA.READ.P не выполнила правильного чтения четырех
;17833  ;                  длинных слов.
;17834  ;
;17835  ;
;17836  ; НАЛАДКА:
;17837  ;
;17838  ;       ПРИМЕЧАНИЕ: следующие ошибки с найбольшей вероятностью указывают на от-
;17839  ;       каз в потоках микрокодов. Аппаратура, используемая микрокодами, ранее
;17840  ;       проверялась. Чтобы локализовать неисправность, необходимо проверить последо-
;17841  ;       вательности, указанные ниже. Если последовательности правильные, подо-
;17842  ;       зревается дефектный бит в одном из ПЗУ управляющей памяти и необходимо
;17843  ;       проверить каждое состояние на правильные сигналы разрешений.
;17844  ;
;17845  ;          ОШИБКА 1 - Это первый случай, когда проверяется чтение четырехкратных
;17846  ;       слов, поэтому любая инструкция в микропрограмме может вызывать отказ.
;17847  ;       Путь, который необходимо наблюдать, является следующим: из QUAD READ V
;17848  ;       RCHK через OCTA ARY CYC C4, через OCTA C5, через OCTA C6, через OCTA C7,
;17849  ;       через OCTA C8, через OCTA C9, через OCTA C10A, через OCTA C11A к CPU RD
;17850  ;       V CONT C7. Отсюда микропрограмма уже проверялась ранее.
;17851  ;
;17852  ;          ОШИБКА 2 - Эта ошибка использует другую ветвь из OCTA C8. Микропрограмма
;17853  ;       та же, как и для ошибки 1 до OCTA C8. Отсюда должен выбираться путь не L
;17854  ;       CPU DR и не P2 к циклу, который зацикливается на себя в ожидании CPU DR.
;17855  ;       Затем через OCTA WAIT CPU DR C1A и через OCTA WAIT REF A к циклу, в ко-
;17856  ;       тором происходит ветвление при IC0 и MEM CYC к OCTA C10A, через OCTA
;17857  ;       C11A к CPU RD V CONT C7.
;17858  ;
;17859  ;          ОШИБКА 3 - Эта ошибка указывает на отказ в ветвлении при ERRSUM из OCTA
;17860  ;       C6. Путь,который выбирается из OCTA C6, должен проходить по ветви ERRSUM
;17861  ;       к циклу, который переходит к WAIT CPU DR C1. Первое чтение (по адресу
;17862  ;       200) должно быть выполнено правильно. Второе чтение (по адресу 204) дол-
;17863  ;       жно быть прекращено.
;17864  ;
;17865  ;          ОШИБКА 4 - Эта ошибка  указывает на отказ в пути, выбираемом из OCTA ARY
;17866  ;       CYC C4. Должен выбираться путь при 2 MEM CYC. Отсюда микропрограмма
```

```
;17867  ;       проходит через OCTA TWO ARY CYC C5, через OCTA TWO ARY CYC C6, через
;17868  ;       OCTA TWO ARY CYC C7, через OCTA TWO ARY CYC C8 к циклу, который перехо-
;17869  ;       дит обратно к OCTA ARY CYC C4. Снова должна выбираться ветвь при 2 MEM
;17870  ;       CYC к OCTA TWO CYC C5. Выбирается ветвь 2ND CYC к циклу, который прове-
;17871  ;       ряет ошибку системы памяти ERRSUM, и затем к OCTA C7. Отсюда путь такой
;17872  ;       же, как и при ошибке 1.
;17873  ;
;17874  ;          ОШИБКА 5 - То же, что и для ошибки 4, за исключением того, что из OCTA
;17875  ;       C8 выбирается путь не CPU DR и не P2.
;17876  ;
;17877  ;          ОШИБКА 6 - Для этой ошибки должен был выбираться путь при ERRSUM из OCTA
;17878  ;       TWO ARY CYC C6. Первый цикл должен был считать данные по адресу 200 и
;17879  ;       поместить их на шину до прекращения (OTHER (другие данные)=200). Второй
;17880  ;       цикл должен быть прекращен, поэтому LS 10 должна получить те же данные.
;17881  ;       Путь из OCTA TWO ARY CYC C6 должен был проходить к циклу, который пере-
;17882  ;       ходит к WAIT CPU DR C1.
;17883  ;
;17884  ;          ОШИБКА 7 - Эта ошибка является первой, которая использует микропрограмму
;17885  ;       чтения восьмикратных слов, которая продолжается после того, как завершена
;17886  ;       микропрограмма чтения четырехкратных слов. Микропрограмма начинается с
;17887  ;       OCTA READ V RCHK и идет к OCTA ARY CYC C4. Отсюда она продолжается как
;17888  ;       при ошибке 1 до OCTA C9. В этой точке выбирается путь при не IC0 и
;17889  ;       1MEM CYC. микропрограмма через один цикл переходит к OCTA C11, затем
;17890  ;       через OCTA C12, через OCTA C13, через OCTA C14, через OCTA C15 к одному
;17891  ;       циклу, который через OCTA C17 возвращает назад к OCTA C6. Отсюда микро-
;17892  ;       программа продолжается также, как из OCTA C6 для ошибки 1.
;17893  ;
;17894  ;          ОШИБКА 8 - Эта ошибка аналогична ошибке 6 выше, за исключением того,что
;17895  ;       в двух точках выбирается путь при не CPU DR и не P2: в OCTA C8 и OCTA C14.
;17896  ;       Эти пути и цикл, в котором имеется ветвление по IC0 и 1MEM CYC, являют-
;17897  ;       ся наиболее подозрительными.
;17898  ;
;17899  ;          ОШИБКА 9 - Для этой ошибки первоначально выбирается тот же путь, как и
;17900  ;       при ошибке 4 выше (после OCTA READ V RCHK к OCTA ARY CYC C4), за исклю-
;17901  ;       чением того, что из OCTA C9 выбирается ветвь не IC0 и 2MEM CYC и из
;17902  ;       OCTA C15 выбирается путь 2 MEM CYC.
;17903  ;
;17904  ;          ОШИБКА A - То же, что и для ошибки 9, за исключением того, что из OCTA C8
;17905  ;       выбирается путь не CPU DR и не P2.
;17906  ;
;17907  ;          ОШИБКА B - Эта ошибка, по-существу, такая же, как и ошибка 9, за исклю-
;17908  ;       чением того, что из циклов OCTA TWO ARY CYC C7, OCTA C7 и OCTA C13
;17909  ;       выбирается путь при ERR. Если микропрограмма проходит по пути ошибки, то во
;17910  ;       всех трех случаях должна выбираться ветвь SERR и ошибка должна корректи-
;17911  ;       роваться.
;17912  ;
;17913  ;          ОШИБКА C - Эта ошибка проверяет ветвление из OCTA 2 ARY CYC WSDE C8.
;17914  ;       микропрограмма до этой точки такая же, как и для ошибки B. Из OCTA 2
;17915  ;       ARY CYC WSDE C8 должно выбираться ветвление при UNC ERR к OCTA 2 ARY CYC
;17916  ;       WUDE C9.
;17917  ;
;17918  ;          ОШИБКА D - Для этой ошибки микропрограмма должна следовать по тому же пу-
;17919  ;       ти, как и для ошибки 4 до OCTA C7. В этой точке должен выбираться путь
;17920  ;       ERR к OCTA ERR C8. Отсюда должно иметь место ветвление при UNC ERR к
;17921  ;       OCTA ERR C9.
```

```
                ;17922  ;
                ;17923  ;          ОШИБКА E - Для этой ошибки микропрограмма должна следовать по тому же
                ;17924  ;        пути, как и для ошибки 9 до OCTA C13. В этой точке должен выбираться путь ERR
                ;17925  ;        к OCTA ERR C14. Отсюда должен выбираться путь UNC ERR к циклу, который
                ;17926  ;        идет к WAIT CPU DR C1.
                ;17927  ;
                ;17928  ;          ОШИБКА F - Эта ошибка проверяет ветвление по ERRSUM из цикла после OCTA
                ;17929  ;        TWO ARY CYC C5 (путь 2ND CYC). До этой точки микропрограмма такая же,
                ;17930  ;        как и для ошибки 4. Затем должен быть выбран путь ERRSUM, заканчивающийся
                ;17931  ;        в WAIT CPU DR C1.
                ;17932  ;
                ;17933  ;          ОШИБКА 10 - Эта ошибка проверяет ветвь ERRSUM из OCTA C12. До этой точ-
                ;17934  ;        ки микропрограмма такая же, как и для ошибки 9. Затем должен выбираться
                ;17935  ;        путь ERRSUM, заканчивающийся в WAIT CPU DR C1.
                ;17936  ;
                ;17937  ;          ОШИБКА 11 - То же, что и для ошибки 7, за исключением того, что точкой
                ;17938  ;        входа является CPU READ PH OCTA.
                ;17939  ;
                ;17940   T.32:
U 1959, B65E,15 ;17941            MOV LS[BEGIN.TEST] TO WR[0]              ; установка в WR0 бита 15 для слова управления и
                ;17942                                                     ; состояния
U 195A, 3E80,15 ;17943            MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 15 в слове управления и состояния. Бит
                ;17944                                                     ; 15 указывает для консольного процессора начало теста
U 195B, 10E0,15 ;17945            MISC [SET.CP.ATTN]                       ; выдача для консольного процессора CPU ATTN.
                ;17946   WAIT.T32.0:
U 195C, 0995,C4 ;17947            JMP [WAIT.T32.0]                         ; цикл для ожидания ответа консольного процессора
U 195D, 0A1A,AC ;17948            JSR [SETUP.1]                            ; установка маски, кода модуля и т.д.
U 195E, 367E,15 ;17949            MOV LS[BIT31] TO WR[0]                   ; установка в рабочем регистре бита 31
U 195F, C77A,15 ;17950            BIS LS[BIT29] TO WR[0]                   ; и бита 29
U 1960, 4772,15 ;17951            BIS LS[BIT25] TO WR[0]                   ; и бита 25
U 1961, C740,15 ;17952            BIS LS[BIT0] TO WR[0]                    ; и бита 0
U 1962, 3E10,15 ;17953            MOV WR[0] TO LS[TB]                      ; запоминание в промежуточной ячейке LS
U 1963, 9B9C,F5 ;17954            MEM.REQ[WRITE.TB] ADRS[#0] DT[LONG]      ; запрос для записи по адресу TB 0
U 1964, 3210,15 ;17955            WRITE.MEM LS[TB]                         ; установка в буфере трансляции битов VALID, PROT C и
                ;17956                                                     ; BYTE OFFSET и отображение виртуального адреса 0 в
                ;17957                                                     ; физический адрес 200
U 1965, 0A17,DC ;17958            JSR [WRITE.CSR1.MME]                     ; запись в CSR1 бита MME
U 1966, B652,15 ;17959            MOV LS[#200] TO WR[0]                    ; выборка в рабочий регистр значения 200
U 1967, C044,15 ;17960            ADD LS[#4] TO WR[0]                      ; рабочий регистр содержит 204
U 1968, BE12,15 ;17961            MOV WR[0] TO LS[T9]                      ; запоминание в LS
U 1969, 1C52,75 ;17962            MEM.REQ[OCTA.WRITE.PJ ADRS[#200] DT[LONG] ; запрос для записи в память 4 длинных слов
U 196A, 329E,15 ;17963            WRITE.MEM LS[ONES]                       ; запись эталона из всех единиц по адресу 200
U 196B, B29C,15 ;17964            WRITE.MEM LS[ZERO]                       ; запись эталона из всех нулей по адресу 204
U 196C, 329E,15 ;17965            WRITE.MEM LS[ONES]                       ; запись эталона из всех единиц по адресу 208
U 196D, B29C,15 ;17966            WRITE.MEM LS[ZERO]                       ; запись эталона из всех нулей по адресу 20C
U 196E, B652,15 ;17967            MOV LS[#200] TO WR[0]                    ; рабочий регистр содержит 200
U 196F, C04B,15 ;17968            ADD LS[#10] TO WR[0]                     ; рабочий регистр содержит 210
U 1970, BE14,15 ;17969            MOV WR[0] TO LS[T10]                     ; запоминания в LS адреса для записи в памяти
U 1971, 9914,75 ;17970            MEM.REQ[WRITE.PJ ADRS[T10] DT[LONG]      ; запрос для записи по адресу 210
U 1972, 329E,15 ;17971            WRITE.MEM LS[ONES]                       ; запись эталона из всех единиц по адресу 210
U 1973, B670,15 ;17972            MOV LS[OTHER.DATA] TO WR[0]              ; установка бита, что в сообщениях об ошибке под OTHER
                ;17973                                                     ; (другие данные) будут печататься данные
U 1974, 3E80,15 ;17974            MOV WR[0] TO LS[CONTROL.STATUS]          ; занесение в слово управления состояния
U 1975, 3653,15 ;17975            MOV LS[#200] TO WR[2]                    ; адрес для печати в поле OTHER (другие данные)
U 1976, 3613,95 ;17976            MOV LS[T9] TO WR[3]                      ; следующий адрес для поля OTHER (другие данные)
```

```
U 1977, E51C,15 ;17977          CLR LS[T14]                             ; очистка ячейки LS
U 1978, FD1C,15 ;17978          COM LS[T14]                             ; ожидаемые данные для первого длинного слова
U 1979, 651E,15 ;17979          CLR LS[T15]                             ; ожидаемые данные для второго длинного слова
                ;17980  LOOP.T32.1:
U 197A, 1F9D,75 ;17981          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#0] DT[LONG] ; запрос для чтения из адресов памяти 200-207
U 197B, 3022,15 ;17982          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 197C, B814,15 ;17983          MOV MEM.DATA TO LS[T10]                 ; выборка результата второго цикла чтения
U 197D, 09A9,4C ;17984          JSR [CHECK.T32.QUAD]                    ; проверка результатов
U 197E, 8997,A4 ;17985          JMP [LOOP.T32.1]                        ; цикл при ошибке, если есть разрешение
U 197F, FF82,15 ;17986          INC LS[ERROR.NUMBER]                    ; ошибка 2
                ;17987  LOOP.T32.2:
U 1980, 1F9D,75 ;17988          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#0] DT[LONG] ; запрос для чтения из адресов памяти 200-207
U 1981, 3022,15 ;17989          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 1982, 0A1B,4C ;17990          JSR [NOP.DELAY]                         ; выполнение задержки из 5 циклов для задержания CPU DR
U 1983, B814,15 ;17991          MOV MEM.DATA TO LS[T10]                 ; выборка результата второго цикла чтения
U 1984, 09A9,4C ;17992          JSR [CHECK.T32.QUAD]                    ; проверка результатов
U 1985, 8998,04 ;17993          JMP [LOOP.T32.2]                        ; цикл при ошибке, если есть разрешение
U 1986, FF82,15 ;17994          INC LS[ERROR.NUMBER]                    ; ошибка 3
U 1987, B676,15 ;17995          MOV LS[MME] TO WR[0]                    ; установка в рабочем регистре бита MME
U 1988, C77A,15 ;17996          BIS LS[TB.PAR.DIAG] TO WR[0]            ; установка в рабочем регистре бита TB PAR DIAG
U 1989, 8A17,FC ;17997          JSR [WRITE.CSR1]                        ; установка в CSR1 битов MME и TB PAR DIAG (для
                ;17998                                                  ; получения при чтении ERRSUM)
U 198A, 3645,95 ;17999          MOV LS[#4] TO WR[3]                     ; увеличение на 4 адреса для второго длинного слова
                ;18000  LOOP.T32.3:
U 198B, 1F9D,75 ;18001          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#0] DT[LONG] ; запрос для чтения из адресов памяти 200-207
U 198C, 3022,15 ;18002          MOV MEM.DATA TO WR[0]                   ; выборка результата
U 198D, B814,15 ;18003          MOV MEM.DATA TO LS[T10]                 ; второй цикл должен быть прерван
U 198E, 3E89,15 ;18004          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати
U 198F, 369E,95 ;18005          MOV LS[ONES] TO WR[1]                   ; выборка ожидаемых данных
U 1990, 0869,3C ;18006          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
U 1991, 0998,B4 ;18007          JMP [LOOP.T32.3]                        ; цикл при ошибке, если есть разрешение
U 1992, 2F82,95 ;18008          CLR WR[1]                               ; ожидаемые данные, если прекращение не состоялось
                ;18009                                                  ; (адрес 204)
U 1993, 3614,15 ;18010          MOV LS[T10] TO WR[0]                    ; выборка второго длинного слова
U 1994, 89AB,2C ;18011          JSR [CHECK.T32.ABORT]                   ; проверка, было ли прекращение
U 1995, 0998,B4 ;18012          JMP [LOOP.T32.3]                        ; цикл при ошибке, если есть разрешение
U 1996, FF82,15 ;18013          INC LS[ERROR.NUMBER]                    ; ошибка 4
U 1997, 0A17,DC ;18014          JSR [WRITE.CSR1.MME]                    ; установка в CSR1 бита MME (без бита ошибки)
U 1998, 4043,15 ;18015          ADD LS[#2] TO WR[2]                     ; значение 202 для печати адреса
U 1999, 2005,95 ;18016          MOV WR[2] TO WR[3]                      ; WR3 содержит 202
U 199A, C045,95 ;18017          ADD LS[#4] TO WR[3]                     ; значение 206 для печати следующего адреса
U 199B, 3628,15 ;18018          MOV LS[#FFFF] TO WR[0]                  ; выборка FFFF в рабочий регистр
U 199C, 3E1C,15 ;18019          MOV WR[0] TO LS[T14]                    ; ожидаемые данные для 1-го длинного слова (0000FFFF)
U 199D, A0C0,15 ;18020          COM WR[0]                               ; рабочий регистр содержит FFFF0000
U 199E, BE1E,15 ;18021          MOV WR[0] TO LS[T15]                    ; ожидаемые данные для 2-го длинного слова
                ;18022  LOOP.T32.4:
U 199F, 1F43,75 ;18023          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для чтения из адресов памяти 202-209
U 19A0, 3022,15 ;18024          MOV MEM.DATA TO WR[0]                   ; прием результата
U 19A1, B814,15 ;18025          MOV MEM.DATA TO LS[T10]                 ; прием результата второго цикла чтения
U 19A2, 09A9,4C ;18026          JSR [CHECK.T32.QUAD]                    ; проверка результатов
U 19A3, 0999,F4 ;18027          JMP [LOOP.T32.4]                        ; цикл при ошибке, если есть разрешение
U 19A4, FF82,15 ;18028          INC LS[ERROR.NUMBER]                    ; ошибка 5
                ;18029  LOOP.T32.5:
U 19A5, 1F43,75 ;18030          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для чтения из адресов памяти 202-209
U 19A6, 3022,15 ;18031          MOV MEM.DATA TO WR[0]                   ; прием результата
```

