ISSN 0135-8588

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА СОЦИАЛИСТИЧЕСКИХ СТРАН

Выпуск
24

Межправительственная
комиссия по сотрудничеству
социалистических стран
в области вычислительной техники

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА СОЦИАЛИСТИЧЕСКИХ СТРАН

Выпуск 24

СБОРНИК СТАТЕЙ
ИЗДАЕТСЯ С 1977 г.

Москва
"Финансы и статистика"
1988

ББК 32.97
В94
УДК 681.3

В $\frac{2404000000-127}{010(01)-88}$ 27—88

I

Технические средства ЕС ЭВМ и СМ ЭВМ

УДК 681.324

ЛОКАЛЬНАЯ СЕТЬ MICRONET-16

В. ФУРНАДЖИЕВ, канд. техн. наук (НРБ),
Г. НАЙДЕНОВ, инженер (НРБ),
П. СТОЯНОВ, инженер (НРБ),
В. ПЕТРОВ, инженер (НРБ)

Проектирование локальных сетей микрокомпьютеров требует решения множества задач. При этом они разделяются на уровни, что способствует последовательному анализу и выполнению этих задач. Можно выделить следующие основные уровни.

Физический уровень. Он связан с обменом электрическими сигналами. Проблемы при проектировании: выбор топологии и среды обмена, способ электрического выделения отдельных микрокомпьютеров из общей среды обмена, выбор кодов обмена информацией и обеспечение правильной битовой и кадровой синхронизации.

Уровень доступа к среде обмена. На этом уровне решаются вопросы получения права доступа к линии. Проблемы при проектировании: выбор метода доступа к среде обмена, способ упаковки информации в кадры при передаче и ее распаковки при приеме.

Уровень логического управления каналом. Он связан с выполнением сетевых операций. Проблемы при проектировании: разработка и реализация протоколов обмена, связь сетевого программного обеспечения с операционной системой микрокомпьютера.

В статье рассматриваются некоторые из перечисленных выше проблем в связи с разработкой локальной сети MICRONET-16 для 16-битовых микрокомпьютеров.

MICRONET-16 представляет собой транспортный контроллер с соответствующим программным обеспечением. Он позволяет соединять в локальную сеть персональные микроЭВМ типа Правец-16. Пользователю предоставляются следующие возможности:

выполнять сетевые операции «память — память». Каждой из микроЭВМ, являющейся абонентом локальной сети, можно обмениваться информацией между своей оперативной памятью и памятью любого из остальных абонентов. При помощи таких сетевых операций осуществляется обмен текстовыми и графическими страницами между отдельными ЭВМ, а также устанавливается связь между процессами и производится обмен оперативными сообщениями между отдельными абонентами;

каждой из микроЭВМ, являющейся абонентом локальной сети, выполнять команды операционной системы, используя периферийные устройства любого из остальных абонентов. При помощи таких сетевых операций обеспечивается «прозрачность» сети по отношению к операционной системе. Каждый абонент предоставляет в распоряжение сети свои периферийные устройства, и они могут быть использованы остальными абонентами.

Технические характеристики локальной сети

Топология	Общая шина
Вид среды обмена	Коаксиальный или оптический кабель
Скорость обмена	800 Кбит/с
Код передачи	MANCHESTER
Метод доступа	Множественный доступ с обнаружением несущей и детерминированным разрешением конфликтов
Формат кадра	SDLC
Операционная система	MS DOS 3.10
Общее число абонентов	до 255
Длина линии передачи	до 1 км при коаксиальном кабеле; до 2 км при оптическом кабеле

Транспортный контроллер MICRONET-16 изготовлен в виде печатной платы. Он реализует физический уровень и уровень доступа к среде обмена.

Блок обмена информацией контроллера (рис. 1) осуществляет связь с микрокомпьютером — абонентом данной сети — при помощи системных сигналов, выведенных на стандартные разъемы для расширения конфигурации микроЭВМ. Из ресурсов микроЭВМ используются один канал прямого доступа к памяти для обмена необходимой информацией, одна шина прерывания и 12 адресов из адресного пространства.

Каналы приема и передачи выполняют преобразование информации из последовательного в параллельный код и наоборот, вычисление контрольных сумм, упаковку и распаковку одного кадра информации. Эти каналы устанавливают также связь с блоком для кодирования и синхронизации следующими сигналами: TxD (выход данных на передатчик, код NRZ), TxC (синхросигнал для выходных данных), RxD (вход данных в приемник, код NRZ), RxC

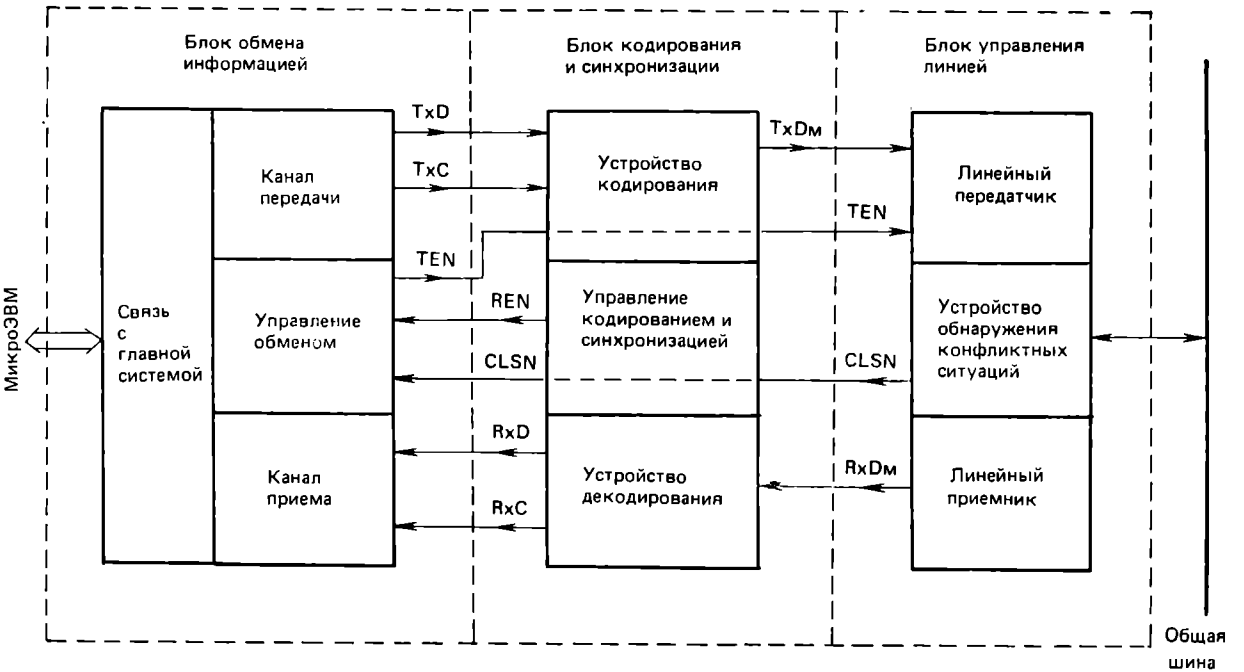


Рис. 1. Блок-схема транспортного контроллера MICRONET-16

(синхросигнал для входных данных). Протокол обмена — битовый, синхронный. Кадры обмениваются в формате SDLC.

Блок управления передачей реализует метод доступа к среде обмена при запросе микроЭВМ на передачу информации. К нему из блока кодирования и синхронизации поступают сигналы о состоянии среды обмена:

Ren, показывающий наличие обмена по общей шине при поступлении сигнала из блока управления линий;

CLSN, показывающий наличие конфликта (одновременной передачи сигнала от нескольких абонентов).

Со своей стороны блок управления передачей посылает к линейному передатчику сигнал Ten, разрешающий передачу сигнала по общей шине.

Блок кодирования и синхронизации осуществляет преобразование данных, поступающих от канала передачи данных из кода NRZ в код MANCHESTER (TxD, TxC → TxDm), и обратное преобразование данных, полученных из линейного приемника (RxDm→RxD, RxC). Сигнал синхронизации RxC формируется аппаратно методом цифровой тактовой синхронизации с плавной автоподстройкой фазы.

При отсутствии обмена информацией по общей шине сигнал Ren к блоку управления передачей не передается.

Блок управления линией осуществляет обмен сигналами между общей шиной и блоком кодирования и синхронизации. В линейном передатчике и приемнике происходят электрическая развязка абонента сети от общей шины и преобразование сигнала из уровня TTL в выбранную форму передачи по общей шине и наоборот. Передача в линейном передатчике становится возможной лишь при получении разрешения (сигнал Ten) блока управления передачей.

Блок осуществляет контроль конфликтной ситуации, следя за параметрами сигналов по линии. Схемы для регистрации конфликта обнаруживают как изменения в амплитуде (если по линии накладывается несколько сигналов), так и в частоте передачи сигналов (в линии появляются импульсы недопустимой частоты). При обнаружении конфликта к блоку управления передачей активируется сигнал CLSN.

Метод доступа к среде обмена, реализованный в локальной сети MICRONET-16, основан на следующих принципах: право доступа к среде определяется децентрализованно; все абоненты имеют равные приоритеты доступа при отсутствии конфликта и различные приоритеты после обнаружения конфликта; участвующие в конфликте абоненты получают более высокий приоритет, чем все остальные абоненты. С учетом перечисленных принципов предлагаемый метод обеспечивает каждому абоненту детерминированное время доступа к среде обмена, обладая при этом алгоритмической простотой и более высоким быстродействием по отношению к известным методам. Детерминированное время доступа дает возможность однозначно определить время реакции в распределенной системе, реализованной на базе локальной сети MICRONET-16.

Граф состояния, реализующий алгоритм доступа к среде обмена, показан на рис. 2. Каждый абонент занимает среду обмена (состояние «сеанс») только при условии, что он имеет такое право (находится в состоянии «разрешенного доступа»). Для этого он должен постоянно наблюдать за средой обмена независимо от того, имеется ли запрос на сетевую операцию (RQ=1) или нет, фик-

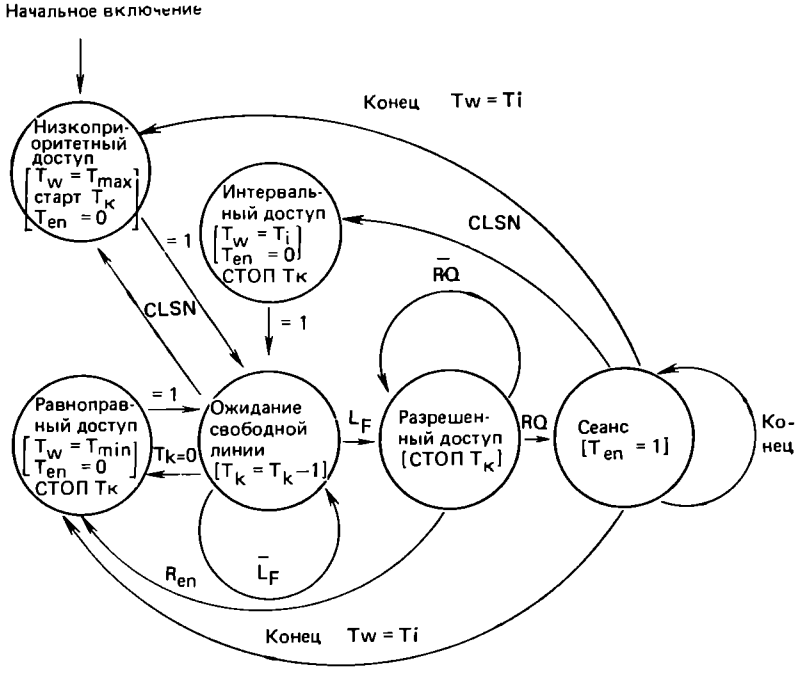


Рис. 2. Граф состояния, реализующий метод доступа

сируя моменты ее занятия (Ren=1) и освобождения (Ren=0). Линия считается свободной только в том случае, если в течение времени T_w по ней не передавались никакие сигналы (Ren=0). Выбранное значение для T_w ($T_{min} \leqslant T_w \leqslant T_{max}$) определяет и приоритет каждого абонента для доступа к среде обмена.

В зависимости от значения T_w каждый абонент может находиться в одной из следующих трех групп с различным приоритетом доступа.

В первой группе находятся все абоненты при нормальной работе, когда по линии нет конфликтов. Абоненты имеют равные приоритеты для занятия среды обмена независимо от их собственных адресов в сети. Значение T_{min} определяется из соображений обеспечения неделимости одного сеанса логической связи.

Во вторую группу переходят все абоненты, вызвавшие конфликт (CLSN=1 в состоянии «сеанс»). Цель интервального доступа —

обеспечить каждому абоненту бесконфликтное занятие линии для проведения одного сеанса. Для этого каждый абонент, участвующий в конфликте, получает разное время ожидания свободной среды в зависимости от своего собственного адреса i:

$$T_i = T_{min} + i \cdot \delta T.$$

Все остальные абоненты получают одинаковое время ожидания свободной среды, которое должно быть больше, чем максимальное из T_i. Это временно переводит их в состояние низкоприоритетного доступа, что освобождает линию для участников конфликта. Таким образом, после конфликта абоненты, которые вызвали его, получают возможность последовательно осуществить свои сетевые операции. Далее они тоже переходят в группу низкоприоритетного доступа. Порядок занятия линии после конфликта соответствует порядку возрастания собственных адресов абонентов.

Значение δT можно определить, анализируя самый неблагоприятный случай конфликта между абонентами с соседними адресами, которые находятся на максимальном расстоянии друг от друга. Временная диаграмма их состояний при конфликте показана на рис. 3.

После обнаружения конфликта абоненты j и i ($i = j+1$) получают право приоритетного доступа в соответствии с временем $T_w = T_j$ и $T_w = T_i$. Из-за наличия времени распространения сигнала r_{ij} эти периоды начинают отсчитываться с максимальной разницей t_{ij}.

По истечении T_j абонент j получает право на доступ к линии, переходя за время δp последовательно в состояние «разрешенного доступа» и «сеанс». В состоянии «сеанс» он начинает передачу по линии. Абонент i узнает о начале передачи абонентом j в наихудшем случае через время t_{ij}. До этого он не должен получать право доступа, для чего необходимо, чтобы:

$$T_i > T_j + \delta p + t_{ij} + t_{ji}.$$

Так как

$$i = j+1 \text{ и } t_{ij} = t_{ji},$$

то

$$T_{min} + (j+1)\delta T > T_{min} + j\delta T + \delta p + 2t_{ij},$$

тогда

$$\delta T = 2t_{ij} + \delta p.$$

При среде обмена «коаксиальный кабель с максимальной длиной 1000 м» t_{ij} приблизительно равно 4 мкс; для контроллера MICRONET-16 $\delta p \leqslant 1$ мкс. В связи с этим выбрано $\delta T = 10$ мкс.

В третью группу переходят после распознавания конфликта все абоненты, не принявшие участия в нем. Таким образом, получая на некоторое время самый низкий приоритет, они дают возможность разрешить конфликт без его дальнейшего углубления. Для

выполнения этого условия в сети с N абонентами необходимо, чтобы

$$T_{max} > T_i \text{ (при } i=N).$$

Каждый абонент из этой группы должен остаться в ней в течение времени T_k, которое достаточно для разрешения конфликта. Значение T_k вычисляется для самого неблагоприятного случая,

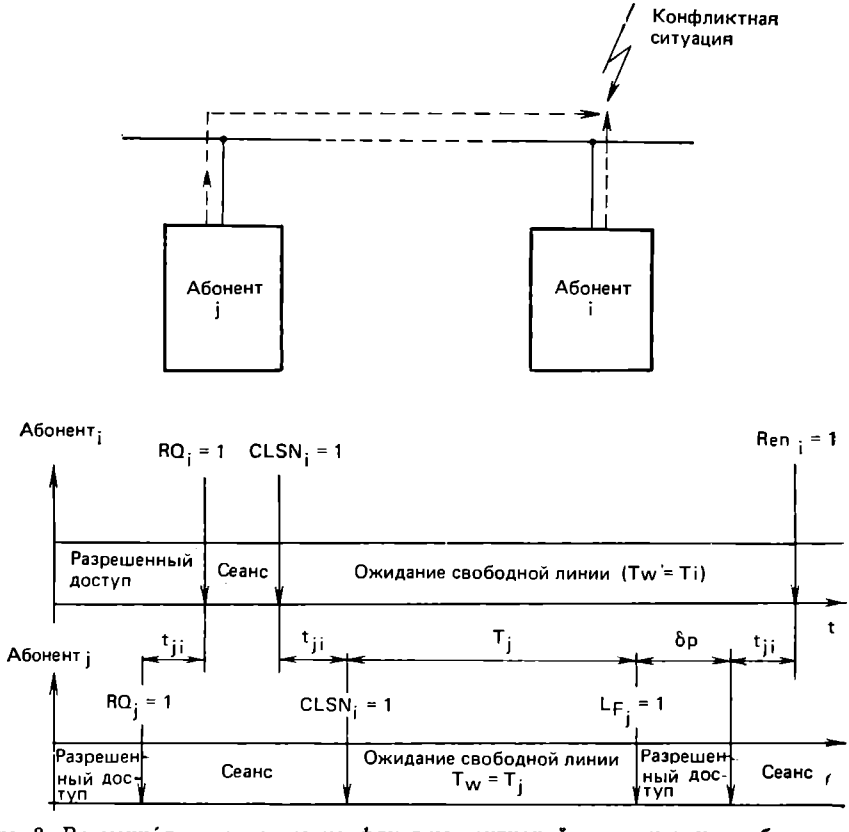


Рис. 3. Временна́я диаграмма конфликтных ситуаций между двумя абонентами

когда в этой группе находится только абонент с номером 0, а все остальные N—1 абоненты в сети имеют приоритетный доступ. Тогда

$$T_k = \sum_{i=1}^{N-1} (T_c + T_i),$$

где T_c — время реализации одного логического сеанса связи.

По сигналу T_k (по истечении времени T_k) каждый абонент, попавший в данную группу, повышает свой приоритет, переходя в

группу равноправного доступа. Если в конфликте не принимали участия все остальные абоненты, то право равноправного доступа возвращается абоненту раньше, после освобождения линии за время T_{max}.

Сетевые операции осуществляются на уровне логической связи. Реализация их протоколов основана на следующих принципах:

каждый абонент локальной сети может посылать запросы любому другому абоненту и одновременно с этим обслуживать запросы любого другого абонента;

каждый абонент в данный момент может выполнять одну собственную сетевую операцию (запрос на которую он послал другому абоненту) и одну чужую сетевую операцию (запрос на которую послал ему другой абонент);

инициатор собственной сетевой операции следит за корректным выполнением логической связи;

если на запрос получен ответ «занят», то заявитель должен повторить запрос через некоторое время. Запросы не сохраняются в исполнителе.

Сетевые операции реализуются в одном или двух сеансах. *Сеанс* — неделимая операция обмена кадрами между двумя абонентами сети. Неделимость означает, что другие абоненты не могут занимать линию между кадрами одного сеанса.

Сеанс начинается занятием среды обмена; при этом устанавливается логическая связь между абонентами и происходит обмен необходимой информацией. Занятие среды обмена начинается передачей последовательности флажков формата SDLC, после чего следует проверка о наличии конфликта и т. д.

Логическая связь между абонентами устанавливается передачей со стороны абонента-инициатора (АИ) кадра «Запрос». С его помощью выбирается абонент-приемник (АП), ему сообщаются тип требуемой сетевой операции и ее глобальные параметры. АП отвечает кадром «Статус», в котором содержится его состояние относительно требуемой операции. Если АП находится в состоянии «занят», сеанс заканчивается и сетевая операция откладывается. При состоянии «свободен» следует обмен параметрами, данными и результатами операции.

Во время всего сеанса доступ других абонентов к линии невозможен, так как это нарушило бы логическую последовательность протокола. Неделимость логического сеанса обеспечивается соответствующим выбором минимального интервала ожидания свободной линии другими абонентами T_{min}. Необходимо соблюдать неравенство

$$T_{min} > T_p,$$

где T_p — максимальная пауза между двумя кадрами в рамках одного логического сеанса.

После этого АП посылает требуемые данные или принимает их в зависимости от типа операции.

Определены два типа сетевых операций: считывание с памяти другого абонента и запись в память другого абонента. Они характерны тем, что выполняются за один сеанс, так как абонент-приемник имеет возможность немедленно выполнить требуемую операцию. Диаграммы протоколов, реализующих эти операции, показаны на рис. 4. В кадре «Запрос» к АП посылаются параметры сетевой операции — начальный адрес и число байтов. Объектом этих сетевых операций могут быть все периферийные устройства, которые поддерживает MS-DOS.

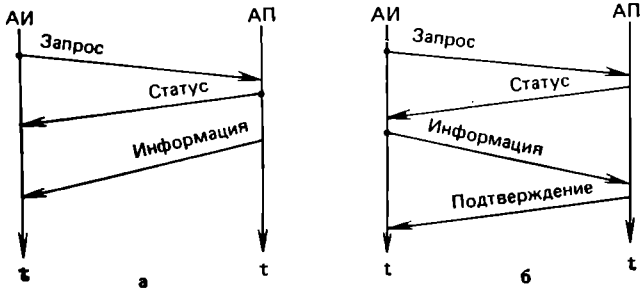


Рис. 4. Сетевые операции «память — память»:
а — считывание; *б* — запись;
АИ — абонент-источник; АП — абонент-приемник

Определены три типа сетевых DOS-операций: считывание с периферийного устройства другого абонента, запись в периферийное устройство другого абонента, управляющая операция, связанная с обменом командами и данными о состоянии с периферийным устройством другого абонента.

Характерным для сетевых DOS-операций является их выполнение в двух сеансах. После получения параметров операции АП не может выполнить сразу нужную операцию. Он должен выждать время, требуемое на переход адресуемого периферийного устройства в состояние готовности, на позиционирование, поиск и выполнение записи или считывания. Разделение операции на два сеанса позволяет освободить среду обмена во время подготовительных работ. В паузе среда обмена может быть использована другими абонентами локальной сети. Диаграмма протоколов, реализующих сетевые DOS-операции, показана на рис. 5.

Локальная сеть MICRONET-16 предоставляет пользователю широкие возможности для доступа к ресурсам всех абонентов в виде виртуальных периферийных устройств. Для корректного выполнения сетевых DOS-операций необходимо, чтобы в режиме выбора конфигурации для каждого виртуального устройства было задано следующее соответствие:

<виртуальное устройство>=<реальное устройство>, <адрес абонента>

Выбор конфигурации устройств осуществляется отдельно и независимо для каждого абонента сети в диалоговом или программном режиме. К сетевому программному обеспечению обращаются выдачей команды прерывания и передачей необходимых пара-

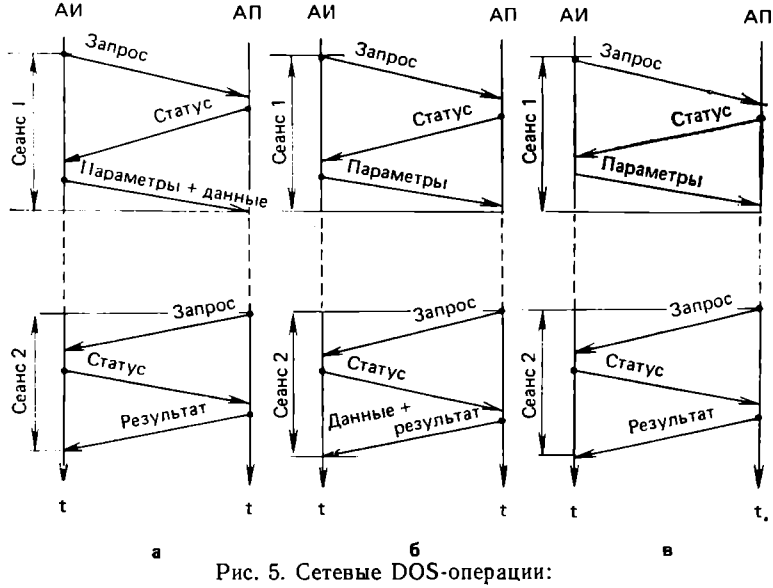


Рис. 5. Сетевые DOS-операции:

а — запись; *б* — считывание; *в* — управление

метров. Соответствие между виртуальными и реальными устройствами может изменяться в любой момент работы локальной сети. После установления конфигурации пользователь может работать со своими виртуальными устройствами, применяя обычные команды MS-DOS, без учета разницы между собственными и чужими периферийными устройствами.

УДК 681.324

СТАНЦИЯ ЛОКАЛЬНОЙ СЕТИ КОЛЬЦЕВОГО ТИПА СЛК-СМ

Ю. Н. ГЛУХОВ,
канд. техн. наук (СССР),
М. С. КОЛОСКОВ,
канд. техн. наук (СССР),
Ю. Б. КОЖЕВНИКОВ, инженер (СССР),
А. Л. КУЗНЕЦОВ, инженер (СССР)

В настоящее время в архитектуре локальных сетей, имеющих различные топологию, скорость передачи, метод доступа, передающую среду и т. д., значительное место занимают

кольцевые локальные сети, обеспечивающие средние скорости передачи данных (до 1—2 Мбит/с) на достаточно значительные расстояния (до 2 км). Эти сети с успехом используются для создания систем массового обслуживания, АСУ, САПР, автоматизации учрежденческой деятельности и в других сферах.

Весьма перспективной областью применения кольцевых локальных сетей является промышленность, так как локальные сети представляют хорошую основу для создания гибких производственных систем, робототехнических комплексов и т. п. При большом количестве абонентов общая протяженность кольцевой сети может достичь нескольких десятков километров, что дает возможность использовать ее на крупных предприятиях. Однако при таком применении локальных сетей особые требования предъявляются к надежности их работы и качеству передачи данных. Кольцевые сети очень уязвимы в отношении отказов, так как выход из строя любого соединения или станции приводит к отказу всей сети и соответственно производственного комплекса, обслуживаемого данной сетью. Кроме того, каналы связи должны обладать хорошей помехозащищенностью.

Обеспечение надежности сводится к дублированию каналов связи и отдельных устройств, как правило, выполняющих функции диспетчера сети. Однако широкого распространения в конкретных локальных сетях эти методы до сих пор не получили в связи с большими аппаратными затратами, сложностью процедур реконфигурации сети, необходимостью изменения программы работы станции.

При разработке станции СЛК-СМ была сделана попытка с помощью минимальных дополнительных аппаратных затрат обеспечить дублирование канала связи с обнаружением отказа основного канала и простой процедурой реконфигурации сети. Назначение станции СЛК-СМ — построение локальных кольцевых сетей с повышенной живучестью и надежностью.

Техническая характеристика станции СЛК-СМ

Количество адресуемых абонентов	124
Скорость передачи информации, Кбит/с	500
Физическая среда, используемая для передачи	Витая пара, коаксиальный кабель
Максимальное расстояние между станциями при передаче по витой паре, км	1,5
Длина информационной части пакета, байт	от 1 до 80
Метод повышения достоверности	Контроль по паритету каждого символа, контрольная сумма пакета
Метод доступа	Вставка регистра
Управление сетью	Децентрализованное
Интерфейс с абонентом	С2-ИС, ИРПС
Скорость обмена данными с абонентом, бит/с	1200, 2400, 4800, 9600, 19 200

Масса в автономном исполнении, кг	2,5
Габаритные размеры, мм	240·295·55
Потребляемая мощность, Вт	10

Станция имеет два конструктивных исполнения: автономное — с внутренним стабилизированным источником питания и встраиваемое — на плате типа Е2 унифицированных конструктивов СМ ЭВМ.

Встраиваемый вариант станции имеет следующие уровни питания и потребления: +5 В —0,8 А; +12 В —0,1А; —12 В —0,04 А.

Для решения задач повышения живучести сети, помехозащищенности передачи, уменьшения времени восстановления сети при отказе в станции СЛК-СМ использованы следующие технические решения:

дублирование сегментов сетевого канала с автоматическим переключением с основного сегмента на резервный. Переключение происходит не только в случае отсутствия сигнала в канале, но и при ухудшении его качества;

полная гальваническая развязка с линией связи;

манчестерское кодирование информации.

Как отмечалось, сопряжение станции с абонентами сети выполняется с помощью интерфейсов С2-ИС и ИРПС, выход на которые имеют все современные мини- и микроЭВМ.

Станция обеспечивает включение в сеть оборудования с различным «интеллектуальным» уровнем благодаря функциональной полноте сетевых протоколов, отрабатываемых внутренним программным обеспечением станции. Таким образом, в вычислительную сеть могут быть объединены ЭВМ различной архитектуры, интерактивные терминалы, устройства ввода-вывода, причем последние приобретают статус сетевых устройств. При этом обеспечивается сетевой обмен между видеотерминалами (типа электронной почты), обмен между терминалами и устройствами ввода-вывода, а также между ЭВМ.