```
U 19A7, 0A1B,4C ;18032          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания CPU DR
U 19A8, B814,15 ;18033          MOV MEM.DATA TO LS[T10]                  ; прием результата второго цикла чтения
U 19A9, 09A9,4C ;18034          JSR [CHECK.T32.QUAD]                    ; проверка результатов
U 19AA, 099A,54 ;18035          JMP [LOOP.T32.5]                        ; цикл при ошибке, если есть разрешение
U 19AB, FF82,15 ;18036          INC LS[ERROR.NUMBER]                    ; ошибка 6
U 19AC, B676,15 ;18037          MOV LS[MME] TO WR[0]                    ; установка в рабочем регистре бита MME
U 19AD, C77A,15 ;18038          BIS LS[TB.PAR.DIAG] TO WR[0]            ; установка в рабочем регистре бита TB PAR DIAG
U 19AE, 8A17,FC ;18039          JSR [WRITE.CSR1]                        ; установка в CSR1 битов MME и TB PAR DIAG (для
                ;18040                                                  ; образования EERSUM при чтении)
U 19AF, 3653,15 ;18041          MOV LS[#200] TO WR[2]                   ; адрес для печати при ошибке
U 19B0, 4143,95 ;18042          SUB LS[#2] FROM WR[3]                   ; фиктивный адрес для второго цикла
                ;18043  LOOP.T32.6:
U 19B1, 1F43,75 ;18044          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для чтения из адресов памяти 202-209
U 19B2, 3022,15 ;18045          MOV MEM.DATA TO WR[0]                   ; выборка результата (первый цикл должен только
                ;18046                                                  ; прочитать адрес 200)
U 19B3, B814,15 ;18047          MOV MEM.DATA TO LS[T10]                  ; второй цикл должен быть прерван
U 19B4, 369E,95 ;18048          MOV LS[ONES] TO WR[1]                   ; выборка ожидаемых данных
U 19B5, 3E89,15 ;18049          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати (200)
U 19B6, 0869,3C ;18050          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
U 19B7, 099B,14 ;18051          JMP [LOOP.T32.6]                        ; цикл при ошибке, если есть разрешение
U 19B8, FF82,15 ;18052          INC LS[ERROR.NUMBER]                    ; ошибка 7
U 19B9, 0A17,DC ;18053          JSR [WRITE.CSR1.MME]                    ; установка в CSR1 бита MME
U 19BA, 3645,95 ;18054          MOV LS[#4] TO WR[3]                     ; занесение в рабочий регистр значения 4 для прибавления
                ;18055                                                  ; к адресу
U 19BB, E51C,15 ;18056          CLR LS[T14]                             ; ячейка LS содержит 0
U 19BC, FD1C,15 ;18057          COM LS[T14]                             ; ячейка LS содерит все 1 (ожидаемые данные для первого
                ;18058                                                  ; и третьего длинных слов. Данные для второго и
                ;18059                                                  ; четвертого длинных слов состоят из всех 0)
                ;18060  LOOP.T32.7:
U 19BD, 9E9D,75 ;18061          MEM.REQ[OCTA.READ.V.RCHK] ADRS[#0] DT[LONG] ; запрос для выполнения чтения восьмикратного
                ;18062                                                  ; слова
U 19BE, 3022,15 ;18063          MOV MEM.DATA TO WR[0]                   ; прием данных первого длинного слова
U 19BF, B814,15 ;18064          MOV MEM.DATA TO LS[T10]                  ; прием данных второго длинного слова
U 19C0, 3816,15 ;18065          MOV MEM.DATA TO LS[T11]                  ; прием данных третьего длинного слова
U 19C1, B818,15 ;18066          MOV MEM.DATA TO LS[T12]                  ; прием данных четвертого длинного слова
U 19C2, 09A9,EC ;18067          JSR [CHECK.T32.OCTA]                    ; проверка результатов
U 19C3, 099B,D4 ;18068          JMP [LOOP.T32.7]                        ; цикл при ошибке, если есть разрешение
U 19C4, FF82,15 ;18069          INC LS[ERROR.NUMBER]                    ; ошибка 8
                ;18070  LOOP.T32.8:
U 19C5, 9E9D,75 ;18071          MEM.REQ[OCTA.READ.V.RCHK] ADRS[#0] DT[LONG] ; запрос для выполнения чтения восьмикратного
                ;18072                                                  ; слова
U 19C6, 3022,15 ;18073          MOV MEM.DATA TO WR[0]                   ; прием первого длинного слова
U 19C7, 0A1B,4C ;18074          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания CPU DR
U 19C8, B814,15 ;18075          MOV MEM.DATA TO LS[T10]                  ; выборка второго длинного слова
U 19C9, 0A1B,4C ;18076          JSR [NOP.DELAY]                         ; выполнение 5 циклов задержки для задержания CPU DR
U 19CA, 3816,15 ;18077          MOV MEM.DATA TO LS[T11]                  ; прием теретьего длинного слова
U 19CB, B818,15 ;18078          MOV MEM.DATA TO LS[T12]                  ; прием четвертого длинного слова
U 19CC, 09A9,EC ;18079          JSR [CHECK.T32.OCTA]                    ; проверка результатов
U 19CD, 099C,54 ;18080          JMP [LOOP.T32.8]                        ; цикл при ошибке, если есть разрешение
U 19CE, FF82,15 ;18081          INC LS[ERROR.NUMBER]                    ; ошибка 9
U 19CF, 4043,15 ;18082          ADD LS[#2] TO WR[2]                     ; в рабочем регистре адрес 202 для печати
U 19D0, 3645,95 ;18083          MOV LS[#4] TO WR[3]                     ; занесение в рабочий регистр значения 4 для прибавления
                ;18084                                                  ; к адресу
U 19D1, 3628,15 ;18085          MOV LS[#FFFF] TO WR[0]                  ; рабочий регистр содержит FFFF
U 19D2, 3E1C,15 ;18086          MOV WR[0] TO LS[T14]                    ; запоминание FFFF в ячейке LS (ожидаемые данные для
```

```
                ;18087                                          ; первого и третьего длинных слов. Данные для второго и
                ;18088                                          ; четвертого длинных слов равны FFFF0000)
                ;18089    LOOP.T32.9:
U 19D3, 9E43,75 ;18090          MEM.REQ[OCTA.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18091                                          ; восьмикратного слова
U 19D4, 3022,15 ;18092          MOV MEM.DATA TO WR[0]            ; прием первого длинного слова данных
U 19D5, B814,15 ;18093          MOV MEM.DATA TO LS[T10]          ; прием второго длинного слова данных
U 19D6, 3816,15 ;18094          MOV MEM.DATA TO LS[T11]          ; прием третьего длинного слова данных
U 19D7, B818,15 ;18095          MOV MEM.DATA TO LS[T12]          ; прием четвертого длинного слова данных
U 19D8, 09A9,EC ;18096          JSR [CHECK.T32.OCTA]             ; проверка результатов
U 19D9, B99D,34 ;18097          JMP [LOOP.T32.9]                 ; цикл при ошибке, если есть разрешение
U 19DA, FF82,15 ;18098          INC LS[ERROR.NUMBER]             ; ошибка A
                ;18099    LOOP.T32.A:
U 19DB, 9E43,75 ;18100          MEM.REQ[OCTA.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18101                                          ; восьмикратного слова
U 19DC, 3022,15 ;18102          MOV MEM.DATA TO WR[0]            ; прием первого длинного слова данных
U 19DD, 0A1B,4C ;18103          JSR [NOP.DELAY]                  ; выполнение задержки из 5 циклов для задержания CPU DR
U 19DE, B814,15 ;18104          MOV MEM.DATA TO LS[T10]           ; прием втрого длинного слова данных
U 19DF, 3816,15 ;18105          MOV MEM.DATA TO LS[T11]           ; прием третьего длинного слова данных
U 19E0, B818,15 ;18106          MOV MEM.DATA TO LS[T12]           ; выборка четвертого длинного слова данных
U 19E1, 09A9,EC ;18107          JSR [CHECK.T32.OCTA]              ; проверка результатов
U 19E2, 099D,B4 ;18108          JMP [LOOP.T32.A]                  ; цикл при ошибке, если есть разрешение
U 19E3, FF82,15 ;18109          INC LS[ERROR.NUMBER]              ; ошибка B
U 19E4, B700,15 ;18110          MOV LS[CKBTS.0111100] TO WR[0]    ; выборка в WR0 контрольных битов для данных 0 или всех
                ;18111                                          ; единиц
U 19E5, A0C0,15 ;18112          COM WR[0]                         ; инвертирование контрольных битов для согласования с
                ;18113                                          ; аппаратурой
U 19E6, 452C,15 ;18114          BIC LS[#FFFFFF00] TO WR[0]        ; очистка всех битов, кроме младшего
U 19E7, C776,15 ;18115          BIS LS[MME] TO WR[0]              ; установка в WR0 бита MME
U 19E8, 8A17,FC ;18116          JSR [WRITE.CSR1]                  ; запись в CSR1 контрольных битов для данных 0 или всех
                ;18117                                          ; 1 и установка бита MME
U 19E9, B652,15 ;18118          MOV LS[#200] TO WR[0]             ; выборка в равочий регистр значения 200
U 19EA, 2040,15 ;18119          INC WR[0]                         ; рабочий регистр содержит 201
U 19EB, BE12,15 ;18120          MOV WR[0] TO LS[T9]               ; адрес для записи в LS (с установленным LVA00)
U 19EC, 5F60,15 ;18121          MCOM LS[#10000] TO WR[0]          ; WR0 содержит FFFEFFFF
U 19ED, BE14,15 ;18122          MOV WR[0] TO LS[T10]              ; запоминание в LS FFFEFFFF в качестве данных для
                ;18123                                          ; записи
U 19EE, 9D12,F5 ;18124          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы
                ;18125                                          ; LVA00 установлен для ветвления
U 19EF, B214,15 ;18126          WRITE.MEM LS[T10]                 ; запись в ячейку 200 кода FFFEFFFF с контрольными
                ;18127                                          ; битами для данных из всех 1
U 19F0, 6C13,95 ;18128          ADD WR[3] TO LS[T9]               ; увеличение адреса на 4
U 19F1, 9D12,F5 ;18129          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы
                ;18130                                          ; LVA00 установлен для ветвления
U 19F2, B260,15 ;18131          WRITE.MEM LS[#10000]              ; запись в ячейку 204 кода 10000(H) с контрольными
                ;18132                                          ; битами для всех 0
U 19F3, 6C13,95 ;18133          ADD WR[3] TO LS[T9]               ; увеличение адреса на 4
U 19F4, 9D12,F5 ;18134          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы.
                ;18135                                          ; LVA00 установлен для ветвления
U 19F5, B214,15 ;18136          WRITE.MEM LS[T10]                 ; запись в ячейку 208 кода FFFEFFFF с контрольными
                ;18137                                          ; битами для данных из всех 1
U 19F6, 6C13,95 ;18138          ADD WR[3] TO LS[T9]               ; увеличение адреса на 4
U 19F7, 9D12,F5 ;18139          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы.
                ;18140                                          ; LVA00 установлен для ветвления
U 19F8, B260,15 ;18141          WRITE.MEM LS[#10000]              ; запись в ячейку 20C кода 10000(H) с контрольными
```

```
                ;18142                                          ; битами для данных из всех 0
U 19F9, 6C13,95 ;18143          ADD WR[3] TO LS[T9]              ; увеличение адреса на 4
U 19FA, 9D12,F5 ;18144          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы.
                ;18145                                          ; LVA00 установлен для ветвления
U 19FB, B214,15 ;18146          WRITE.MEM LS[T10]               ; запись в ячейку 210 кода FFFEFFFF с контрольными
                ;18147                                          ; битами для данных из всех 1
                ;18148  LOOP.T32.B:
U 19FC, 9E43,75 ;18149  .       MEM.REQ[OCTA.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18150                                          ; восьмикратных слов
U 19FD, 3022,15 ;18151          MOV MEM.DATA TO WR[0]           ; прием первого длинного слова данных
U 19FE, BB14,15 ;18152          MOV MEM.DATA TO LS[T10]          ; прием второго длинного слова данных
U 19FF, 3816,15 ;18153          MOV MEM.DATA TO LS[T11]          ; прием третьего длинного слова данных
U 1A00, B818,15 ;18154          MOV MEM.DATA TO LS[T12]          ; прием четвертого длинного слова данных
U 1A01, 09A9,EC ;18155          JSR [CHECK.T32.OCTA]            ; проверка результатов
U 1A02, 099F,C4 ;18156          JMP [LOOP.T32.B]                ; цикл при ошибке, если есть разрешение
U 1A03, FF82,15 ;18157          INC LS[ERROR.NUMBER]            ; ошибка C
U 1A04, B652,15 ;18158          MOV LS[#200] TO WR[0]           ; выборка в рабочий регистр значения 200
U 1A05, 2040,15 ;18159          INC WR[0]                       ; рабочий регистр содержит 201
U 1A06, BE12,15 ;18160          MOV WR[0] TO LS[T9]              ; ячейка LS содержит адрес для записи (с установленным
                ;18161                                          ; LVA00)
U 1A07, 5F60,15 ;18162          MCOM LS[#10000] TO WR[0]        ; WR0 содержит FFFEFFFF
U 1A08, 4562,15 ;18163          BIC LS[BIT17] TO WR[0]          ; WR0 содержит FFFCFFFF
U 1A09, 3E1A,15 ;18164          MOV WR[0] TO LS[T13]            ; ячейка LS содержит FFFCFFFF в качестве данных для
                ;18165                                          ; записи
U 1A0A, 9D12,F5 ;18166          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической программы
                ;18167                                          ; LVA00 установлен для ветвления
U 1A0B, 321A,15 ;18168          WRITE.MEM LS[T13]               ; запись в ячейку 200 кода FFFCFFFF с контрольными
                ;18169                                          ; битами для данных из всех 1.
U 1A0C, DF40,15 ;18170          MCOM LS[#1] TO WR[0]            ; рабочий регистр содержит FFFFFFFE
U 1A0D, C542,15 ;18171          BIC LS[BIT1] TO WR[0]           ; рабочий регистр содержит FFFFFFFC
U 1A0E, 3E1A,15 ;18172          MOV WR[0] TO LS[T13]            ; ячейка LS содержит ожидаемые данные
U 1A0F, 3E89,15 ;18173          MOV WR[2] TO LS[ADDRESS.DATA]   ; адрес для печати при ошибке (202)
                .   ;18174  LOOP.T32.C:
U 1A10, 1F43,75 ;18175          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненнго чтения
                ;18176                                          ; четырехкратного слова
U 1A11, 3022,15 ;18177          MOV MEM.DATA TO WR[0]           ; выборка данных первого слова (после этого чтение
                ;18178                                          ; должно быть прекращено)
U 1A12, BB14,15 ;18179          MOV MEM.DATA TO LS[T10]          ; выборка второго длинного слова данных
U 1A13, 361A,95 ;18180          MOV LS[T13] TO WR[1]            ; выборка ожидаемых данных (FFFFFFFC)
U 1A14, 0B69,3C ;18181          JSR [CHECK.RESULT]              ; выборка результатов и проверка
U 1A15, B9A1,04 ;18182          JMP [LOOP.T32.C]                ; цикл при ошибке, если есть разрешение
U 1A16, 1952,75 ;18183          MEM.REQ[WRITE.P] ADRS[#200] DT[LONG]  ; запрос для записи в ячейку 200 правильных данных
U 1A17, 329E,15 ;18184          WRITE.MEM LS[ONES]              ; запись в память всех единиц
U 1A18, 6C13,95 ;18185          ADD WR[3] TO LS[T9]              ; увеличение адреса на 4
U 1A19, 9D12,F5 ;18186          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической
                ;18187                                          ; программы. LVA00 установлен для ветвления
U 1A1A, B2C0,15 ;18188          WRITE.MEM LS[#3(H)]             ; запись в ячейку 204 кода 3(H) с контрольными битами для
                ;18189                                          ; данных из всех 0
U 1A1B, FF82,15 ;18190          INC LS[ERROR.NUMBER]            ; ошибка D
U 1A1C, 3628,15 ;18191          MOV LS[#FFFF] TO WR[0]          ; рабочий регистр содержит FFFF
U 1A1D, 4760,15 ;18192          BIS LS[BIT16] TO WR[0]          ; рабочий регистр содержит 1FFFF
U 1A1E, C762,15 ;18193          BIS LS[BIT17] TO WR[0]          ; рабочий регистр содержит 3FFFF
U 1A1F, 3E1C,15 ;18194          MOV WR[0] TO LS[T14]            ; ожидаемые данные (3FFFF)
                ;18195  LOOP.T32.D:
U 1A20, 3E89,15 ;18196          MOV WR[2] TO LS[ADDRESS.DATA]   ; адрес для печати при ошибке
```

```
U 1A21, 1F43,75 ;18197          MEM.REQ[QUAD.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18198                                                      ; четырехкратного слова
U 1A22, 3022,15 ;18199          MOV MEM.DATA TO WR[0]                       ; прием первого длинного слова данных
U 1A23, 8814,15 ;18200          MOV MEM.DATA TO LS[T10]                     ; прием второго длинного слова данных (после этого
                ;18201                                                      ; чтение должно быть прекращено)
U 1A24, 361C,95 ;18202          MOV LS[T14] TO WR[1]                        ; выборка ожидаемых данных
U 1A25, 0869,3C ;18203          JSR [CHECK.RESULT]                          ; выборка результата и проверка
U 1A26, 89A2,04 ;18204          JMP [LOOP.T32.D]                            ; цикл при ошибке, если есть разрешение
U 1A27, DF28,95 ;18205          MCOM LS[#FFFF] TO WR[1]                     ; ожидаемые данные, если прекращение не состоялось
                ;18206                                                      ; (адрес 206)
U 1A28, 3614,15 ;18207          MOV LS[T10] TO WR[0]                        ; выборка второго длинного слова
U 1A29, 89AB,2C ;18208          JSR [CHECK.T32.ABORT]                       ; проверка, что чтение было прекращено
U 1A2A, 89A2,04 ;18209          JMP [LOOP.T32.D]                            ; цикл при ошибке, если есть разрешение
U 1A2B, FC12,15 ;18210          DEC LS[T9]                                  ; T9 теперь содержит 204
U 1A2C, 9912,75 ;18211          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]          ; запрос для записи по адресу 204 правильных данных
U 1A2D, B29C,15 ;18212          WRITE.MEM LS[ZERO]                          ; запись в память всех нулей
U 1A2E, FF12,15 ;18213          INC LS[T9]                                  ; ячейка LS теперь содержит 205
U 1A2F, 6C13,95 ;18214          ADD WR[3] TO LS[T9]                         ; увеличение адреса на 4
U 1A30, 9D12,F5 ;18215          MEM.REQ[MAINT.ECC.DATA] ADRS[T9] DT[LONG] ; запрос для использования диагностической
                ;18216                                                      ; программы. LVA00 установлен для ветвления
U 1A31, 321A,15 ;18217          WRITE.MEM LS[T13]                           ; запись в ячейку 208 кода FFFFFFFC с контрольными битами
                ;18218                                                      ; для всех 1
U 1A32, FF82,15 ;18219          INC LS[ERROR.NUMBER]                        ; ошибка E
U 1A33, 5F28,15 ;18220          MCOM LS[#FFFF] TO WR[0]                     ; рабочий регистр содержит FFFF0000
U 1A34, C560,15 ;18221          BIC LS[BIT16] TO WR[0]                      ; рабочий регистр содержит FFFE0000
U 1A35, 4562,15 ;18222          BIC LS[BIT17] TO WR[0]                      ; рабочий регистр содержит FFFC0000
U 1A36, 3E1A,15 ;18223          MOV WR[0] TO LS[T13]                        ; ожидаемые данные
                ;18224  LOOP.T32.E:
U 1A37, 3EB9,15 ;18225          MOV WR[2] TO LS[ADDRESS.DATA]               ; адрес для печати равен 202
U 1A38, 9E43,75 ;18226          MEM.REQ[OCTA.READ.V.RCHK] ADRS[#2] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18227                                                      ; восьмикратного слова
U 1A39, 3022,15 ;18228          MOV MEM.DATA TO WR[0]                       ; прием первого длинного слова данных
U 1A3A, 8814,15 ;18229          MOV MEM.DATA TO LS[T10]                     ; прием второго длинного слова данных
U 1A3B, 3816,15 ;18230          MOV MEM.DATA TO LS[T11]                     ; прием третьего длинного слова данных
U 1A3C, 8818,15 ;18231          MOV MEM.DATA TO LS[T12]                     ; выборка четвертого длинного слова данных
U 1A3D, B628,95 ;18232          MOV LS[#FFFF] TO WR[1]                      ; выборка ожидаемых данных (0000FFFF)
U 1A3E, 0869,3C ;18233          JSR [CHECK.RESULT]                          ; выборка результата и проверка
U 1A3F, 89A3,74 ;18234          JMP [LOOP.T32.E]                            ; цикл при ошибке, если есть разрешение
U 1A40, 6C89,95 ;18235          ADD WR[3] TO LS[ADDRESS.DATA]               ; прибавление 4 к адресу для печати
U 1A41, 361A,95 ;18236          MOV LS[T13] TO WR[1]                        ; ожидаемые данные (FFFF0000)
U 1A42, 3614,15 ;18237          MOV LS[T10] TO WR[0]                        ; полученные данные
U 1A43, 0869,3C ;18238          JSR [CHECK.RESULT]                          ; выборка полученых данных и проверка
U 1A44, 89A3,74 ;18239          JMP [LOOP.T32.E]                            ; цикл при ошибке, если есть разрешения
U 1A45, B628,95 ;18240          MOV LS[#FFFF] TO WR[1]                      ; ожидаемые данные, если прекращения нет
U 1A46, B616,15 ;18241          MOV LS[T11] TO WR[0]                        ; полученные данные
U 1A47, 89AB,2C ;18242          JSR [CHECK.T32.ABORT]                       ; проверка, что чтение прекращено
U 1A48, 89A3,74 ;18243          JMP [LOOP.T32.E]                            ; цикл при ошибке, если есть разрешение
U 1A49, FF82,15 ;18244          INC LS[ERROR.NUMBER]                        ; ошибка F
U 1A4A, B610,15 ;18245          MOV LS[T8] TO WR[0]                         ; выборка данных для буфера трансляции
U 1A4B, 4540,15 ;18246          BIC LS[BIT0] TO WR[0]                       ; отображение в адрес 0
U 1A4C, 3E10,15 ;18247          MOV WR[0] TO LS[T8]                         ; запоминание данных в LS
U 1A4D, 989C,F5 ;18248          MEM.REQ[WRITE.T8] ADRS[#0] DT[LONG]         ; запрос для записи по адресу T8 0
U 1A4E, 3210,15 ;18249          WRITE.MEM LS[T8]                            ; установка в T8 битов VALID, PROT, C, BYTE OFFSET и
                ;18250                                                      ; отображение вирт.адреса 0 в физ.адрес 0
U 1A4F, B610,15 ;18251          MOV LS[T8] TO WR[0]                         ; выборка данных для буфера трансляции
```

```
U 1A50, C57E,15 ;18252          BIC LS[BIT31] TO WR[0]                  ; очистка бита VALID
U 1A51, C740,15 ;18253          BIS LS[BIT0] TO WR[0]                   ; отображение вирт.адреса 0 в физ.адрес 200
U 1A52, 3E10,15 ;18254          MOV WR[0] TO LS[TB]                     ; запоминание данных в LS
U 1A53, 1B52,F5 ;18255          MEM.REQ[WRITE.TB] ADRS[BIT9] DT[LONG]   ; запрос для записи по адресу TB 1
U 1A54, 3210,15 ;18256          WRITE.MEM LS[TB]                        ; установка в TB битов PROT C и BYTE OFFSET и
                ;18257                                                  ; отображение вирт.адреса 0 в физ.адрес 200
U 1A55, B652,15 ;18258          MOV LS[#200] TO WR[0]                   ; рабочий регистр содержит 200
U 1A56, 4144,15 ;18259          SUB LS[#4] FROM WR[0]                   ; рабочий регистр содержит 1FC
U 1A57, BE12,15 ;18260          MOV WR[0] TO LS[T9]                     ; запоминание адреса 1FC в LS
U 1A58, 9912,75 ;18261          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи по адресу 1FC
U 1A59, B29C,15 ;18262          WRITE.MEM LS[ZERO]                      ; запись нулей по адресу 1FC
U 1A5A, C042,15 ;18263          ADD LS[#2] TO WR[0]                     ; рабочий регистр содержит 1FE
U 1A5B, BE12,15 ;18264          MOV WR[0] TO LS[T9]                     ; адрес для чтения (невыравненное чтение)
U 1A5C, 2001,15 ;18265          MOV WR[0] TO WR[2]                      ; запоминание адреса в WR2
                ;18266  LOOP.T32.F:
U 1A5D, 3E89,15 ;18267          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати
U 1A5E, 1F13,75 ;18268          MEM.REQ[QUAD.READ.V.RCHK] ADRS[T9] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18269                                                  ; четырехкратных слов по адресу 1FE
U 1A5F, 3022,15 ;18270          MOV MEM.DATA TO WR[0]                   ; выборка первого длинного слова данных
U 1A60, B814,15 ;18271          MOV MEM.DATA TO LS[T10]                 ; выборка второго длинного слова данных
U 1A61, DF28,95 ;18272          MCOM LS[#FFFF] TO WR[1]                 ; выборка ожидаемых данных (FFFF0000)
U 1A62, 0869,3C ;18273          JSR [CHECK.RESULT]                      ; выборка результата и проверка
U 1A63, 89A5,D4 ;18274          JMP [LOOP.T32.F]                        ; цикл при ошибке, если есть разрешение
U 1A64, B628,95 ;18275          MOV LS[#FFFF] TO WR[1]                  ; ожидаемые данные, если прекращения нет
U 1A65, 3614,15 ;18276          MOV LS[T10] TO WR[0]                    ; полученные данные
U 1A66, 89A8,2C ;18277          JSR [CHECK.T32.ABORT]                   ; проверка, что чтение прекращено
U 1A67, 89A5,D4 ;18278          JMP [LOOP.T32.F]                        ; цикл при ошибке, если есть разрешение
U 1A68, FF82,15 ;18279          INC LS[ERROR.NUMBER]                    ; ошибка 10
U 1A69, B652,15 ;18280          MOV LS[#200] TO WR[0]                   ; рабочий регистр содержит 200
U 1A6A, C146,15 ;18281          SUB LS[#8] FROM WR[0]                   ; рабочий регистр содержит 1F8
U 1A6B, BE12,15 ;18282          MOV WR[0] TO LS[T9]                     ; запоминание в LS адреса 1F8
U 1A6C, 9912,75 ;18283          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи по адресу 1F8
U 1A6D, 329E,15 ;18284          WRITE.MEM LS[ONES]                      ; запись единиц по адресу 1F8
U 1A6E, C042,15 ;18285          ADD LS[#2] TO WR[0]                     ; рабочий регистр содержит 1FA
U 1A6F, BE12,15 ;18286          MOV WR[0] TO LS[T9]                     ; адрес для чтения (невыравненное чтение)
U 1A70, 2001,15 ;18287          MOV WR[0] TO WR[2]                      ; запоминание адреса в WR2
                ;18288  LOOP.T32.10:
U 1A71, 3E89,15 ;18289          MOV WR[2] TO LS[ADDRESS.DATA]           ; пересылка 1FA на место адреса для печати
U 1A72, 9E13,75 ;18290          MEM.REQ[OCTA.READ.V.RCHK] ADRS[T9] DT[LONG] ; запрос для выполнения невыравненного чтения
                ;18291                                                  ; восьмикратного слова с адреса 1FA
U 1A73, 3022,15 ;18292          MOV MEM.DATA TO WR[0]                   ; выборка первого длинного слова данных
U 1A74, B814,15 ;18293          MOV MEM.DATA TO LS[T10]                 ; выборка второго длинного слова данных (после этого
                ;18294                                                  ; чтение должно быть прекращено)
U 1A75, 3816,15 ;18295          MOV MEM.DATA TO LS[T11]                 ; выборка третьего длинного слова данных
U 1A76, B818,15 ;18296          MOV MEM.DATA TO LS[T12]                 ; выборка четвертого длинного слова данных
U 1A77, B628,95 ;18297          MOV LS[#FFFF] TO WR[1]                  ; выборка ожидаемых данных (0000FFFF)
U 1A78, 0869,3C ;18298          JSR [CHECK.RESULT]                      ; выборка результата и проверка
U 1A79, 09A7,14 ;18299          JMP [LOOP.T32.10]                       ; цикл при ошибке, если есть разрешение
U 1A7A, 6C89,95 ;18300          ADD WR[3] TO LS[ADDRESS.DATA]           ; прибавление 4 к адресу для печати (1FE)
U 1A7B, DF28,95 ;18301          MCOM LS[#FFFF] TO WR[1]                 ; выборка ожидаемых данных (FFFF0000)
U 1A7C, 3614,15 ;18302          MOV LS[T10] TO WR[0]                    ; полученные данные
U 1A7D, 0869,3C ;18303          JSR [CHECK.RESULT]                      ; выборка ожидаемых данных и проверка
U 1A7E, 09A7,14 ;18304          JMP [LOOP.T32.10]                       ; цикл при ошибке, если есть разрешение
U 1A7F, B628,95 ;18305          MOV LS[#FFFF] TO WR[1]                  ; ожидаемые данные, если прекращение не состоялось
                ;18306                                                  ; (адрес 202)
```

```
U 1AB0, B616,15 ;18307          MOV LS[T11] TO WR[0]                    ; полученные данные
U 1AB1, B9AB,2C ;18308          JSR [CHECK.T32.ABORT]                   ; проверка, что чтение прекращено
U 1AB2, 09A7,14 ;18309          JMP [LOOP.T32.10]                       ; цикл при ошибке, если есть разрешение
U 1AB3, FF82,15 ;18310          INC LS[ERROR.NUMBER]                    ; ошибка 11
U 1AB4, 1C52,75 ;18311          MEM.REQ[OCTA.WRITE.P] ADRS[#200] DT[LONG] ; запрос для записи в память 4 длинных слов
U 1AB5, 329E,15 ;18312          WRITE.MEM LS[ONES]                      ; запись по адресу 200 тестовых данных из всех 1
U 1AB6, B29C,15 ;18313          WRITE.MEM LS[ZERO]                      ; запись по адресу 204 тестовых данных из всех 0
U 1AB7, 329E,15 ;18314          WRITE.MEM LS[ONES]                      ; запись по адресу 208 тестовых данных из всех 1
U 1AB8, B29C,15 ;18315          WRITE.MEM LS[ZERO]                      ; запись по адресу 20C тестовых данных из всех 0
U 1AB9, E51C,15 ;18316          CLR LS[T14]                             ; ячейка LS содержит 0
U 1ABA, FD1C,15 ;18317          COM LS[T14]                             ; ячейка LS содержит все 1 (ожидаемые данные для
                ;18318                                                  ; первого и третьего длинных слов. Данные для второго и
                ;18319                                                  ; четвертого длинных слов состоят из всех 0)
U 1ABB, 3653,15 ;18320          MOV LS[#200] TO WR[2]                   ; адрес начала чтения
                ;18321  LOOP.T32.11:
U 1ABC, 9E52,75 ;18322          MEM.REQ[OCTA.READ.P] ADRS[#200] DT[LONG] ; запрос для чтения восьмикратного слова по
                ;18323                                                  ; физическому адресу
U 1ABD, 3022,15 ;18324          MOV MEM.DATA TO WR[0]                   ; прием первого длинного слова данных
U 1ABE, B814,15 ;18325          MOV MEM.DATA TO LS[T10]                 ; прием второго длинного слова данных
U 1ABF, 3816,15 ;18326          MOV MEM.DATA TO LS[T11]                 ; прием третьего длинного слова данных
U 1AC0, B818,15 ;18327          MOV MEM.DATA TO LS[T12]                 ; прием четвертого длинного слова данных
U 1AC1, 09A9,EC ;18328          JSR [CHECK.T32.OCTA]                    ; проверка результатов
U 1AC2, 89A8,C4 ;18329          JMP [LOOP.T32.11]                       ; цикл при ошибке, если есть разрешение
U 1AC3, 09A8,E4 ;18330          JMP [END.T32]                           ; конец теста
                ;18331   ;
                ;18332   ;    Эта программа проверяет 2 длинных слова данных, полученных при чтении четы-
                ;18333   ;    рехкратного слова. Ожидаемый результат для первого длинного слова находится
                ;18334   ;    в ячейке LS T14, а для второго длинного слова в LS T15.
                ;18335   ;
                ;18336   CHECK.T32.QUAD:
U 1AC4, 3E89,15 ;18337          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати
U 1AC5, 361C,95 ;18338          MOV LS[T14] TO WR[1]                    ; выборка ожидаемых данных
U 1AC6, 0869,3C ;18339          JSR [CHECK.RESULT]                      ; проверка, что данные правильные
U 1AC7, 5B00,14 ;18340          RETURN                                  ; цикл при ошибке, если есть разрешение
U 1AC8, BE89,95 ;18341          MOV WR[3] TO LS[ADDRESS.DATA]           ; адрес для печати
U 1AC9, B61E,95 ;18342          MOV LS[T15] TO WR[1]                    ; ожидаемые данные для второго длинного слова
U 1ACA, 3614,15 ;18343          MOV LS[T10] TO WR[0]                    ; выборка результата из LS
U 1ACB, 0869,3C ;18344          JSR [CHECK.RESULT]                      ; проверка правильности данных
U 1ACC, 5B00,14 ;18345          RETURN                                  ; цикл при ошибке, если есть разрешение
U 1ACD, DB00,16 ;18346          RETURN+1                                ; возврат, если цикл при ошибке не выполняется
                ;18347   ;
                ;18348   ;    Эта подпрограмма проверяет 4 длинных слова данных, полученных при чтении
                ;18349   ;    восьмикратного слова. ожидаемый результат (для первого длинного слова) нахо-
                ;18350   ;    дится в ячейке LS T14.
                ;18351   ;
                ;18352   CHECK.T32.OCTA:
U 1ACE, 3E89,15 ;18353          MOV WR[2] TO LS[ADDRESS.DATA]           ; адрес для печати при ошибке
U 1ACF, 361C,95 ;18354          MOV LS[T14] TO WR[1]                    ; выборка ожидаемых данных
U 1AD0, 0869,3C ;18355          JSR [CHECK.RESULT]                      ; выборка результата и проверка
U 1AD1, 5B00,14 ;18356          RETURN                                  ; цикл при ошибке, если есть разрешение
U 1AD2, 6CB9,95 ;18357          ADD WR[3] TO LS[ADDRESS.DATA]           ; адрес для печати при ошибке
U 1AD3, 5F1C,95 ;18358          MCOM LS[T14] TO WR[1]                   ; выборка ожидаемых данных
U 1AD4, 3614,15 ;18359          MOV LS[T10] TO WR[0]                    ; полученные данные
U 1AD5, 0869,3C ;18360          JSR [CHECK.RESULT]                      ; проверка наличия ошибок
U 1AD6, 5B00,14 ;18361          RETURN                                  ; цикл при ошибке, если есть разрешение
```