Станция имеет три режима работы: интерактивный, мультиплексный и базовый.

Интерактивный режим используется для сопряжения станции с видеотерминалом пользователя сети. В этом режиме пользователю предоставляется набор из 13 сетевых команд, который позволяет: входить в сеть, устанавливая свой адрес, и выходить из сети; устанавливать соединения с абонентами сети для выполнения обмена; устанавливать режимы обмена; программировать режимы работ станции, в том числе параметры обмена с видеотерминалом; программировать режимы работы удаленной станции; проводить анализ состояния сети; вызывать список абонентов сети, список установленных соединений и т. д., а также проводить анализ состояния станции.

Мультиплексный режим работы станции предназначен для сопряжения с ЭВМ. Основное отличие этого режима состоит в том, что станция позволяет создавать до 64 портов и соответствующее количество виртуальных каналов, обеспечивая возможность одно-

временной работы ЭВМ с 63 абонентами (один канал — служебный). Набор команд в основном совпадает с набором интерактивного режима.

Базовый режим используется для работы станции с устройствами ввода-вывода, не имеющими возможности выполнять самостоятельно сетевые процедуры. Работа в этом режиме обеспечивается специальными командами дистанционного программирования, с помощью которых оператор или ЭВМ может задавать режим работы удаленной станции, к которой подключено устройство ввода-вывода.

Архитектурные особенности сети СЛК-СМ, связанные с созданием неоднородной сети ЭВМ, терминалов, устройств ввода-вывода, наложили специфический отпечаток на систему сетевых протоколов, выполняемых станцией СЛК-СМ, хотя в этой системе протоколов могут быть очерчены все уровни, необходимые для самостоятельного сетевого взаимодействия. Например, можно выполнить сетевое взаимодействие терминалов типа электронной почты без дополнительных программных средств.

Для универсального применения станции необходимо специальное программное обеспечение, которое базируется на внутреннем программном обеспечении станции. Сеть СЛК-СМ программно совместима с получившими распространение в СССР локальными сетями ИЗОТ — РИНГ и «Эстафета», поэтому могут использоваться разработанные для этих сетей пакеты «Локал» и «Эстафета». Разработан пакет программ «Колос», который обеспечивает сетевой сервис для машин СМ ЭВМ. Внутреннее программное обеспечение станции позволяет также решать задачи контроля и диагностики сети. С этой целью в случае отсутствия в сети информативных пакетов предусмотрена периодическая посылка станциями тестовых пакетов, с помощью которых проверяется целостность сети и определяется место неисправности.

Тесты самодиагностики, которыми дополнено внутреннее программное обеспечение станции, позволяют проверить основные узлы станции. Программа самодиагностики, включающая шесть тестов, выполняется в автономном режиме после отключения станции от сетевого канала. Для выполнения самодиагностики станцию необходимо подключить к видеотерминалу. Благодаря повышению живучести сети СЛК-СМ за счет введения автоматического переключения на резервный канал, а также росту помехозащищенности передачи существенно расширяется область применения сети, в частности, ее можно использовать в промышленности. Действительно, при отказе одного из сегментов в сети с одиночным каналом на восстановление работоспособности сети может потребоваться до 0,5 ч, что в свою очередь может привести к простою обслуживаемого сетью производственного участка. Время восстановления работоспособности сети СЛК-СМ составляет всего 2 с.

Станция СЛК-СМ прошла в 1986 г. межведомственные испытания. В 1987 г. выпущена первая промышленная партия. Серийное производство станции СЛК-СМ начато в 1988 г.

УДК 681.3.02.57

АРХИТЕКТУРА ВЫСОКОПРОИЗВОДИТЕЛЬНОГО КОМПЛЕКСА МОДЕЛИРОВАНИЯ ЦИФРОВЫХ СХЕМ

Н. Л. ПРОХОРОВ,
генеральный конструктор
СМ ЭВМ (СССР),
Б. Г. СЕРГЕЕВ,
канд. техн. наук (СССР)

В практике проектирования средств цифровой техники широко используется моделирование, которое является эффективным и во многих случаях единственным инструментом верификации проекта до изготовления образцов проектируемых средств. Особая роль отводится моделированию при разработке БИС и СБИС, где возможности физического макетирования схем весьма ограничены или вообще отсутствуют. В связи с этим средства моделирования представляют собой обязательную часть любой современной САПР СБИС.

Традиционный способ моделирования основывается на использовании программ, представляющих объект на соответствующем уровне детализации (функциональном, логическом, схемотехническом). Эти программы прогоняются на ЭВМ общего назначения. Однако решение задач верификации сложных схем и синтеза для них тестов производственного и эксплуатационного контроля требует значительных затрат машинного времени даже при использовании ЭВМ высокой производительности. В наибольшей степени это относится к логическому (на уровне логических элементов — вентилей) и схемотехническому моделированию. Уже для БИС, содержащих до 10 тыс. логических вентилей, временны́е затраты составляют сотни часов. Для СБИС, сложность которых оценивается сотнями тысяч вентилей, требуемое машинное время возрастает на несколько порядков. Таким образом, системы моделирования на базе универсальных ЭВМ оказываются малоэффективными даже для существующих объектов, сложность которых определяется возможностями микроэлектронной технологии сегодняшнего дня.

Существенное увеличение скорости моделирования, позволяющее резко повысить порог сложности схем, может быть достигнуто в результате применения специализированных вычислительных средств. Первый опыт создания экспериментальных образцов таких средств относится к концу 70-х годов. Начало коммерческому выпуску специализированных процессоров-ускорителей и комплексов моделирования было положено в 1982 г. Выпускаемые в на-

стоящее время в США лучшие образцы такого оборудования обеспечивают ускорение моделирования по сравнению с универсальными ЭВМ в 1000 и более раз. В этом плане показательна продукция фирмы *Zycad Corp.*, в течение ряда лет занимающей лидирующее положение на рынке высокопроизводительных средств моделирования. С 1982 г. эта фирма выпустила систему LE1000, представляющую собой модульный комплекс, сопрягаемый на правах периферийного процессора с ЭВМ типов VAX II и Microvax фирмы *DEC* и с ЭВМ некоторых других фирм. Младшая модель LE 1001 этой системы имеет один процессор моделирования, модель LE 1002 — один сдвоенный процессор, старшая модель — LE 1032 — 16 сдвоенных процессоров. В 1985 г. фирма объявила о выпуске одноплатного процессора *Sprintor*, предназначенного для инженерных АРМ или персональных компьютеров с шиной *Multibus*, и комплекса EXPEDITOR 100/200, включающего пакет прикладных программ. Комплекс рассчитан на работу с ЭВМ VAX II, Microvax, с рабочими станциями фирмы *Apollo*, с персональными ЭВМ PC AT фирмы *IBM*. Наконец, в 1985 г. *Zycad Corp.* анонсировала разработку 256-процессорной суперсистемы моделирования SDE 8000 сверхвысокой производительности. Основные характеристики перечисленного оборудования фирмы *Zycad Corp.* [1] приведены в таблице.

Наименование	Год начала производства	Объем моделируемых схем, тыс. 3-входовых элементов	Производительность, млн. событий/с	Цена, тыс. долл.
LE1001	1982	32	0,5	250
LE1002	1983	64	1,0	400
LE1032	1983	1 000	16	2 100
EXPEDITOR 100/200	1985	16/32	1,0	95/120
	1986	16	0,2	20
SDE8032 (модификация, содержащая 256 процессоров)	1986	1500	1 000	2 500

Примечание. Все указанные в таблице средства фирмы *Zycad Corp.* обеспечивают событийное моделирование, при котором вычисляются состояния только тех элементов схемы, у которых изменяется хотя бы один вход, поэтому производительность этих средств измеряется числом событий, обрабатываемых в 1 с. Событием считается изменение состояния (активизация) какого-либо выхода элемента или внешнего входа схемы. Обработка одного события включает вычисление новых состояний всех элементов, входы которых связаны с источником события. При этом предполагается, что среднее количество таких элементов-приемников события равно 2,5.

Технико-экономические исследования, проведенные в рамках программы СМ ЭВМ, показали, что потребность ускорения проектирования средств ВТ и обеспечения надлежащего уровня их качества диктует необходимость значительного увеличения масштабов применения САПР и улучшения их показателей. Поскольку при широком применении САПР важное значение приобретает фактор

стоимости, основной технической базой этих систем вместо традиционно используемых больших ЭВМ общего назначения и даже суперЭВМ всюду, где это возможно, должны стать дешевые средства СМ ЭВМ, дополненные сравнительно недорогими специализированными сопроцессорами, периферийными процессорами и субкомплексами.

Специализация этого дополнительного оборудования на конкретные задачи проектирования помимо очевидного выигрыша в стоимости во многих случаях обеспечивает высокую производительность моделирования, недостижимую даже при использовании суперЭВМ. Таким образом, разработка указанных средств является необходимым шагом в создании нового поколения САПР, предназначенных для проектирования современных и будущих средств ВТ. Примером такой разработки является создание в рамках СМ ЭВМ 4-й очереди комплекса специализированных средств моделирования для САПР СБИС и других объектов цифровой техники.

Технические средства этого комплекса совместно с типовыми вычислительными комплексами СМ ЭВМ должны обеспечивать высокоскоростное моделирование цифровых объектов на функциональном, логическом и переключательном (транзисторном) уровнях с учетом реальных задержек элементов, а также моделирование неисправностей и анализ полноты тестов. Комплекс должен решать следующие типовые задачи САПР:

верификация схем проектируемых средств ВТ;

отладка микропрограмм и подготовка данных, необходимых для программирования ПЗУ, ПЛМ и других программируемых схем;

проверка корректности и полноты тестов;

автоматизированный синтез тестов;

экспертиза проектируемых схем на контролепригодность;

оценка эффективности средств встроенного контроля.

Комплекс должен иметь модульную структуру, позволяющую изменять его основные характеристики (вид моделирования, производительность, объем моделируемых схем, стоимость) путем изменения состава технических и программных средств в зависимости от требований конкретного применения.

Комплекс в максимальной модификации будет рассчитан на объем моделируемых схем на вентильном уровне порядка 1 млн. логических элементов и иметь производительность не ниже 10—20 млн. событий/с.

Наивысшая из всех встречающихся в литературе оценка скорости событийных программ вентильного уровня, реализуемых на универсальных ЭВМ, получена для суперЭВМ Cyber 176 фирмы *CDC* и равна 90 тыс. событий/с [2]. Для ЭВМ среднего класса (IBM 370/168, EC1061, EC1066 и др.) скорость событийного моделирования составляет 10—20 тыс. событий/с. Для супермини-ЭВМ типа VAX II/780 аналогичная оценка равна 1—5 тыс. событий/с. Таким образом, при производительности специализированного

комплекса 10 млн. событий/с моделирование ускоряется по сравнению с суперЭВМ в 100 раз, а по сравнению с супермини-ЭВМ — в 2000 раз.

Для разработки комплекса моделирования СМ ЭВМ в 1984—1985 гг. проведены исследования, в результате которых определены требования к комплексу, особенности его архитектуры и принципы построения. При сохранении основных черт классической (фон-Неймана) архитектуры возможны следующие пути повышения производительности спецпроцессоров моделирования:

сокращение списка команд (как в RISC-машинах общего назначения), упрощение процесса выполнения команды и сжатие на этой основе цикла команд до нескольких микроопераций;

использование многоадресных команд, позволяющих обрабатывать состояния большинства вентилей одной командой;

применение раздельных ЗУ для команд и данных, приспособление организации ЗУ к структуре информации, копирование одних и тех же данных в нескольких ЗУ для совмещения операций многоадресной выборки;

полная конвейеризация процесса выполнения команды;

исключение потерь времени при передаче данных с одной стадии обработки на другую путем использования синхронного конвейера, в котором каждая стадия выполняется за один такт синхронизации.

Реализация всех этих возможностей в совокупности позволяет ускорить моделирование по сравнению с обычным процессором на элементах того же быстродействия примерно на порядок.

Более высокая производительность процессоров моделирования достигается при использовании специальных архитектур, реализующих событийные алгоритмы. В наиболее простом варианте такая архитектура обеспечивается введением в обычный процессор дополнительного сопроцессора — аппаратного событийного диспетчера программы при сохранении программного способа вычисления элементов схемы. Максимальное ускорение в этом варианте получается при полном совмещении вычисления элементов с операциями анализа событий и событийного управления процессом вычислений. Выигрыш в скорости по сравнению с несобытийным (сплошным) компилятивным моделированием определяется величиной 100 С/АК, где С — скорость сплошного моделирования, выраженная числом вентилей, вычисляемых в секунду, А — средний процент активных (изменяющих состояние выхода) вентилей в каждый момент времени, К — средний коэффициент разветвления выхода вентиля. При типовых значениях $A=2-5\%$ и $K=2{,}5$ скорость моделирования по отношению к ускоренному фон-неймановскому процессору увеличивается в 8—20 раз.

Дальнейшее повышение производительности спецпроцессоров событийного моделирования обеспечивается более жесткой их специализацией и достигается:

оптимизацией алгоритма моделирования;

приспособлением структуры процессора (вплоть до топологического соответствия) к выбранному алгоритму;

использованием распределительной памяти, исключающей разделение одних и тех же ЗУ разными этапами алгоритма;

аппаратной реализацией управления;

использованием синхронного конвейера операций для выполнения всех шагов алгоритма.

Осуществление перечисленных мер позволяет повысить скорость вентильного моделирования на 2—3 порядка по сравнению с процессорами классической архитектуры.

При моделировании на функциональном уровне такое же увеличение производительности требует не только введения механизма аппаратного событийного управления процессом, но и значительного сокращения времени вычисления сложных функциональных узлов.

В случае функционального моделирования блоков и устройств ЭВМ нужный эффект дает программно-аппаратная обработка, при которой в качестве моделей функциональных узлов используются образцы реальных БИС, применяемых в моделируемом объекте. При этом время вычисления узла определяется только длиной программы обмена данными со схемой-моделью, практически не зависит от его функций и сложности и сокращается в сотни раз по сравнению с чисто программным способом.

Наконец, еще один способ ускорения как вентильного, так и функционального моделирования состоит в параллельном вычислении разных подсхем моделируемой схемы с помощью нескольких процессоров, объединенных в многопроцессорный комплекс. Так как между подсхемами, обрабатываемыми разными процессорами, существуют связи, в комплексе должен предусматриваться межпроцессорный обмен данными в каждый момент (микротакт) дискретного модельного времени.

При N параллельно работающих процессоров (архитектура типа «N — потоков команд, N — потоков данных») производительность комплекса, выраженная числом элементов, вычисляемых в 1 с, или числом обрабатываемых в 1 с событий, в N раз выше, чем у одного процессора. Однако реальное ускорение моделирования равно N/P, где $P>1$ из-за дополнительных затрат времени на межпроцессорный обмен и неравномерной загрузки процессоров вследствие различия потоков событий и трудностей такого разделения схемы между процессорами, при котором эффективно используется параллелизм вычислений. С увеличением N возрастает и P.

По экспериментальным данным фирмы IBM, для машины сплошного моделирования YSE $P\approx 1$ при $N<8$; при $N=32$ $P\approx 2$; при $N=64$ $P\approx 3$. По сведениям фирмы *NEC* для событийного комплекса HAL наращивание количества процессоров с 7 до 25 увеличивает реальную производительность комплекса в 1,6 раза, т. е. при $N=25$ $P=2{,}2$. Таким образом, начиная с некоторого значения N (по-видимому, около 20—30) для достижения заданной реальной производительности многопроцессорного комплекса выгоднее уве-

личивать быстродействие процессоров, чем их количество. С этой точки зрения представляется проблематичной возможность реального использования вычислительной мощности комплексов с 256 процессорами, таких, как YSE IBM и SDE 8032 Zycad.

На основе изложенных здесь принципов разработан предварительный проект комплекса моделирования СМ ЭВМ [3 и 5], который может служить основой для опытно-конструкторских работ. Комплекс имеет параллельную многопроцессорную архитектуру с распределенной по процессорам памятью исходных, обрабатываемых данных, данных (команд) управления и с централизованным управлением временем модели. В максимальной конфигурации комплекс содержит 16 специализированных процессоров моделирования и процессор обмена, обеспечивающий передачу данных между моделирующими процессорами и их сопряжение с ВК САПР. Связи между процессорами унифицированы и осуществляются с помощью внутренней асинхронной магистрали комплекса; соединение процессора обмена с ВК обеспечивается через системный интерфейс ВК.

Межпроцессорный обмен в комплексе основывается на событийном принципе. При этом из одного процессора в другие передаются сведения только о тех элементах или внешних входах моделируемой схемы, которые изменяют свое состояние в текущем микротакте моделирования. Инициатором обмена всегда является процессор, моделирующий источник сигнала, передаваемого в другие процессоры. После захвата магистрали процессор-инициатор сам адресует процессоры-приемники данных и обеспечивает требуемую передачу.

Арбитраж запросов процессоров моделирования и предоставление им магистрали являются функциями процессора обмена. Он же использует эту магистраль для обмена данными между ВК и моделирующими процессорами. Для перекрытия обмена с процессом моделирования связь всех процессоров с магистралью осуществляется через буферы данных типа FIFA.

Основные возможности и характеристики комплекса определяются используемыми в его составе процессорами моделирования. В первую очередь разработки входят два типа таких процессоров: процессор вентильного моделирования (ПВМ) и процессор функционального моделирования (ПФМ).

ПВМ предназначен для моделирования схем на уровне базовых логических элементов (вентили И, ИЛИ, И-НЕ, ИЛИ-НЕ, И-ИЛИ, ИЛИ-И и т. п.) и на переключательном (транзисторном) уровне. Кроме того, ПВМ обеспечивает функциональное моделирование входящих в схему оперативных и постоянных ЗУ. В своей основной конфигурации базовый логический элемент имеет 1 выход и не более 6 входов, базовый элемент ЗУ — не более 4 выходов и 22 входов. Существуют также конфигурации логических элементов с внутренней обратной связью и с динамическим (фронтовым) входом, которые служат для функционального моделирования триггеров.

Архитектура ПВМ ориентирована на многозначное (значения 1, 0, неопределенное и высокоимпедансное с характеристиками логической «силы») логическое и логико-временное событийное моделирование с индивидуально задаваемыми задержками элементов, отдельно для переключения выхода с 0 на 1 и с 1 на 0. Процессор позволяет моделировать схемы произвольной конфигурации, в том числе содержащие встроенные автогенераторы, функции И, ИЛИ, образуемые объединением выходов элементов, элементы с тремя состояниями выхода, двунаправленные линии передачи данных, двунаправленные вентили (транзисторы), аттенюаторы (резисторы), элементы задержки.

Максимальное количество значений любого сигнала в схеме при многозначном моделировании — 16. Функции элементов (примитивы) в многозначной логике произвольные и задаются с помощью таблиц истинности, загружаемых от ВК. Диапазон времени задержки для каждого элемента 0—255 микротактов. Максимальный объем моделируемой схемы — 64 тыс. элементов и, кроме того, 512 Кбит оперативной и постоянной памяти, производительность при моделировании — 1 млн. событий/с.

По своей архитектуре ПВМ представляет 8-стадийный синхронный конвейер операций с распределенной между стадиями памятью. Каждая стадия конвейера соответствует определенному шагу событийного алгоритма моделирования. На вход конвейера подаются данные, определяющие события в схеме в текущем микротакте ее дискретного времени. Событием считается изменение состояния (активизация) выхода элемента или входа схемы. Обработка одного события включает: выборку активного элемента из списка событий, обновление состояния его выхода, определение элементов-приемников, входы которых связаны с выходом активного элемента, вычисление нового состояния приемников и сравнение его с предыдущим состоянием, планирование моментов времени изменения состояния приемников исходя из данных значений их задержки, запись активизированных приемников в список событий.

Список событий организуется на основе так называемого «колеса времени» [5], реализованного с помощью набора стеков событий, каждый из которых соответствует определенному микротакту. Для обработки активизированных логических элементов используется однопроходовый алгоритм, для элементов ЗУ — двухпроходовый. Оба алгоритма интерпретируются на основе аппаратного управления.

Память ПВМ обеспечивает хранение описания моделируемой схемы и данных самого процесса моделирования. Она организована с помощью 12 отдельных ЗУ с независимым управлением, обеспечивающим одновременное обращение к 190 битам данных. Общая емкость всех ЗУ — 1,5 Мбайт.

Другой процессор комплекса — ПФМ служит для двоичного событийного моделирования схем на уровне вентилей, функциональных узлов и на смешанном вентильно-функциональном уровне. Входящие в схему вентили моделируются программно. Функцио-

нальное моделирование производится на уровне интегральных схем, в том числе СИС и БИС. При этом в качестве моделей СИС и БИС используются их реальные образцы, входящие в состав встроенной в ПФМ аппаратной библиотеки либо подключаемые пользователем к специальным внешним каналам процессора. Определяемые схемой объекта соединения СИС и БИС между собой и с программно-моделируемыми элементами устанавливаются с помощью команд передачи значений соответствующих входных и выходных сигналов. При этом эмулируются синхронные или асинхронные интерфейсы БИС, подключенных к внешним каналам ПФМ. Многоразрядные информационные, адресные и командные тракты объекта, включая линии передачи, регистры и другие схемы, моделируются параллельно как один сложный (векторный) сигнал или элемент с использованием команд, оперирующих с данными переменного формата (1, 4, 8 или 16 разрядов).

Максимальный объем моделируемых схем, предусматриваемый архитектурой ПФМ: на вентильном уровне — 16 тыс. элементов, на функциональном уровне — 2 тыс. СИС и БИС. Количество внешних каналов для подключения сменных БИС-моделей — до 4096. Максимальный объем встроенной аппаратной библиотеки СИС и БИС — 500 типов схем. Производительность: при вентильном моделировании — 0,5 млн. событий/с, при функциональном — 0,1 млн. событий/с.

ПФМ состоит из двух работающих параллельно конвейерных сопроцессоров: вычислителя, выполняющего программу моделирования, и событийного диспетчера этой программы. Сопроцессор-вычислитель во многом идентичен процессору RISC-архитектуры. Событийный диспетчер обладает специальной архитектурой, оптимизированной для реализации двухпроходового алгоритма событийного моделирования с нулевыми и единичными задержками элементов и интегральных схем.

ПВМ и ПФМ имеют одинаковый интерфейс для сопряжения с внутренней магистралью комплекса моделирования, поэтому они могут использоваться в его составе в любых сочетаниях при общем количестве процессоров до 16. Тип интерфейса связан с ВК, реализованным в процессоре обмена, — общая шина ЭВМ. Однако заменой соответствующего модуля в этом процессоре или всего процессора может быть обеспечен перевод комплекса моделирования на другие системные интерфейсы, например на интерфейсы ЕС ЭВМ.

Литература

1. Zucad announces high — speed simulation for every design//Computer Design.— V. 4.— 1985.— № 7.— P. 32—33.
2. Abramovic M., Levendel Y. H., Monon P. R. A logic Simulation Machine//ACM IEEE 19-th Design Automation Conference.— Las Vegas.— 1982.— P. 65—73.
3. Прохоров Н. Л. Повышение производительности вычислительных комплексов СМ ЭВМ при решении задач автоматизации проектирования цифровой техники//Приборы и системы управления. — 1987. — № 6. — С. 3—5.

4. Сергеев Б. Г. Многопроцессорный комплекс моделирования для САПР цифровых объектов. Моделирование, контроль и диагностика цифровых устройств//Сб. трудов ИНЭУМ.— М.— 1986.— С. 4—16.
5. Теория и методы автоматизации проектирования вычислительных систем/Под ред. М. Брейера.— М.: Мир, 1977.

УДК 681.324

ТЕХНИЧЕСКИЕ И ПРОГРАММНЫЕ СРЕДСТВА ЛОКАЛЬНОЙ СЕТИ ROLANET1

И. РИХТЕР, инженер (ГДР),
Б. ТЕРПЕ, инженер (ГДР)

Локальные сети (ЛС) получили распространение во всем мире. Их внедрению способствовала разработка стандартов на основные типы ЛС. В ГДР некоторые стандартные решения внедрены главным образом в ЛС вузов. На НП «Роботрон» ведутся работы над реализацией возможности включения всех средств вычислительной техники, выпускаемой комбинатом, в локальную сеть. Такая ЛС должна удовлетворять следующим требованиям:

в ней должна использоваться выпускаемая в настоящее время в странах СЭВ элементная база;

стоимость средств подключения устройства должна быть значительно ниже стоимости самого устройства;

скорость передачи данных в ЛС должна удовлетворять требованиям большого круга пользователей.

Ниже описывается ЛС такого типа — ROLANET1 (Robotron local area network). Система разработана в результате широкого сотрудничества с вузами, в особенности с Центром информатики и с ВЦ Дрезденского политехнического института, с факультетом электроники Берлинского университета им. Гумбольдта и с Академией наук ГДР.

Технические средства ЛС ROLANET1. Сеть (рис. 1) состоит из следующих частей:

коаксиального кабеля с нагрузочными резисторами в качестве среды передачи;

транссивера (или блока подключения) для пассивной связи станции со средой передачи данных;

кабеля транссивера для соединения транссивера с подключенным ЛС устройством;

контроллера ЛС, называемого также интерфейсом сети;

подключенных к сети устройств (станций) — максимально 100.

Контроллеры локальной сети (КЛС) для большинства устройств выполнены в виде ТЭЗов. КЛС непосредственно размещены в устройствах. Перед передачей информационные сигналы кодируются путем модуляции базовой частоты. В результате созда-

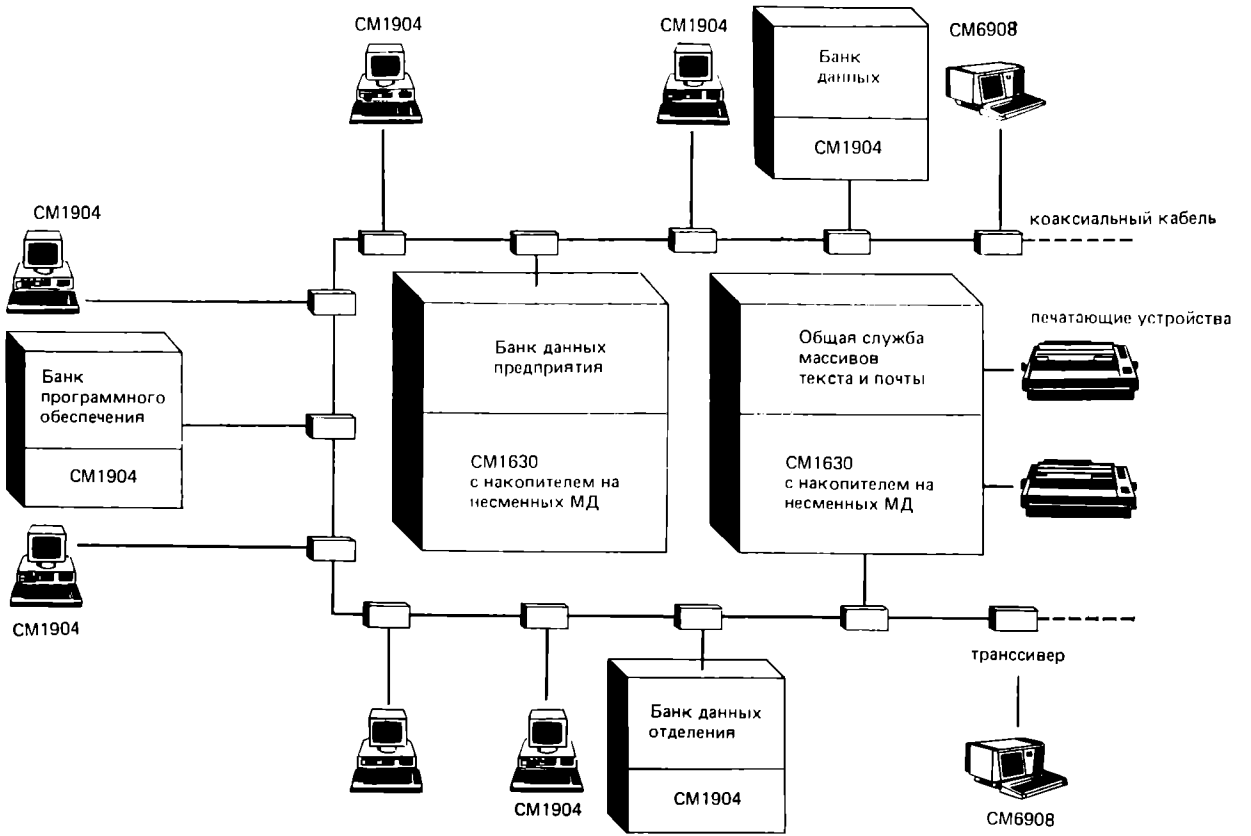


Рис. 1. Пример конфигурации локальной сети на основе средств ROLANET1 и пакетов программ SCPNET, NETSERV

ется поток сигналов, позволяющий выполнять в приемнике восстановление такта и декодирование. В соответствии с международным стандартом применяется манчестерское кодирование. В качестве коаксиального кабеля используется обычный 75-омный кабель. Максимальная длина коаксиального кабеля 1000 м, на концах кабеля ставятся нагрузочные резисторы в 75 Ом. В качестве кабеля транссивера используется обычный телефонный кабель. Максимальная длина кабеля транссивера — 50 м. По одной паре проводов проходят передаваемые и принимаемые данные, а также происходит наложение данных. Четвертая пара проводов служит для электропитания транссивера. Для идентификации канала передачи необходим соответствующий метод доступа. На основе стандартов ИСО 8802.2 «Common logical link control protocol» и ИСО 8802.3 «CSMA/CO Access method» для локальной сети «Роботрон» был выбран метод CSMA (Carrier Sense Multiple Access with Collision Detection). Этот метод получил международное распространение, в особенности для решения задач САПР и систем автоматизации конторских работ, благодаря своей надежности и простоте реализации.