```
U 1AA7, 6C89,95 ;18362          ADD WR[3] TO LS[ADDRESS.DATA]          ; адрес для печати при ошибке
U 1AA8, 361C,95 ;18363          MOV LS[T14] TO WR[1]                   ; выборка ожидаемых данных
U 1AA9, B616,15 ;18364          MOV LS[T11] TO WR[0]                   ; полученные данные
U 1AAA, 0869,3C ;18365          JSR [CHECK.RESULT]                     ; проверка наличия ошибок
U 1AAB, 5B00,14 ;18366          RETURN                                 ; цикл при ошибке, если есть разрешение
U 1AAC, 6C89,95 ;18367          ADD WR[3] TO LS[ADDRESS.DATA]          ; адрес для печати при ошибке
U 1AAD, 5F1C,95 ;18368          MCOM LS[T14] TO WR[1]                  ; выборка ожидаемых данных
U 1AAE, 3618,15 ;18369          MOV LS[T12] TO WR[0]                   ; полученные данные
U 1AAF, 0869,3C ;18370          JSR [CHECK.RESULT]                     ; проверка наличия ошибок
U 1AB0, 5B00,14 ;18371          RETURN                                 ; цикл при ошибке, если есть разрешение
U 1AB1, DB00,16 ;18372          RETURN+1                               ; возврат при отсутствии цикла по ошибке
                ;18373  ;
                ;18374  ;   Эта программа проверяет, что чтение было прекращено, путем сравнения полу-
                ;18375  ;   ченных данных с данными, какие были бы получены, если чтение не прекращено.
                ;18376  ;   Если данные равны, вызывается фиктивная ошибка и дается сообщение.
                ;18377  ;
                ;18378  CHECK.T32.ABORT:
U 1AB2, 6C89,95 ;18379          ADD WR[3] TO LS[ADDRESS.DATA]          ; увеличение на 4 адреса для печати
                ;18380            CMP WR[0] WITH WR[1],                ; проверка, что полученные данные равны тем, какие были
                ;18381                                                 ; бы получены, если чтение не прекращено
U 1AB3, 2640,B5 ;18382          DT(LONG)&SET.ALU.CC                    ; установка кодов условий
U 1AB4, B9AB,91 ;18383          JMP.IF[NEQ] TO [T32.RET+1]             ; переход, если не равны. Данные правильные
U 1AB5, B66E,15 ;18384          MOV LS[NA.EXP.REC] TO WR[0]            ; установка бита для запрета печати ожидаемых и
                ;18385                                                 ; полученных данных
U 1AB6, 6D80,15 ;18386          BIS WR[0] TO LS[CONTROL.STATUS]         ; установка этого бита в слове управления и состояния
U 1AB7, 0869,3C ;18387          JSR [CHECK.RESULT]                     ; вызов программы обработки ошибок с данными,
                ;18388                                                 ; установленными для вызова ошибки
U 1AB8, 09AB,B4 ;18389          JMP [CLEAR.T32.NA]                     ; очистка запрета печати ожидаемых и полученных данных
                ;18390  T32.RET+1:
U 1AB9, B9AB,BC ;18391          JSR [CLEAR.T32.NA]                     ; очистка бита запрета, если установлен
U 1ABA, DB00,16 ;18392          RETURN+1                               ; возврат для отсутствия цикла при ошибке
                ;18393  CLEAR.T32.NA:
U 1ABB, B670,15 ;18394          MOV LS[OTHER.DATA] TO WR[0]            ; установка бита, разрешающего печать под OTHER (другие
                ;18395                                                 ; данные) в сообщении об ошибке. Очистка бита запрета
U 1ABC, 3E80,15 ;18396          MOV WR[0] TO LS[CONTROL.STATUS]         ; занесение в слово управления и состояния
U 1ABD, 5B00,14 ;18397          RETURN                                 ; возврат, если выполняется цикл при ошибке, или
                ;18398                                                 ; возврат при T32.RET+1
                ;18399  END.T32:
```

```
;18400  .PAGE "ТЕСТ 33 - тест временных соотношений модуля и системы памяти (модули MOS или MCT)"
;18401  ;
;18402  ;
;18403  ; ОПИСАНИЕ ТЕСТА:
;18404  ;
;18405  ;       Этот тест выполняет тестирование основной памяти при помощи бегущей инвер-
;18406  ;   сии. Тест бегущей инверсии похож на тест "МАРШ", за исключением метода увели-
;18407  ;   чения адреса. Адрес увеличивается таким образом, что каждый адресный бит пере-
;18408  ;   ключается поочередно с максимальной скоростью. Это осуществляется путем при-
;18409  ;   бавления константы со сдвигаемой единицей при каждом новом проходе теста.
;18410  ;   Первый проход теста записывает фон из нулей, затем считываются нули и запи-
;18411  ;   сываются единицы по адресу 0. Затем к адресу прибавляется 4 и чтение/запись
;18412  ;   повторяется. Это продолжается до тех пор, пока не произойдет доступ ко всей
;18413  ;   памяти. Затем модуль памяти проверяется с использованием адресов, уменьшаю-
;18414  ;   щихся таким же способом. Это переключает бит 2 при максимальной скорости.
;18415  ;       Второй проход аналогичен первому за исключением того, что в качестве конс-
;18416  ;   танты, прибавляемой к адресу, используется 8. Когда достигнут наибольший ад-
;18417  ;   рес памяти, увеличивается бит 2 и наращивание повторяется. Адресация выглядит
;18418  ;   следующим образом (16-ричн.): 0,8,10,18... до максимального адреса памяти, за-
;18419  ;   тем 4,C,14,1C... снова до максимального адреса памяти. Таким образом, с мак-
;18420  ;   симальной скоростью переключается бит 3. Затем тест повторяется с использо-
;18421  ;   ванием записи и чтения по байтам, причем биты 0 и 1 переключаются с макси-
;18422  ;   мальной скоростью для них.
;18423  ;       Оба теста повторяются в следующем проходе с инвертированными данными.
;18424  ;       Выполняется тест "МАРШ", использующий обращения невыравненными длинными
;18425  ;   словами и восьмикратными словами, для обнаружения неполадок, связанных с вре-
;18426  ;   менными соотношениями в выполнении этих микропрограмм. Эти тесты выполняются
;18427  ;   без переключения различных битов для поддержания в разумных пределах времен
;18428  ;   выполнения. Доступ восьмикратными словами осуществляется с начального адреса
;18429  ;   4 для обеспечения максимального пересечения границ страницы.
;18430  ;
;18431  ; ПРЕДПОЛОЖЕНИЯ:
;18432  ;
;18433  ;       Принимается, что все предыдущие тесты прошли успешно.
;18434  ;
;18435  ; ШАГИ ТЕСТА:
;18436  ;
;18437  ;       1)Установка в LS номера ошибки и номера модуля (для распечатки ошибок) и
;18438  ;   очистка в LS номера предыдущей ошибки.
;18439  ;       2)Определение размера и инициализация памяти. Сообщение о числе обнару-
;18440  ;   женных блоков по 256K байт.
;18441  ;       3)Запись фона единиц во всей доступной памяти.
;18442  ;       4)Начиная с адреса 0, чтение единиц, проверка результата, затем запись ну-
;18443  ;   лей.Увеличение адреса на X (X-текущий бит, переключаемый с максимальной ско-
;18444  ;   ростью).
;18445  ;       5)Повторение, пока не будет достигнут максимальный адрес памяти. Затем
;18446  ;   увеличение на 4 младших 18 битов и очистка битов, старше бита 17. Если пере-
;18447  ;   ключаемый бит теперь установлен, то продолжение. Иначе повторение шагов 4 и
;18448  ;   5, начиная с текущего адреса.
;18449  ;       6)Повторение шагов 4 и 5 до тех пор, пока каждый бит от 2 до 17 пройдет
;18450  ;   переключение со своей максимальной скоростью (т.е. X уже был равен каждому би-
;18451  ;   ту от 2 до 17).
;18452  ;       7)Повторение шагов от 4 до 6, начиная с максимального адреса памяти и дви-
;18453  ;   гаясь вниз.
;18454  ;       8)Повторение шагов от 4 до 8, используя биты 0 и 1 в качестве переключа-
```

```
                #18455  ;   емых битов и выполняя запись и чтение по байтам.
                #18456  ;      9)Повторение шагов от 4 до 8 с инвертированными данными.
                #18457  ;     10)Выполнение обычного теста "МАРШ" с использованием обращений к невырав-
                #18458  ;   ненным длинным словам. Выполнение одного прохода вверх и одного прохода вниз.
                #18459  ;     11)То же, что и в шаге 10, но с обращениями к восьмикратным словам.
                #18460  ;
                #18461  ; ОШИБКИ:
                #18462  ;
                #18463  ;   ПРИМЕЧАНИЕ: если происходит отказ, адрес, по которому было обращение, находится
                #18464  ;   в LS9 и может просматриваться посредством EX LS 9. Бит, переключаемый с мак-
                #18465  ;   симальной скоростью, находится в WR3 и может просматриваться путем печати EX
                #18466  ;   WR 3. OTHER (другие данные) не используются для того, чтобы обеспечить, насколько
                #18467  ;   это возможно, быстрое выполнение теста.
                #18468  ;
                #18469  ; ошибка 1 - тест бегущей инверсии не прошел успешно на восходящем пути с фоном из
                #18470  ;            единиц. Переключаемым является один из битов от 2 до 17.
                #18471  ; ошибка 2 - тест бегущей инверсии не прошел успешно на нисходящем пути с фоном из
                #18472  ;            единиц. Переключаемым является один из битов от 2 до 17.
                #18473  ; ошибка 3 - тест бегущей инверсии не прошел успешно на восходящем пути с фоном из
                #18474  ;            единиц. Переключаемым битом является либо бит 0, либо 1.
                #18475  ; ошибка 4 - тест бегущей инверсии не прошел успешно на нисходящем пути с фоном из
                #18476  ;            единиц. Переключаемым является либо бит 0, либо 1.
                #18477  ; ошибка 5 - тест бегущей инверсии не прошел успешно на восходящем пути с фоном из
                #18478  ;            0. Переключаемым битом является один из битов от 2 до 17.
                #18479  ; ошибка 6 - тест бегущей инверсии не прошел успешно на нисходящем пути с фоном из
                #18480  ;            0. Переключаемым битом является один из битов от 2 до 17.
                #18481  ; ошибка 7 - тест бегущей инверсии не прошел успешно на восходящем пути с фоном из
                #18482  ;            0. Переключаемым битом является либо бит 0, либо 1.
                #18483  ; ошибка 8 - тест бегущей инверсии не прошел успешно на нисходящем пути с фоном из
                #18484  ;            0. Переключаемым битом является либо бит 0, либо 1.
                #18485  ; ошибка 9 - тест "МАРШ" для невыравненных длинных слов не прошел успешно на вос-
                #18486  ;            ходящем пути с фоном из 0.
                #18487  ; ошибка A - тест "МАРШ" для невыравненных длинных слов не прошел успешно на нис-
                #18488  ;            ходящем пути с фоном из 0.
                #18489  ; ошибка B - тест"МАРШ" для обращений к восьмикратным словам не прошел успешно на
                #18490  ;            восходящем пути с фоном из 0.
                #18491  ; ошибка C - тест "МАРШ" для обращений к восьмикратным словам не прошел успешно на
                #18492  ;            нисходящем пути с фоном из 0.
                #18493  ;
                #18494  ; НАЛАДКА:
                #18495  ;
                #18496  ;      ПРИМЕЧАНИЕ: если MCT работает в пошаговом режиме, регенерация модуля памя-
                #18497  ;   ти не выполняется. Таким образом данные модуля памяти будут потеряны.
                #18498  ;
                #18499  ;      ОШИБКИ ОТ 1 ДО C - любая из этих ошибок свидетельствует о возможной неис-
                #18500  ;   правности в плате модуля памяти, к которому происходит обращение, что указыва-
                #18501  ;   ется адресом в LS9. Этот адрес и бит данных, в котором проявляется отказ, мо-
                #18502  ;   жет помочь в определении неисправной микросхемы ЗУПВ. Если не срабатывают все
                #18503  ;   биты, то проблема может быть связана с платой MCT, с медленной микросхемой в
                #18504  ;   логике регистра виртуального адреса. Так как этот тест проверяет работу
                #18505  ;   логических схем в динамике, то трудно указать место неисправности без наблю-
                #18506  ;   дения адресных линий.
                #18507   T.33:
U 1ABE, B65E,15 #18508           MOV LSCBEGIN.TEST] TO WR[0]              ; установка в WR0 бита 15 для слова управления и
                #18509                                                    ; состояния
```

```
U 1ABF, 3E80,15 ;18510          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка бита 15 в слове управления и состояния. Бит
                ;18511                                                   ; 15 указывает для консольного процессора начало теста
U 1AC0, 10E0,15 ;18512          MISC [SET.CP.ATTN]                       ; выдача для консольного процессора CPU ATTN
                ;18513  WAIT.T33.0:
U 1AC1, 89AC,14 ;18514          JMP [WAIT.T33.0]                         ; цикл для ожидания ответа консольного процессора
U 1AC2, B690,95 ;18515          MOV LS[ERROR.CONTROL] TO WR[1]           ; выборка слова управления ошибками
                ;18516            BIT LS[APT.PRESENT] WITH WR[1],        ; проверка наличия APT
U 1AC3, D94A,B5 ;18517          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1AC4, 09AC,89 ;18518          JMP.IF[BITS.CLR] TO [T33.START]          ; если не APT, переход к выполнению теста
U 1AC5, 3692,95 ;18519          MOV LS[APT.PARAM] TO WR[1]               ; иначе выборка параметров APT
                ;18520            BIT LS[#10000] WITH WR[1],             ; проверка, установлен ли бит программного обеспечения
                ;18521                                                   ; для работы под APT
U 1AC6, 5960,B5 ;18522          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1AC7, 09BC,D1 ;18523          JMP.IF[BIT.SET] TO [END.T33]             ; если бит установлен, пропуск теста, иначе выполнение
                ;18524                                                   ; этого теста
                ;18525  T33.START:
U 1AC8, 0A1A,9C ;18526          JSR [SETUP]                              ; установка маски, кода модуля и т.д. и очистка CSR1
                ;18527                                                   ; MCT
U 1AC9, 4748,15 ;18528          BIS LS[ARRAY] TO WR[0]                   ; добавление платы модуля памяти в качестве тестируемого
                ;18529                                                   ; модуля
U 1ACA, 3E8C,15 ;18530          MOV WR[0] TO LS[MODULE.NUM]              ; запоминание в LS (тестируемыми модулями являются
                ;18531                                                   ; модуль памяти или модуль MCT)
                ;18532            MOV LS[MM.LASTADDR+1] TO WR[2],        ; выборка из LS последнего адреса+1
U 1ACB, 3625,35 ;18533          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1ACC, 89AC,E1 ;18534          JMP.IF[NEQ] TO [T33.CONTINUE]            ; продолжение тестирования, если последний адрес+1 не
                ;18535                                                   ; нуль
U 1ACD, 8A19,4C ;18536          JSR [SIZE.MEMORY]                        ; иначе определение размера памяти, печать результата и
                ;18537                                                   ; продолжение
                ;18538  T33.CONTINUE:
U 1ACE, 3628,15 ;18539          MOV LS[#FFFF] TO WR[0]                   ; рабочий регистр содержит FFFF
U 1ACF, 4760,15 ;18540          BIS LS[#10000] TO WR[0]                  ; рабочий регистр содержит 1FFFF
U 1AD0, C762,15 ;18541          BIS LS[#20000] TO WR[0]                  ; рабочий регистр содержит 3FFFF
U 1AD1, 3E10,15 ;18542          MOV WR[0] TO LS[T8]                      ; запоминание в LS для маскирования младших битов
U 1AD2, 6518,15 ;18543          CLR LS[T12]                              ; очистка ячейки LS
U 1AD3, 7D18,15 ;18544          COM LS[T12]                              ; данные в LS T12 содержат все единицы
U 1AD4, E51A,15 ;18545          CLR LS[T13]                              ; данные в LS T13 содержат все нули
U 1AD5, 89BC,1C ;18546          JSR [BACK.T33]                           ; переход к записи фона единиц
U 1AD6, 89AE,BC ;18547          JSR [T33.1.5]                            ; переход к тесту с переключением битов 2 и выше
U 1AD7, B62C,15 ;18548          MOV LS[#FFFFFF00] TO WR[0]               ; пересылка в рабочий регистр FFFFFF00
U 1AD8, 3E8A,15 ;18549          MOV WR[0] TO LS[ERROR.MASK]              ; проверка только одного байта
U 1AD9, 09BC,AC ;18550          JSR [T33.INC.ERR]                        ; установка номера ошибки 3
U 1ADA, 09B1,DC ;18551          JSR [T33.3.7]                            ; переход к выполнению теста с переключением битов
                ;18552                                                   ; 0 и 1
U 1ADB, E58A,15 ;18553          CLR LS[ERROR.MASK]                       ; проверка всех битов
U 1ADC, 09BC,AC ;18554          JSR [T33.INC.ERR]                        ; установка номера ошибки на 5
U 1ADD, 7D18,15 ;18555          COM LS[T12]                              ; инвертирование данных из всех 1 в данные из всех 0
U 1ADE, FD1A,15 ;18556          COM LS[T13]                              ; инвертирование данных из всех 0 в данные из всех 1
U 1ADF, 89BC,1C ;18557          JSR [BACK.T33]                           ; запись фона нулей
U 1AE0, 89AE,BC ;18558          JSR [T33.1.5]                            ; выполнение теста с переключением битов 2 и выше с
                ;18559                                                   ; инвертированными данными
U 1AE1, B62C,15 ;18560          MOV LS[#FFFFFF00] TO WR[0]               ; рабочий регистр содержит FFFFFF00
U 1AE2, 3E8A,15 ;18561          MOV WR[0] TO LS[ERROR.MASK]              ; проверка только одного байта
U 1AE3, 09BC,AC ;18562          JSR [T33.INC.ERR]                        ; установка номера ошибки на 7
U 1AE4, 09B1,DC ;18563          JSR [T33.3.7]                            ; выполнение теста с переключением битов 0 и 1 с
                ;18564                                                   ; инвертированными данными
```