Транссивер TCR (Robotron K 8601) выполняет функции пассивной связи с каналом передачи (рис. 2), связи между устройствами с безопасным сверхнизким и низким напряжением, распознавание наложений (конфликтов) данных. Питание транссивера осуществляется через кабель транссивера от блока электропитания станции ЛС. В транссивере поданное напряжение в 12 В преобразуется в напряжение 5 В для логической части транссивера. Основной составной частью приемной схемы является разностный усилитель с источником стабильного тока. Добавочный преобразователь полного сопротивления обеспечивает низкое вносимое затухание транссивера. Разделительный конденсатор осуществляет гальваническую развязку коаксиального кабеля от транссивера.

Информационные каналы передаваемых данных, принимаемых данных и сигнализации наложений данных кабеля транссивера гальванически отделены транслятором от логической части транссивера. Конструкция трансляторов информационных каналов и трансвертора соответствует условиям безопасности по СТ СЭВ 3743—82. Таким образом, возможно соединение в одну сеть устройств с безопасным низким напряжением, например CM1904 (Robotron PC 1715) и CM1910 (Robotron A 7100), и устройств с низким напряжением, например CM1630 (Robotron K 1600).

Для распознавания конфликтов в транссивере собственный передаваемый сигнал сравнивается с сигналом, принятым через схему приема от коаксиального кабеля. В случае наложения данных в коаксиальном кабеле происходит искажение передаваемого сигнала наложением сигнала станции-партнера. При сравнении передаваемого и принимаемого сигналов такое искажение распознается и в контроллере ЛС и передается прямоугольный сигнал.

Транссивер находится в отдельном корпусе с размерами 220× ×120×40 мм. Кабель подключается к транссиверу при помощи

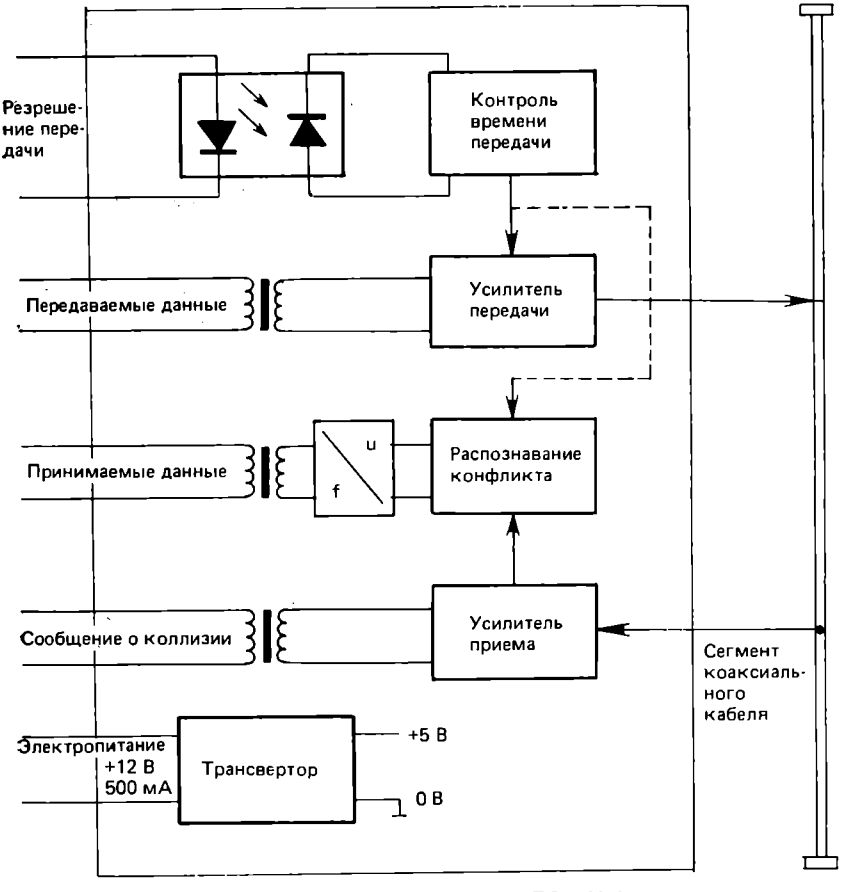


Рис. 2. Блок-схема транссивера TCR К 8601

15-полюсного разъема типа Кэнон. Коаксиальный кабель подключается к транссиверу при помощи высокочастотного разъема.

Блок-схема контроллера LNC1 для всех вычислительных систем с системной шиной СМ1626 представлена на рис. 3. Контроллер подсоединяется непосредственно к внутренней шине главной ЭВМ. Интерфейс контроллера со стороны станции состоит из командного интерфейса (схема PIO) и интерфейса передачи данных (памяти с двумя портами). Через схему PIO осуществляется обмен командами для возврата контроллера в исходное состояние, включение и отключение памяти с двумя портами (DPM) и обмен информацией о состоянии. В перепрограммируемом постоянном запоминающем устройстве (ППЗУ) хранится аппаратно-программное обеспечение передачи и приема данных по описанному в ИСО 8802.3 методу CSMA/CD. Запоминающее устройство с произвольным доступом служит в качестве оперативной памяти (стек, оперативные

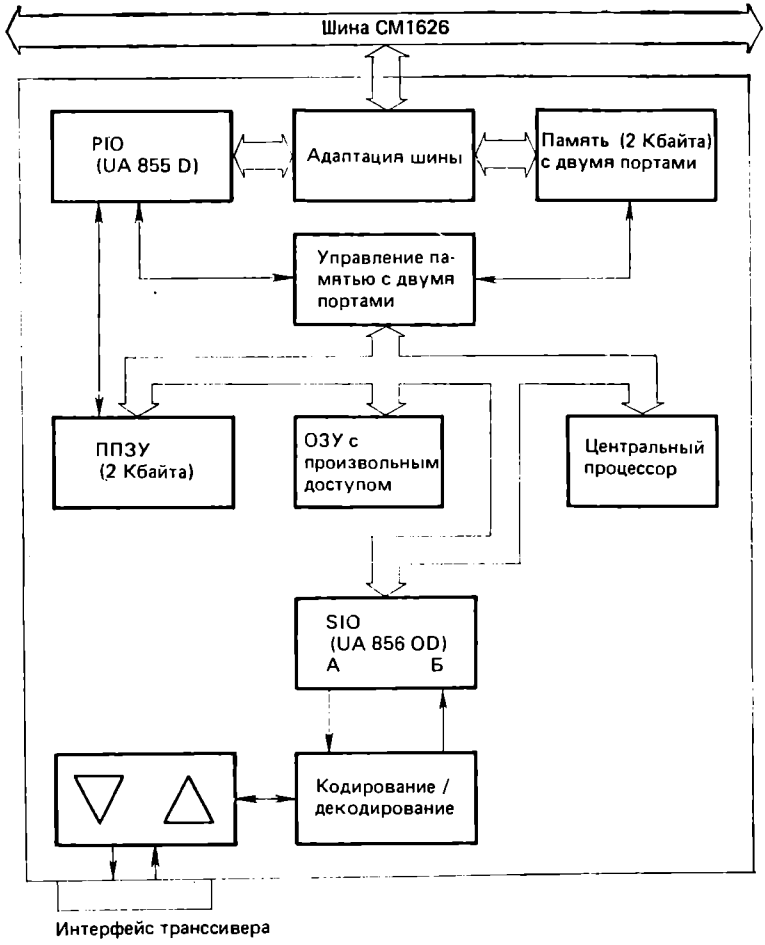


Рис. 3. Блок-схема контроллера локальной сети ROLANET1 LNC-1520

ячейки). SIO выполняет параллельно-последовательное преобразование данных и обеспечивает правильность передачи данных с помощью циклического контроля по избыточному коду. Драйверные и приемные каскады, расположенные за схемой кодирования-декодирования, действуют совместно с соответствующими противовключенными каскадами транссивера.

ЗУ с двумя портами разбито на 6 приемных и один передаточный буферы. Контроллер в режиме опроса или прерываний может управляться станцией локальной сети. Структура передаваемых фреймов определяется схемой U 856. По аналогии с контроллером ЛС LNC1-1520 контроллеры рассчитаны на подключение нескольких устройств к сети ROLANET1. Разница состоит только в разме-

рах печатных плат и в конкретной логике для подключения к шине соответствующей станции.

Ниже приведены контроллеры ЛС LNC1 и станции ЛС, подключаемые к сети ROLANET1.

Контроллеры ЛС	Локальные станции
LNC1-1715	СМ1904
LNC1-1520	СМ6908 (ВС А 5120) СМ6907 (А 5130), другие устройства системы микроЭВМ СМ1626
LNC1-7100	ППЭВМ СМ1910 (А 7100)
LNC1-1600	Все варианты микроЭВМ СМ 1630 (К 1630)
LNC1-1834	ППЭВМ ЕС1834

Подключение ЭВМ типа ЕС. Для подключения ЭВМ типа ЕС к сети, построенной на основе микроЭВМ СМ1626, было разработано специальное устройство. Оно подключается непосредственно к селекторному каналу ЭВМ Единой системы. Для управления используется метод доступа GAM.

ЭВМ типа ЕС рассматривает данное устройство как графический дисплейный терминал ЕС7066. Эмуляция графического терминала реализована в отдельной микроЭВМ. Вторая микроЭВМ управляет контроллером ЛС. Обе ЭВМ сопряжены друг с другом и размещены в одном корпусе.

Программное обеспечение ЛС ROLANET1. При эксплуатации локальных сетей особое значение имеет производительность применяемого базового и пользовательского программного обеспечения. Но самое важное для пользователя — это предоставляемые программным обеспечением услуги (метод доступа и среда передачи, которая используется для реализации обмена данными, для пользователя играют меньшую роль). Особое значение в настоящее время приобретают базовые услуги ЛС, необходимые для обмена данными между подключенными к сети станциями. Это прежде всего услуги, обеспечивающие доступ к удаленным и централизованным массивам данных и реализующие передачу массивов. На международном рынке предлагается ряд пакетов программ ЛС, выполняющих такой обмен данными. Основными характеристиками качества подобных пакетов являются:

максимальная загрузка технических средств передачи данных и малая потребность базовых услуг в ресурсах сети;

высокопроизводительное обслуживание массивов, в особенности малое время доступа даже к большим массивам данных и при большом количестве пользователей — коллективный доступ к массивам при помощи метода «record locking»;

широкое «прозрачное» включение базовых услуг в операционную систему;

переносимость, доступность относительно разных типов ЭВМ и операционных систем;

соблюдение международных стандартов.

По аналогии со стандартизацией открытых сетей дальней связи разрабатываются международные стандарты и для локальных сетей. При этом за основу берется модель архитектуры Международной организации стандартизации, так называемая модель OSI по ИСО 7498 «Information processing systems — Open systems interconnection — Basic reference model». Для каждого из 7 уровней этой модели Международная организация стандартизации и другие организации разработали или уже приняли проекты стандартов. Их подавляющая часть касается программного обеспечения. Проекты стандартов для более высоких уровней (5, 6, 7) эталонной модели взаимосвязи открытых сетей одинаково распространяются на ЛС и СДС. На НП «Роботрон — Проект — Дрезден» разработка базового программного обеспечения ЛС ведется по программе, состоящей из двух этапов. Разграничение этапов происходит в соответствии со сроками и объемами выпуска и характеристиками основных компонентов технических средств (главных ЭВМ, соответствующих контроллеров локальной сети, трансиверов). Первый этап рассчитан на 1984—1988 гг. Основными характеристиками этого этапа являются:

использование 8-разрядных ЭВМ СМ1904, СМ6907, СМ6908 НП «Роботрон» и их усовершенствованных моделей;

адаптация протоколов передачи данных к возможностям 8-разрядных ЭВМ (оперативная память — 64 Кбайта, процессор U880);

реализация более высокопроизводительных рабочих станций и централизованных сетевых услуг (управление массивами, печатью, электронная почта) на базе 16-разрядных ЭВМ;

использование операционных систем SCP (совместима с CP/M V2.2), МООС 1600 и ОМОС 1600 (совместимы с RSX-11MV3.1 или V4.1) и DCP (совместима с MS-DOS V3.2).

Общая архитектура ЛС этого этапа соответствует эталонной модели ИСО. Предусмотренные на уровнях 5 и 6 услуги не были осуществлены (отсутствовала необходимость преобразования данных) из-за недостаточной производительности 8-разрядных ЭВМ. Их реализация предусмотрена на одном из следующих этапов.

Пакеты базовых программ SCPNET и NETSERV для ROLANET1. Это первые программные продукты из набора программ локальной сети ROLANET. На основе описанных технических средств они служат для связи между следующими ЭВМ:

ППЭВМ СМ1904 (РС1715) с операционной системой SCP-1715, бюрокомпьютер СМ6908 (А 5120), СМ6907 (А 5130) с операционной системой SCP-1526;

СМ1630 с операционной системой МООС 1600 или ОМОС 1600.

Пакеты SCPNET и NETSERV поддерживают связь станций сети с обменом сообщениями и массивами. Для этого предусмотрены:

транспортная служба, т. е. надежная передача сообщений между станциями ЛС;

связь системы с оператором для интерактивной передачи оператору сообщений и запросов;

система массивов данных сети для выполнения простых операций над массивами в удаленно расположенных массивах;

система виртуальных дисководов, имитирование 16 виртуальных дисководов на основе системы массивов данных сети.

Пакеты SCPNET и NETSERV представляют собой загружаемые расширения соответствующей операционной системы (в настоящее время SCP-1526, SCP-1715, OMOC 1600 и MOOC 1600). Они построены так, что со стороны пользователя создается тесная связь с базовой операционной системой. По возможности при разработке старались избежать новых, специфических для сетей интерфейсов. Вместо этого функции существующих интерфейсов были расширены в отношении доступа к ресурсам сети, поэтому рассчитанное первоначально только на локальную работу программное обеспечение может использоваться и при обращении к централизованным массивам данных ЛС. Важнейшим результатом такого подхода является наличие системы виртуальных дисководов. В результате значительно сокращается этап подготовки эксплуатации сети у пользователя.

Пакеты NETSERV-1715 (CM1904) или NETSERV-1600 (CM1630) предоставляют в распоряжение пользователя следующие сетевые службы:

обслуживание массивов данных (программа с обслуживанием оглавлений);

программа печатания;

электронная почта;

коммуникация с пользователем.

Самой важной службой является обслуживание массивов данных, потому что оно, с одной стороны, позволяет создавать звенья данных между всеми станциями ЛС, а с другой — представляет собой основу для реализации других служб. Кроме того, разрабатываются службы типа Gateway, устанавливающие связи через общественную сеть с другими ЛС.

Варианты NETSERV-1600 и NETSERV-1715 обладают различными функциями, но последний совместим «снизу вверх» с первым. Пакет программ NETSERV-1715 ограничен по объему функций прежде всего из-за малой производительности операционной системы и степени оснащения дисководами, поэтому он рассчитан в первую очередь для обучения пользователей, занимающихся первый раз локальными сетями, и служит базой для разработки специфических прикладных решений. При создании более крупных проектов следует ориентироваться на NETSERV-1600 или на разрабатываемые пакеты программ для 16-разрядных ППЭВМ (например, ЕС1834), причем при реализации подпроектов дополнительно можно применять пакет NETSERV-1715.

Пакеты SCPNET и NETSERV позволяют разрабатывать разные структуры системы. Распределение функций (на рабочие станции и выполняющие службы станции) в локальной сети может быть без особого труда приспособлено к условиям конкретного применения, причем пользователь может изменять структуру в соответ-

ствии со своими техническими средствами. На основе базового программного обеспечения можно внедрять системы, состоящие максимально из 254 станций. В зависимости от применения количество подключаемых станций колеблется от 30 до 50.

Функции и характеристики пакетов программ SCPNET и NETSERV. Компоненты пакетов программ SCPNET и NETSERV можно разделить на резидентные для основной памяти (они допол-

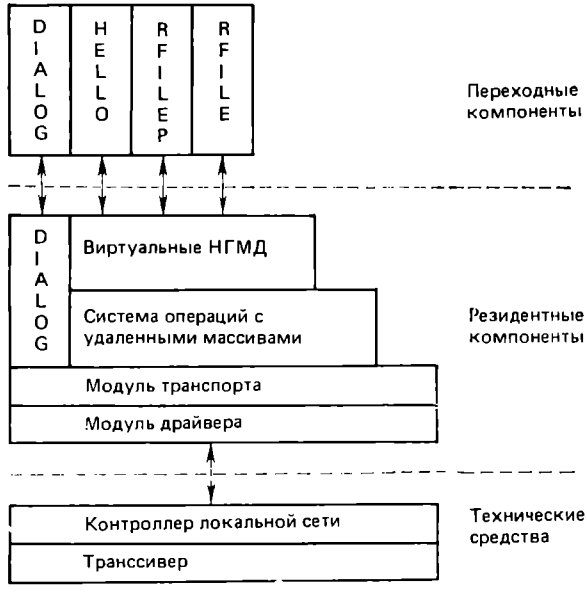


Рис. 4. Компоненты пакетов программ SCPNET

няют операционную систему), реализующие основные функции сети, и на так называемые переходные компоненты, загружаемые в случае необходимости из внешней памяти. При этом наличие резидентных компонентов является условием выполнения переходных.

Резидентные компоненты выполнены в виде программных модулей, загружаемых дополнительно к соответствующей операционной системе. Таким образом, резидентные модули могут быть получены независимо от самой операционной системы. Они занимают приблизительно 8 Кбайт оперативной памяти. На рис. 4 показаны структура резидентной части пакета SCPNET, так называемого программного обеспечения сети, а также возможности доступа пользователя. Область основной памяти, оставшаяся незанятой после загрузки резидентных компонентов SCPNET (для применения пользователем), позволяет одновременно использовать любые стандартные прикладные программы (например, REDABAS/M8 [1] и TEXT 30 [2]). С помощью виртуальной системы, расположенной на НГМД, программы имеют доступ к массивам данных, хранящимся в других терминалах. Дополнительно к физически существующим

внешним накопителям (как правило, НГМД) этот компонент реализует еще не более 16 виртуальных НГМД. Массивы данных, доступ к которым обеспечивается с помощью этих виртуальных накопителей, реально записаны на носителях, принадлежащих удаленным станциям локальной сети. Соответствующие идентификаторы указываются в базе данных, которая генерируется в соответствии с требованиями пользователя и динамически модифицируется.

В состав переходных компонентов входят следующие макрокоманды: SCPNETx — инициализация работы сети, RFILEP — вызов режима сервисной программы простой передачи данных, HELLO — начало обращения пользователя, BYE — конец обращения пользователя, NWSTAT — определение и индикация связей в сети, DIALOG — связь с оператором, RFILE — программа манипуляции массивами данных, RABO — закрытие всех массивов данных и сброс виртуальных НГМД. Все переходные компоненты реализованы в виде отдельных законченных программ. Каждому компоненту, вызываемому по имени, соответствует файл программ с таким же именем.

Литература

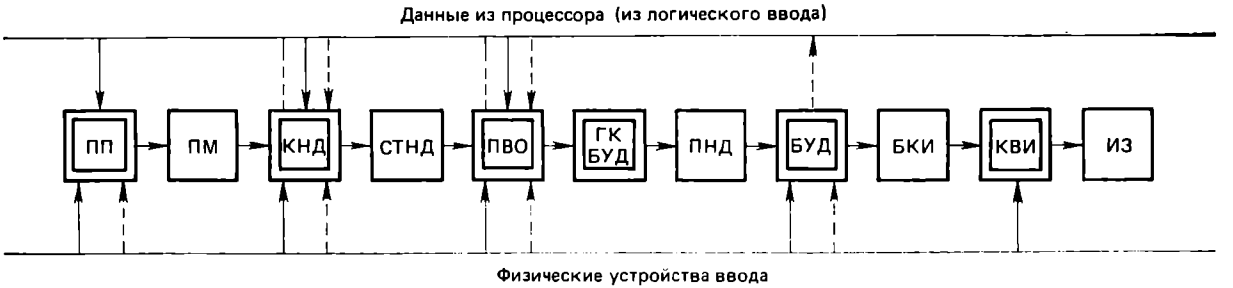
1. Хайнеманн Р., Хемпель Y. REDABAS — реляционная система управления базами данных для персональной ЭВМ// В кн.: Вычислительная техника соц. стран.— Вып. 20.— М.: Финансы и статистика, 1986.— С. 81—86.
2. Heydenbluth. Das Textprogramm des robotron 1715//Neue Technik im Büro.— 1986.— H. 1.— № 30.— S. 11.

УДК 681.326

ИНТЕРАКТИВНЫЕ ГРАФИЧЕСКИЕ СИСТЕМЫ ИГС-2 И ИГС-3 ПРОИЗВОДСТВА ЧССР

В. КОПЕЦКИЙ, канд. техн. наук (ЧССР)

Машинной графике как одному из эффективных средств совершенствования процессов проектирования объектов техники сейчас уделяется особое внимание. По программе СМ ЭВМ в ЧССР были разработаны и внедрены в производство графические станции (ГС) и интерактивные графические системы (ИГС) нескольких поколений. Для объяснения различий между ГС разных поколений производства ЧССР на рис. 1 приведена функциональная схема обработки графической информации в обобщенной системе. На следующих рисунках представлены способы реализации отдельных логических процессоров на конкретной ГС (от способа реализации графических процессоров в графической системе зависит мощность системы).




Логический процессор


Представление модели объекта

Данные модификации процесса ввода

Данные

Рис. 1. Функциональная схема процесса обработки графической информации:

БУД — блок управления дисплеем; ПП — прикладная программа; ГК БУД — генератор блока управления дисплеем; КВИ — контроллер вывода изображения; КНД — компилятор набора дисплея; ПВО — процессор видеооперации; БКИ — битовая карта изображения; ИЗ — изображение; ПМ — прикладная модель; ПНД — последовательный набор дисплея; СТНД — структурный набор дисплея.

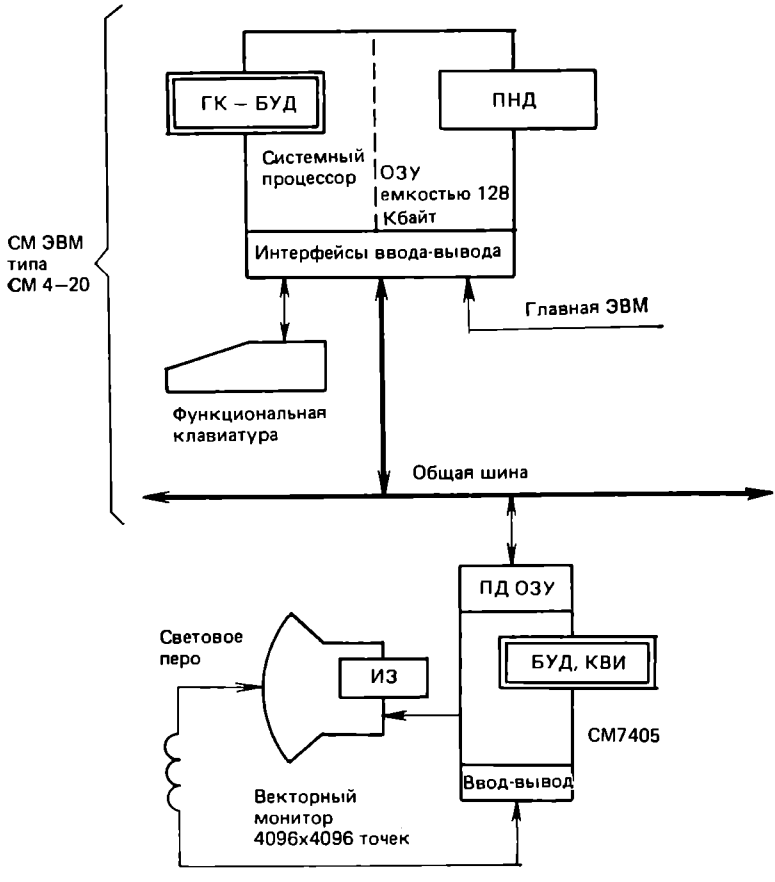


Рис. 2. Архитектура графической станции ГС-1:
ПД ОЗУ — память данных ОЗУ

Графическая станция первого поколения (ГС-1), представленная на рис. 2, имеет следующие характерные черты:

все логические процессоры (кроме БУД — блока управления дисплеем) реализуются программным способом в центральной ЭВМ (например, в ЭВМ СМ 4—20). К логическим процессорам относятся: прикладная программа (ПП), компилятор набора дисплея (КНД), процессор видеоопераций (ПВО), генератор команд блока управления дисплеем (ГК БУД);

все уровни представления модели объекта размещаются и хранятся только в памяти центральной ЭВМ;

в блоке управления дисплеем предусмотрен прямой доступ к оперативной памяти центральной ЭВМ;

устройство для взаимодействия пользователя с системой — функциональная клавиатура (селектор), применяемая в качестве центральной ЭВМ, и световое перо, связанное с БУД;

векторный монитор располагает адресным пространством размером 4096×4096 точек.

Характерные особенности графической станции второго поколения — ГС-2 (рис. 3):

размещение памяти битовой карты (БК) в блоке управления дисплеем (БУД). Битовая карта служит для восстановления изображения на растровом дисплее;

более мощный процессор в БУД;

расширение набора графических команд БУД (управление памятью пикселей, создание окон, сегментация экрана и т. д.);

использование локатора интерактивного управляющего устройства (вместо светового пера), которое подключено прямо к БУД;

растровый монитор с разрешающей способностью 512×512 пикселей.

Все остальные черты архитектуры ГС-2 (реализация основных графических функций в системной ЭВМ, прямой доступ к памяти из БУД, использование селектора и т. д.) такие же, как и у ГС-1.

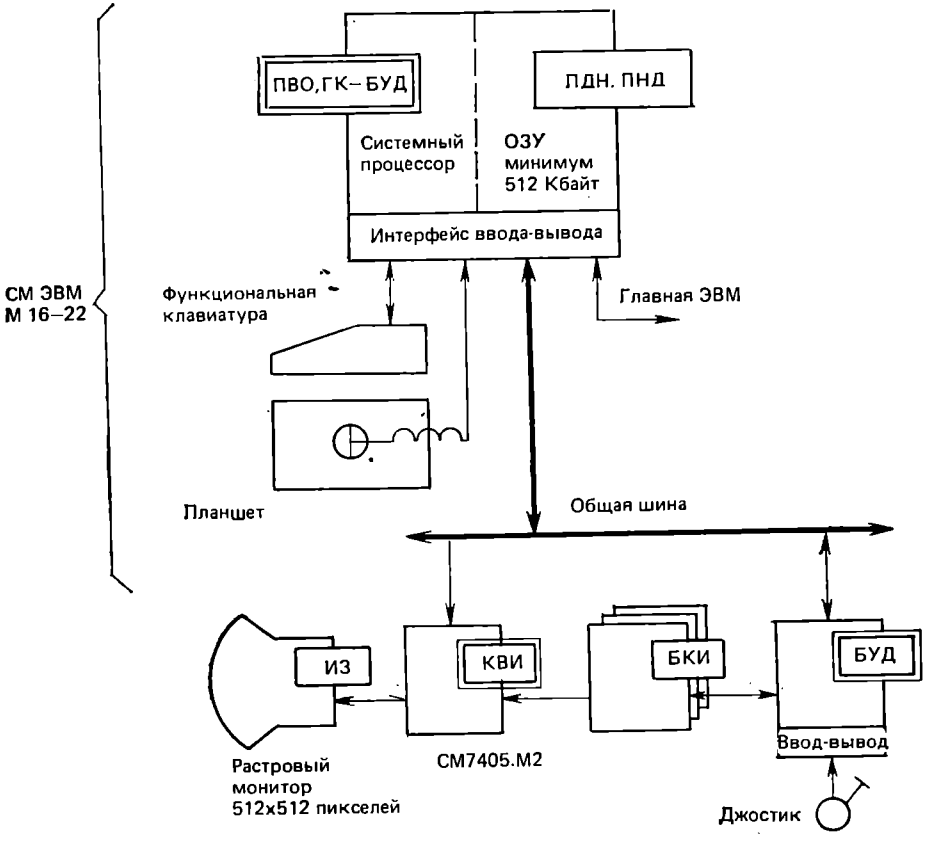


Рис. 3. Архитектура графической станции ГС-2

Архитектура ГС-3 (рис. 4) разработана с учетом принципов мультипроцессорной системы с магистральной архитектурой. Это связано с необходимостью реализации функций логических процессоров функциональной модели (см. рис. 1) автономными универсальными или специализированными вычислительными средствами.