```
U 1AE5, 09BC,AC ;18565          JSR [T33.INC.ERR]                   ; установка номера ошибки 9
U 1AE6, E58A,15 ;18566          CLR LS[ERROR.MASK]                  ; проверка всех битов
U 1AE7, 09B5,3C ;18567          JSR [T33.9]                         ; выполнение теста с обращением к невыравненным длинным
                ;18568                                              ; словам
U 1AE8, FF82,15 ;18569          INC LS[ERROR.NUMBER]                ; установка номера ошибки 8
U 1AE9, 89B7,5C ;18570          JSR [T33.8]                         ; выполнение теста с обращением к восьмикратным словам
U 1AEA, 09BC,D4 ;18571          JMP [END.T33]                       ; конец теста
                ;18572  T33.1.5:
U 1AEB, 3645,95 ;18573          MOV LS[BIT2] TO WR[3]               ; бит, который будет переключаться с максимальной
                ;18574                                              ; скоростью. Первым переключаемым битом является бит 2
                ;18575                                              ; ЗУПВ
U 1AEC, B645,15 ;18576          MOV LS[BIT2] TO WR[2]               ; установка инкремента
U 1AED, 09BB,DC ;18577          JSR [CALL.MONITOR]                  ; вызов монитора для извещения, что тест продолжается
                ;18578  REPEAT.T33.1.5:
U 1AEE, 6512,15 ;18579          CLR LS[T9]                          ; запуск с адреса 0 модуля памяти
                ;18580  LOOP.T33.1.5:
U 1AEF, B618,95 ;18581          MOV LS[T12] TO WR[1]                ; ожидаемые данные
U 1AF0, 1813,F5 ;18582          MEM.REQ[READ.P] ADRS[T9] DT[LONG]   ; запрос для чтения из модуля памяти
U 1AF1, 3022,15 ;18583          MOV MEM.DATA TO WR[0]               ; прием результата
U 1AF2, 0869,3C ;18584          JSR [CHECK.RESULT]                  ; проверка наличия ошибок
U 1AF3, 89AE,F4 ;18585          JMP [LOOP.T33.1.5]                  ; цикл при ошибке, если есть разрешение
U 1AF4, 9912,75 ;18586          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в модуль памяти инвертированных
                ;18587                                              ; данных
U 1AF5, 321A,15 ;18588          WRITE.MEM LS[T13]                   ; запись в текущую ячейку инвертированных данных
U 1AF6, 6C13,95 ;18589          ADD WR[3] TO LS[T9]                 ; переключение бита в адресе
U 1AF7, 3612,15 ;18590          MOV LS[T9] TO WR[0]                 ; выборка откорректированного адреса
U 1AF8, 4510,15 ;18591          BIC LS[T8] TO WR[0]                 ; очистка младших 18 битов
                ;18592            CMP WR[0] WITH LS[MM.LASTADDR+1],  ; проверка, что увеличение превысило максимальный адрес
U 1AF9, 4F24,35 ;18593          DT(LONG)&SET.ALU.CC                 ; и установка кодов условий
U 1AFA, 89AE,F1 ;18594          JMP.IF[NEQ] TO [LOOP.T33.1.5]       ; если не превышает максимального адреса памяти, то
                ;18595                                              ; повторение со следующим адресом
U 1AFB, EC13,15 ;18596          ADD WR[2] TO LS[T9]                 ; иначе увеличение адреса ниже переключаемого бита
U 1AFC, B610,15 ;18597          MOV LS[T8] TO WR[0]                 ; подготовка к очистке битов выше бита 17
U 1AFD, EE12,15 ;18598          AND WR[0] TO LS[T9]                 ; очистка битов выше бита 17
                ;18599            BIT WR[3] WITH LS[T9],            ; если переключаемый бит теперь установлен, то проход
                ;18600                                              ; выполнен
U 1AFE, D913,B5 ;18601          DT(LONG)&SET.ALU.CC                 ; установка кодов условий
U 1AFF, 09AE,F9 ;18602          JMP.IF[BITS.CLR] TO [LOOP.T33.1.5]  ; если проход не выполнен, то повторение теста при новом
                ;18603                                              ; адресе. Иначе переход к запуску по убывающим адресам
U 1B00, 09BB,DC ;18604          JSR [CALL.MONITOR]                  ; вызов монитора для извещения, что тест продолжается
U 1B01, FF82,15 ;18605          INC LS[ERROR.NUMBER]                ; следующий номер ошибки
U 1B02, 3624,15 ;18606          MOV LS[MM.LASTADDR+1] TO WR[0]      ; запуск со старшего адреса памяти
U 1B03, 4144,15 ;18607          SUB LS[#4] FROM WR[0]               ; уменьшение на 4 для получения истинного старшего
                ;18608                                              ; адреса
U 1B04, BE14,15 ;18609          MOV WR[0] TO LS[T10]                ; запоминание в LS
U 1B05, BE12,15 ;18610          MOV WR[0] TO LS[T9]                 ; текущий адрес в T9
U 1B06, AE46,15 ;18611          SUB WR[3] FROM WR[0]                ; вычисление адреса, на котором должен произойти останов
U 1B07, 3E16,15 ;18612          MOV WR[0] TO LS[T11]                ; запоминание его в LS
                ;18613  LOOP.T33.2.6:
U 1B08, 361A,95 ;18614          MOV LS[T13] TO WR[1]                ; ожидаемые данные
U 1B09, 1813,F5 ;18615          MEM.REQ[READ.P] ADRS[T9] DT[LONG]   ; запрос для чтения из модуля памяти
U 1B0A, 3022,15 ;18616          MOV MEM.DATA TO WR[0]               ; выборка результата
U 1B0B, 0869,3C ;18617          JSR [CHECK.RESULT]                  ; проверка наличия ошибок
U 1B0C, 09B0,84 ;18618          JMP [LOOP.T33.2.6]                  ; цикл при ошибке, если есть разрешение
U 1B0D, 9912,75 ;18619          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в модуль памяти инвертированных
```

```
                ;18620                                  ; данных
1B0E, B218,15   ;18621          WRITE.MEM LSCT12]       ; запись в данную ячейку инвертированных данных
                ;18622            SUB WRC3] FROM LSCT9],; переключение бита в адресе
1B0F, 6B13,B5   ;18623          DT(LONG)&SET.ALU.CC     ; и установка кодов условий
1B10, 89B0,80   ;18624          JMP.IFCN,CLR] TO CLOOP.T33.2.6]  ; если не меньше чем 0, повторение с новым адресом
1B11, 3614,15   ;18625          MOV LSCT10] TO WRC0]    ; иначе выборка последнего адреса, с которого начинался
                ;18626                                  ; проход
1B12, 2E44,15   ;18627          SUB WRC2] FROM WRC0]    ; уменьшение адреса
1B13, BE14,15   ;18628          MOV WRC0] TO LSCT10]    ; запоминание адреса
1B14, BE12,15   ;18629          MOV WRC0] TO LSCT9]     ; адрес в T9
                ;18630            CMP LSCT11] WITH WRC0],; если адрес равен адресу в T11, конец выполнения
1B15, 4E16,35   ;18631          DT(LONG)&SET.ALU.CC     ; установка кодов условий
1B16, 09B0,81   ;18632          JMP.IFCNEQ] TO CLOOP.T33.2.6]  ; повторение теста с новым адресом, если не конец
1B17, 09BB,DC   ;18633          JSR CCALL.MONITOR]      ; вызов монитора для извещения, что тест продолжается
1B18, FC82,15   ;18634          DEC LSCERROR.NUMBER]    ; уменьшение номера ошибки
1B19, A3C1,95   ;18635          ROL WRC3]               ; следующий бит для переключения с максимальной
                ;18636                                  ; скоростью
                ;18637            BIT LSC#40000] WITH WRC3],; проверка, что бит 18 установлен
1B1A, 5965,B5   ;18638          DT(LONG)&SET.ALU.CC     ; и установка кодов условий
1B1B, 89AE,E9   ;18639          JMP.IFCBITS.CLR] TO CREPEAT.T33.1.5]  ; повторение теста с новым переключаемым битом, если бит
                ;18640                                  ; 18 не установлен
1B1C, 5B00,14   ;18641          RETURN                  ; иначе конец выполнения
                ;18642  T33.3.7:
1B1D, 3641,15   ;18643          MOV LSCBIT0] TO WRC2]   ; установка инкремента (инкрементирует адресный бит 0
                ;18644                                  ; микросхемы ЗУПВ)
1B1E, B641,95   ;18645          MOV LSCBIT0] TO WRC3]   ; бит, который будет переключаться с максимальной
                ;18646                                  ; скорость». Первым переключаемым битом является бит 0
                ;18647                                  ; адреса ЗУПВ
1B1F, 09BB,BC   ;18648          JSR CSETUP.T33.COUNTER&CALL.MONITOR]  ; установка счетчика и вызов монитора для извещения, что
                ;18649                                  ; тест продолжается
                ;18650  REPEAT.T33.3.7:
1B20, 6512,15   ;18651          CLR LSCT9]              ; запуск с адреса 0 модуля памяти
                ;18652  LOOP.T33.3.7:
1B21, B618,95   ;18653          MOV LSCT12] TO WRC1]    ; ожидаемые данные
1B22, 1813,95   ;18654          MEM.REQCREAD,P] ADRSCT9] DTCBYTE]  ; запрос для чтения из модуля памяти
1B23, 3022,15   ;18655          MOV MEM.DATA TO WRC0]   ; выборка результата
1B24, 0869,3C   ;18656          JSR CCHECK.RESULT]      ; проверка наличия ошибок
1B25, 89B2,14   ;18657          JMP CLOOP.T33.3.7]      ; цикл при ошибке, если есть разрешение
1B26, 9912,15   ;18658          MEM.REQCWRITE,P] ADRSCT9] DTCBYTE]  ; запрос для записи в модуль памяти инвертированных
                ;18659                                  ; данных
1B27, 321A,15   ;18660          WRITE.MEM LSCT13]       ; запись в данную ячейку инвертированных данных
                ;18661            DEC LSCT14],          ; уменьшение счетчика
1B28, FC1C,35   ;18662          DT(LONG)&SET.ALU.CC     ; и установка кодов условий
1B29, 89B2,B1   ;18663          JMP.IFCNEQ] TO CCONT.T33.3.7]  ; перехода, если счетчик еще не 0
1B2A, 09BB,BC   ;18664          JSR CSETUP.T33.COUNTER&CALL.MONITOR]  ; установка счетчика и вызов монитора для извещения, что
                ;18665                                  ; тест продолжается
                ;18666  CONT.T33.3.7:
1B2B, 6C13,95   ;18667          ADD WRC3] TO LSCT9]     ; переключение бита в адресе
1B2C, 3612,15   ;18668          MOV LSCT9] TO WRC0]     ; выборка откорректированного адреса
1B2D, 4510,15   ;18669          BIC LSCT8] TO WRC0]     ; очистка младших 18 битов
                ;18670            CMP WRC0] WITH LSCMM.LASTADDR+1],  ; проверка, что превышен старший адрес памяти
1B2E, 4F24,35   ;18671          DT(LONG)&SET.ALU.CC     ; и установка кодов условии
1B2F, 99B2,11   ;18672          JMP.IFCNEQ] TO CLOOP.T33.3.7]  ; если не превышен старший адрес памяти, повторение со
                ;18673                                  ; следующим адресом
1B30, EC13,15   ;18674          ADD WRC2] TO LSCT9]     ; иначе увеличение адреса ниже переключаемого бита
```

```
U 1B31, B610,15 ;18675          MOV LS[T8] TO WR[0]                     ; подготовка к очистке битов выше 17
U 1B32, EE12,15 ;18676          AND WR[0] TO LS[T9]                     ; очистка битов выше 17
                ;18677            BIT WR[3] WITH LS[T9],                ; если теперь установлен переключаемый бит, то конец
                ;18678                                                  ; выполнения
U 1B33, D913,B5 ;18679          DT(LONG)&SET.ALU.CC                     ; установка кодов условий
U 1B34, 09B2,19 ;18680          JMP.IF[BITS.CLR] TO [LOOP.T33.3.7]      ; если нет, то повторение теста с нового адреса, иначе
                ;18681                                                  ; перехода к запуску по нисходящим адресам
U 1B35, FF82,15 ;18682          INC LS[ERROR.NUMBER]                    ; ошибка 4 или 8
U 1B36, 3624,15 ;18683          MOV LS[MM.LASTADDR+1] TO WR[0]           ; запуск со старшего адреса памяти
U 1B37, 2100,15 ;18684          DEC WR[0]                               ; уменьшение на 1 для получения истинного старшего
                ;18685                                                  ; адреса
U 1B38, BE14,15 ;18686          MOV WR[0] TO LS[T10]                     ; запоминание в LS
U 1B39, BE12,15 ;18687          MOV WR[0] TO LS[T9]                      ; текущий адрес в T9
U 1B3A, AE46,15 ;18688          SUB WR[3] FROM WR[0]                     ; также вычисление адреса, по которому должен произойти
                ;18689                                                  ; останов
U 1B3B, 3E16,15 ;18690          MOV WR[0] TO LS[T11]                     ; запоминание его в LS
                ;18691    LOOP.T33.6.8:
U 1B3C, 361A,95 ;18692          MOV LS[T13] TO WR[1]                     ; ожидаемые данные
U 1B3D, 1813,95 ;18693          MEM.REQ[READ.P] ADRS[T9] DT[BYTE]        ; запрос для чтения из модуля памяти
U 1B3E, 3022,15 ;18694          MOV MEM.DATA TO WR[0]                    ; выборка результатов
U 1B3F, 0869,3C ;18695          JSR [CHECK.RESULT]                       ; проверка на ошибки
U 1B40, 89B3,C4 ;18696          JMP [LOOP.T33.6.8]                       ; цикл при ошибке, если есть разрешение
U 1B41, 9912,15 ;18697          MEM.REQ[WRITE.P] ADRS[T9] DT[BYTE]       ; запрос для записи в модуль памяти инвертированных
                ;18698                                                  ; данных
U 1B42, B218,15 ;18699          WRITE.MEM LS[T12]                        ; запись в текущую ячейку инвертированных данных
                ;18700            DEC LS[T14],                          ; уменьшение счетчика
U 1B43, FC1C,35 ;18701          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1B44, 09B4,61 ;18702          JMP.IF[NEQ] TO [CONT.T33.6.8]            ; перехода, если счетчик еще не нуль
U 1B45, 09BB,BC ;18703          JSR [SETUP.T33.COUNTER&CALL.MONITOR]     ; установка счетчика и вызов монитора для извещения, что
                ;18704                                                  ; тест продолжается
                ;18705    CONT.T33.6.8:
                ;18706           SUB WR[3] FROM LS[T9],                  ; переключение бита в адресе
U 1B46, 6B13,B5 ;18707          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1B47, 09B3,C0 ;18708          JMP.IF[N.CLR] TO [LOOP.T33.6.8]          ; если адрес не меньше, чем 0 повторение со следующим
                ;18709                                                  ; адресом
U 1B48, 3614,15 ;18710          MOV LS[T10] TO WR[0]                     ; иначе перехода к последнему адресу, с которого был
                ;18711                                                  ; запуск\
U 1B49, 2E44,15 ;18712          SUB WR[2] FROM WR[0]                     ; уменьшение адреса
U 1B4A, BE14,15 ;18713          MOV WR[0] TO LS[T10]                     ; запоминание адреса
U 1B4B, BE12,15 ;18714          MOV WR[0] TO LS[T9]                      ; адрес в T9
                ;18715            CMP LS[T11] WITH WR[0],               ; если адрес равен адресу в T11, конец выполнения
U 1B4C, 4E16,35 ;18716          DT(LONG)&SET.ALU.CC                      ; установка кода условий
U 1B4D, 89B3,C1 ;18717          JMP.IF[NEQ] TO [LOOP.T33.6.8]            ; повторение теста с новым адресом, если не конец
U 1B4E, FC82,15 ;18718          DEC LS[ERROR.NUMBER]                     ; увеличение номера ошибки
U 1B4F, A3C1,95 ;18719          ROL WR[3]                                ; следующий бит для переключения с максимальной
                ;18720                                                  ; скоростью
                ;18721            BIT LS[#4] WITH WR[3],                ; проверка, что бит 2 установлен
U 1B50, D945,B5 ;18722          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1B51, 89B2,09 ;18723          JMP.IF[BITS.CLR] TO [REPEAT.T33.3.7]     ; повторение теста с новым переключаемым битом, если бит
                ;18724                                                  ; 2 не был установлен
U 1B52, 5B00,14 ;18725          RETURN                                   ; иначе конец выполнения
                ;18726    T33.9:
U 1B53, 09BB,BC ;18727          JSR [CALL.MONITOR]                       ; вызов монитора для извещения, что тест продолжается
U 1B54, B645,15 ;18728          MOV LS[#4] TO WR[2]                      ; подготовка увеличения на 4
U 1B55, 3625,95 ;18729          MOV LS[MM.LASTADDR+1] TO WR[3]           ; рабочий регистр содержит последний адрес памяти +1 для
```