Особенности архитектуры ГС-3:

мощный универсальный микропроцессор, который реализует функции логического процессора для связи с ЭВМ высшего уровня, логического системного процессора ГС и логического процессора операций изображений;

встроенное программное обеспечение для автоматического тестирования ГС, графические функции, связь с ЭВМ высшего уровня и связь с графическими периферийными устройствами;

большая оперативная память для размещения структурного набора дисплея, псевдодисплейного набора, последовательного набора дисплея, сегментов набора дисплея;

специализированные процессоры для работы с числами с плавающей точкой, реализующие функции копроцессора процессора команд;

большой ассортимент интерфейсов: интерфейс внешних ЗУ — ИВП (гибкие диски — ГД, ЗУ типа «винчестер» — ПВ); интерфейс для минимум трех устройств ввода-вывода последовательного типа; интерфейс прямого доступа к памяти ЭВМ высшего уровня; специальные интерфейсы для цветного графического вывода на цветной регистратор и цветную графическую печать.

В случае работы ГС-3 в локальном режиме логические процессы ПП и КНД выполняются в системном процессоре графической станции. При работе в локальном режиме составной частью ГС-3 являются также модули ИВП, ГД и ПВ. При работе ГС-3 в качестве «интеллектуального» графического терминала 32-разрядной ЭВМ высшего уровня из графической системы исключаются ПП, КНД, ИВП, ГД, ПВ и др.

Технические средства графической станции ГС-2: системный процессор М16—22; оперативная память емкостью 0,5 до 2 Мбайт; буквенно-цифровой дисплей СМ7202.М1—А; дисковый кассетный блок СМ5400 (СМ5410); блок гибкого диска СМ5626; графический растровый блок СМ7405.М2 с рычаговым управляющим устройством (диапазон изображения 512×512 точек); мозаичное печатающее устройство с графическим режимом С 2111-; дополнительные устройства — цветной графопостроитель планшетного типа СМ6413; рулонный графопостроитель 930; устройство ввода графической информации СМ6412.

Программное обеспечение графической станции ГС-2: операционная система ДОС РВ версия 3; основное графическое программное обеспечение (ОГПО), интерактивная графическая библиотека (ИГБ), ОГПО, разработанное на базе стандарта ГКС (ДГКС).

Графическая станция ГС-2 может работать либо автономно, либо быть составной частью интерактивной графической системы (ИГС-2). Структура ИГС-2 приведена на рис. 5.

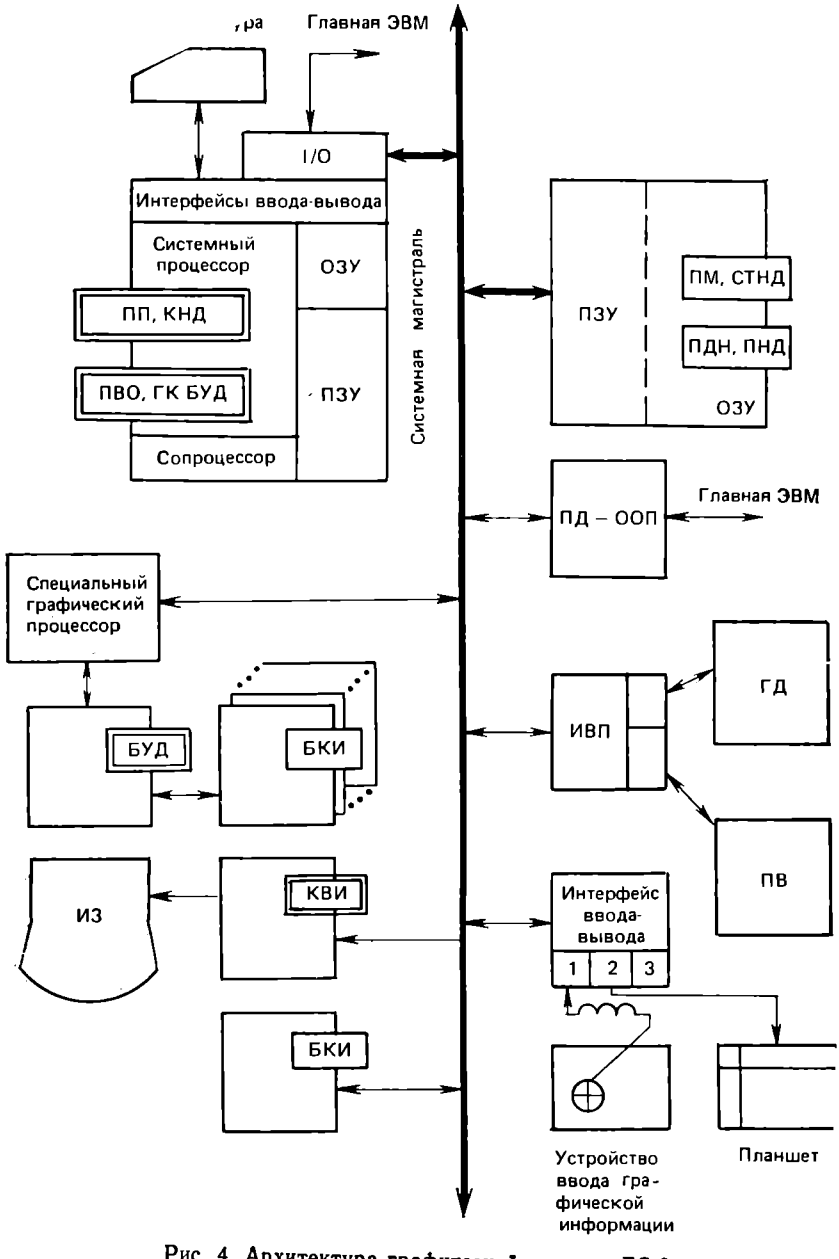


Рис. 4. Архитектура графической станции ГС-3

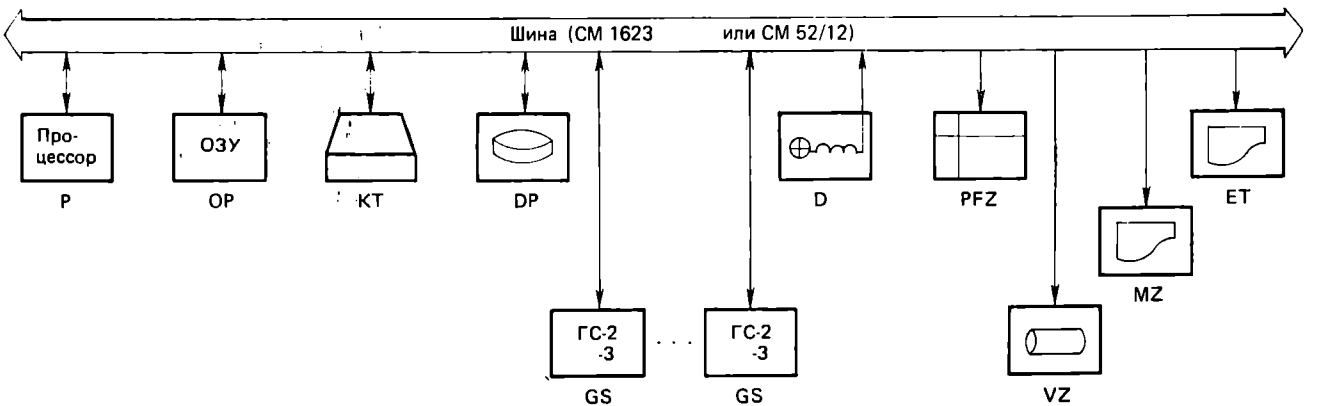


Рис. 5. Структура интерактивных графических систем ИГС-2 и ИГС-3:

Р — процессор М16—23 для ИГС-2 и СМ 52/12 для ИГС-3; ОР — оперативная память; КТ — консольный терминал; DP — дисковая память; GS — графическая станция; D — планшет; PFZ — цветной графопостроитель; VZ — рулонный графопостроитель; MZ, ET — печатающие устройства с графическим режимом

Вычислительная система М 16—23 в ИГС-2 управляется операционной системой ДОСРВ версии 3. Связь между системами М 16—23 и М 16—22 в ИГС-2 обеспечена системой управления вычислительными сетями СИРПОС.

Графическая станция ГС-3, использующая 16-разрядные микропроцессоры, дает возможность изображать на мониторе 1280×1024 точек. Он может также работать в автономном режиме или быть составной частью интерактивной графической системы ИГС-3.

В программное обеспечение интерактивной графической системы ИГС-3 входят: операционная система ВОС с модулями управления графическими периферийными устройствами; основное графическое программное обеспечение на базе стандарта ГКС уровня 2б и СУБД реляционного типа; системное программное обеспечение графической станции ГС-3.

Дальнейшая разработка ГС и ИГС в программе СМ ЭВМ в ЧССР будет ориентироваться на повышение мощности этих систем путем реализации логических процессоров техническими средствами с ориентацией на стандарты ГКС, ЦГМ и ЦГИ.

УДК 681.324

ТЕСТЕР ПЕЧАТНЫХ ПЛАТ Р 3040

Х. ЗИБЕРТ, доктор (ГДР),
Х.-Д. ПРАНГЕ, доктор (ГДР)

Тестер печатных плат Р 3040 представляет собой новый автомат для контроля монтажа и элементов цифровых и аналоговых электронных узлов. Он позволяет выполнять внутрисхемный контроль дискретных и интегральных, аналоговых и цифровых элементов схем, а также обнаруживать короткие замыкания и обрывы проводников на печатных платах после монтажа и пайки в соответствии с возможностями контактирующего устройства и диапазонами измерения тестера. Кроме того, электроника измерительных каналов тестера дает возможность выполнять частичный цифровой функциональный контроль с частотой стимулирующих сигналов и опроса измерительных точек 200 кГц. Подготовка текстов программ контроля осуществляется вручную на языке высокого уровня. Запоминание правильных реакций проверяемого образца происходит в режиме LERN с применением исправного образца.

Аналоговый функциональный контроль возможен благодаря встроенному интерфейсу ИИС-2 в сочетании с 9 двуполюсными входами для измерительных приборов. Программирование этих устройств обеспечивается операторами тестового языка.

Основной частью тестера (рис. 1) является высокоэффективная мини-ЭВМ типа «Электроника 60—1» (16 бит, 256 Кбайт ОЗУ)

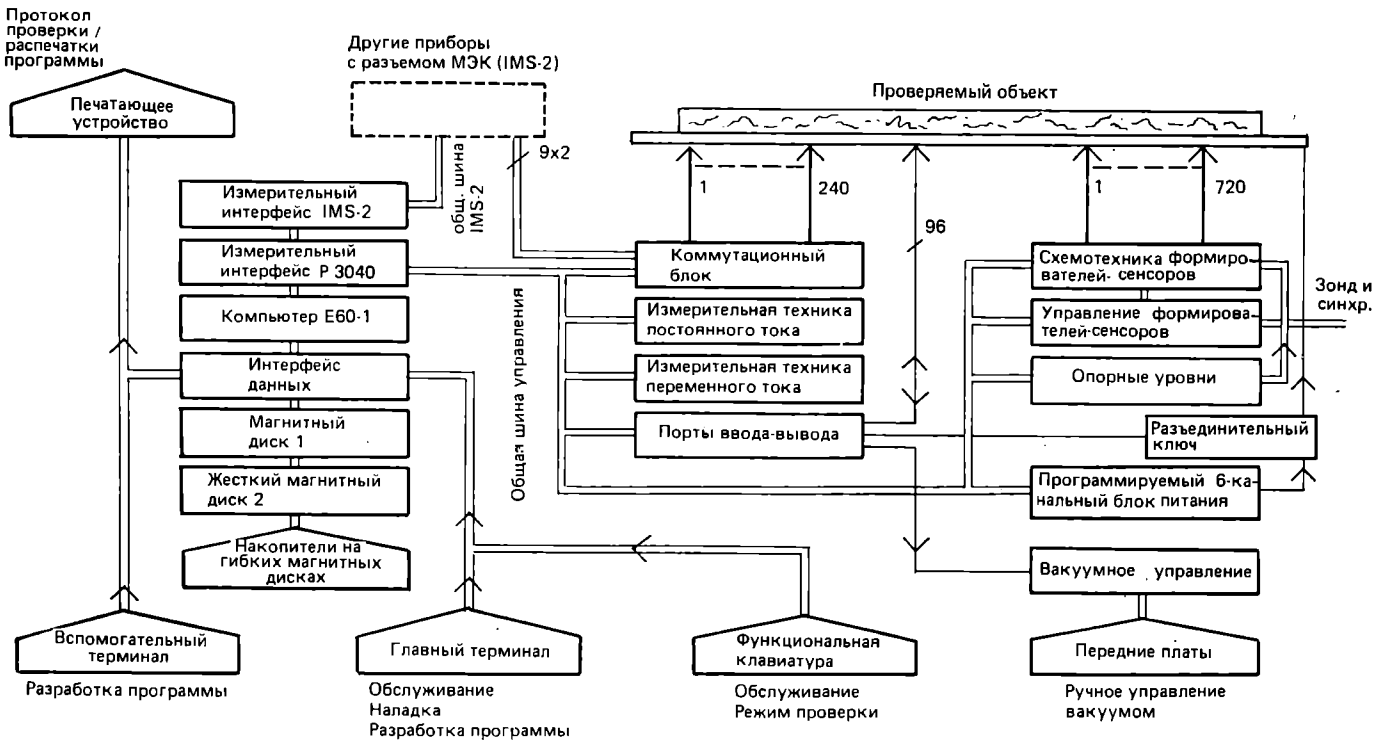


Рис. 1. Автомат для испытания печатных плат P 3040

в сочетании с двумя НМД (35 Мбайт каждый). Спектр периферийных устройств дополняют два накопителя на гибких магнитных дисках (200 мм, двойная плотность записи), два программируемых терминала, используемых в качестве пульта оператора и пульта программиста (К 8911, модифицированный вариант), и печатающее устройство (К 6313). Для управления основными функциями в режиме тестирования предусмотрен дополнительный пульт оператора, расположенный вблизи контактирующего устройства.

Интерфейс СИ ИИС-2 (контроллер С4, источник информации с расширением TE5, приемник информации с расширением LE3, дистанционно-местное управление RL1, запуск устройства DT, параллельный опрос PP2, установка устройств в исходное состояние DC1) позволяет управлять внешними устройствами, оснащенными интерфейсом ИИС-2. Аналоговый измерительный комплекс позволяет производить измерения в 240 аналоговых измерительных точках проверяемого объекта. Цифровой измерительный комплекс позволяет подключить 720 измерительных точек. Кроме того, параллельное подключение одного аналогового и одного цифрового измерительного канала к общей измерительной точке дает возможность проводить смешанное тестирование максимально в 240 точках.

Аналоговая измерительная техника. Каждый узел схемы проверяемого объекта соединяется с релейным коммутатором через контактирующее устройство. С помощью программы узел схемы, необходимый для контроля определенного элемента, подключается к общей аналоговой шине, состоящей из четырех пар измерительных линий. Электрическая изоляция элемента схемы достигается подключением дополнительных узлов к общей линии (потенциал 0). За счет разделения общей шины на сигнальную и сенсорную линии эффективность электрической изоляции элементов схемы по сравнению с ранее использованными методами возросла приблизительно на один порядок.

Величина сопротивления и импеданса может измеряться в следующих диапазонах:

омическое сопротивление:	от 21 Ом до 6,5 МОм ±1% от 0,8 Ом до 100 МОм ±10%
реактивные элементы:	частота 0,1; 1; 10 кГц напряжение 0,2; 1 В
пределы измерения емкости:	от 15 до 200 пФ ±10% от 200 пФ до 100 мкФ ±1,5% от 100 мкФ до 2 мкФ±10%
пределы измерения индуктивности:	от 30 мкГ до 500 мкГ ±10% от 500 мкГ до 130 Г ±1,5% от 130 Г до 2,4 кГ±10%

Измеритель импеданса на переменном токе измеряет активные и реактивные компоненты тока через этот импеданс и напряжение, падающее на него. Из полученных значений для эквивалентной параллельной или последовательной схемы (по выбору) рассчитывается величина активного и реактивного компонента.

Кроме того, аналоговая измерительная техника постоянного тока используется для измерения:

коэффициента усиления транзисторного каскада;
крутизны характеристики полевого транзистора;
прямого напряжения диодов;
параметров тиристоров и тринисторов;
параметров АЦП и ЦАП;
параметров схем сравнения;
параметров операционных усилителей.

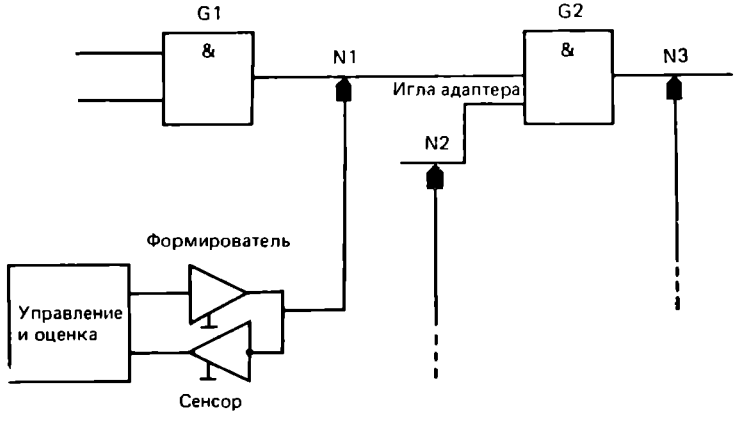


Рис. 2. Принцип внутрисхемного цифрового теста

Помимо тестирования элементов схемы проверяемого объекта с помощью измерительной техники постоянного тока могут производиться также измерения напряжений в диапазоне 0,25 В...32 В (7 поддиапазонов) и токов в диапазоне 10 мкА...160 мА (8 поддиапазонов) с разрешением 14 бит.

Прецизионные источники напряжения и тока служат для формирования программируемых сигналов постоянного тока (напряжение устанавливается в пределах 0,25...16 В, 4 диапазона; ток — в пределах 0,125 мА...128 мА, 6 диапазонов) с переключаемой полярностью.

Цифровая измерительная техника. Принцип внутрисхемного цифрового тестирования наглядно представлен на рис. 2. На входы проверяемой цифровой функциональной группы подается тестирующая последовательность сигналов через иглы N1 и N2. Одновременно производится контроль выхода этой группы G2 через иглу N3, соединенную с быстрым компаратором. Сравнение измеренного уровня с заданным дает информацию о наличии неисправности.

Искусственное создание определенных входных уровней на G2 является причиной электрической изоляции этой группы. При этом выход предыдущей функциональной группы G1 должен быть

кратковременно переписан. Это и дало методу проверки название «Nodeforcing» или «Backdriving». Автоматическое ограничение времени воздействия повышенного тока от формирователей стимулирующих сигналов исключает повреждение выходных цепей предыдущих микросхем.

Цифровой измерительный комплекс (рис. 3) состоит из контроллера, 45 печатных плат с электроникой формирователей и сенсоров, каждая из которых образует 16 цифровых каналов, и платы формирования опорных напряжений, которая вырабатывает

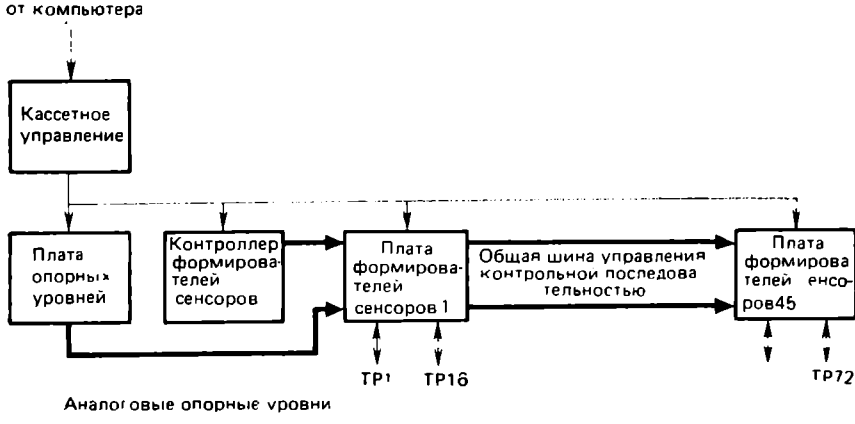


Рис. 3. Упрощенная блок-схема электроники формирователей-сенсоров

8 программируемых контрольных уровней для двух групп логических уровней. Контроллер служит для управления обменом данными между ЭВМ и ЗУ на платах формирователей-сенсоров и для управления автономным выполнением теста (BURST — MODE) после инициализации, выполняемой ЭВМ.

На рис. 4 представлена внутренняя схема каналов формирователей-сенсоров. Измерительные точки, участвующие в одном тестировании, подключаются к соответствующим каналам с помощью реле. Формирователи могут быть отключены по программе. В активном состоянии они формируют синхронно с тактом уровни high и low заданной группы логических уровней при максимальном токе около 300 мА, текущем через входы проверяемого элемента схемы. Соответствующая последовательность логических уровней для тестирования загружается во вспомогательные ЗУ емкостью 1 Кбит для каждого проверяемого выхода микросхемы.

Сенсоры представляют собой двойные компараторы, пороговые напряжения которых для уровней high и low соответствуют одной из групп логических уровней. Компараторы следят при подключенном формирователе за уровнем напряжения или за реакцией проверяемого объекта. Устройство сравнения истинного и регистрируемого результата выдает сигнал ошибки при обнаружении отклоне-

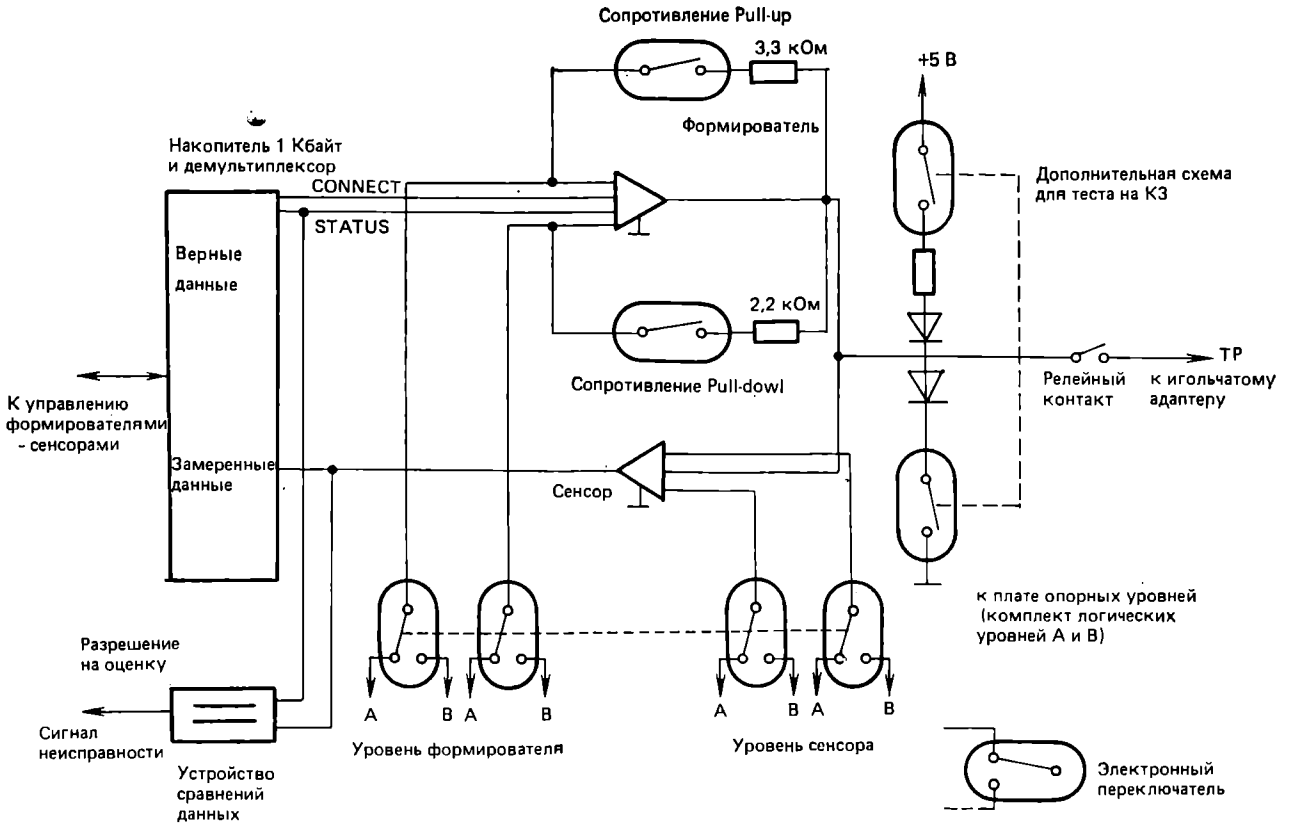


Рис. 4. Схематичное устройство канала формирователя-сенсора

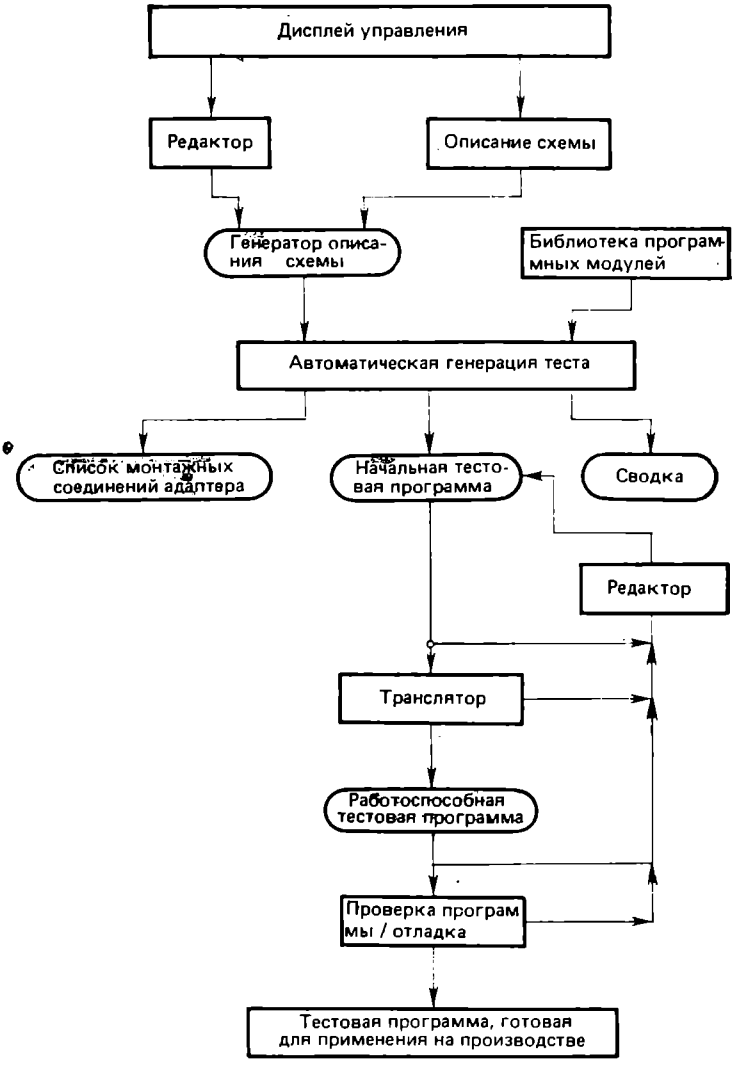


Рис. 5. Технология программирования Р 3040 (схема)

ния, который записывается в ЗУ. Устройство сравнения может быть отключено, чтобы предотвратить ложное срабатывание, которое может быть вызвано, например, инициализацией проверяемого объекта. После автоматической отработки тестовой последовательности ЭВМ считывает результаты сравнения и выводит сообщения об обнаруженных ошибках на печатающее устройство. Для анализа причины ошибки все наблюдавшиеся на проверяемой схеме уровни и их изменение во времени могут быть представлены на экране пульта оператора. При этом отклонения показываются

инверсными участками диаграммы (темные линии на светлом фоне). К каждой контрольной точке по программе могут подключаться резисторы, соединяющие проверяемый узел схемы с шиной питания или с общим проводом. Это подключение может производиться перед циклом тестирования, а также при его выполнении.

Техническая характеристика цифрового блока тестера Р 3040

Количество измерительных каналов	720
Емкость вспомогательного ОЗУ	1024 бит на канал
Уровни формирователей-сенсоров	—8 ... +16 В
Ток формирователя	макс. 300 мА
Длина тестирующей последовательности	макс. 500 шагов
Частота стимулирующего сигнала	200 кГц

Программное обеспечение. Обширное программное обеспечение позволяет составлять тексты тестирующих программ с помощью библиотеки. Тестовые программы могут составляться автоматически в режиме «автоматическое генерирование теста» или вручную с помощью мощных программ-редакторов.

Для составления тестирующих программ используется проблемно-ориентированный язык, английская мнемоника которого легко запоминается. Для работ с программным обеспечением в распоряжение пользователя предоставляется множество вспомогательных меню и информационных текстов.

На рис. 5 представлены отдельные этапы составления тестирующей программы. Исходным материалом служит электрическая схема проверяемого объекта, по узлам которой распределяются аналоговые и цифровые измерительные каналы. С помощью генератора описания схемы в диалоговом режиме составляется соответствующее описание. Тестирующая система получает при этом все необходимые данные об элементах схемы: тип, номинал, допуски, сопоставление выводов элемента с узлами схемы и т. д.

После ввода описания по анализу избыточной информации, требуемой при вводе, определяется правильность указанных значений, что обеспечивает безошибочность описания.

Автоматический генератор теста (ATG) анализирует описание схемы и на основе библиотеки программных модулей для отдельных элементов конструирует текст тестирующей программы. Одновременно составляются схема подключения адаптера и отчет об эффективности найденных моделей теста для каждого элемента. С помощью редактора вручную изменяются или дополняются те тесты, для которых автоматический генератор не нашел одной модели. После глубокого синтаксического контроля текст программы с помощью транслятора переводится в объектный код.

Физическая проверка программы и, при необходимости, ее изменение производятся с помощью исправного образца проверяемого объекта. При этом оптимальный вариант теста может записываться на уровне текста исходной программы. Исправленная

тестовая программа может отрабатываться без повторного использования транслятора.

Эффективность цифрового внутрисхемного контроля в основном зависит от построения библиотеки программных модулей для цифрового теста. Для каждой микросхемы такой модуль имеет следующую структуру:

HEAD — описание конфигурации выводов проверяемой схемы;

INHIBIT — содержит последовательности, которые устанавливают на выходах микросхемы постоянные уровни, что исключает помехи за счет обратных связей;

DISABLE — содержит стимулирующие последовательности для отключения микросхем от общей шины;

H-FORCE, L-FORCE — содержат последовательности для генерации выходных уровней high и low, применяются для установки микросхем, блокирующих общую шину;

TEST — стимулирующие последовательности, используемые непосредственно для функционального теста микросхем.

Выбор годных для тестирования стимулирующих последовательностей осуществляется генератором тестовых программ.

Другими важными функциями монитора являются:

LIBRARY MANAGEMENT — внесение изменений и расширение библиотек;

BATCH — автономная отработка заранее введенной последовательности директив;

LEARN — запоминание правильных реакций цифровой схемы или ее частей;

PROBING — организация работы с тестовым зондом;

STATISTIC — статистический анализ результатов тестов;

SETUP — определение конфигурации аппаратной части тестера;

HELP — множество информационных текстов для всех функций монитора.

Высокая надежность тестера (среднее время работы между двумя отказами более 400 ч) была достигнута путем тщательного подбора всех его компонентов.

По инструкции пользователь должен еженедельно проводить работы по техническому обслуживанию тестера, для которых предусмотрен большой пакет диагностических программ.

Высокий коэффициент готовности тестера (более 95%) достигается с помощью множества вспомогательных программ, которые позволяют определить неисправную плату. Большой комплект запасных частей, в состав которого входят почти все платы тестера, дает возможность быстро восстановить работоспособность тестера заменой неисправной платы на новую.

Изготовитель тестеров проводит для покупателей курсы подготовки сервисного персонала. Эти специалисты смогут сами отремонтировать отдельные функциональные узлы.

УДК 681.84

НАКОПИТЕЛИ НА МАГНИТНЫХ ДИСКАХ ДЛЯ МАЛЫХ И ПЕРСОНАЛЬНЫХ ЭВМ

Б. ЦОНЕВ, ст. науч. сотр. (НРБ),
Б. ЦЕНКУЛОВ, ст. науч. сотр. (НРБ),
К. МИТЕВ, науч. сотр. (НРБ)

Производство накопителей на магнитных дисках (НМД) — одна из важнейших отраслей современной вычислительной техники. Значительная часть НМД для стран — членов СЭВ поставляется НРБ, которая с 1975 г. выпускает накопители, предназначенные для больших и малых ЭВМ.

В последние годы в связи с развитием малых и персональных профессиональных ЭВМ (ППЭВМ) резко повысились потребности в НМД на гибких сменных и на жестких несменных носителях. Эти накопители характеризуются качественно новыми требованиями к надежности, габаритным размерам, массе и др. Внедренная на предприятиях страны технология производства НМД типа «винчестер» вытеснила НМД на сменных носителях (пакетах и кассетах) и ускорила развитие НМД на несменных носителях с уменьшенным диаметром диска. В период с 1984 по 1986 г. НРБ выпустила серийно ряд новых накопителей, соответствующих требованиям современных ЭВМ:

ЕС5063 емкостью 317,5 Мбайт на несменном носителе с диаметром диска 365 мм для ЕС ЭВМ;

ЕС5300 емкостью 5 Мбайт на несменном носителе с диаметром диска 130 мм для ЕС и СМ ЭВМ;

СМ5508 емкостью 11/10 Мбайт на несменном носителе с диаметром 130 мм для ЕС ЭВМ;

семейство накопителей ЕС5088, ЕС5321, ЕС5323 емкостью 0,125; 0,5 и 1,0 Мбайт на дискете с диаметром гибкого диска 130 мм.

В статье рассматриваются некоторые проблемы, связанные с проектированием накопителей на жестких магнитных дисках для малых и ППЭВМ.

Интерфейсы. Емкость НМД для малых и ППЭВМ удваивается каждые три года. В то же время производительность ЭВМ увеличивается ежегодно на 20%, а необходимость в увеличении емкости внешней памяти нарастает с 40 до 75% за год. Эффективным способом преодоления этой диспропорции является увеличение «интеллектуальности» накопителей.

Приняты 4 уровня стандартизации интерфейсов в вычислительной машине: процессорный, системный, подсистемный и периферийный (рис. 1). Ниже приводятся некоторые наиболее распрост-

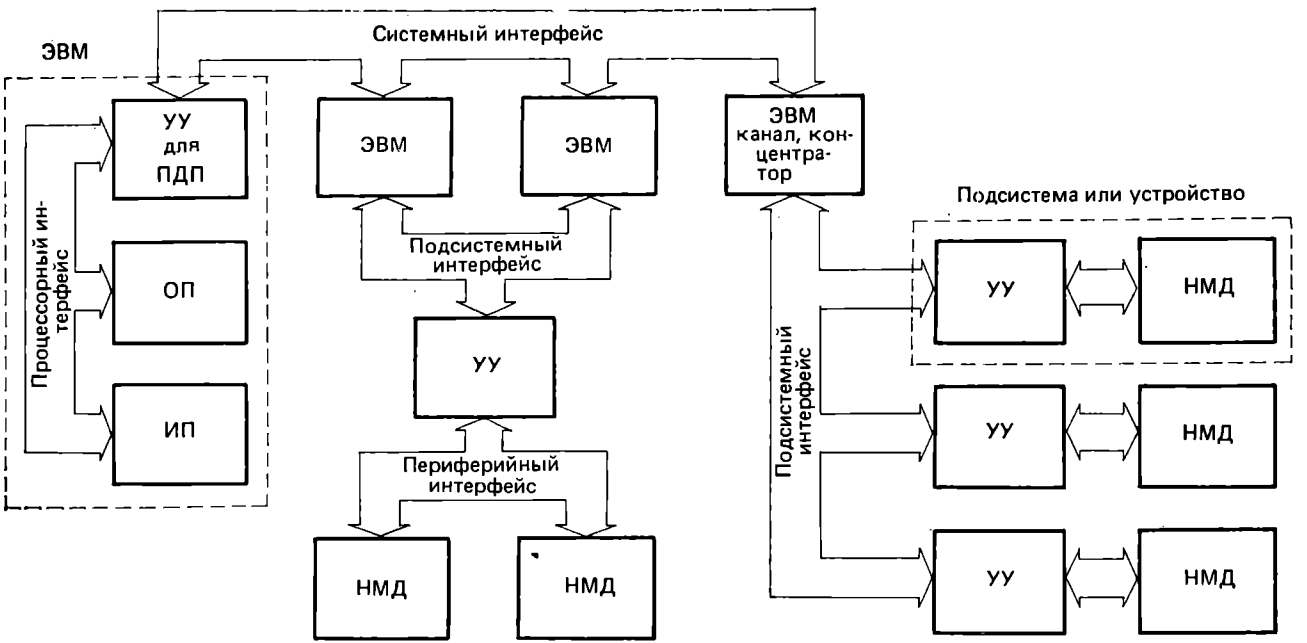


Рис. 1. Интерфейсные уровни в структуре ЭВМ:

ЦП — центральный процессор; ОП — оперативная память; ПДП — прямой доступ к памяти

раненные интерфейсы для малых и ППЭВМ (в скобках указаны интерфейсы, стандартизованные в странах СЭВ).

Интерфейс	Для малых ЭВМ	Для ППЭВМ
Процессорный	СМ 1 SM 1	MULTIBUS VME BUS SCSI
Системный	UNI BUS (ОШ) Q-BUS (МПИ) MASS-BUS (ИМП)	типа RDSX
Периферийный	типа DIABLO (ММ СМ ЕИМ 007-76) SMD (СМД)	ST506 1412 (ИМД-М) SAUGART (ИГМД) STU12HP ESDI

Выбор интерфейса — один из основных вопросов, стоящих перед конструкторами НМД. Интерфейс должен обеспечивать максимальную скорость передачи данных, функциональную гибкость подключения разных типов НМД и других внешних устройств, работу с разными протоколами обмена, простую аппаратную реализацию и др. НРБ принимает активное участие в выборе и стандартизации интерфейсов в рамках СЭВ. Ниже приведены некоторые из используемых в Болгарии интерфейсов.

Интерфейс	НМД
ММ СМ ЭВМ 007-76	СМ5400, СМ5410
СМД	СМ5412
ИГМД	ЕС5074, ЕС5088, ЕС5323, ЕС5321 и др.
ИМДМ	ЕС5300, СМ5508
ИМП	СМ5404, СМ5416

С появлением 32-битовых малых и ППЭВМ требования к НМД, а отсюда и к соответствующим подсистемным интерфейсам повысились. Однако выяснилось, что экономически выгоднее подключать НМД большой производительности к существующим вычислительным системам. Использованные до сих пор интерфейсы не удовлетворяют новым требованиям как со стороны системы, так и внешних устройств. Это обусловливает создание и развитие так называемых «интеллектуальных» интерфейсов.

В процессе повышения интеллектуальности НМД можно выделить несколько этапов:

кодирование и декодирование данных и управление позиционированием головок выполняется НМД;

НМД берет на себя функции форматирования дисковых поверхностей и преобразования получаемых логических адресов в физические;

почти полное высвобождение ЭВМ от функций по обмену данными между разными внешними устройствами в подсистеме.

Первыми разработками в направлении «интеллектуализации» стали НМД СМ5404 и СМ5416. Существует тенденция к переводу

НМД для малых ЭВМ (мини-НМД) на носителях с диаметром 200 мм на интерфейсы типа SDI, IPI, ISI и другие, а НМД для ППЭВМ (микроНМД) на носителях диаметром 89 мм — на интерфейсы типа ESDI и SCSI.

Позиционирование в НМД. Такое позиционирование осуществляется посредством шагового двигателя без датчика положения каретки с головками (открытая сервосистема) или посредством линейного (ротационного) двигателя с датчиком положения (замкнутая сервосистема). Замкнутые сервосистемы являются единст-

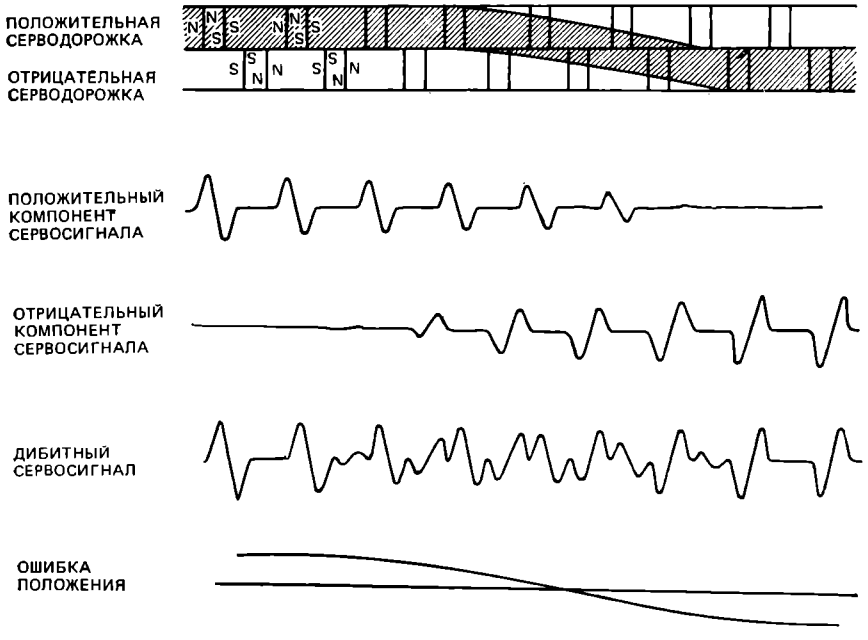


Рис. 2. Дибитный сервосигнал

венным решением для мини-НМД. Их развитие представляет собой смену четырех поколений. Первое поколение характеризуется механической установкой магнитных головок. Во втором используются индуктивные, или оптические, датчики положения, и устанавливается электронная головка. В третьем поколении накопителей одна из дисковых поверхностей (сервоповерхность) содержит закодированную информацию, которая записывается изготовителем и воспроизводится отдельной головкой, так называемой сервоголовкой. К четвертому относятся позиционирующие системы с распределенной сервоинформацией.

Широкое распространение получила «дибитная» конфигурация сервозаписи (рис. 2). Используются и другие конфигурации серво-

сигнала — трехбитные и квадратурные, характерные для современных НМД. Сервоповерхность записывается в трех зонах. Внешняя защитная зона содержит определенное число однотипных серводорожек (например, отрицательных). Сервозона состоит из чередующихся положительных и отрицательных серводорожек, причем посредством кодирования сервозаписи обозначается начало дорожки («индекс»). Ширина сервозоны соответствует ширине зоны для записи данных на поверхности для данных.

Внутренняя защитная зона имеет определенное число однотипных серводорожек (например, положительных).

При технологии типа «винчестер» головки «садятся» и «взлетают» с одной из защитных зон, а их позиционирование управляется сервосистемой. Она представляет собой сложную систему автоматического регулирования с множеством обратных связей различного типа, осуществляющую разные режимы позиционирования: поиск, восстановление, установку, смещение с центра дорожки, вытягивание головок из сервозоны при остановке, аварийное вытягивание при пропадании напряжения питания и др. Сервосистемы этого типа требуют большого объема электроники и обычно используются в мини-НМД большой емкости.

В микроНМД емкостью до 20 Мбайт для позиционирования головок обычно используется открытая сервосистема с шаговым двигателем. Последний поддерживает постоянную скорость, не зависящую (в определенных границах) от нагрузки.

Для управления двигателем смешанного типа используются двуполярные мостовые схемы, которые переключают в определенной последовательности направление тока через каждую из двух обмоток. Вращение вала двигателя осуществляется с помощью 4-шаговой последовательности. Переключением соответствующих цепей обычно управляет микропроцессорная система.

В микроНМД применяется вращение двигателя и с полушаговой (8-шаговой) последовательностью, при помощи которой получается перенос на полшага. Задерживающий момент (сила, которую нужно приложить к валу двигателя для его вращения на один шаг), разный для четных и нечетных позиций, так как на каждом четном шаге ток протекает только через одну из обмоток. Точность позиционирования и вибростойкость на «слабых» (четных) шагах ниже по сравнению с 4-шаговой последовательностью.

Выбор шагового двигателя для микроНМД производится на основе зависимости между рабочим моментом и частотой подачи шаговых импульсов. Для того чтобы достигнуть максимальной скорости вращения, применяют разные законы наращивания скорости. Наиболее часто используются линейное ускорение и линейное замедление, так как они легче реализуются.

При экспоненциальном ускорении время достижения максимальной (постоянной) скорости уменьшается, но экспоненциальное замедление очень крутое, из-за чего возможно нарушение синхронности вращения двигателя (потеря шагов). По этой при-

чине ускорение выполняется по экспоненциальному закону, а замедление — по обратноэкспоненциальной зависимости. Наиболее полное использование момента двигателя может быть достигнуто ускорением и замедлением по параболе.

В микроНМД емкостью более 20 Мбайт применяются замкнутые сервосистемы.

Тракт записи и воспроизведения. Если для позиционирующих систем преобладают конструктивные проблемы, то для записи и воспроизведения мини- и макроНМД — технологические. Они связаны с оптимизацией параметров магнитных головок и дискового покрытия. Главной целью является повышение плотности записываемой информации. В массовых НМД плотность записи порядка 10 000 бит/дюйм и 1 000 дорожек/дюйм. Такой плотности можно достичь при ферролаковом или тонкослойном покрытии диска с помощью композитных или интегральных магнитных головок с высотой плавания от 0,25 до 0,5 мкм. Основные проблемы в области цифровой магнитной записи в НМД:

получение сигнала вопроизведения с достаточно большой амплитудой и соотношением «сигнал/помехи» не менее 20 дБ;

получение очень тонких и гладких покрытий с коэрцитивной силой 400—1 300 Э (для продольной записи);

нанесение тонкого защитного покрытия с большой стартстопной износостойкостью;

достижение устойчивого плавания магнитных головок на малой высоте;

использование специализированных интегральных схем с высокой степенью интеграции для кодирования, корректирования и компенсации сигнала записи и усиления, формирования и кодирования воспроизведенного сигнала;

обеспечение сверхчистого воздуха с низкой относительной влажностью в рабочей области головок и дисков посредством фильтрования, рециркуляции и др.

Для технологии «винчестер» характерно то, что блок, состоящий из магнитных головок и дисков, работает в сверхчистой, герметически закрытой области, причем в состоянии покоя головки имеют контакт с дисковыми поверхностями. Блок с головками и дисками конструктивно оформлен в виде несменного информационного модуля. В нем находится также интегральный предусилитель, который может работать с одной или несколькими головками. При записи информационный поток направляется через драйвер записи к выбранной головке, а при воспроизведении осуществляется предварительное усиление сигнала, в результате чего повышается помехоустойчивость в случае передачи для дальнейшей обработки.

Тракт записи и воспроизведения в микроНМД (емкостью до 100 Мбайт) упрощен прежде всего из-за небольшого объема всей электроники. Предкомпенсация и декодирование данных выполняются в устройстве управления микрокомпьютерной системы.

Использование групповых кодов (RLL) типа 2/7 (или 4/7) позволяет увеличивать объем записываемой на НМД информации при сохранении плотности записи. Это достигается с помощью группового кодирования, при котором уменьшается число магнитных переходов на дисковом покрытии. Кодом 2/7, например, можно уменьшить на 33% длину дорожки, записанной RLL-данными, т. е. увеличить на $^1/_3$ полезную емкость НМД.

Управление шпиндельным двигателем. В мини-НМД с 14-дюймовыми носителями привод шпиндельного двигателя осуществляется обычным способом — асинхронными двигателями и ременной передачей. С уменьшением диаметра дисков (8 и 5,25 дюймов) полностью переходят на применение привода интегрированным нетоковым бесколлекторным двигателем постоянного тока.

В типичной схеме управления интегрированным трехфазным шпиндельным двигателем сигналы о положении ротора формируются тремя датчиками, подключенными к трем обмоткам двигателя. Сигналы поступают на три входа микропроцессора, который формирует управляющие сигналы для трех драйверов. Реальная скорость вращения ротора определяется при помощи индексных импульсов, поступающих на вход прерывания. Двоичное число разности между реальной и заданной скоростью подается к цифроаналоговому преобразователю (ЦАП). Выходной сигнал ЦАП поступает на вход схемы сравнения, выход которой влияет (в нужном направлении) на драйвер, увеличивая или уменьшая ток, протекающий через соответствующую обмотку двигателя. Информация о величине тока поступает с выхода драйвера к схеме сравнения.

Коррегирующее воздействие продолжается до выравнивания сигналов на двух входах схемы сравнения. Таким образом, при прецизионном двигателе можно достичь точности поддержания скорости вращения до +0,1%.

Надежность. Основной параметр надежности — среднее время между отказами (СВМО) для НМД на сменных дисках имеет значение в пределах 1—2 тыс. ч. С помощью технологии «винчестер» значение этого параметра можно увеличить не менее чем в два раза. Однако следует отметить, что надежность НМД и, в частности, СВМО определяется не только технологией изготовления. В равной степени влияют такие факторы, как коэффициент загрузки НМД (обычно 0,20—0,25) и внешние климатические и механические воздействия (температура и влажность окружающего воздуха, ударные и вибрационные нагрузки).

Для достижения максимально возможной надежности НМД пользователю следует соблюдать определенные правила обращения с изделием, например избегать ненужных включений/выключений накопителя и обеспечивать хорошие условия вентиляции (охлаждения) изделия.

В заключение приведем основные параметры двух накопителей, которые наиболее полно удовлетворяют требованиям малых и ППЭВМ.

	Мини-НМД	МикроНМД
Объем памяти, Мбайт	>30	>40
Скорость обмена, Мбит/с	>9,00	5,00
Плотность данных, бит/дюйм	>6 500	>8 500
Плотность дорожек, дорожка/дюйм	>700	>600
Диаметр носителя, мм	356	130
Интерфейс	ИМП	ИМДМ
Кодирование данных	MFM	MFM
Число дисков	5	4
Число головок	19 (+1 серво)	7 (+1 серво)
Позиционер	Линейный двигатель	Линейный двигатель

УДК 681.3.181.5

СТАНДАРТИЗАЦИЯ ИНТЕРФЕЙСОВ МИКРОЭВМ

А. МИХАЛЬСКИЙ,
канд. техн. наук (ПНР)

Для выяснения ситуации, которая складывается при стандартизации интерфейсов микроЭВМ, необходимо рассмотреть технические и психологические факторы. К техническим факторам можно отнести архитектуру интерфейса и использование существующих стандартов, к психологическим — наличие хорошего поставщика (разработчика) и рекламу. Рассмотрим эти факторы подробнее.

Архитектура интерфейса. Чтобы наилучшим образом удовлетворить потребности различных пользователей, в архитектуру интерфейсов вводится несколько уровней (шин), выполняющих разные функции. Современный интерфейс микроЭВМ должен иметь шину данных с большой шириной, высокой пропускной способностью (скорость обмена), возможностью многопроцессорной работы. Кроме того, интерфейс должен быть процессорно-независимым.

Существует несколько удачно разработанных интерфейсов, удовлетворяющих этим требованиям (табл. 1), но только два из них — Multibus II и VME-bus — получили широкое распространение и стали фактическими стандартами. Между этими двумя стандартами идет постоянная конкурентная борьба за завоевание рынка. Многоуровневая структура шин встречается в различных интерфейсах (табл. 2). Гибкостью структуры отличаются только два интерфейса: Multibus II и VME-bus.

Использование стандартов. Рассмотрим стандарты, играющие решающую роль в процессе производства и сборки системы. К ним относятся стандарты на платы и разъемы. Как правило, во всех интерфейсах используется печатная плата стандарта EUROCARD (IEC 297—3). Различаются интерфейсы по способу расположения разъемов (табл. 3).

Таблица 1. **Современные интерфейсы**

Параметр	Тип интерфейса					
	Fastbus	Futurebus	Multi-bus II	Nubus	VME-bus	BI-bus
Статус стандарта	IEEE (проект)	IEEE (проект)	IEEE (проект)	IEEE (проект)	IEEE (проект)	—
Главный разработчик	NIM Cimmit-tee	IEEE	INTEL	TEXAS Instn	Motorola MOSTEC SIGNETICS	DEC
Связанные фирмы	—	—	Siemens TEC, HP, AMD	—	Philips, Thomson	—
Интерфейсные схемы	В разработке	Нет	Есть	В разработке	Есть	В разработке

Таблица 2. **Архитектура интерфейсов**

Шина	Тип интерфейса					
	Fastbus	Futurebus	Multi-bus II	Nubus	VME-bus	BI-bus
Параллельная	Fastbus	Futurebus	IPSB	Nubus	VME	BI
Локальная	—	—	iLBX II	—	VMX	—
Последовательная	Serialbus	Serialbus (CS MVC)	iSSB (CS MA/CD)	—	VMS (CS MA/CD)	—
Внутреняя	—	—	iSEX	—	—	—
Ввода-вывода	—	—	Multi-channel DMA Bus	—	G64 Bus I/O Channel	—

Таблица 3. **Механические стандарты модулей**

Параметр	Fastbus	Futurebus	Multi-bus II	Nubus	VME-bus	BI-bus
Размер платы	366,7×400	366,7×280	233,4×220	366,7×280	233,4×160	?
Разъем	130 AMP	DIN-96 (IEC 603-2)	DIN-96 (IEC 603-2)	DIN-96 (IEC 603-2)	DIN-96 (IEC 603-2)	Zero-force
Расположение разъемов	Нестандартное	Стандартное	Стандартное	Стандартное	Стандартное	?
Модуль расстояния	16	20,32 (0,8″)	20,32 (0,8″)	20,32 (0,8″)	20,32 (0,8″)	20,32 (0,8″)

Как видно из табл. 3, общепризнанные стандарты не соблюдаются только в двух интерфейсах — Fastbus и BI-bus.

Наличие хорошего поставщика. Самым оптимальным случаем можно считать тот, когда разработкой интерфейса занимается фирма-изготовитель средств вычислительной техники, или, что еще лучше, несколько таких фирм вместе (см. табл. 1). Стандарт, разработанный непромышленной организацией, сложнее внедрять, хотя бы из-за отсутствия необходимых интерфейсных интегральных схем. В связи с этим наименее годны для внедрения интерфейсы Fastbus и Futurebus.

Реклама. В условиях жесткой конкурентной борьбы реклама, как известно, играет определяющую роль в завоевании рынка сбыта продукции. Например, слишком поздняя и не очень широкая рекламная деятельность фирмы TEXAS Instr. привела к тому, что в настоящее время интерфейс Nubus не пользуется популярностью. С другой стороны, есть фирмы, как, например, DEC, которые не заинтересованы в распространении своего стандарта и не ведут рекламной деятельности.

Самая высокая оценка рекламы стандартов принадлежит двум интерфейсам: Multibus II и VME, что совпадает с ситуацией на мировом рынке микроЭВМ [2—7].

Из-за того, что до сих пор развивалась линия специализированных интегральных схем-аналогов изделий фирмы Intel, интерфейс Multibus II имеет современную, многоуровневую архитектуру, очень похожую на структуру локальной сети. На примере этого интерфейса наглядно просматривается тенденция широкого использования имеющихся стандартов. В данном случае фирма оставила название интерфейса, который пользовался большой популярностью, указав в новом изделии лишь признак модернизации. Несмотря на кажущуюся сложность структуры интерфейса, в нем используется принцип, который был с самого начала заложен в Multibus [1]: введение двух типов шин — локальной и глобальной. В новом решении обеспечивается лишь однородное программное обслуживание всех уровней и устраняются неудобства, связанные с обслуживанием прерываний, путем использования сообщений.

На отдельных уровнях интерфейса используются, как правило, стандарты, проверенные на практике, к которым пользователи уже привыкли (iSBX — это P959; iLBX, Multichannel — это DMA).

При конструктивном исполнении применяются только хорошо зарекомендовавшие себя международные или фирменные стандарты. Плата выполнена по стандарту EVROCARD (IEC 297—3). На уровнях iPCB, iSSB и iLBX II используется разъем DIN 96 (IEC 603—2), на уровне iSBX — разъем INTEL (P959) и на уровне Multichannel — разъем Scotch Amphenol.