```
                ;18730                                          ; сравнения
U 1B56, 41C1,95 ;18731          SUB LS[#3(H)] FROM WR[3]        ; адрес для останова при восходящем проходе
U 1B57, 6512,15 ;18732          CLR LS[T9]                      ; очистка ячейки LS
U 1B58, FF12,15 ;18733          INC LS[T9]                      ; начало с адреса 1 модуля памяти (невыравненный адрес)
                ;18734  LOOP.T33.9:
U 1B59, B618,95 ;18735          MOV LS[T12] TO WR[1]            ; ожидаемые данные
U 1B5A, 1813,F5 ;18736          MEM.REQ[READ.P] ADRS[T9] DT[LONG]  ; запрос для чтения из модуля памяти
U 1B5B, 3022,15 ;18737          MOV MEM.DATA TO WR[0]           ; прием результата
U 1B5C, 0869,3C ;18738          JSR [CHECK.RESULT]              ; проверка наличия ошибок
U 1B5D, 89B5,94 ;18739          JMP [LOOP.T33.9]                ; цикл при ошибке, если есть разрешение
U 1B5E, 9912,75 ;18740          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в модуль памяти инвертированных
                ;18741                                          ; данных
U 1B5F, 321A,15 ;18742          WRITE.MEM LS[T13]               ; запись в эту ячейку инвертированных данных
U 1B60, EC13,15 ;18743          ADD WR[2] TO LS[T9]             ; увеличение на 4
                ;18744            CMP WR[3] WITH LS[T9],        ; проверка, что восходящий прохоа выполнен
U 1B61, 4F13,B5 ;18745          DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 1B62, 89B5,91 ;18746          JMP.IF[NEQ] TO [LOOP.T33.9]     ; если не старший адрес памяти, повторение для
                ;18747                                          ; следующего адреса, иначе переход к запуску по
                ;18748                                          ; нисходящим адресам
U 1B63, 09BB,DC ;18749          JSR [CALL.MONITOR]              ; вызов монитора для извещения, что тест продолжается
U 1B64, FF82,15 ;18750          INC LS[ERROR.NUMBER]            ; ошибка A
U 1B65, 3624,15 ;18751          MOV LS[MM.LASTADDR+1] TO WR[0]  ; запуск с верхнего адреса памяти
U 1B66, 41C0,15 ;18752          SUB LS[#3(H)] FROM WR[0]        ; уменьшение на 3 для получения истинного старшего
                ;18753                                          ; адреса
U 1B67, BE12,15 ;18754          MOV WR[0] TO LS[T9]             ; текущий адрес в T9
U 1B68, B641,95 ;18755          MOV LS[#1] TO WR[3]             ; адрес, по которому должен произойти останов
                ;18756  REPEAT.T33.A:
U 1B69, 6B13,15 ;18757          SUB WR[2] FROM LS[T9]           ; уменьшение на 4
                ;18758  LOOP.T33.A:
U 1B6A, 361A,95 ;18759          MOV LS[T13] TO WR[1]            ; ожидаемые данные
U 1B6B, 1813,F5 ;18760          MEM.REQ[READ.P] ADRS[T9] DT[LONG]  ; запрос для чтения из модуля памяти
U 1B6C, 3022,15 ;18761          MOV MEM.DATA TO WR[0]           ; выборка результата
U 1B6D, 0869,3C ;18762          JSR [CHECK.RESULT]              ; проверка наличия ошибок
U 1B6E, 89B6,A4 ;18763          JMP [LOOP.T33.A]                ; цикл при ошибке, если есть разрешение
U 1B6F, 9912,75 ;18764          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи в модуль памяти инвертированных
                ;18765                                          ; данных
U 1B70, B218,15 ;18766          WRITE.MEM LS[T12]               ; запись в текущую ячейку инвертированных данных
                ;18767            CMP LS[T9] WITH WR[3],        ; адрес, по которому должен произойти останов
U 1B71, CE13,B5 ;18768          DT(LONG)&SET.ALU.CC             ; и установка кодов условий
U 1B72, 89B6,91 ;18769          JMP.IF[NEQ] TO [REPEAT.T33.A]   ; если не самый младший адрес, повторение со следующим
                ;18770                                          ; адресом
U 1B73, 09BB,DC ;18771          JSR [CALL.MONITOR]              ; вызов монитора для извещения, что тест продолжается
U 1B74, 5B00,14 ;18772          RETURN                          ; конец выполнения
                ;18773  T33.B:
U 1B75, 3625,95 ;18774          MOV LS[MM.LASTADDR+1] TO WR[3]  ; рабочий регистр содержит последний адрес+1
U 1B76, C147,95 ;18775          SUB LS[#8] FROM WR[3]           ; вычитание 8 (десятичн.)
U 1B77, 4145,95 ;18776          SUB LS[#4] FROM WR[3]           ; вычитание 4 (десятичн.) для получения адреса останова
                ;18777                                          ; при восходящем проходе
U 1B78, B649,15 ;18778          MOV LS[#10] TO WR[2]            ; увеличение на 16 (десятичн.)
U 1B79, 3644,15 ;18779          MOV LS[#4] TO WR[0]             ; рабочий регистр содержит 4
U 1B7A, BE12,15 ;18780          MOV WR[0] TO LS[T9]             ; запуск с адреса 4 (для максимального пересечения
                ;18781                                          ; границ страницы)
                ;18782  LOOP.T33.B:
U 1B7B, B618,95 ;18783          MOV LS[T12] TO WR[1]            ; ожидаемые данные
U 1B7C, 1E12,75 ;18784          MEM.REQ[OCTA.READ.P] ADRS[T9] DT[LONG]  ; запрос для чтения из модуля памяти
```

```
U 1B7D, 3022,15 ;18785          MOV MEM.DATA TO WR[0]                    ; прием длинного слова 1 результата
U 1B7E, 38AE,15 ;18786          MOV MEM.DATA TO LS[T57]                   ; прием длинного слова 2 результата
U 1B7F, 38B0,15 ;18787          MOV MEM.DATA TO LS[T58]                   ; прием длинного слова 3 результата
U 1B80, B8B2,15 ;18788          MOV MEM.DATA TO LS[T59]                   ; прием длинного слова 4 результата
U 1B81, 0869,3C ;18789          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1B82, 89B7,B4 ;18790          JMP [LOOP.T33.B]                         ; цикл при ошибке, если есть разрешение
U 1B83, B618,95 ;18791          MOV LS[T12] TO WR[1]                     ; ожидаемые данные
U 1B84, B6AE,15 ;18792          MOV LS[T57] TO WR[0]                     ; полученные данные длинного слова 2
U 1B85, 0869,3C ;18793          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1B86, 89B7,B4 ;18794          JMP [LOOP.T33.B]                         ; цикл при ошибке, если есть разрешение
U 1B87, B618,95 ;18795          MOV LS[T12] TO WR[1]                     ; ожидаемые данные
U 1B88, B6B0,15 ;18796          MOV LS[T58] TO WR[0]                     ; полученные данные длинного слова 3
U 1B89, 0869,3C ;18797          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1B8A, 89B7,B4 ;18798          JMP [LOOP.T33.B]                         ; цикл при ошибке, если есть разрешение
U 1B8B, B618,95 ;18799          MOV LS[T12] TO WR[1]                     ; ожидаемые данные
U 1B8C, 36B2,15 ;18800          MOV LS[T59] TO WR[0]                     ; полученные данные длинного слова 4
U 1B8D, 0869,3C ;18801          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1B8E, 89B7,B4 ;18802          JMP [LOOP.T33.B]                         ; цикл при ошибке, если есть разрешение
U 1B8F, 9C12,75 ;18803          MEM.REQ[OCTA.WRITE.P] ADRS[T9] DT[LONG]  ; запрос для записи инвертированных данных
U 1B90, 321A,15 ;18804          WRITE.MEM LS[T13]                        ; запись длинного слова 1
U 1B91, 321A,15 ;18805          WRITE.MEM LS[T13]                        ; запись длинного слова 2
U 1B92, 321A,15 ;18806          WRITE.MEM LS[T13]                        ; запись длинного слова 3
U 1B93, 321A,15 ;18807          WRITE.MEM LS[T13]                        ; запись длинного слова 4
U 1B94, EC13,15 ;18808          ADD WR[2] TO LS[T9]                      ; увеличение на 16 (десятичн.)
                ;18809            CMP WR[3] WITH LS[T9],                 ; проверка, закончен ли восходящий проход
U 1B95, 4F13,B5 ;18810          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 1B96, 89B7,B1 ;18811          JMP.IF[NEQ] TO [LOOP.T33.B]              ; если не старший адрес памяти, повторение со следующим
                ;18812                                                   ; адресом, иначе переход к запуску по нисходящим адресам
U 1B97, 09BB,DC ;18813          JSR [CALL.MONITOR]                       ; вызов монитора для извещения, что тест продолжается
U 1B98, FF82,15 ;18814          INC LS[ERROR.NUMBER]                     ; ошибка С
U 1B99, 3624,15 ;18815          MOV LS[MM.LASTADDR+1] TO WR[0]           ; запуск со старшего адреса памяти
U 1B9A, C146,15 ;18816          SUB LS[#8] FROM WR[0]                    ; уменьшение на 8
U 1B9B, 4144,15 ;18817          SUB LS[#4] FROM WR[0]                    ; уменьшение на 4 для получения старшего адреса для
                ;18818                                                   ; обращения
U 1B9C, BE12,15 ;18819          MOV WR[0] TO LS[T9]                      ; загрузка адреса в Т9
U 1B9D, 3645,95 ;18820          MOV LS[#4] TO WR[3]                      ; адрес для останова на нисходящем проходе
                ;18821  REPEAT.T33.C:
U 1B9E, 6B13,15 ;18822          SUB WR[2] FROM LS[T9]                    ; уменьшение на 16 (десятичн.)
                ;18823  LOOP.T33.C:
U 1B9F, 361A,95 ;18824          MOV LS[T13] TO WR[1]                     ; ожидаемые данные
U 1BA0, 1E12,75 ;18825          MEM.REQ[OCTA.READ.P] ADRS[T9] DT[LONG]   ; запрос для чтения из модуля памяти
U 1BA1, 3022,15 ;18826          MOV MEM.DATA TO WR[0]                    ; прием длинного слова 1 результата
U 1BA2, 38AE,15 ;18827          MOV MEM.DATA TO LS[T57]                   ; прием длинного слова 2 результата
U 1BA3, 38B0,15 ;18828          MOV MEM.DATA TO LS[T58]                   ; прием длинного слова 3 результата
U 1BA4, B8B2,15 ;18829          MOV MEM.DATA TO LS[T59]                   ; прием длинного слова 4 результата
U 1BA5, 0869,3C ;18830          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1BA6, 89B9,F4 ;18831          JMP [LOOP.T33.C]                         ; цикл при ошибке, если есть разрешение
U 1BA7, 361A,95 ;18832          MOV LS[T13] TO WR[1]                     ; ожидаемые данные
U 1BA8, B6AE,15 ;18833          MOV LS[T57] TO WR[0]                     ; полученные данные длинного слова 2
U 1BA9, 0869,3C ;18834          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1BAA, 89B9,F4 ;18835          JMP [LOOP.T33.C]                         ; цикл при ошибке, если есть разрешение
U 1BAB, 361A,95 ;18836          MOV LS[T13] TO WR[1]                     ; ожидаемые данные
U 1BAC, B6B0,15 ;18837          MOV LS[T58] TO WR[0]                     ; полученные данные длинного слова
U 1BAD, 0869,3C ;18838          JSR [CHECK.RESULT]                       ; проверка наличия ошибок
U 1BAE, 89B9,F4 ;18839          JMP [LOOP.T33.C]                         ; цикл при ошибке, если есть разрешение
```

```
U 1BAF, 361A,95 ;18840          MOV LS[T13] TO WR[1]                    ; ожидаемые данные
U 1BB0, 36B2,15 ;18841          MOV LS[T59] TO WR[0]                    ; полученные данные длинного слова 4
U 1BB1, 0869,3C ;18842          JSR [CHECK.RESULT]                      ; проверка наличия ошибок
U 1BB2, 89B9,F4 ;18843          JMP [LOOP.T33.C]                        ; цикл при ошибке, если есть разрешение
U 1BB3, 9C12,75 ;18844          MEM.REQ[OCTA.WRITE.P] ADRS[T9] DT[LONG];запрос для записи инвертированных данных
U 1BB4, B218,15 ;18845          WRITE.MEM LS[T12]                       ; запись длинного слова 1
U 1BB5, B218,15 ;18846          WRITE.MEM LS[T12]                       ; запись длинного слова 2
U 1BB6, B218,15 ;18847          WRITE.MEM LS[T12]                       ; запись длинного слова 3
U 1BB7, B218,15 ;18848          WRITE.MEM LS[T12]                       ; запись длинного слова 4
                ;18849            CMP LS[T9] WITH WR[3],                ; адрес, по которому должен произойти останов
U 1BB8, CE13,B5 ;18850          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 1BB9, 09B9,E1 ;18851          JMP.IF[NEQ] TO [REPEAT.T33.C]           ; если не самый младший адрес, повторение со следующим
                ;18852                                                  ; адресом
U 1BBA, 5B00,14 ;18853          RETURN                                  ; конец выполнения
                ;18854    ;
                ;18855    ; Эта программа вызывает монитор для предотвращения таймаута. Если вызвана из кон-
                ;18856    ; ца прохода, также устанавливает счетчик для теста по байтам. Счетчик позволяет
                ;18857    ; вызывать монитор после обращений к каждому из 1 мбайт памяти.
                ;18858    ;
                ;18859    SETUP.T33.COUNTER&CALL.MONITOR:
U 1BBB, B668,15 ;18860          MOV LS[#100000] TO WR[0]                ; установка счетчика на 1 миллион обращений. Монитор
                ;18861                                                  ; будет вызываться после обращений к каждому 1 мегабайту
                ;18862                                                  ; памяти
U 1BBC, 3E1C,15 ;18863          MOV WR[0] TO LS[T14]                    ; запоминание в LS
                ;18864    CALL.MONITOR:
U 1BBD, E580,15 ;18865          CLR LS[CONTROL.STATUS]                  ; очистка слова управления и состояния
U 1BBE, 10E0,15 ;18866          MISC [SET.CP.ATTN]                      ; выдача для консольного процессора CPU ATTN с целью
                ;18867                                                  ; извещения, что тест продолжается
                ;18868    WAIT.T33.CALL:
U 1BBF, 09BB,F4 ;18869          JMP [WAIT.T33.CALL]                     ;
U 1BC0, 5B00,14 ;18870          RETURN                                  ; конец выполнения
                ;18871    ;
                ;18872    ; Эта программа записывает фон из данных в T12 во всей доступной памяти
                ;18873    BACK.T33:
U 1BC1, 3624,15 ;18874          MOV LS[MM.LASTADDR+1] TO WR[0]          ; рабочий регистр содержит последний адрес+1 для
                ;18875                                                  ; сравнения
U 1BC2, B645,15 ;18876          MOV LS[#4] TO WR[2]                     ; WR2 содержит значение инкремента
U 1BC3, 6512,15 ;18877          CLR LS[T9]                              ; начало с адреса 0
                ;18878    BACK.T33.REP:
U 1BC4, 9912,75 ;18879          MEM.REQ[WRITE.P] ADRS[T9] DT[LONG]      ; запрос для записи фона в модуль памяти
U 1BC5, B218,15 ;18880          WRITE.MEM LS[T12]                       ; запись данных в модуль памяти
U 1BC6, EC13,15 ;18881          ADD WR[2] TO LS[T9]                     ; увеличение адреса на 4
                ;18882            CMP WR[0] WITH LS[T9],                ; проверка, что вся память записана
U 1BC7, 4F12,35 ;18883          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 1BC8, 09BC,41 ;18884          JMP.IF[NEQ] TO [BACK.T33.REP]           ; повторение, пока не будет записана вся память
U 1BC9, 5B00,14 ;18885          RETURN                                  ; запись фона выполнена
                ;18886    ;
                ;18887    ; Эта программа увеличивает номер ошибки на 2
                ;18888    ;
                ;18889    T33.INC.ERR:
U 1BCA, FF82,15 ;18890          INC LS[ERROR.NUMBER]                    ; увеличение номера ошибки
U 1BCB, FF82,15 ;18891          INC LS[ERROR.NUMBER]                    ; еще одно увеличение
U 1BCC, 5B00,14 ;18892          RETURN                                  ; конец выполнения
                ;18893    END.T33:
```