Приведенные выше данные показывают, что учет существующих стандартов оказывает решающее влияние на распространение нового изделия, в том числе интерфейса, определяющего в свою очередь на длительное время основные свойства разрабатываемой микрокомпьютерной системы.

Литература

1. Beaston J. Multiprocessor bus is ready to meet 32-bit applications of future//Electronics.— 1984.— № 22.— P. 126—131.
2. Beaston J., Shuen A. Multibus II designs exploit advances bus concept//Computer Design.— 1985.— Feb.— P. 171—178.
3. Hindin H. Thirty-two bit system designers face decision time//Computer Design.— 1984.— Feb.— P. 27—38.
4. Muller K. D. Momparison and status of 32-bit backplane bus architectures//IEEE Trans. on Nucl. Sc. — 1985. — Vol. NS-32.— № 1.— P. 262—268.
5. Nicolson B. 32-bit Futurebus nears IEEE approval//EDN.— 1984.— № 23.— P. 55—74.
6. Rosenberg R. Battle of the buses//Electronics.— 1985.— № 25.— P. 48—51.
7. Williams T. Silicon handles avbitration and message interrupts on Multibus II//Computer Design.— 1984.— Oct. 1.— P. 38—40.

УДК 681.324

ТЕХНИЧЕСКИЕ И ПРОГРАММНЫЕ СРЕДСТВА ВЫЧИСЛИТЕЛЬНЫХ СЕТЕЙ СМ ЭВМ В ЧССР

Я. ГЛУШЕК, инженер (ЧССР),
В. ГРИЦ, инженер (ЧССР)

Совместное использование вычислительной техники и техники связи значительно расширило применение вычислительных систем. Для чехословацких СМ ЭВМ создано большое количество средств телеобработки как технических, так и программных. Ниже подробно описываются только средства телеобработки для вычислительных систем с общей шиной.

Технические средства сетевой телеобработки

К настоящему времени в ЧССР для вычислительных систем с общей шиной (СМ 4—20, СМ 50/50, СМ 52/11, СМ 52/11.М1, М 16—22, СМ 52/12) созданы следующие технические средства телеобработки:

асинхронные — асинхронный адаптер СМ6002, 4-входовой асинхронный адаптер СМ8512, асинхронный мультиплексор СМ8511, телетайпный адаптер;

синхронные — синхронный адаптер СМ8506, синхронный адаптер-Б (СМ8517), синхронный адаптер-Д, высокоскоростной синхронный адаптер (СМ0510.8525);

специальные — процессор телеобработки СМ2401.0510.

Асинхронные модули. Они позволяют подключать терминал или другую ЭВМ. Передача данных осуществляется стартстопным

способом с помощью интерфейса С2 или ИРПС. Интерфейс С2 определяется рекомендациями V24 и V28 МККТТ. Отдельные модули используют разные подмножества связных цепей. Интерфейс ИРПС определяется в СМ ЭВМ и обеспечивает передачу по 4-проводному кабелю в дуплексном режиме на расстояние до 500 м при максимальной скорости 9600 бит/с.

Асинхронный адаптер (АСАД) позволяет присоединить к вычислительной системе одно устройство через интерфейс ИРПС или С2. Оба интерфейса выводятся на 30-контактный разъем, и выбор одного из них обеспечивается соответствующим кабелем. АСАД позволяет управлять модемом и в полудуплексном режиме.

Техническая характеристика адаптера АСАД

Способ передачи	Стартстопный
Режим передачи	Полудуплексный, дуплексный
Скорость передачи, бит/с	50, 100, 200, 300, 600, 1200, 2400, 4800, 9600 (скорость выбирается переключателями отдельно для передатчика и приемника)
Количество информационных битов	5, 6, 7 или 8
Количество старт-бит	1
Количество стоп-бит	1, 1,5 или 2
Контроль передачи	По четности/нечетности (или без контроля)
Питание	+5 В/2А +12 В/0,1 А —12 В/0,15 А
Конструктивное исполнение	Плата 2/3 СМ ЭВМ (280×240 мм), которую можно поставить в любую позицию универсального блока интерфейсов

Асинхронный 4-входовой адаптер позволяет присоединить к вычислительной системе 4 устройства через интерфейс С2 или ИРПС; при этом типы интерфейсов определяются типом платы интерфейса на распределительной панели. Один из каналов адаптера можно использовать для подключения консольного периферийного устройства.

Техническая характеристика асинхронного 4-входового адаптера

Интерфейс	С2 или ИРПС
Способ передачи	Стартстопный
Режим передачи	Дуплексный
Скорость передачи, бит/с	50, 75, 100, 200, 300, 600, 1200, 2400, 4800, 9600 и 19 200 (скорость выбирается переключателем для каждого канала; передатчик и приемник работают с одинаковой скоростью)
Количество информационных битов	5, 6, 7 или 8
Количество старт-бит	1
Количество стоп-бит	1 или 2
Контроль передачи	По четности/нечетности (или без контроля)

Питание	+5 В/2,5 А +12 В/0,2 А —12 В/0,45 А
Конструктивное исполнение	Плата 2/3, распределительная панель и переключающий кабель

Асинхронный мультиплексор (АМУ) позволяет присоединить к вычислительной системе 8 (с возможностью расширения до 16) устройств через интерфейс ИРПС или С2.

Техническая характеристика 8-канального АМУ

Способ передачи	Стартстопный		
Режим передачи	Дуплексный		
Скорость передачи, бит/с	9 600		
Количество информационных бит	5, 6, 7 или 8		
Количество старт-бит	1		
Количество стоп-бит	1 или 2		
Контроль передачи	По четности/нечетности (или без контроля)		
Питание, В	+5	—12	+12
АМУ-А, Б, А	5,5	0,2	0,15
АМУ-Д, А	6	1	—
Конструктивное исполнение	Плата 2/4, плата 3/3 и распределительная панель		

Производится несколько кодификаций асинхронных мультиплексоров с разным количеством каналов и разными интерфейсами.

Телетайпный адаптер позволяет подключить СМ ЭВМ с общей шиной к телеграфной станции DAU-62. Адаптер при соответствующем программном обеспечении позволяет выполнять автоматическое соединение, идентификацию абонентов соединения, передачу данных, логическое отключение, физическое отключение и возврат в исходное состояние.

Техническая характеристика телетайпного адаптера

Количество каналов	2
Способ передачи	Стартстопный
Скорость передачи, бит/с	50
Количество информационных бит	5
Потребляемая мощность, ВА	макс. 15
Конструктивное выполнение	2/3 плата СМ1237, на которой разрешается логика для 2 каналов и плата преобразователя уровней СМ1238

Синхронные модули. Синхронная передача эффективнее асинхронной, поэтому она используется для соединений между ЭВМ. Разработано 3 типа синхронного адаптера, которые различаются типом протокола передачи (рис. 1). Синхронный адаптер (САД) СМ8506 может работать с протоколами HDLC, DDCMP и BSG. САД предназначен для открытых вычислительных сетей, созданных на основе рекомендаций МКТТ ряда X. Программное обеспечение для приведенного типа сети разработано Институтом прикладной кибернетики в Братиславе.

САД-Б предназначен для соединения машин СМ ЭВМ и ЕС ЭВМ. При этом используется один из эмуляторов ЕС7921 или ЕС8514. Этот синхронный адаптер совместно с эмулятором ЕС8514 также позволяет соединять две машины СМ ЭВМ.

САД-Д предназначен для работы по протоколу DDCMP. В этом случае он вполне заменяет САД и имеет по сравнению с ним преимущества в бóльшей надежности. САД-Д реализуется на плате 2/3 (САД — это плата 3/3), что позволяет вставлять его в терминальную станцию СМ 50/50 и в персональную ЭВМ РР 04.

Техническая характеристика	САД	САД-Б	САД-Д
Количество присоединенных устройств	1	1	1
Способ передачи	Синхронный	Синхронный	Синхронный
Интерфейс	С2	С2	С2
Максимальная скорость передачи данных, бит/с	9 600	9 600	9 600
Протоколы	HDLC	DDCMP	BSC
Конструктивное исполнение	Плата 3/3	Плата 2/3	Плата 2/3

Высокоскоростной синхронный адаптер (РСАД) предназначен для локального соединения вычислительных систем СМ ЭВМ с общей шиной со скоростью передачи данных 1 Мбит/с (рис. 2). Для передачи используется коаксиальный кабель максимальной длины 2000 м.

Техническая характеристика РСАД

Режим работы	Полудуплексный, дуплексный
Способ передачи	Синхронный
Протокол передачи	HDLC
Питание, В/А	+5/9,5; +12/0,1
Конструктивное исполнение	Процессор телеобработки и канальный блок платы 3/3 (СМ122), на которой помещен и локальный модем

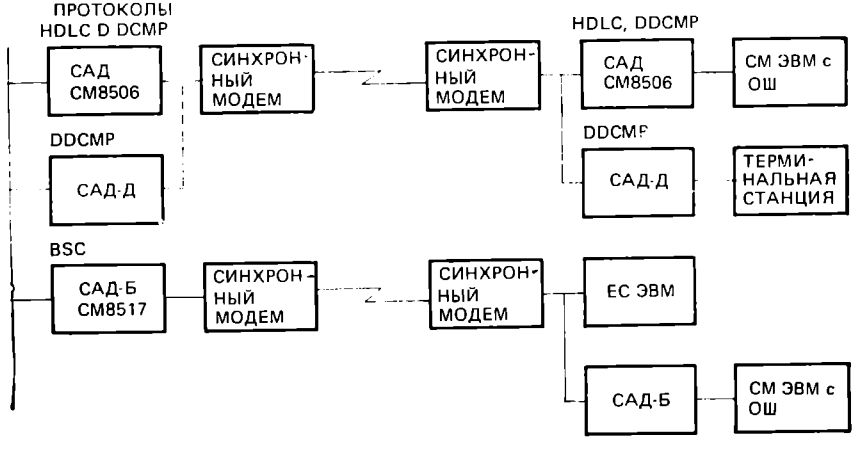


Рис. 1. Синхронный адаптер САД СМ8506

Процессор телеобработки (КОМПРО). При обслуживании большого количества асинхронных или синхронных линий связи возникают высокие требования к производительности процессора вычислительной системы. В систему выгодно включать процессор телеобработки, который обслуживает коммуникационные устройства — синхронные адаптеры или асинхронные мультиплексоры (см. рис. 2).

В результате соединения КОМПРО с синхронными адаптерами (САД-Д или САД), которых может быть 16, возникает состав КОМ-САД, который управляется микропрограммой. Связь между КОМПРО и синхронными адаптерами реализуется через общую шину вычислительной системы, что обеспечивает простое резервирование процессора телеобработки.

При соединении КОМПРО с асинхронными мультиплексорами образуется комплекс, который посредством микропрограмм КОМ-АМУ может управлять шестью 8-канальными мультиплексорами.

Конфигурация интерфейсов (ИРПС и С2) определяется только использованными типами АМУ.

Комплексы КОМ-САД и КОМ-АМУ не требуют прямого физического соединения между модулями, поэтому оба состава легко встраиваются в систему.

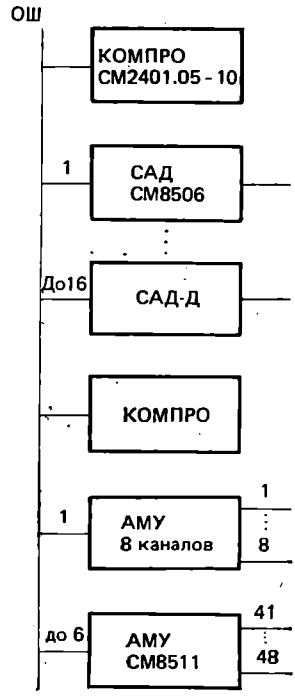


Рис. 2. Коммуникационный процессор КОМПРО СМ2401.05-10

Техническая характеристика КОМПРО

Емкость управляющей памяти, Кслов	1
Емкость памяти данных, Кбайт	1
Работа с оперативной памятью	Прямой доступ
Подключение к вычислительной системе	Посредством общей шины
Максимальное количество управляемых линий	48 асинхронных (6 АМУ) или 16 синхронных (16 САД-Д или САД)
Питание, В/А	+5/8,5
Конструктивное исполнение	Плата 3/3 (СМ1220), плата 2/3 (СМ1221) и коммутирующие платы (СМ1227 и СМ1228)

Модемы. В настоящее время в ЧССР развиваются два типа модемов: МДЕ 1222; МДЕ 2426.

Модем МДЕ 1222 позволяет вести дуплексную передачу данных со скоростью 1200 вит/с по 2-проводным выделенным или коммутируемым телефонным цепям. В цепях низшего звена можно применять скорость передачи 600 бит/с. Параметры модема соответ-

ствуют рекомендации V22 МККТТ, интерфейс С2 между модемом и оконечным оборудованием содержит цепи связи согласно рекомендации V24 МККТТ. Модем обеспечивает синхронный и асинхронный способ передачи, когда знак может содержать 8, 9, 10 или 11 бит (включая стартстопные биты).

Передача данных по телефонным проводам осуществляется по двум частотно разделенным каналам. Основная частота нижнего канала — 1200 Гц, верхнего — 2400 Гц. Дифференциальная фазовая модуляция используется в 2 состояниях при скорости 600 бит/с и в 4 состояниях — при 1200 бит/с. Модем включает автосинхронизирующий скремблер и дескремблер.

Модем МДЕ 2426 обеспечивает передачу данных со скоростью 2400 бит/с, но на линиях низшего качества можно работать со скоростью 1200 бит/с. Модем вполне отвечает рекомендации V26 МККТТ при работе на выделенных 4-приводных линиях и рекомендации V26bis при работе на коммутируемых линиях. Модем обеспечивает синхронную передачу данных полным дуплексом на 4-проводных телефонных линиях и полудуплексом на 2-проводных линиях. При этом используется дифференциальная фазовая модуляция.

Оба типа модемов, оснащенные блоком автоматического ответа, обеспечивают работу в режимах «телефон», «данные» и «тест».

Программные средства телеобработки

Ниже приводится перечень основных программных средств телеобработки, которые находятся или в ближайшем будущем будут находиться в распоряжении пользователя.

SYPROS2 — сетевое программное обеспечение, предназначенное для операционной системы ДОС РВ ВЗ. Оно представляет собой высший уровень сетевого программного обеспечения, основной особенностью которого является возможность образовывать транзитные узлы. Благодаря этому свойству пользователь может логически соединяться и с такими узлами, с которыми не устанавливается прямое физическое соединение. Более того, SYPROS2 обеспечен автоматическим поиском наиболее подходящего пути передачи информации. Практически можно использовать все доступные коммуникационные средства (SAD, ASAD, AMUX, KOMPRO, RSAD).

Среди возможностей, которые предоставляет программисту и пользователю SYPROS2, можно назвать:

связь с другими терминалами. С помощью служебной программы TLK/TALK (Terminal Communication Utility) пользователь может передать сообщение с одного терминала на другой в любом узле сети, который обеспечен целевой задачей;

работа с файлами. Пользователь, работающий с NFT (Network File Transfer) или с FTS (File Transfer Spooler), может передавать файлы с одного узла сети в другой. Например, файл, содержащий командную процедуру, может быть передан в удаленный узел или в случае, когда он находится в этом узле, может быть использован в удаленном узле пользователем местного узла. FTS может либо

обрабатывать запросы сразу, либо ставить их в очередь на обработку в определенное время;

доступ к удаленным источникам. Вызовом задачи RMT (Remote Terminal Utility) пользователь может в местном узле соединиться с другим узлом сети и подключиться к этой системе. В таком случае пользователь имеет доступ к средствам удаленной системы;

межзадачная связь. С помощью вызовов, написанных на языках Фортран, Кобол и Бейсик плюс-2, задачи могут обмениваться в разных узлах с сообщениями и данными посредством общего канала данных;

доступ к удаленным файлам. Способность программировать доступ к файлам с использованием языков Фортран, Кобол и Бейсик плюс-2 обеспечивает чтение, запись и дополнения записи любого файла в сети. Пользовательские программы могут также удалять файлы из дисков в удаленных узлах;

управление удаленными задачами. На языке Фортран можно написать программу, которая способна управлять выполнением задач в узлах. Пользователь может решать задачи немедленно или планировать периодичное выполнение задач в удаленном узле.

SYPROS/ФОБОС2. Система управления вычислительной сетью SYPROS представляет собой программное обеспечение, которое обеспечивает работу операционной системы ФОБОС2 в узле вычислительной сети. Она состоит из двух или более взаимно связанных независимых систем, называемых узлами, которые соединяются линиями передачи данных с целью эффективного использования средств передачи информации.

В рамках одной вычислительной сети SYPROS несколько узлов может быть соединено с разными операционными системами (ФОБОС, ДОС РВ, МОС РВ). Узлы сети могут разделять файлы программы, устройства, области памяти и средства обработки.

SYPROS/ФОБОС2 обеспечивает узлу с операционной системой ФОБОС2 соединение только с одним узлом сети, с которым имеется физическое соединение, поэтому этот узел всегда конечный.

Система SYPROS/ФОБОС2 предоставляет пользователю следующие возможности:

вести передачу сообщений с пользователем, находящимся в другом узле сети (межтерминальный диалог);

копировать и удалять файлы в другом узле сети;

передавать командные процедуры в другой узел сети для выполнения или хранения их в этом узле;

получить информацию о состоянии сети;

обеспечивать соединение между программой в узле пользователя и программой, соединенной с ним в другом узле сети.

Для образования физического соединения система SYPROS/ФОБОС2 может использовать два коммуникационных устройства, синхронный адаптер САД (или САД-Д) и асинхронный адаптер АСАД.

Программное обеспечение SYPROS/ФОБОС2 включает следующие служебные программы: NIP — информационную программу

сети, TLK — программу терминальной связи, NFT — программу передачи файлов в сети, FAL — программу доступа к файлам.

Протокол KERMIT предназначен для знако-ориентированной передачи сообщений. В качестве носителя данных используется асинхронный последовательный канал связи, который способен обслуживать почти все ЭВМ. В настоящее время проверяется соединение для мини-ЭВМ с операционной системой ДОС РВ и ФОБОС и для персональных ЭВМ РР04 и РР06.

Передача сообщений полудуплексная — одна ЭВМ передает данные, вторая их принимает и подтверждает прием; в случае ошибки передача пакета повторяется. В обоих концах двухточечного соединения должна быть программа с протоколом KERMIT. Программы поочередно передают одна другой данные или управляющие сообщения. При таком соединении возможны передача файлов, работа с файлами в местной и удаленной системе, работа с удаленным терминалом.

Текстовые и двоичные файлы можно передавать отдельно и группами. Преимуществом такого деления является единый синтаксис команд для связи и работы с файлами в местной и удаленной системе.

Программное обеспечение телетайпного адаптера включает тестирующие программы, устанавливающие процедуры, управляющие программы ОС ФОБОС2, ДОС РВ В2, ДОС РВ В3.

Тестирующие программы обеспечивают полное тестирование адаптера и работают под управлением операционной системы ТМОС.

Устанавливающие процедуры — это командные процедуры, которые генерируют драйвер (управляющую программу) телетайпного адаптера под управлением соответствующей операционной системы.

Управляющие программы позволяют работать с адаптером так, что ЭВМ может представлять телетайп, который передает и принимает сообщения. Управляющие программы обеспечивают следующие операции:

чтение с телетайпа. Управляющая программа ждет, пока любой абонент телетайпной сети попытается соединиться с СМ ЭВМ. С этого момента управляющая программа принимает знаки с линии и передает их пользователю;

запись на телетайп. Управляющая программа обеспечивает выбор удаленного абонента, его проверку и образование соединения, потом автоматически передает сообщение, находящееся в файле, удаленному абоненту.

Перспективные средства телеобработки СМ ЭВМ

В настоящее время происходит разработка синхронно-асинхронного мультиплексора локальной сети LOPOS, локальной сети с методом доступа CSMA/CD. Синхронно-асинхронный мультиплексор (САМУКС) подключается к общей шине вычислительной системы и позволяет работать с 8 асинхронными линиями, 1 синхронной ли-

нией, 1 параллельным интерфейсом для подсоединения печатающего устройства. Мультиплексор работает со скоростью передачи данных до 19 200 бит/с.

В настоящее время в ВУВТ (г. Жилина) развиваются два типа локальных сетей: магистральная сеть с методом доступа «token bus» и магистральная сеть с методом доступа CSMA/CD.

В ВУВТ проходит испытание локальная сеть LOPOS, предназначенная для подключения мини-ЭВМ СМ ЭВМ (СМ4-20, 52/11, 52/11+, М16-22) к локальной сети с помощью автономного коммуникационного модуля (АКМ).

Техническая характеристика сети LOPOS

Скорость передачи, Кбит/с	500
Максимальное количество узлов в одном сегменте	32
Максимальное расстояние между узлами	1 500
Физическая среда	Коаксиальный кабель
Длина пакета, байт	1—256
Топология сети	Магистраль

В АКМ используется сетевой метод доступа с передачей знака («token bus»), который предупреждает конфликтные ситуации и обеспечивает для каждой станции (узел сети) доступ в сеть в определенный момент времени, в связи с чем данный метод предназначен для применения в реальном масштабе времени. Метод основан на постоянной циркуляции знака между узлами сети, которые используют канал передачи (магистраль) без конфликтных ситуаций. В каждый момент времени в сети находится 1 знак. Станция, в которой находится знак, временно становится управляющей. При этом можно использовать канал передачи данных любым способом до окончания заранее установленного временного промежутка. Далее станция передает знак следующей станции. Эта последовательность передачи называется «логическим кольцом».

LOPOS — локальная сеть магистральной топологии со сложной архитектурой, которая в основном соответствует архитектуре сети SYPROS в сети LOPOS. Это дает возможность применять пользовательские программы сети SYPROS в сети LOPOS. В результате обеспечиваются: междузадачная связь, управление логическими линиями, связь «терминал — терминал», совместное использование местных и удаленных средств, доступ к удаленным файлам и манипуляция с ними, присоединение в качестве терминала удаленной системы. Программное обеспечение сети LOPOS работает в среде ОС ДОС РВ В3 и состоит из нескольких модулей; для своей работы оно постоянно требует минимум 8 Кслов оперативной памяти.

Ниже приведена техническая характеристика магистральной сети с методом доступа CSMA/CD (Carrier Sence Multiple Access with Collision Delection).

Скорость передачи данных, бит/с	10
Максимальное расстояние между узлами, м	1 500
Длина пакета, байт	1 500

Метод доступа	CSMA/CD
Физическая среда	Коаксиальный кабель
Топология сети	Магистраль

В локальную сеть на первом этапе ее внедрения можно будет подключать вычислительные системы с общей шиной, в том числе СМ 52/12, а на втором этапе — персональные ЭВМ.

Программное обеспечение для этой локальной сети обеспечит все возможности, которые в настоящее время предоставляет SYPROS. Это значит, что в LOPOS можно будет использовать как локальную сеть, так и удаленное соединение. При этом пользователь будет применять одни и те же команды. На первом этапе внедрения сети такое программное обеспечение будет развито для ОС ДОСРВ и МОСВП.

II

Программное обеспечение ЭВМ

УДК 681.3.06

СИСТЕМА SOFTORG — ИНТЕГРИРОВАННАЯ СИСТЕМА ВСПОМОГАТЕЛЬНЫХ СРЕДСТВ ДЛЯ СОВРЕМЕННОЙ РАЗРАБОТКИ ПРИКЛАДНЫХ СИСТЕМ

К. ФЮЛЕ, инженер (ВНР)

Комплекс средств SOFTORG предоставляет разработчику программного обеспечения, во-первых, такую готовую технологию, которая распространяется на все виды деятельности, на все этапы жизненного цикла программного средства, и, во-вторых, такие вспомогательные средства, с помощью которых он может в своей каждодневной работе эффективно использовать эту технологию.

Система SOFTORG состоит из семи подсистем (рис. 1).

Подсистема SOFTMAN служит в качестве вспомогательного средства для руководителя разработки программного обеспечения и поддерживает проектирование, организацию, управление, контроль, оценку временны́х и финансовых ресурсов, качества изготовленных полуготовых и готовых средств, а также связанные с ними функции «системного хозяина».

Шесть остальных подсистем являются вспомогательными средствами для разработчика программного обеспечения:

SOFTSPEC обеспечивает запись спецификации требуемой системы на семантическом уровне, не зависимом от целевого системного окружения, с помощью формальных средств, позволяющих выполнять автоматическую верификацию;

SOFTCON поддерживает проектирование требуемой системы на семантическом уровне, учитывающем целевое системное окружение, с помощью формальных средств, позволяющих выполнять автоматическую верификацию содержания и его техническую оценку;

SOFTGEN поддерживает программирование требуемой системы на языках ПЛ/1 и Кобол с помощью автоматического генерирования соответствующих исполнительных элементов (структуры

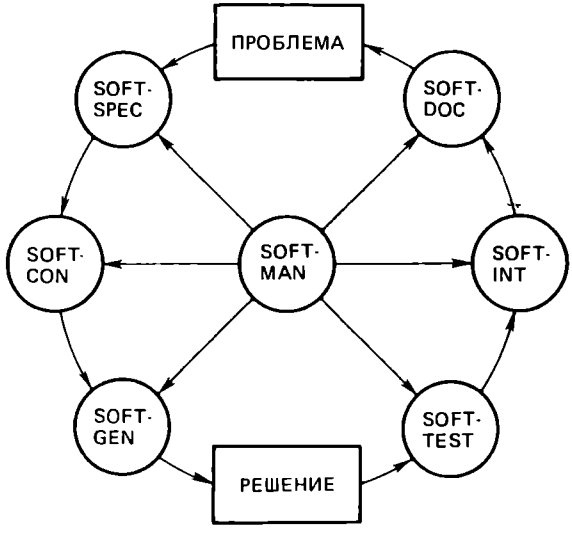


Рис. 1. Система SOFTORG

модулей, макросы данных, описания коммуникации данных и банков данных, совокупности JCL);

SOFTEST поддерживает тестирование и динамический анализ, выполняемые в окружении на модульном уровне (по единицам трансляции);

SOFTINT поддерживает тестирование и динамический анализ, выполняемые на системном уровне в реальном системном окружении;

SOFTDOC поддерживает трехуровневое (на уровне модуля, программы и системы) документирование и статистический анализ, выполняемый на отдельных уровнях.

Наиболее важные характеристики системы SOFTORG: она ориентирована на задачи по разработке программного обеспечения для управления предприятиями и в рамках этого — в основном на задачи комплексной обработки информации (менее подходит для научного или технико-математических применений, а для небольших систем является менее экономичной). Система SOFTORG учитывает аппаратно-программное окружение, наиболее соответствую-

щее классу поставленных задач, т. е. ограничивается большими ЭВМ, имеющими виртуальную память, языками программирования Кобол и ПЛ/1, операционными системами DOS и OS, а также применяемыми в этой среде средствами работы с данными (QSAM, VSAM), СУБД (IMS, IDMS) и средствами телеобработки (CICS, CMS, IMS, SHADOW). Таким образом, систему SOFTORG можно применять и в системной среде ЕС ЭВМ. Отдельные подсистемы SOFTORG могут применяться и самостоятельно, но качественные преимущества можно получить только в случае, если совместно используется несколько подсистем или все подсистемы применяются интегрированно.

В системе SOFTORG под программным обеспечением понимаются не только сами программы в виде, пригодном для выполнения на ЭВМ, возможно, в исходных кодах, но и вся документация, которая создается в фазах жизненного цикла программного обеспечения (рис. 2).

Фазовую концепцию SOFTORG интерпретирует не только статически, но и динамически. Жизненный цикл программного обеспечения в повседневной практике может быть процессом постоянно движущимся в одном направлении, но возможен и возврат на одну или несколько фаз. В то же время после первого прохождения жизненного цикла может потребоваться новое прохождение, вероятно, и несколько раз. Динамическая интерпретация фазовой концепции, естественно, должна действовать так, чтобы технология (средство) заставляла специалиста выполнять возврат, или же повторять процесс даже с первой фазы. Можно сказать, что фазовая концепция повышает требования к технологии во времени, а концепция управления — в пространстве.

При разработке программного обеспечения в организациях параллельно выполняются несколько проектов, поэтому прикрепление ресурсов к проектам необходимо планировать, контролировать и в случае необходимости регулировать динамически. Для каждого проекта нужно оценить затраты и время на его выполнение, непрерывно вести учет фактических затрат, в случае необходимости своевременно вмешиваться в процесс. В рамках каждой фазы необходимо обеспечить распределение и планирование по времени отдельных задач, действенную проверку (контроль качества) полуготовых и готовых продуктов и т. п.