```
;18894  .PAGE "МИКРОДИАГНОСТИКА С ПРИМЕНЕНИЕМ ТЕСТЕРА ШИНЫ (UBE)"
;18895  ;
;18896  ;    Микродиагностика с UBE предназначена для проверки аппаратуры в МСТ, связанной
;18897  ;    с общей шиной, которая не могла быть проверена в предыдущих тестах без уст-
;18898  ;    ройства общей шины. Эта микродиагностика не предназначена для использования
;18899  ;    в качестве средства проверки логических схем общей шины.
;18900  ;      Первый тест "ПРОЩУПЫВАЕТ" наличие UBE путем записи по адресу общей шины,
;18901  ;    который должен ответить, если UBE установлен и конфигурирован так, как опи-
;18902  ;    сывается в последующем разделе. Затем программа диагностики проверяет бит
;18903  ;    NXM в CSR1. Если бит NXM установлен, это означает, что при записи в устройс-
;18904  ;    тво общей шины не был получен ответ, и оставшиеся тесты с UBE пропускают-
;18905  ;    ся. Если бит NXM не установлен, что означает, что ответ при записи был полу-
;18906  ;    чен, тогда тесты с UBE начинаются. В любом случае для информирования опера-
;18907  ;    тора о результатах теста "ПРОЩУПЫВАНИЯ" печатается сообщение.
;18908  ;      Если UBE не обнаружен, но он установлен и конфигурирован правильно, опера-
;18909  ;    тор может вызвать зацикливание теста "ПРОЩУПЫВАНИЯ" и, таким образом, непо-
;18910  ;    ладка может быть выявлена. Это осуществляется путем установки признака SER
;18911  ;    (SE SE) и выполнением теста. Эта процедура подробно описывается в описании
;18912  ;    теста "ПРОЩУПЫВАНИЕ".
;18913  ;      Последний тест UBE, в действительности, вобще не является тестом, а только
;18914  ;    средством отладки. Он не выполняется, если его выполнение не задано конкрет-
;18915  ;    но посредством команды DI TE 43. Тест только сообщает состояние регистров
;18916  ;    UBE. Более подробно смотри описание теста.
;18917  ;
;18918  ; ОПИСАНИЕ ПОДГОТОВКИ UBE И РЕГИСТРОВ
;18919  ;
;18920  ;      Эти тесты предназначены для выполнения, по крайней мере, с одним UBE, уста-
;18921  ;    новленным в монтажной плате и отвечающим на адреса от FFFFF000 (H) до FFFFF0
;18922  ;    0E (H) (восмеричные от 770000 до 770016). Может устанавливаться более одного
;18923  ;    UBE, но этот тест будет использовать только тот, который отвечает на выше ука-
;18924  ;    занные адреса. Вектор прерывания на этом UBE должен быть установлен на реак-
;18925  ;    цию по адресу 510 (восмеричн.).
;18926  ;      Этим требованиям отвечает такая установка переключателей:
;18927  ;        ячейка     S1     S2     S3     S4     S5     S6     S7     S8
;18928  ;         D125     вкл    вкл    вкл    вкл    вкл    вкл    вкл    вкл
;18929  ;         D88      вкл    вкл     X     вкл    выкл   выкл   вкл    выкл
;18930  ;
;18931  ;    X=безразлично. D125 является переключателем адреса, D88 является переключа-
;18932  ;    телем вектора.
;18933  ;
;18934  ;      Регистры UBE загружаются функцией WRITE.F к соответсвующему адресу. Адрес
;18935  ;    интерпретируется контроллером памяти как физический адрес общей шины. Далее
;18936  ;    следует краткое описание каждого регистра с соответствующим 16-ричным адре-
;18937  ;    сом. Более подробную информацию следует искать в материалах по UBE.
;18938  ;
;18939  ;       Регистр данных (BEDB) - 16-ричный адрес FFFFF000
;18940  ;
;18941  ;      Этот регистр содержит 16 битов данных, которые подлежат записи в модуль па-
;18942  ;    мяти во время операции UBE DATO или данные, считанные из модуля памяти во
;18943  ;    время операции UBE DATI.
;18944  ;
;18945  ;       Регистр счетчика циклов (BECC) - 16-ричный адрес FFFFF002
;18946  ;
;18947  ;    Этот регистр загружается дополнением до 2 числа пересылок, которые надлежит
;18948  ;    выполнить UBE. Для большинства этих тестов этот регистр загружается значени-
```

```
;18949  ;   ем -1 для выполнения одной пересылки.
;18950  ;
;18951  ;       Регистр адреса (BEBA) - 16-ричный адрес FFFFF004
;18952  ;
;18953  ;      Этот регистр загружается адресом модуля памяти, к которому происходит обра-
;18954  ;   щение при записи или чтении из UBE. Он увеличивается на 2 после каждой пере-
;18955  ;   сылки (увеличивается на единицу в цикле DATOB).
;18956  ;
;18957  ;       Регистр управления 1 (BECR1) - 16-ричный адрес FFFFF006
;18958  ;
;18959  ;      Этот регистр содержит управляющую информацию UBE. Он указывает UBE, какая
;18960  ;   функция должна выполняться. Бит 15 является битом ошибки, который устанавли-
;18961  ;   вается, если установлены какие-либо биты ошибок в регистре управления 2. Бит
;18962  ;   0 является битом GO (старт), который устанавливается в единицу при записи в
;18963  ;   регистр с целью запуска UBE. Ниже приводится краткий обзор функций битов ре-
;18964  ;   гистра. Более подробное описание следует искать в материалах по UBE.
;18965  ;
;18966  ;    15    14     13     12     11   10    9    8    7    6    5    4    3   2   1   0
;18967  ;   ---------------------------------------------------------------------------------
;18968  ;   :ERR  :DATO  : INR  :DATA  :FUN :FUN :    :    :    :INT :    : BR :BR :BR :BR :   :
;18969  ;   :FLAG : OR   :DATIP : OR   : B  : A  :C01 :C00 :RDY : ON :NPR : 7  : 6 : 5 : 4 :GO :
;18970  ;   :     :DATOB : ROL  :CYCLE :    :    :    :    :    :DONE:    :    :   :   :   :   :
;18971  ;   ---------------------------------------------------------------------------------
;18972  ;   15 - признак ошибки
;18973  ;   14 - DATO или DATOB
;18974  ;   13 - запрет DATIP
;18975  ;   12 - данные или зацикливание
;18976  ;    6 - прерывание по окончанию
;18977  ;
;18978  ;   C01 - C00: 0=DATI    1=DATIP   2=DATO    3=DATOB
;18979  ;   FUNB - FUNA:  0 = выполнение прерывания
;18980  ;                1 = выполнение пересылки слова
;18981  ;                2 = выполнение пересылки длинного слова
;18982  ;                3 = освобождение шины
;18983  ;
;18984  ;       Очистка адреса ошибки (CLEAR.ERR.ADDR) - 16-ричный адрес FFFFF00A
;18985  ;
;18986  ;      В действительности это не регистр. Если выполняется DATO по этому адресу,
;18987  ;   очищаются все биты ошибок, установленные в регистре управления 2.
;18988  ;
;18989  ;       Регистр управления 2 (BECR2) - 16-ричный адрес FFFFF010
;18990  ;
;18991  ;      Этот регистр содержит некоторые биты управления и индикаторы ошибок. Бит 13
;18992  ;   и бит 2 являются единственными управляющими битами, используемыми этими тес-
;18993  ;   тами. Бит 13 является битом переполнения счетчика циклов и устанавливается,
;18994  ;   когда выполнено требуемое число циклов. Бит 2 представляет собой бит запрета
;18995  ;   наращивания адреса и битов регистра счетчика циклов и предотвращает в UBE
;18996  ;   увеличение регистров счетчика циклов и адреса после каждой пересылки. Этот
;18997  ;   бит используется тестом, который проверяет функцию блокировки, заставляющую
;18998  ;   UBE выдавать непрерывные запросы прямого доступа NPR, до тех пор, пока не
;18999  ;   будет установлен очисткой этого бита. Другие управляющие биты в этих тестах
;19000  ;   всегда очищены. Ниже представлена диаграмма битов индикации ошибок (биты от
;19001  ;   11 до 5), которые выдаются в сообщении об ошибке, если происходит ошибка UBE.
;19002  ;
;19003  ;     11     10     9    8     7     6    5
```

```
;19004  ;    ------------------------------------------
;19005  ;    : NO   :      : W :  NO  : NO,  : BUS : W :
;19006  ;    : INTR : OTOG : A : SSYN : NO   : LAT : G :
;19007  ;    : SSYN :      : L :      : SACK :     : B :
;19008  ;    ------------------------------------------
;19009  ;
;19010  ;
;19011  ;    11 = не получен SSYN после выдачи в центр.процессор сигнала прерывания (INTR)
;19012  ;    10 = нет разрешения или более одного разрешения (не одно разрешение)
;19013  ;     9 = ошибка в адресных линиях при сравнении выданного адреса с полученным адресом
;19014  ;     8 = не получен SSYN после MSYN
;19015  ;     7 = центр.процессор не зафиксировал таймаута, когда подтверждение не было
;19016  ;         передано после получения разрешения
;19017  ;     6 = ошибка задержки шины
;19018  ;     5 = неправильно возвращено разрешение
```

```
;19019  .PAGE "ТЕСТЫ С UBE"
;19020  .TOC  "ТЕСТ 34 - проверка наличия UBE"
;19021  ;
;19022  ; ОПИСАНИЕ ТЕСТА:
;19023  ;
;19024  ;       Этот тест выполняет проверку, позволяющую удостовериться, что в системе аппа-
;19025  ;   ратных средств содержится UBE. Если UBE найден, печатается сообщение (UBE PRE-
;19026  ;   SENT - UBE присутствует) и выполнение переходит к следующему тесту. Если UBE не
;19027  ;   обнаружен, печатается сообщения (UBE NOT PRESENT - UBE отсутствует) и тесты с UBE
;19028  ;   пропускаются.
;19029  ;       Этот тест "ПРОЩУПЫВАЕТ" наличие UBE путем выполнения функции READ.P из адре-
;19030  ;   са FFF000 (регистр данных UBE) и проверки, установлен ли бит NXM в CSR1 контро-
;19031  ;   ллера памяти. Установленный бит NXM указывает, что произошел таймаут, означа-
;19032  ;   ющий, что UBE не обнаружен. Очищенный бит NXM указывает, что UBE обнаружен и
;19033  ;   отвечает.
;19034  ;
;19035  ; ПРЕДПОЛОЖЕНИЯ:
;19036  ;
;19037  ;   Предполагается, что все предыдущие тесты микродиагностики прошли успешно. Если
;19038  ;   UBE присутствует, то принимается, что он исправен.
;19039  ;
;19040  ;   ПРИМЕЧАНИЕ: признак SP (специальный бит) должен быть очищенным, если не желатель-
;19041  ;   но зацикливание на этом тесте (см.наладку).
;19042  ;
;19043  ; ШАГИ ТЕСТА:
;19044  ;
;19045  ;   1)Выдача DCLO для инициализации UBE, если он присутствует.
;19046  ;   2)Выдача инструкции MEM.REQCREAD.PJ к буферу данных UBE. Завершение чикла па-
;19047  ;     мяти выдачей инструкции MOV MEM.DATA TO WR0.
;19048  ;   3)Проверка признака SER. Если установлен, повторение шага 1 (позволяет зацик-
;19049  ;     ливание.
;19050  ;   4)Чтение CSR1 и проверка бита NXM (несуществующая память). Если очищен, сообщение
;19051  ;     UBE PRESENT. Если установлен, сообщение UBE NOT PRESENT (отсутствует).
;19052  ;   5)Если UBE отсутсвует, перехоа к концу сегмента. Иначе перехоа к следующему
;19053  ;     тесту.
;19054  ;
;19055  ; ОШИБКИ:
;19056  ;
;19057  ;   ошибка 1 - результат "ПРОЩУПЫВАНИЯ" UBE отличается от того, какой указан из
;19058  ;              APT (эта ошибка возможна только при выполнении под управлением из APT)
;19059  ;
;19060  ; НАЛАДКА:
;19061  ;
;19062  ;       Если UBE присутствует, но не обнаружен этим тестом, чтение регистра данных UBE
;19063  ;   можно зацикливать путем печати SE SP (установка специального управляющего би-
;19064  ;   та), SE LO (установка зацикливания при ошибке для предовращения таймаута) и пов-
;19065  ;   торным запуском этого теста. Следует помнить, что необходимо выполнить CL SP,
;19066  ;   если желательно продолжить следующие тесты. Для выхода из этого цикла необхо-
;19067  ;   димо печатать YC/C или YC/P. При выходе из этого цикла возможно "ЗАВИСАНИЕ"
;19068  ;   МСТ. Если необходимо освободить МСТ, следует печатать I (INIT).
;19069  ;     При зацикливании необходимо проверить SSYN, поступающий в приемник и про-
;19070  ;   ходящий через 2 буфера в качестве L SSYN H. Если сигнал правильный, следует
;19071  ;   проверить сигнал, поступающий в мультиплексор ветвления BEN1. Если SSYN здесь
;19072  ;   отсутствует, можно подозревать неисправность печатных проводников к контак-
;19073  ;   там, связанным с адресными линиями общей шины, выходом сигнала MSYN или входом
```

```
                ;19074  ;   SSYN.
                ;19075  ;--
                ;19076
                ;19077   T.34:
U 1BCD, B65E,15 ;19078           MOV LSCBEGIN.TESTC TO WRC0C                    ; установка в WR0 бита 15 для слова управления и
                ;19079                                                          ; состояния
U 1BCE, 3E80,15 ;19080           MOV WRC0C TO LSCCONTROL.STATUSC                ; установка бита 15 в слове управления и состояния. Бит
                ;19081                                                          ; 15 указывает для конс.процессора начало теста
U 1BCF, 10E0,15 ;19082           MISC CSET.CP.ATTNC                             ; выдача для конс.процессора CPU ATTN
                ;19083   WAIT.T34.0:
U 1BD0, 09BD,04 ;19084           JMP CWAIT.T34.0C                               ; цикл для ожидания ответа конс.процессора
U 1BD1, 0A1B,8C ;19085           JSR CISSUE.DCLOC                               ; выдача DCLO для инициализации UBE
                ;19086   LOOP.T34.1:
U 1BD2, 98B5,B5 ;19087           MEM.REQCREAD.PC ADRSCBEDBC DTCWORDC            ; запрос для чтения из буфера данных UBE
U 1BD3, 3022,15 ;19088           MOV MEM.DATA TO WRC0C                          ; завершение обращения к памяти
U 1BD4, 3690,15 ;19089           MOV LSCERROR.CONTROLC TO WRC0C                 ; выборка слова управления ошибками
                ;19090             BIT LSCSPECIALC WITH WRC0C,                  ; проверка, является ли активным SE SP
U 1BD5, D94E,35 ;19091           DT(LONG)&SET.ALU.CC                            ; и установка кодов условий
U 1BD6, 89BD,21 ;19092           JMP.IFCBIT.SETC TO CLOOP.T34.1C                ; зацикливание, если SE SP активный
U 1BD7, 9D45,75 ;19093           MEM.REQCREAD.CSRC ADRSCCSR1C DTCLONGC          ; запрос для чтения из CSR
U 1BD8, 3022,15 ;19094           MOV MEM.DATA TO WRC0C                          ; выборка содержимого CSR 1
U 1BD9, E50E,15 ;19095           CLR LSCUBE.FOUNDC                              ; очистка LS 7
                ;19096             BIT LSCNXMC WITH WRC0C,                      ; проверка, что бит NXM установлен
U 1BDA, D960,35 ;19097           DT(LONG)&SET.ALU.CC                            ; и установка кодов условий
U 1BDB, 2F80,15 ;19098           CLR WRC0C                                      ; подготовка для сообщения, что UBE отсутсвует
U 1BDC, 09BE,01 ;19099           JMP.IFCBIT.SETC TO CCHECK.APTC                 ; если бит NXM был установлен, не устанавливается бит в
                ;19100                                                          ; LS7
U 1BDD, 7F0E,15 ;19101           INC LSCUBE.FOUNDC                              ; иначе установка в LS 7 бита 0
U 1BDE, 3650,15 ;19102           MOV LSCBIT8C TO WRC0C                          ; установка в рабочем регистре бита 8
U 1BDF, 6D0E,15 ;19103           BIS WRC0C TO LSCUBE.FOUNDC                     ; установка в LS бита 8, если UBE обнаружен (не
                ;19104                                                          ; установлен NXM)
                ;19105   CHECK.APT:
U 1BE0, 3640,95 ;19106           MOV LSC#1C TO WRC1C                            ; занесение 1 в рабочий регистр
U 1BE1, 3E82,95 ;19107           MOV WRC1C TO LSCERROR.NUMBERC                  ; ошибка 1 (только при APT)
U 1BE2, B690,95 ;19108           MOV LSCERROR.CONTROLC TO WRC1C                 ; выборка слова управления ошибками
                ;19109             BIT LSCAPT.PRESENTC WITH WRC1C,              ; проверка, что работа под управлением APT
U 1BE3, D94A,B5 ;19110           DT(LONG)&SET.ALU.CC                            ; и установка кодов условий
U 1BE4, 09BE,B9 ;19111           JMP.IFCBITS.CLRC TO CT34.PRINTC                ; если не APT, переход к печати результатов
                ;19112                                                          ; "ПРОЩУПЫВАНИЯ"
U 1BE5, 3692,95 ;19113           MOV LSCAPT.PARAMC TO WRC1C                     ; иначе выборка параметров APT
U 1BE6, DF29,15 ;19114           MCOM LSC#FFFFC TO WRC2C                        ; занесение FFFF0000 в WR2
U 1BE7, 2F44,95 ;19115           BIC WRC2C TO WRC1C                             ; очистка старшего слова в параметрах APT
U 1BE8, C526,95 ;19116           BIC LSC#FFC TO WRC1C                           ; очистка младшего байта в параметрах APT
U 1BE9, 0869,3C ;19117           JSR CCHECK.RESULTC                             ; сообщение об ошибке, если конфигурация аппаратуры в APT
                ;19118                                                          ; отличается от результата "ПРОЩУПЫВАНИЯ"
U 1BEA, 09BE,F4 ;19119           JMP CEND.T34C                                  ; зацикливание на ошибке, конец теста
                ;19120   T34.PRINT:
U 1BEB, 364C,95 ;19121           MOV LSCPRINT.UBEC TO WRC1C                     ; установка в рабочем регистре бита 6
U 1BEC, BE80,95 ;19122           MOV WRC1C TO LSCCONTROL.STATUSC                ; установка бита 6 в слове управления
U 1BED, 10E0,15 ;19123           MISC CSET.CP.ATTNC                             ; выдача для конс.процессора CPU ATTN
                ;19124   WAIT.T34.1:
U 1BEE, 89BE,E4 ;19125           JMP CWAIT.T34.1C                               ; цикл для ожидания ответа конс.процессора
                ;19126   END.T34:
                ;19127           MOV LSCUBE.FOUNDC TO WRC0C,                    ; выборка результатов из LS7
U 1BEF, 360E,35 ;19128           DT(LONG)&SET.ALU.CC                            ; и установка кодов условий
```

```
U 1BF0, 8A17,99 ;19129          JMP.IF[BITS.CLR] TO [END.SECTION]      ; конец, если нет UBE
                ;19130   2179:
                ;19131   END.SECTION:
U 2179, 365C,15 ;19132          MOV LS[EOS] TO WR[0]                   ; установка в WR0 конца сегмента (бит 14)
U 217A, 3E80,15 ;19133          MOV WR[0] TO LS[CONTROL.STATUS]        ; установка слова управления и состояния для
                ;19134                                                 ; информирования о конце сегмента
U 217B, 10E0,15 ;19135          MISC [SET.CP.ATTN]                     ; выдача для конс.процессора CPU ATTN
                ;19136   WAIT.EOS:
U 217C, 0A17,C4 ;19137          JMP [WAIT.EOS]                         ; цикл для ожидания ответа конс.процессора
                ;19138    ;
```

```
                    #19139    .PAGE "ПОДПРОГРАММЫ МСТ"
                    #19140    ;
                    #19141    ;       Общие подпрограммы для тестов МСТ
                    #19142    ;
                    #19143    WRITE.CSR1.MME:
                    #19144            MOV LS[MME] TO WR[0],                    ; установка бита MME
U 217D, 3676,1E     #19145            SKIP                                     ; и пропуск следующей инструкции
                    #19146    CLEAR.CSR1:
U 217E, 2F80,15     #19147            CLR WR[0]                                ; занесение 0 в рабочий регистр
                    #19148    WRITE.CSR1:
U 217F, BE1E,15     #19149            MOV WR[0] TO LS[T15]                     ; загрузка кода данных в промежуточную ячейку
U 2180, 1D45,F5     #19150            MEM.REQ[WRITE.CSR] ADRS[CSR1] DT[LONG]   ; начало записи в CSR1
                    #19151              WRITE.MEM LS[T15],                     ; запись кода данных из ячейки LS F(H)
U 2181, 321E,14     #19152            RETURN                                   ; возврат к основной программе
                    #19153    NEXT.TB.ADDRESS:
U 2182, 3612,15     #19154            MOV LS[T9] TO WR[0]                      ; выборка старого адреса
U 2183, FF88,15     #19155            INC LS[ADDRESS.DATA]                     ; увеличение адреса, который будет распечатан
                    #19156              ADD LS[BIT9] TO WR[0],                 ; увеличение старого адреса
U 2184, C052,55     #19157            DT(SIZE)&SET.ALU.CC                      ; и установка кодов условий
U 2185, 8A18,A0     #19158            JMP.IF[N.CLR] TO [RET.NEXT]              ; повторение, если бит 15 все еще очищен
U 2186, 455E,15     #19159            BIC LS[BIT15] TO WR[0]                   ; иначе очистка бита 15
                    #19160              XOR LS[BIT31] TO WR[0],                ; переключение бита 31
U 2187, 437E,35     #19161            DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 2188, DB00,08     #19162            SKIP.IF[N.SET]                           ; пропуск следующей инструкции, если бит 31 установлен
                    #19163                                                     ; иначе конец выполнения
U 2189, DB00,16     #19164            RETURN+1                                 ; переход к следующей части
                    #19165    RET.NEXT:
U 218A, BE12,15     #19166            MOV WR[0] TO LS[T9]                      ; запоминание нового адреса в LS 9
U 218B, 5B00,14     #19167            RETURN                                   ; повторение теста с новым адресом
                    #19168    NEXT.UB.ADDRESS:
U 218C, 3612,15     #19169            MOV LS[T9] TO WR[0]                      ; выборка старого адреса
U 218D, FF88,15     #19170            INC LS[ADDRESS.DATA]                     ; увеличение адреса, который будет распечатан
U 218E, 4052,15     #19171            ADD LS[BIT9] TO WR[0]                    ; увеличение старого адреса
                    #19172              BIT LS[BIT18] WITH WR[0],              ; проверка, что бит 18 установлен
U 218F, 5964,35     #19173            DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 2190, 5B00,09     #19174            SKIP.IF[BITS.CLR]                        ; повторение, если бит 18 все еще очищен, иначе конец
                    #19175                                                     ; выполнения
U 2191, DB00,16     #19176            RETURN+1                                 ; переход к следующей части
U 2192, BE12,15     #19177            MOV WR[0] TO LS[T9]                      ; запоминание нового адреса в LS 9
U 2193, 5B00,14     #19178            RETURN                                   ; повторение теста при новом адресе
                    #19179    ;
                    #19180    ;       Эта программа "прощупывает" и инициализирует основную память и печатает
                    #19181    ;       число обнаруженных блоков по 256K
                    #19182    ;
                    #19183    SIZE.MEMORY:
U 2194, 6524,15     #19184            CLR LS[MM.LASTADDR+1]                    ; очистка ячейки LS (ячейка 12(H)) для хранения адреса
U 2195, B645,15     #19185            MOV LS[#4] TO WR[2]                      ; значение инкремента для адреса памяти
U 2196, B640,15     #19186            MOV LS[#1] TO WR[0]                      ; загрузка 1 в рабочий регистр
                    #19187    REPEAT.SIZE.1:
U 2197, 9924,75     #19188            MEM.REQ[WRITE.P] ADRS[MM.LASTADDR+1] DT[LONG] ; запрос для записи нулей во все ячейки памяти
U 2198, B29C,15     #19189            WRITE.MEM LS[ZERO]                       ; запись в память всех нулей
U 2199, 5B00,1D     #19190            SKIP.IF[MEM.REF.OK]                      ; пропуск следующей инструкции, если нет ошибки системы
                    #19191                                                     ; памяти
U 219A, 0A19,D4     #19192            JMP [PRINT.SIZE]                         ; переход к печати вычисленного размера памяти
U 219B, EC25,15     #19193            ADD WR[2] TO LS[MM.LASTADDR+1]           ; увеличение адреса памяти на 4
```

```
U 219C, 0A19,74 #19194          JMP [REPEAT.SIZE.1]                     ; повторение до получения ошибки системы памяти
                #19195  PRINT.SIZE:
U 219D, B625,15 #19196          MOV LS[MM.LASTADDR+1] TO WR[2]           ; занесение в рабочий регистр последнего адреса+1
U 219E, 3648,15 #19197          MOV LS[#10] TO WR[0]                    ; загрузка в WR0 16(десятичн.)
U 219F, C042,15 #19198          ADD LS[#2] TO WR[0]                     ; WR0 содержит 18(десятичн.)
                #19199  SIZE.SHIFT.LOOP:
U 21A0, A341,15 #19200          ROR WR[2]                               ; сдвиг вправо значения последнего адреса+1
                #19201            DEC WR[0],                            ; уменьшение счетчика
U 21A1, A100,35 #19202          DT(LONG)&SET.ALU.CC                     ; и установка кодов условий
U 21A2, 8A1A,01 #19203          JMP.IF[NEQ] TO [SIZE.SHIFT.LOOP]        ; повторение, пока не будет выполнен сдвиг на 18
                #19204                                                  ; позиций
U 21A3, BE0F,15 #19205          MOV WR[2] TO LS[MEMORY.SIZE]             ; занесение размера в блоках по 256K в LS 7 для печати
U 21A4, 3656,15 #19206          MOV LS[PRINT.MEM.SIZE] TO WR[0]          ; установка в рабочем регистре бита 11
U 21A5, 3E80,15 #19207          MOV WR[0] TO LS[CONTROL.STATUS]          ; бит 11 информирует консоль, что будет печать размера
                #19208                                                  ; памяти
U 21A6, 10E0,15 #19209          MISC [SET.CP.ATTN]                      ; выдача для конс.процессора CPU ATTN
                #19210  WAIT.SIZE.1:
U 21A7, 0A1A,74 #19211          JMP [WAIT.SIZE.1]                       ; цикл для ожидания ответа конс.процессора
                #19212            MOV LS[MM.LASTADDR+1] TO WR[2],       ; запоминание размера в WR2
U 21A8, 3625,14 #19213          RETURN                                  ; конец выполнения
                #19214   ;
                #19215   ;       Эта программа выполняет подготовку в начале каждого теста. SETUP.1 не
                #19216   ;       очищает CSR1, в то время, как SETUP эту очистку выполняет.
                #19217   ;
                #19218  SETUP:
U 21A9, 0A17,EC #19219          JSR [CLEAR.CSR1]                        ; очистка CSR1 МСТ
                #19220  SETUP.1:
U 21AA, 65A0,15 #19221          CLR LS[PREVIOUS.ERROR]                  ; очистка в LS номера предыдущей ошибки
U 21AB, E58A,15 #19222          CLR LS[ERROR.MASK]                      ; установка маски ошибок на проверку всех битов
U 21AC, 3022,15 #19223          MOV MEM.DATA TO WR[0]                   ; используется для освобождения памяти, если она
                #19224                                                  ; находится в состоянии ожидания CPU DR
U 21AD, 3022,15 #19225          MOV MEM.DATA TO WR[0]                   ; требуется 4 этих инструкции, если было "ЗАВИСАНИЕ" при
                #19226                                                  ; работе с восьмикратными данными
U 21AE, 3022,15 #19227          MOV MEM.DATA TO WR[0]                   ;
U 21AF, 3022,15 #19228          MOV MEM.DATA TO WR[0]                   ;
U 21B0, B640,15 #19229          MOV LS[#1] TO WR[0]                     ; загрузка 1 в рабочий регистр
U 21B1, BE82,15 #19230          MOV WR[0] TO LS[ERROR.NUMBER]           ; установка в LS номера ошибки
U 21B2, 3644,15 #19231          MOV LS[MCT] TO WR[0]                    ; установка бита 2
                #19232            MOV WR[0] TO LS[MODULE.NUM],          ; запоминание в LS кода модуля для индикации платы МСТ
U 21B3, BE8C,14 #19233          RETURN                                  ; конец подготовки
                #19234   ;
                #19235   ;       Эта программа выполняет задержку на 4 цикла центрального процессора
                #19236   ;
                #19237  NOP.DELAY:
U 21B4, DB00,15 #19238          NOP                                     ; задержка на 1 цикл
U 21B5, DB00,15 #19239          NOP                                     ; еще один цикл
U 21B6, DB00,15 #19240          NOP                                     ; еще один цикл
U 21B7, 5B00,14 #19241          RETURN                                  ; последний цикл задержки
                #19242   ;
                #19243   ;       Общие подпрограммы для тестов UBE
                #19244   ;
                #19245  ISSUE.DCLO:
U 21B8, 3680,95 #19246          MOV LS[CONTROL.STATUS] TO WR[1]          ; запоминание старого слова управления и состояния
U 21B9, 3660,15 #19247          MOV LS[INTERUPT.EN] TO WR[0]             ; установка в рабочем регистре бита 16
U 21BA, C77A,15 #19248          BIS LS[UBS.DCLO] TO WR[0]                ; кроме того установка бита 29
```

```
U 21B8, 3E80,15 ;19249          MOV WR[0] TO LS[CONTROL.STATUS]          ; установка битов в слове управления и состояния для
                ;19250                                                   ; указания монитору, что требуется установка, а затем
                ;19251                                                   ; снятие сигнала DCLO общей шины
U 21BC, 10E0,15 ;19252          MISC [SET.CP.ATTN]                       ; выдача для конс.процессора сигнала CPU ATTN
                ;19253  WAIT.ISSUE.0:
U 21BD, 8A1B,D4 ;19254          JMP [WAIT.ISSUE.0]                       ; цикл для ожидания ответа конс.процессора
U 21BE, BE80,95 ;19255          MOV WR[1] TO LS[CONTROL.STATUS]          ; восстановление старого слова управления и состояния
U 21BF, 5B00,14 ;19256          RETURN                                   ; возврат к основной программе
                ;19257  SETUP.DATO.NPR:
U 21C0, B646,15 ;19258          MOV LS[BR6] TO WR[0]                     ; установка в рабочем регистре бита 3 (прерывание через
                ;19259                                                   ; BR6)
U 21C1, C74C,15 ;19260          BIS LS[INT.DONE] TO WR[0]                ; установка в рабочем регистре бита 6 (прерывание по
                ;19261                                                   ; окончании выполнения)
U 21C2, C74A,15 ;19262          BIS LS[NPR] TO WR[0]                     ; установка в рабочем регистре бита 5 (выдача NPR)
U 21C3, C752,15 ;19263          BIS LS[C01] TO WR[0]                     ; установка в рабочем регистре бита 9 (выполнение цикла
                ;19264                                                   ; DATO)
U 21C4, C754,15 ;19265          BIS LS[FUNA] TO WR[0]                    ; установка в рабочем регистре бита 10 (выполнение 1
                ;19266                                                   ; цикла до переполнения счетчика циклов)
U 21C5, C740,15 ;19267          BIS LS[GO] TO WR[0]                      ; установка бита 0 (бит GO для UBE)
                ;19268            MOV WR[0] TO LS[T14],                  ; запоминание в LS T14
U 21C6, BE1C,14 ;19269          RETURN                                   ; конец
                ;19270  SETUP.DATI.NPR:
U 21C7, B646,15 ;19271          MOV LS[BR6] TO WR[0]                     ; установка в рабочем регистре бита 3 (прерывание через
                ;19272                                                   ; BR6)
U 21C8, C74C,15 ;19273          BIS LS[INT.DONE] TO WR[0]                ; установка в рабочем регистре бита 6 (прерывание по
                ;19274                                                   ; окончании выполнения)
U 21C9, C74A,15 ;19275          BIS LS[NPR] TO WR[0]                     ; установка в рабочем регистре бита 5 (выдача NPR)
U 21CA, C754,15 ;19276          BIS LS[FUNA] TO WR[0]                    ; установк  в рабочем регистре бита 10 (выполнение 1
                ;19277                                                   ; цикла до переполнения счетчика циклов)
U 21CB, C740,15 ;19278          BIS LS[GO] TO WR[0]                      ; установка бита 0 (бит GO для UBE)
                ;19279            MOV WR[0] TO LS[T14],                  ; запоминание в LS T14
U 21CC, BE1C,14 ;19280          RETURN                                   ; конец
                ;19281  CHECK.UBE.ERR:
U 21CD, 36FC,15 ;19282          MOV LS[INTERRUPT.VEC] TO WR[0]           ; выборка идентификатора прерывания BR
U 21CE, B710,95 ;19283          MOV LS[IDENT.MASK] TO WR[1]              ; рабочий регистр содержит FFFFFFC3
U 21CF, AF42,15 ;19284          BIC WR[1] TO WR[0]                       ; очистка в идентификаторе всех битов, кроме 5-2
                ;19285            CMP LS[BR7.IDENT] WITH WR[0],          ; проверка, что идентификатор BR7 (ошибка UBE)
U 21D0, CECC,35 ;19286          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 21D1, 8A1D,99 ;19287          JMP.IF[EQL] TO [UBE.ERROR]               ; переход к выдаче BG 7 и сообщению об ошибке, если BR7
U 21D2, 9F46,35 ;19288          MEM.REQ[ISSUE.BG] ADRS[BG6] DT[WORD]     ; подготовка для выдачи BG на уровне 6
U 21D3, 3022,15 ;19289          MOV MEM.DATA TO WR[0]                    ; завершение цикла BG
U 21D4, 3642,15 ;19290          MOV LS[#2] TO WR[0]                      ; счетчик задержки
                ;19291  LOOP.NOP:
U 21D5, DB00,15 ;19292          NOP                                      ; обеспечение времени для прерывания через BR7, если
                ;19293                                                   ; ошибка UBE
                ;19294            DEC WR[0],                             ; уменьшение счетчика
U 21D6, A100,35 ;19295          DT(LONG)&SET.ALU.CC                      ; и установка кодов условий
U 21D7, 0A1D,51 ;19296          JMP.IF[NEQ] TO [LOOP.NOP]                ; повторение, пока счетчик не будет исчерпан
U 21D8, 8A1E,27 ;19297          JMP.IF[NO.INTERRUPT] TO [CHECK.UBE.ERROR.RET+1] ; возврат, если нет ожидающего BR7. Иначе
                ;19298                                                   ; сообщение об ошибке UBE и содержимом регистра ошибок
                ;19299                                                   ; UBE
                ;19300  UBE.ERROR:
U 21D9, 9FCE,35 ;19301          MEM.REQ[ISSUE.BG] ADRS[BG7] DT[WORD]     ; запрос для выдачи BG на уровне 7
U 21DA, 3022,15 ;19302          MOV MEM.DATA TO WR[0]                    ; завершение цикла BG
U 21DB, 5F28,15 ;19303          MCOM LS[#FFFF] TO WR[0]                  ; рабочий регистр содержит FFFF0000
```

```
U 21DC, 3E8A,15 ;19304          MOV WR[0] TO LS[ERROR.MASK]             ; проверка битов 15-0 в регистре управления UBE
U 21DD, 98BF,B5 ;19305          MEM.REQ[READ.P] ADRS[BECR2] DT[WORD]    ; чтение регистра управления 2 UBE
U 21DE, 3022,15 ;19306          MOV MEM.DATA TO WR[0]                   ; выборка содержимого регистра управления 2
U 21DF, 2F82,95 ;19307          CLR WR[1]                               ; ожидаемые данные (биты ошибок очищены)
U 21E0, 0869,3C ;19308          JSR [CHECK.RESULT]                      ; сообщение об ошибке UBE
U 21E1, 5B00,14 ;19309          RETURN                                  ; возврат для цикла при ошибке
                ;19310  CHECK.UBE.ERROR.RET+1:
U 21E2, DB00,16 ;19311          RETURN+1                                ; возврат для отсутствия цикла при ошибке
                ;19312  CLEAR.CSR2:
U 21E3, 9D47,F5 ;19313          MEM.REQ[WRITE.CSR] ADRS[CSR2] DT[LONG]  ; запрос для записи в CSR2
                ;19314            WRITE.MEM LS[ZERO],                   ; очистка ошибок в регистре ошибок ОШ CSR2
U 21E4, 329C,14 ;19315          RETURN                                  ; конец выполнения
                ;19316
```

Лист регистрации изменений

Изм	Номера листов (стр.)			Всего листов в док.	N док.	Входящий N сопр. док. и дата	Подпись	Дата
	измененных	новых	аннулированных					