Декомпозиция. Для этого чтобы система SOFTORG для каждой подсистемы (в последнее время — даже внутри подсистемы для каждой программы) образовывала отдельную единицу, применяется процедура декомпозиции. Отдельные подсистемы — в результате небольшой дополнительной работы — можно переделать в самостоятельно используемые средства, и таким образом спектр применяемости может выйти за рамки общей концепции SOFTORG. Систему желательно внедрять постепенно, по подсистемам.

Принцип декомпозиции действует следующим образом. Легко можно сделать самостоятельным этап специфицирования, так как внутри жизненного цикла программного обеспечения это первая

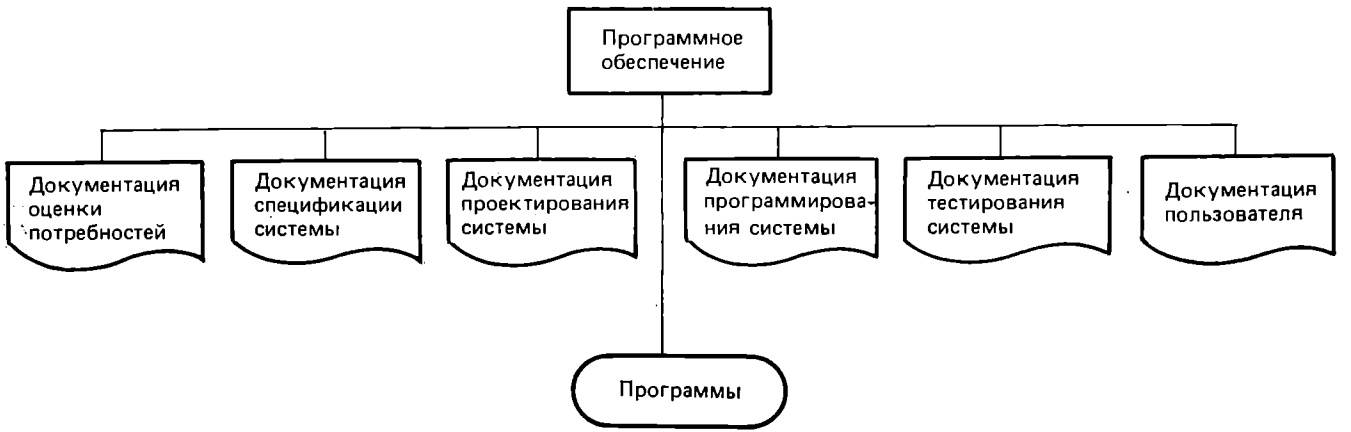


Рис. 2. Состав программного обеспечения

фаза и для подсистемы SOFTSPEC еще не нужно учитывать свойства рабочей среды разрабатываемой подсистемы. Сделать самостоятельной фазу проектирования сложнее. Однако концепцией SOFTORG поддерживается идея проектирования с помощью подсистемы SOFTCON без спецификации.

Временной порядок этих трех фаз тестирования-документирования в принципе не зафиксирован, поэтому все три подсистемы (SOFTDOC, SOFTEST, SOFTINT) могут использоваться и самостоятельно.

Функцию управления также легко можно сделать самостоятельной, так как она хотя и имеет связь с каждой фазой, но органически не привязана ни к одной из них. Для подсистемы SOFTMAN (как и для подсистемы SOFTSPEC) безразлично, в какой среде будут применяться разрабатывающие системы.

Степень свободы. Технологические предписания, введенные фирмами, и технологические вспомогательные средства фактически выводят разработчика на вынужденный маршрут. Но, как известно, чрезмерное ограничение свободы разработчика является такой же ошибкой, как и чрезмерная уступчивость.

От исходного кода, получаемого с помощью системы SOFTORG, мы не ожидаем, что уже при первом проходе он будет совпадать с исходным кодом будущей эксплуатируемой системы, а только то, что он будет как бы основой будущего исходного кода. Эта уступка означает, что в предыдущих фазах проектирования и специфицирования не нужно стремиться любой ценой дорабатывать несущественные фрагменты, а надо концентрировать внимание на более высоких уровнях.

От документации (представления), полученной с помощью системы SOFTORG, не требуется, чтобы только непосредственно из нее генерировался код.

Разработчик имеет возможность применять традиционные средства проектирования, создаваемые в базе данных, без использования подсистемы SOFTCON. Следовательно, проектировщик сначала «в основном» создает проект, а затем улучшает его. Применяя традиционные средства, специалист может дополнить также спецификацию, созданную с помощью подсистемы SOFTSPEC.

Свобода разработчика проявляется в том, что база данных подсистем SOFTSPEC, SOFTCON и SOFTGEN открыта для разработчика, описание их внутренней организации — структуры содержится в описании пользователя.

Интеграция. Система SOFTORG — не совокупность изолированных друг от друга технологических средств: интеграция органически связанных друг с другом (но не являющихся неразделимыми) подсистем «во времени» распространяется на 6 фаз жизненного цикла программного обеспечения, а «в пространстве» кроме функций выполнения охватывает функции управления и контроля качества. В этом смысле семейство продуктов SOFTORG можно сравнить с интегрированными системами производства, управления и контроля, используемыми в промышленности.

Интеграция SOFTORG возможна в первую очередь благодаря тому, что отдельные подсистемы «знают» базу данных их коллег, могут непосредственно принимать информацию друг от друга.

Декомпозиция и степень свободы пользователя значительно усложняют решение внутренней интеграции.

Необходимо предусмотреть дополнительные функции для того, чтобы каждая подсистема в случае самостоятельного применения могла получить информацию, требуемую от другой подсистемы и непосредственно от пользователя, или создавать ее некоторым автоматическим способом.

Требуется также встроить соответствующие функции обратной связи и блокировки для того, чтобы свобода пользователя не нарушала формальную полноту и согласованность внешней документации (внутренних представлений) различного уровня.

Контроль качества. От технологии в настоящее время мы ожидаем того, чтобы она не только поддерживала контроль качества, но и гарантировала требуемое качество разрабатываемого программного обеспечения.

Новая задача контроля качества заключается в контроле внутренней согласованности всех фазовых продуктов, относящихся к одному и тому же программному обеспечению, в проверке их формальной полноты. В процессе создания программного обеспечения переход из предыдущих фаз в следующие должен разрешаться или запрещаться контролем качества; злоупотребление свободой разработчика также должно обнаруживаться контролем качества и он должен обеспечивать сравнения, имеющие характер обратной связи.

Автоматизация традиционных функций контроля качества становится возможной благодаря тому, что каждый отдельный фазовый продукт имеется и в виде внутреннего представления, анализируемого (оцениваемого) машинным путем.

Согласно технологии SOFTORG достигаемая надежность создаваемой системы зависит не только от профессиональной подготовки специалистов, но и от таких факторов, как количество (комплексность программного обеспечения), время (количество затраченных человеко-месяцев) и затраты. Все четыре фактора с точки зрения «ресурсов» обеспечиваются возможностями разрабатывающей организации: аппаратными и программными возможностями, профессиональной ориентацией (практика участвующих разработчиков) и, наконец, сложностью, новизной решаемой задачи. Возможности организации по разработке в совокупности определяют производительность, выражаемую фиксированным количеством баллов, и это количество баллов необходимо разделить между количеством, качеством, временем и денежными затратами с учетом интересов заказчика-предпринимателя.

Разработка системы. Система SOFTORG (технология и средства) не дает совершенно новой технологии, а только является объединением технологий, более или менее знакомых нам из профессиональной литературы. В SOFTORG эти готовые технологические

приемы слиты в единую технологическую систему, которая эффективно поддерживает пользователя. В большинстве случаев адаптация означала не только простое встраивание готовой технологии, но и ее доработку и расширение, особенно в тех случаях, когда данная технология стала известной только в рамках какого-либо университета.

За прошедшие 6 лет для разработки SOFTORG потребовалось более чем 90 чел.-лет интеллектуального труда. Здесь мы отмечаем только тот опыт разработки, который представляет общий интерес:

разработка отдельных подсистем выполнялась не одновременно: сначала были созданы подсистемы SOFTSPEC и SOFTDOC, а последними — SOFTMAN и SOFTINT. Это объяснялось тем, что продажа законченных подсистем отчасти позволила финансировать разработку последующих подсистем;

мы стремились к поставке на рынок отдельных продуктов в форме прототипа. Это воздействовало на разработку: в действительности только в ходе «живого» применения проясняются требования, ожидания, уточняется спецификация системы;

по мере расширения круга покупателей выдвигались новые требования. Таким образом произошла разработка следующих друг за другом вариантов подсистем (и эта тенденция, вероятно, пока будет оставаться).

Система средств SOFTORG получила высокое признание у сотрудничающих фирм и специалистов; пока она не является прибыльной, но в ближайшем будущем положение должно измениться.

УДК 681.3.06:519.6

НЕОРТОГОНАЛЬНОСТЬ И ОРТОГОНАЛЬНОСТЬ В СОВРЕМЕННЫХ ЯЗЫКАХ ПРОГРАММИРОВАНИЯ

Г. ШТИЛЛЕР, профессор (ГДР)

Языки программирования представляют собой готовые элементы для написания программ, чем-то напоминающие детские кубики. Архитектура языка косвенно влияет на свойства программ, определяя эффективность программирования и надежность. Одним из признаков, принимаемых во внимание при создании языка, является его ортогональность. Ортогональность представляет общие знания (знание нескольких языков) и никак не влияет на стиль программирования [3, с. 5, 4, с. 52]. Ортогональность позволяет оптимизировать конструктивные свойства языка: язык должен содержать по возможности небольшое число основных концепций, которые могут свободно сочетаться между собой для представления точно определенных новых действий. Основные

концепции должны быть типовыми для проблемно-ориентированного программирования и независимыми друг от друга.

Технология разработки программного обеспечения направлена на создание процесса разработки, не зависящего от конкретного языка, с последующей адаптацией к языку программирования [1]. При соответствующей реализации ортогональность является средством, поддерживающим такую стратегию.

Ортогональность являлась целью при разработке языка Алгол 68 [5, 6, 7].

Ортогональные концепции Алгол 68 основаны на концепции типов, концепции описания тождества (identity declaration), концепции основного предложения (unitary clause) и его почти неограниченной комбинируемости. Ортогональные компоненты можно найти в разных языках, например в ПЛ/1 — включение меток (LABEL) в концепцию данных и переменных, хотя это и противоречит принципам структурного программирования, в Паскале — типы индексов массива не ограничены целочисленным типом (integer).

Представления синтаксиса тоже подлежат оценке по критерию ортогональности. Одинаковые элементы должны быть и одинаково представлены. Различия в синтаксисе следующих примеров на ПЛ/1 и Фортране не обоснованы (см. [8]):

```
DCL X CHAR (4) DEFINED B;        DCL Y CHAR(4) BASED(P)
GO TO (10, 11, 12) I             IF(I) 10, 11, 12
```

Различный уровень ортогональности языков программирования вызван определенной сложностью реализации этого принципа. При очень простых языках он не имеет значения (например, LISP имеет крайне простое ядро языка).

В языке Модула-2 [4, 9] недостаток неполной спецификации процедур устранен. Процедуры являются данными и имеют тип. Функции являются процедурами со специфицированным типом результата. Существуют процедурные переменные. Их использование обеспечивает текстовое распознавание рекурсивных описаний процедур при текстовом отделении как заголовка, так и тела процедуры.

Эта попытка ортогональности является очевидным заимствованием из Алгола 68 с неортогональным включением в язык Модула-2: в описании языка [9] ничего не сказано о процедурных константах (понятие не существует в перечне слов; пользователь должен сам понимать, что описание процедур действует как описание константы; описание процедур не имеет никакого сходства с описанием константы); в языке не существует неименованных процедурных констант, похожих на литерал; обработка границ индексов при параметрах типа «массив» неодинакова относительно типа процедуры, спецификации параметров и переменных типа «массив» в программе.

Последний пример касается общих вопросов концепции типов языков программирования, а именно соотношений между описанием типов, эквивалентностью типов и определенными образованиями типов данных.

В Алголе 60 используется сравнительно простая концепция типов и в качестве средства диагностики ошибок. Присваивание значения в форме REALx; x:=TRUE,... указывается транслятором при переводе программы как ошибка (контроль контекста, контроль типов во время трансляции). В этом смысле концепция типов содержит такие свойства объекта, которые можно проверить во время трансляции («статическая» концепция типов). Свойства объектов программы, которые можно проверить только при выполнении программы (проверка во время выполнения), не относятся к типу объекта (границы индексов переменных типа «массива»). Алгол 68 последовательно принимал это статическое понимание при существенно расширенном количестве типов. В описании языка EDISON четко сказано: «a range is not a type» [7, с. 377].

В Паскале [10] и в некоторых последующих языках (например, [11, 9]) указания области значений являются составной частью типа. Это относится конкретно к диапазонам (subrange types) и к области индексов массива (index types). В процессе разработки языка Паскаль ожидалось, что проверки станут возможными как бы статически с помощью верификации во время трансляции программы. Это не подтвердилось.

Концепция диапазонов играет довольно важную роль в языке Паскаль. Ее нельзя исключить без существенных изменений языка. Она вызывает ряд проблем, которые частично решаются в Паскале с помощью упрощающих (часто неортогональных) ограничений, но при обобщении приводят к дискуссиям. Язык Aga, например, не преодолевал связанное с этим увеличение комплектности, несмотря на некоторые интересные особенности (например, range constraints, некоторые представления).

При рассмотрении выражения на языке Паскаль

TYPE t1=1 ... 5; t2=2 9

возникает ряд вопросов.

Какой тип присваивается определенному значению от t1? Очевидно, это должен быть указанный тип, т. е. t1 (Паскаль, Модула определяют, что надо присвоить базисный тип, т. е. в этом случае integer).

Какой тип операндов идентифицирует оператор +, например, в формуле 3+4? Какой тип имеет результат (Паскаль, Модула определяют integer)?

Являются ли типы ссылок ↑t1, ↑t2 и ↑integer совместимыми между собой или нет? По правилам определения языков Паскаль и Модула они не должны быть совместимыми. Но это очень нецелесообразное ограничение может быть и не реализовано в трансляторах (незначительность указаний области значений в атрибуте базиса данных ссылок).

Должен ли быть при формальном параметре типа t1 и фактический параметр такого же типа или принимается и фактический параметр типа integer (или типа t2)?

Должна ли обязательно соблюдаться совместимость типов в качестве направленного соотношения к отношению «содержится в» соответствующих наборах значений рассмотренных типов? Это значит t1 совместим с integer (обратное недействительно). Определения, которые это соотношение делают симметричным [9], противоречат толкованию диапазона в качестве типа.

Становится ясно, что концепция диапазонов в качестве типа лишена смысла. Ее полезность заключается в проверке области значений при присваивании значения переменной типа «диапазон». Если, например, описывается VAR subrange:t1; то subrange должен принимать только значения из интервала 1 ... 5. Автор в свою очередь должен констатировать, что соответствующие проверки во время выполнения программы были реализованы не полностью в двух трансляторах Паскаля, с которыми он работал.

Нет оснований связывать желаемую проверку при присваивании значения переменным типа «диапазон» с типом переменной: можно воспринимать ее как свойство переменных, которое осуществляется дополнительно к указанию типа (в этом случае integer). Можно, например, написать

VAR subrange : int<1 : 5>

и рассматривать subrange в качестве переменной типа integer с ограниченной областью значений. Хотя это действие аналогично описанному, но что касается понятия и концепции, то здесь есть существенное различие, освобождающее понятие типа. Здесь выражено отношение к концепции range constraints в языке Ада. Эту концепцию можно очевидно расширить в очень обобщенной форме при соблюдении так называемой эквивалентности типов, так что дополнительной концепции subtipe не требуется.

Для областей индексов массивов справедливо аналогичное утверждение: их можно понимать как свойство переменных типа массива, но не как составную часть типа массива. Можно, например, написать

VAR vector : ARRAY [1 : 10] real

рассматривая vector в качестве переменной типа ARRAY[] real с областью значений индекса, ограниченных на интервал 1:10 (допускаются ли еще другие типы помимо integer, для индекса не так важно). Вопросы при концепции типа диапазона можно задавать и отвечать на них по смыслу. Концепция описателя (declarer) в Алголе 68 является даже предпосылкой для принципиального решения. Введенные на основе описания типа новые идентификаторы типов, например TYPE sign=int<—1:+1>; array=ARRAY [m:n]real, можно использовать в разном контексте: в описаниях констант, для спецификации параметров процедур, в качестве базиса данных для ссылок. В процессе трансляции программы использованные подобным образом идентификаторы типов заменяются (на правой стороне описания типов) своим предписанием имплементации. Эта замена текста выполняется относительно сущест-

вующих параметров областей значений обычно в зависимости от контекста по правилам языка. В связи с описаниями переменных они всегда требуются и, следовательно, надо их заменять. В базисе данных для ссылок они имеют значения и могут быть игнорированы.

Если параметры областей значений указываются в описании типов, то это возможный (но не необходимый) сокращенный способ написания. Он обычный, даже если такие параметры не действуют в качестве признака типа. Этот способ написания соответствует неявной параметризации описания типов: необходимый фактический параметр указывается вместо формального параметра в описании типа и заменяется в соответствующем случае по тексту. Это, например, является действием механизма параметров call by name в Алголе-60. Точно также можно было бы предусмотреть формальные параметры, например

```
TYPE array = (RANGE r): ARRAY [r] real
VAR vector : array (m : n)
```

Фактические параметры тогда появляются явно в описании переменных. Вопросы синтаксического представления дают здесь обширный материал для дискуссии, но не могут быть обсуждены в рамках этой статьи.

С помощью связи механизма замены описания типов с механизмом проверки типов во время трансляции реализуются различные виды эквивалентности типов.

При эквивалентности по структуре новый описанный идентификатор типа и его предписание имплементации (правая сторона описания типа) являются равноправными на уровне исходного языка. Замена идентификатора типа предписанием имплементации осуществляется перед проверкой типов во время трансляции.

При эквивалентности по имени замена осуществляется после проверки. Этим достигается то, что идентификатор типа описывает качественно новый тип, который при проверке типов во время трансляции оказывается несовместимым с предписанием имплементации на правой стороне.

При последовательном использовании эквивалентности по имени рекомендуется допускать для спецификации типа объектов программ только синтаксическую форму идентификатора типа. Объекты программы будут совместимыми по типу в том случае, если специфицированы одним и тем же идентификатором типа. Надо учитывать, что при этом операторы не могут быть автоматически идентифицированы новым типом. Если описывается, например, TYPE event=int, то операции для типа integer не предоставляются для данных типа event; для этого требуются особые указания. В разных языках программирования имеются очень различные указания относительно эквивалентности по типам, иногда они вообще отсутствуют.

В Алголе 68 действует эквивалентность по структуре: определение языка требует проверки эквивалентности типов.

В Паскале эквивалентность по типам не определена, поэтому она зависит от имплементации и реализована неединообразно. Аналогично и в языке Модула [11].

Что же касается языка Модула-2, то в нем оба вида эквивалентностей действуют в зависимости от контекста в том же описании типа (зависят от синтаксической формы правой стороны): если там стоит идентификатор или тип диапазона, то действует эквивалентность по структуре, в других случаях — эквивалентность по имени [9].

В Аде имеются два разных описания типов: описание TYPE вызывает эквивалентность по имени, описание SUBTYPE вызывает эквивалентность по структуре.

В языке EDISON действует эквивалентность по имени; синтаксис заставляет составлять описание для структурированных типов и ограничивает одновременно их применение в этих случаях; тип объектов программы можно специфицировать только идентификатором.

Благодаря своим преимуществам (лучшее распознавание ошибок, проверки на эквивалентность не требуется) принцип эквивалентности по имени находит довольно большое признание. Недостатком этого принципа является возникающая неединообразность, что непросто воспринимается пользователями, особенно тогда, когда оба механизма эквивалентностей действуют одновременно или, как, например, в Модула-2, даже в одном и том же описании типа. При этом надо учитывать, что такие правила в Модула-2 вызваны решением проблемы переноса операторов и ограничением многообразия типов, которое возникает из-за частых переименований при эквивалентности типов, например

```
TYPE event=int; piece=event; person=piece; consumer=person;
```

Такое представление является, кроме того, излишним. То же самое достигается представлением

```
TYPE event=int; piece=int; person=int; consumer=int;
```

Здесь обоснованы ограничения, которые по возможности можно распространить на синтаксис языка.

Надо учитывать, что некоторые типы данных связаны по содержанию с определенным механизмом эквивалентности.

Тип перечисления (enumeration type) надо рассматривать как качественно новый (элементарный) тип в соответствии с концепцией эквивалентности по имени.

Относящиеся к данным ссылки при одинаковом атрибуте базиса данных совместимы в соответствии с эквивалентностью по структуре.

Если диапазоны принимаются в качестве типов, то эквивалентность по структуре уместна; если они не принимаются в качестве типов, то можно создать желательную совместимость объектов с одинаковой областью значений (область индексов) также при эквивалентности по имени.

Целью попытки концептуального упорядочения представленных фактов является увеличение ортогональности в базисных концепциях (понятие типа, эквивалентность типов) зачастую за счет определений особых случаев, например в области синтаксиса.

Предполагается, что концепция типов данных статистическая. Области значения не являются типом и составной частью типа.

Эквивалентность по имени рекомендуется в качестве стандартного вида эквивалентности типов. Если требования эквивалентности в отдельном случае обусловливают эквивалентность по структуре, то желательный эффект надо моделировать механизмом эквивалентности по имени. Это можно решить при областях значений, руководствуясь правилами замены или с помощью подходящей параметризации при описаниях типов. При этом следует стремиться к единой обработке со стороны языка программирования областей значений и областей индексов. При значениях перечисления из описания типа должно быть видно, что определяется качественно новый вид значений и что его предписание имплементации задано неявно, например записью

TYPE colour (red, green, blue)

или

TYPE colour: enumeration (red, green, blue)

Речь идет о том, чтобы избежать формы уравнения в Паскале, которая говорит о наличии несуществующей эквивалентности по структуре. При относящихся к данным ссылках можно моделировать необходимую совместимость при одинаковом базисе данных с помощью эквивалентности по имени, если спецификация типа ссылки (примерно в виде REF_real, REF_REF_int) считается специальной формой идентификатора типа и одновременно исключается, что они встречаются на правой стороне описания типа в качестве предписания имплементации. В таком случае, например, при TYPE reference=REF_real; adress=REF_real образовались бы несовместимые ссылки с одним и тем же базисом данных, что кажется нецелесообразным. Определение вида

TYPE node=STRUCT (value : atom, next : REF_node)

было бы допустимым. С такой обработкой типов ссылок достигается действие, которое похоже на collection в языке CLU (например, [6, с. 96]).

Литература

1. Ершов А. П. Предварительные соображения о лексиконе программирования//Кибернетика и вычислительная техника.— 1985.— Вып. 1.
2. Stiller G. Teaching with ALGOL 68 in Dresden//Proc. Int. Conf. on ALGOL 68.— Amsterdam: Mathematisch Centrum, Math. Centre Tracts.— 1981.— V. 134.— P. 45—58.
3. Hahn R. Höhere Programmiersprachen im Vergleich.— Wiesbaden: Akad. Verlagsgesellschaft. — 1981.

4. Wirth N. Programming in MODULA-2.— Heidelberg: Springer-Verlag.— 1983.
5. Wijngaarden A. van et al. Revised Report on the Algorithmic Language ALGOL 68//ACTA INFORMATICA.— 1975.— V. 5.— Fasc. 1—3.
6. Ghezzi C., Jazayeri M. Programming Language Concepts.— New York: J. Wiley & Sons.— 1982.
7. Brinch Hansen P. The Design of EDISON//Software Practice and Experience.— 1981.— V. 11.— P. 363—396.
8. Wupper H. Zur Spezifikation und Struktur gröberer Systeme — Untersuchungen am Beispiel von ALGOL 68 und GKS//Rechenzentrums-Schriften.— Ruhr-Universität Bochum.— 1985.— V. 4.— ISSN 0721—2186.
9. Wirth N. MODULA—2.— Zürich: Eidg. Technische Hochschule. Inst. für Informatik, Bericht Nr. 36.—1980.
10. Jensen K., Wirth N. Pascal-User Manual and Report// Lecture Notes in Computer Science.— Berlin—W.—Heidelberg—New York: Springer—Verlag.—1972.—V. 18.
11. Wirth N. MODULA: A Language for Modular Programming//Software Practice and Experience.— 1977.—Vol. 7.— P. 3—35.
12. Lindsey C. H., van der Meulen S. G. Informal Introduction to ALGOL 68.—Amsterdam: North Holland Publ. Co.— 1977.
13. Stiller G. Zur Evolution des Wortartbegriffs in höheren Programmiersprachen//IIR — Report. — Ins. für Informatik und Rechentechnik. — 1985. — N. 1.

УДК 681.3.06

ПРИМЕНЕНИЕ ЯЗЫКА ПРОЛОГ ДЛЯ ЭКСПЕРТНЫХ СИСТЕМ

У. ПЕТЕРСОН, д-р техн. наук (ГДР),
Ю. ПИЧКЕ, инженер (ГДР)

Экспертная система (ЭС) — это способная к решению задач и получению выводов система баз знаний, применяемая для решения задач, требующих специализированных знаний (т. е. знаний эксперта-человека).

Экспертные системы характеризуются в основном [1]:

способностью к решению задач и получению выводов как основного компонента предоставления знаний;

способностью к объяснению и оценке;

удобным для пользователя диалогом с системой;

возможностью применения ЭС для решения специфических комплексных задач, требующих специальных знаний;

способностью к обучению.

Не каждый из этих признаков должен быть ярко выраженным, но основы его обязательно должны существовать.

Ниже кратко описаны функции отдельных блоков основной структуры экспертной системы (рис. 1).

База знаний. База знаний состоит из базы фактографических знаний, базы знаний о правилах и аксиомах и базы процедурных знаний. База знаний содержит специализированные знания (т. е. описания, определения, множества, отношения, правила принятия решений, ограничения, эвристические процедуры, гипотезы, неточные факты) в приспособленном к ЭВМ коде.

Блок решения задач. Этот блок обрабатывает заданные пользователем задачи. Кроме алгоритмов интерпретации запросов, блок имеет алгоритмы, представляющие способы решения задач человеком. Такие алгоритмы используются в процессе работы с базами знаний. Сюда относятся механизмы получения выводов, механизмы поиска, механизмы описания.

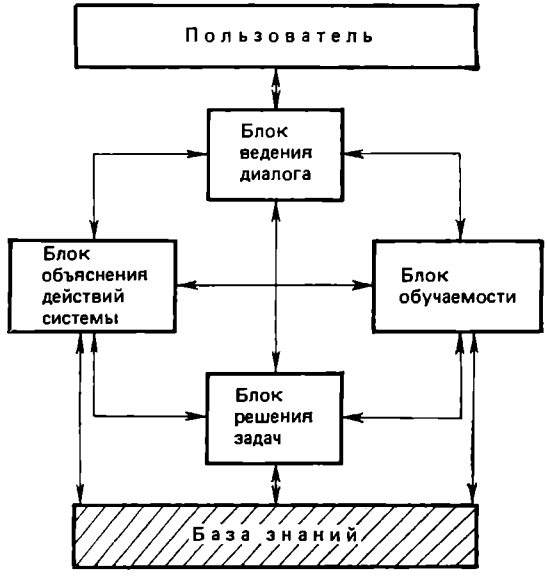


Рис. 1. Общая структура экспертной системы

Блок ведения диалога. В данном блоке содержится человеко-машинный интерфейс с соответствующими техническими средствами (например интерфейс естественного языка, возможность подключения сенсоров, графические средства). Для широкого использования многообразных возможностей экспертных систем человеко-машинный интерфейс должен вести гибкий диалог.

Блок обучаемости. Задача этого блока состоит в поддержке создания и модифицирования базы знаний, ее постоянной актуализации.

Блок объяснения действий системы. Этот блок должен протоколировать ход процесса решения задачи и в случае необходимости объяснять пользователю найденное блоком решение задачи. Это может быть реализовано протоколированием знаний, на основе которых получено решение, объяснением использованных для решения задачи алгоритмов, объяснением понятий.

Как уже было упомянуто, экспертные системы разделяются на базу знаний и механизм получения выводов. Такая структура преобразуется в логический образец программирования:

база знаний представляет собой множество отношений, кодирующих знания предметной области. Подобные отношения описываются на языке Пролог;

в экспертных системах первого поколения механизм получения выводов в общем случае представляет собой интерпретатор множества правил вывода типа IF A THEN B.

Новое качество систем баз знаний достигается следующими приемами. С целью получения выводов систематически анализируется при помощи метода «проб и ошибок» совокупность информационных данных и знаний. Вид представления совокупности знаний дает возможность расширять и модифицировать знания, причем не требуется вмешательства в систему. Такой подход соответствует принципам работы с образцами в области искусственного интеллекта [2].

Рассмотрим простой пример экспертной системы [3]. Задача состоит в определении химического состава неизвестной жидкости.

Система содержит 8 правил на основе программирования правил вывода на языке Пролог. Применяется способ представления, принятый в языке МикроПролог. Получение выводов происходит за счет интерпретатора МикроПролог.

```
((composition X sulphuric_ acid)
    (analysis X sulphate_ ion_ test positive))
((химическое соединение X серная кислота)
    (анализ X проба на серную соль положительна))
((composition X gasoline)
    (spill_ type X oil)
    (analysis X odour gasoline))
((химическое соединение X бензин)
    (тип жидкости X масло)
    (анализ X запах бензина))
((composition X disel_ oil)
    (spill_ typ X oil)
    (analysis X odour oil))
((химическое соединение X дизельное топливо)
    (тип жидкости X масло)
    (анализ X запах масла))
((composition X acetic_ acid)
    (spill_ typ X acid)
    (analysis X odour vinegar))
((химическое соединение X уксусная кислота)
    (тип жидкости X кислота)
    (анализ X запах винного уксуса))
((composition X hydrochloric _acid)
    (spill_ typ X acid)
    (analysis X odour pungent))
((химическое соединение X соляная кислота)
    (тип жидкости X кислота)
    (анализ X едкий запах)
((spill_ type X base 7)
    (analysis X solubility high)
    (analysis X spill _PHY)
    (LESS 8 Y))
((тип жидкости X щелок)
    (анализ X высокая растворимость)
    (анализ X водородный показатель — Y)
    (LESS 8 Y))
((spill _type X acid)
    (analysis X solubility high)
```

```
    (analysis X spill_PH Y)
    (LESS Y 6))
((тип жидкости X кислота)
    (анализ X высокая растворимость)
    (анализ X водородный показатель — Y)
    (LESS Y 6))
((spill_type X oil)
    (analysis X solubility low)
    (analysis X spill_PH Y)
    (LESS Y 9)
    (LESS 5 Y)
((тип жидкости X масло)
    (анализ X низкая растворимость)
    (анализ X водородный показатель — Y)
    (LESS Y 9)
    (LESS 5 Y)
```

Правило R2, например, интерпретируется следующим образом:
IF тип жидкости пробы — масло &;
THEN запах пробы — бензин;
химическое соединение — бензин.

Для определения типа жидкости применяется правило R8. Жидкость в общем определяется при помощи последовательности полученных выводов. Из этого видно, что базу знаний можно расширить несложным образом. А если добавить к такой базе знаний фактографическую информацию о неизвестной жидкости, то может быть определен химический состав неизвестной жидкости. В результате анализа, например, получено:

```
((analysis sample 1 solubility low))
((analysis sample 1 spill_PH 6))
((analysis sample 1 odour oil))
((analysis sample 2 sulphate_ion_test positive))
((анализ пробы 1 — низкая растворимость))
((анализ пробы 1 — водородный показатель — 6))
((анализ пробы 1 — запах масла))
((анализ пробы 2 — проба на сульфатные ионы положительна))
На запрос
?((composition sample 1 X) (PP X))
?((химическое соединение пробы 1 X) (PP X)))
выдается ответ «дизельное топливо».
```

По такому образу функционирования построены экспертные системы первого поколения для диагностики и интерпретации данных. Часто они программируются и на других языках. Но Пролог прямым образом поддерживает представление и обработку знаний.

Очень важным компонентом, обеспечивающим эффективность работы экспертной системы, является блок ведения диалога, т. е. человеко-машинный интерфейс. Создание человеко-машинного интерфейса необходимо, потому что синтаксис языка МикроПролог неудобен для пользователя, что осложняет общение с системой.

Человеко-машинный интерфейс осуществляет прием информации от пользователя (ввод в систему), вывод пользователю ответов типа искусственного интеллекта. Кроме человеко-машинных интерфейсов, требующих от пользователя хорошего владения языком

Пролог (например, SIMPLE), известны следующие виды человеко-машинных интерфейсов: графические, управляемые при помощи меню, на естественном языке, комбинированные виды (например, управляемые при помощи меню графические человеко-машинные интерфейсы).

Для традиционных 8- и 16-разрядных ЭВМ предлагаются управляемые при помощи меню человеко-машинные интерфейсы, потому что существующие технические средства очень слабо поддерживают графические человеко-машинные интерфейсы и человеко-машинные интерфейсы естественного языка.

В пользу применения меню говорит также следующее: пользователю не требуется досконально изучать операционную систему и язык Пролог; применение модульного принципа позволяет реализовать комплексные системы. Недостаток этого способа состоит в том, что пользователь-непрофессионал почти не имеет возможности активно участвовать в программировании системы на языке Пролог. Программирование управляемого при помощи меню человеко-машинного интерфейса не выходит за рамки известных решений. Использование новых возможностей языка МикроПролог дает возможность быстрее создать систему с управляющим при помощи меню человеко-машинным интерфейсом и снабдить ее большим объемом искусственного интеллекта.

Для реализации вышеприведенных функций человеко-машинного интерфейса целесообразно кроме системных предикатов определить пользовательские предикаты для управления экраном, для формирования кадров меню, а также предикаты, которые часто используются, например: предикаты позиционирования курсора, предикаты передачи управляющих символов, предикаты вывода перечней, предикаты вывода инверсных изображений и т. д.

Перечисленные предикаты целесообразно объединить в модуль управления экраном. При помощи основных предикатов можно определить другие предикаты для создания удобных для пользователя кадров меню. С помощью встроенного предиката «?» можно в любое время воспользоваться накопленной в базе знаний информацией. Но, с одной стороны, это трудно неопытному пользователю, а с другой — способ довольно трудоемок при выполнении постоянно повторяющихся комплексных запросов. В связи с этим диалогом лучше управлять при помощи меню. В таком случае вызываются лишь предварительно определенные целевые условия. Они позволяют, например, ввести запросы о существовании, свойствах и отношениях объектов. В определении целевых условий помогают нижеприведенные предикаты.

```
((IS X)
   (? X)
/
(PP YES))
((IS X)
   (PP NO))
((WHICH))
   (FORALL Y ((PP X)))
```

Предикат IS проверяет существование объекта, его свойства и отношения с другими объектами и отвечает YES или NO

При помощи предиката WHICH можно реализовать комплексные запросы типа «ка-

```
    (PP No (more) answers))
((SHOW X)
    (true__of X Y)
    (PP Y)
    (PP more (Y/N)7)
    (RDCH "KBD:" Z)
    (P Z)
    (PP)
    (CHECK Z)
    (FAIL))
((SHOW X)
    (PP No (more) answers))
((CHECK N)
/
    (ABORT))
((CHECK X))
((true__of X Y)
    (X/Y)
```

кие X выполняют условие Y?», после чего выводятся все соответствующие ответы
SHOW проверяет, существует ли решение на предикат X, и выводит аргументы найденного решения. После этого система спрашивает о том, должны ли быть найдены другие решения

При помощи таких предикатов можно вести простой диалог. Для несложной экспертной системы (см. пример) можно, например, сформулировать следующие запросы:

```
&IS ((analysis sample2 spill__PH 7))
NO
        анализ пробы 2 — водородный показатель — 7
&WHICH (X (analysis X spill__ PH 6))
sample 1
&WHICH (X (composition sample1 X))
diesel __oil
        (анализ X — водородный показатель — 6))
проба 1
        (химическое соединение пробы 1))
дизельное топливо
&SHOW analysis
(sample1 solubility low)
more (Y/N) ?Y
(sample1 spill__PH 6)
more (Y/N) ?Y
(sample1 odour oil)
more (Y/N) ?N
        анализ
(проба 1 — низкая растворимость)
(проба 1 — водородный показатель — 6)
(проба 1 — запах масла)
```

Применение отношений в качестве специального человеко-машинного интерфейса дает пользователю возможность работы с релевантными для него понятиями на любом уровне абстракции, потому что всякие отношения могут быть определены.

Экспертные системы требуют очень большой емкости оперативной памяти, что затрудняет программирование даже сравнительно небольших экспертных систем. Каким образом избежать такой ситуации? Одна из возможностей состоит в загрузке в оперативную память отдельных частей экспертной системы с последующим их уничтожением. Например, в случае простых экспертных систем за-

грузить в оперативную память блок обучаемости (см. рис. 1), а позже ее разгрузить.

Другая возможность состоит в создании частей экспертной системы (например, базы процедурных знаний как часть базы знаний) в виде структуры дерева, причем в оперативную память загружается только часть дерева, а позже она выводится. Исходя из такого типа решений соответствующие блоки системы можно программировать в виде общего графа, после чего только часть графа (например, узел) загружается в оперативную память. Первым необходимым для этого шагом является разделение общей задачи на подзадачи (подструктуры). Далее подструктуры объединяются в модулях, которые при необходимости загружаются в память, а позже уничтожаются. Особенно удобным для программирования является разделение задач на структуру дерева. Это возможно для многих классов задач. Если система создается как управляемая при помощи меню, то сразу получается структура дерева: из главного меню можно выбирать подменю, далее из выбранных подменю опять можно выбирать новые подменю или функции. Таким образом получается структура дерева (рис. 2).

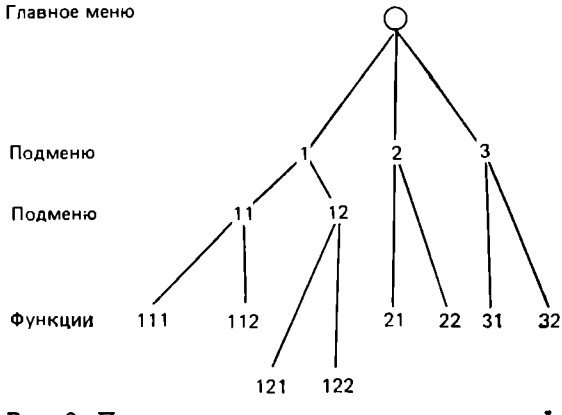


Рис. 2. Пример структуры дерева, управляемой при помощи меню системы на языке Пролог

Каждый узел дерева (подменю или функция) удобно программировать как модуль, который содержит правила реализации соответствующей функции или меню. Для обеспечения широкой применяемости базу фактографических знаний рекомендуется разделять на файлы. Пользователь задает, с каким файлом базы фактографических знаний соответствующая функция должна работать. Такой режим работы имеет следующие преимущества:

модули получаются не слишком большими, что обеспечивает обозримость их содержания;

программы создаются гибкими: если изменяется модуль (узел), то это влияет только на предыдущие и последующие узлы. Поскольку связь модулей устанавливается при помощи экспортных и импортных перечней, модули в значительной степени независимы друг от друга, что положительно влияет на процесс разработки, так как модули могут разрабатываться независимо друг от друга. Однако при этом увеличивается продолжительность обработки из-за повторной загрузки и уничтожения отдельных модулей. Следует создать также управляющий модуль, отвечающий за загрузку и

уничтожение модулей. Управляющий модуль не идентичен с главным меню. По возможности он должен быть довольно маленьким, поскольку постоянно находится в оперативной памяти.

Существуют задачи, для решения которых простые типы правил вывода и формы представления знаний недостаточны. Для них можно сформулировать более сложные правила или метаправила (правила, управляющие выполнением правил) или значительно усовершенствовать базу знаний. Для такого подхода механизм получения выводов целесообразно оформить как метаинтерпретатор с целью улучшения таких характеристик, как модульность и расширяемость. Отношение solve показывает простую структуру метаинтерпретатора. Каждое условие формально рассматривается как перечень:

```
((solve()))
((solve (NOT/X) (solve X)/(FAIL))
((solve (NOT/X)))
((solve X)  (SYS X)/X)
((solve X) (CL X/Y)) (solve Y))
((solve (X/Y)) (solve X) (solve Y))
```

Метаинтерпретатор отвечает на запрос, заданный в примере экспертной системы, следующим образом:

```
&? ((solve ((химическое соединение пробы/X)) (PPX)) дизельное топливо
&.
```

Таким образом, создается блок объяснений действий системы за счет расширения простого способа решения задачи.

Другой возможностью расширения метаинтерпретатора является, например, включение ненадежных выводов. Метаинтерпретатор вычисляет фактор очевидности конъюнкции в соответствии с определенными правилами его размножения. Вывод выполняется введением порогового значения. Если значение очевидности ниже порогового значения, то правило применять нельзя. Другими словами, целевое условие считается выполненным, если полученный фактор очевидности выше порогового значения.

В состав структуры экспертных систем входит блок объяснений действий системы. Он необходим в связи с тем, что:

экспертные системы используются для решения специальных, очень сложных задач большого объема, требующих специализированных знаний;

в экспертных системах применяются эвристические процедуры и неточные знания;

экспертные системы работают неалгоритмически и частично недетерминистически.

Однако для пользователя экспертной системы найденный системой путь получения решения является почти непостижимым (из этого вытекает недоверие к найденным решениям). Иногда пользователь не может понять найденные системой решения без дополнительной информации, потому что у него нет специальных знаний в этой области или пользователь не знает, в какой дополнительной информации нуждается система для решения задачи (большой объем задачи).

Блок объяснений действий системы должен выполнить следующие функции [2]:

(1) объяснять значение использованных понятий (например, при помощи определения);
(2) объяснять процессы или структуры;
(3) показывать пути решения путем изложения причин, законов и т. п.

В [2] такие типы объяснений называются: (1) — интерпретирующими, (2) — описывающими, (3) — обосновывающими.

Объяснения (1) и (2) реализуются функциями HELP. Для этого определения и объяснения понятий необходимо запоминать как часть базы знаний. Определения и объяснения могут быть накоплены, например, во внешних запоминающих устройствах. Кроме того, требуются механизмы, реализующие эффективный доступ к такой информации и ее отображение на экране. Относительно объяснения (3) необходимо следить за процедурой получения выводов и показать его пользователю. Ниже представляется предикат, реализующий такую функцию на самом низком уровне.

```
((test (FORALL X Y))                          Объяснить
//                                            FORALL
   (FORALL x ((why X) (Why Y) (PP))))
((test X)                                     Объяснить встроенные предикаты
   (SYS X)
   /(PP X is a built-in-predicate)
((test X)                                     Объяснить пользовательские предикаты
   (CL X/Y)
   X
   (IF (EQ Y ())
       (PP X is a fact))
       ((? Y) (PP X is proved using)
           (PP y ))))
((why1 ())                                    Условие прерывания: все условия
   /)                                         объяснены
((why1 (X/Y))                                 Объяснить последовательность це-
   (?(X))                                     левых условий
   /
   (test X)
   (why1 Y))
((why X)                                      Объяснить X, если X правильно
   (? X)
   /
   (why1 X))                                  Объяснить, почему X неправильно
((why (X/Z))
   (PP (X/Z) false)
   (LF (EQ Z ()) (P there is
       no substitution for X)
       ((PP!))
   ((PP) (PP proving by Z))))
CLMOD.
```

Для объяснения решений загрузить модуль mwhy. Вызов блока объяснений действий системы происходит при помощи &why (вызов условия). В приведенной выше простой экспертной системе из &why (химическое соединение пробы 1X) получается: (химическое соединение пробы 1 — дизельное топливо) правильно, потому что

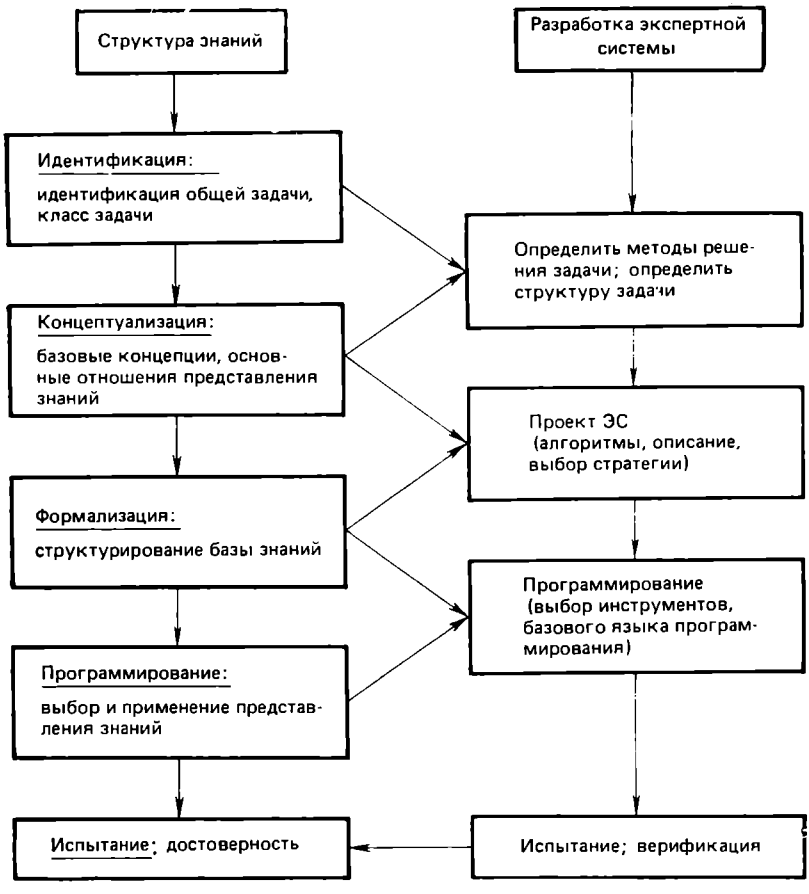


Рис. 3. Схема подхода к разработке экспертных систем

((тип жидкости пробы 1 — масло) (анализ пробы 1 — запах масла)) правильно.

Для создания более удобных блоков объяснений действий системы надо ответить и на другие вопросы. Например: какие объяснения ожидает пользователь, какой объем должно иметь объяснение, чтобы достаточно проинформировать пользователя, не выдавая ему лишней или известной информации и др.

Следует найти также эффективные возможности протоколирования использованных правил. Это особенно важно для экспертных систем, работающих недетерминистическим способом, потому что иначе невозможно показать путь получения решения.

Экспертные системы — это программные продукты, имеющие, как и все традиционные программные продукты, жизненный цикл. Однако у них есть некоторые особенности. На рис. 3 представлен

подход к созданию экспертных систем. При их разработке следует обратить внимание на следующие особенности:

большое значение имеет идентификация и анализ проблемы. Инженер-интерпретатор, имеющий специальные знания и о предметной области, и о разработке математического обеспечения, должен обобщать необходимые знания экспертов (людей). Анализ задачи влияет в наибольшей степени на применяемость и эффективность экспертной системы, потому что ЭС представляют собой сложные системы, подверженные постоянным изменениям;

на этапе формализации необходимо обратить большое внимание на изображение знаний в машинной форме. Инженер-интерпретатор должен перевести знания базы данных в подходящую для ЭВМ форму. Форма представления знаний имеет большое значение и для эффективного применения системы;

особое значение в экспертных системах имеет достоверность, поэтому в экспертных системах модификация играет гораздо большее значение, чем в других программных системах (например, в математических). В ходе проектирования экспертной системы следует обратить внимание на то, чтобы позже можно было легко и просто выполнять исправления, дополнения и модификацию. Для поддержания актуальности базы знаний желательно, чтобы экспертная система по крайней мере была удобной при обучении приемам работы с нею.

В заключение перечислим некоторые известные из литературы области применения экспертных систем:

химия (например, синтез органических веществ, получение структурных формул из спектральных данных);

медицина (например, установление диагноза, анализ рентгеноснимков и снимков микроскопа);

биология (манипуляция генами, прогноз урожайности и т. д.);

геология (например, разведка полезных ископаемых);

военное дело (планирование операций, управление, тренировка полета, распознавание объектов и т. п.);

прогнозирование (определение будущей ситуации на основе измеренных или полученных наблюдением данных и т. д.);

процессы контроля (например, соблюдение параметров процессов в атомных электростанциях, контроль работы аппаратов «сердце — легкие»);

управление и контроль (контроль полета космических летательных аппаратов, управление воздушным сообщением, управление производственными процессами и т. д.);

обнаружение ошибок (устранение ошибок в комплексных технических системах, например, обнаружение ошибок в телефонных сетях и т. д.);

конструирование (например, объединение элементов в общую конструкцию, удовлетворяющую определенным требованиям, например, установление конфигурации ЭВМ, проектирование схем и т. д.);

консультационные системы (правоведение, страхование и др.).

Литература

1. Appelrath H.— J. Die Erweiterung von DB-und IR-Systemen zu wissensbasierten Systemen//in Nachrichten fur Dokumentation.— 1985.— 36.— № 1.— S. 13.
2. Sell P. S. Expert Systems — A Practical Introduction Macmillan Publishers LTD.— 1985.
3. Hayes-Roth F., Waterman D., Lenat D. Building Expert Systems-Addison-Wesley.— 1983.

УДК 681.3.06

ВЕНГЕРСКИЕ СИСТЕМЫ ПРОЛОГ

И. ФУТО, инженер (ВНР)

В Прологе в отличие от традиционных языков программирования дается не алгоритм решения задачи, а описание задачи, ее определение. Способы решения, его ход, алгоритм решения находит сам Пролог, поэтому его называют языком нового типа, являющимся одним из языков программирования, используемых в исследованиях по искусственному интеллекту (ИИ). В отличие от остальных языков ИИ его синтаксис прост, а семантика хорошо определена благодаря тому, что он основывается на математической логике.

Первый интерпретатор Пролога был создан в Марселе в 1973 г. на основе теоретических работ Р. Ковальски и П. Найес под руководством А. Колмерауэр [1]. До начала 80-х годов Прологом занимались в основном академические круги. Начиная с 1981 г., когда появился японский проект ЭВМ пятого поколения, где Пролог, а более точно его расширенный вариант, был выбран в качестве основного машинного языка,— данный язык становится очень популярным. Это доказывается тем, что библиография о Прологе к 1984 г. содержала более 1000 публикаций.

В Венгрии интерпретатор Пролога, написанный на Фортране, был получен в 1974 г. Однако данный вариант не мог быть реализован на имеющейся машине, поэтому в 1975 г. был разработан новый вариант, написанный на языке CDL [11].

Таким образом, разработкой интерпретатора Пролога Венгрия весьма быстро включилась в исследования, связанные с этим языком. Эта деятельность не осталась на уровне простых исследований, ее результаты сразу же стали использоваться на практике [4], в первую очередь в области фармакологии, защиты окружающей среды и проектирования квартир [2, 5, 9]. Это объясняется двумя причинами. Во-первых, разработка проходила в среде, где тесно сотрудничали друг с другом исследователи и пользователи; во-вторых, в Венгрии не было других языков ИИ для исследований такого характера.

В 1978 г. работы, связанные с усовершенствованием Пролога, были переведены в Институт по координации вычислительной техники, где началось создание новой модульной системы Пролог на языке CDL2, используемой в промышленных масштабах. Первый вариант был создан в 1981—1982 гг., по времени удачно совпав с появлением японского проекта 5-го поколения. Этим и современностью продукта, его хорошим качеством можно объяснить тот факт, что МПролог (модульный Пролог) был применен на практике в 18 странах четырех континентов, у более чем 100 пользователей. В 1982 г. был разработан Т-Пролог [8], дальнейшее развитие Пролога для моделирования дискретных систем, а затем в 1985 г. появились мини-Пролог для персональных вычислительных машин, а также Т/ТС-Пролог [6, 7], пригодный для моделирования комбинированных дискретно-непрерывных систем. Эти системы работают практически на всех распространенных типах машин, таких, как IBM, VAX, Siemens, Tektronix и совместимых с ними ЭВМ, а также на различных машинах с микропроцессором 68 000.

Рассмотрим основные элементы программ на Прологе. В программах на Прологе существуют три типа команд: факт, правило вывода и цель.

Факт соответствует описанию какого-либо простого утверждения, выражающего для нас истину, с заданием отношений, например, (1) «дует ветер», (2) «страна (венгрия)», «любит (собака, кость)».

Синтактически правильное толкование описанных выше частей программы на Прологе: (1): дует ветер, (2): Венгрия — страна, (3): собака любит кость.

Следующий элемент программы — *правило вывода.* Правило вывода описывает утверждения, зависящие от условий, например:

```
(1) плохая_погода: — дует_ветер («:—» обозначает «если»)
(2) есть _столица (X):— страна (X)
(3) едет _отдыхать (Кто _то):— есть (отпуск, Кто _то),
                                любит (отдыхать, Кто_ то),
                                стоит _хорошая_погода.
```

Приведенные выше синтактически правильные части программы означают следующее: (1): плохая погода, если дует ветер, (2): у X есть столица, если X — страна, (3): кто-то едет отдыхать, если у кого-то есть отпуск, этот кто-то любит отдыхать и к тому же стоит хорошая погода. Правило вывода, таким образом, означает, что утверждение, стоящее по левую сторону от знака «:—», истинно, если и все утверждения, стоящие в правой стороне, истинны. Утверждения, стоящие по правую сторону знака «:—», будем называть условиями утверждения, стоящего по левую сторону. В случае

```
a: — b, c
```

выполнение «в» и «с»— условие выполнения «а».

В этих примерах, однако, встречается и новое понятие — понятие *переменной.* Переменные в Прологе пишутся с большой бук-

вы или большими буквами, этим они отличаются от констант. Таким образом, в рассматриваемом примере «X» «Кто-то»— переменные, а «венгрия», «собака», «кость»— константы (поэтому Венгрия в примере была написана с маленькой буквы).

В заключение рассмотрим последний основной элемент — *цель*, с помощью которого может быть поставлен интересующий нас вопрос.

(1) ? плохая_погода.
(2)? есть_столица (X).
(3) ? едет_отдыхать (Y).

Описанные выше цели обозначают:

(1): плохая погода?; (2) у X есть столица?
(3): едет отдыхать Y?

На эти вопросы Пролог пробует ответить с помощью фактов, хранимых в базе данных, и правил вывода.

Рассмотрим следующую полную программу на Прологе:

```
(1) есть (отпуск, мария).
(2) любит (отдыхать, мария).
(3) стоит_ хорошая_погода.
(4) едет_отдыхать (Кто_то):—
                        есть (отпуск, Кто_то),
                        любит_отдыхать (Кто_то),
                        стоит_хорошая_погода.
(5)? едет_отдыхать (мария).
```

Здесь факты (1)—(3) описывают настоящее положение Марии и погоды, (4) определяет факт, когда кто-то едет отдыхать, а (5) формулирует вопрос, едет ли отдыхать Мария.

Пролог решает задачу по так называемому правилу сверху вниз, слева направо. Это означает, что Пролог начинает решение задачи с цели и ищет такой факт или правило вывода, с помощью которого может быть достигнута цель. Если такой факт есть, после соответствующей подстановки переменных Пролог переходит к решению следующей цели при том условии, что предыдущая цель была доказана. Если эта цель была последней, тогда доказательство заканчивается: цель найдена. Если же не было такого факта, но нашлось правило вывода, которое могло быть примерно, тогда цель заменяется на условие этого правила вывода с соответствующей подстановкой переменных, так как условием выполнения цели теперь становится доказательство условий правила вывода.

Программа (1)—(5) выполняется в следующей последовательности:

```
(5) ? едет_отдыхать (мария)
(4) Кто_то:— мария
    ?есть (отпуск, мария), любит (отдыхать, мария),
(1)                                  стоит_хорошая_погода.
    ? любит (отдыхать, мария), стоит_ хорошая_погода
```

(2)
? стоит_ хорошая _погода.
(3)

Цель (5) можно сравнить с левой стороной правила (4) с присвоением значения «Кто-то:— мария» (в Прологе это называется *сравнением по образцу*). После этой подстановки получим новую цель, доказательство которой теперь является условием того, едет ли Мария отдыхать:

? есть (отпуск, мария)
любит (отдыхать, мария),
стоит_ хорошая_ погода.

Первый вопрос цели можно сравнить с фактом (1), и так как он выполняется, т. е. ответом на вопрос является «да», оставшаяся цель формулируется так:

? любит (отдыхать, мария), стоит_ хорошая _погода.

Ответ на вопрос, любит ли отдыхать Мария на основе факта (2), тоже положительный. Таким образом, остается последнее условие отдыха Марии: «стоит ли хорошая погода»

? стоит_ хорошая_ погода.

Так как на основе факта (3) это условие тоже выполняется, появляется сообщение YES.

Одна из особенностей Пролога состоит в том, что в правилах вывода, содержащих элементы, определенные только пользователем (т. е. не содержащие встроенных процедур, например, вычислительных операций, операций с файлами и процедур ввода-вывода), нет разницы между параметрами ввода-вывода. Например, в рассматриваемой выше программе в вопросе «? едет_ отдыхать (мария)» «мария» являлась входным параметром. Та же самая программа могла быть вызвана и с помощью вопроса:

? едет _отдыхать (X).

Последовательность выполнения программы будет иметь следующий вид:

(5) ? едет_ отдыхать (X)
X:— Кто_то.
(4)
? есть (отпуск, Кто_то), любит (отдыхать, Кто _то),
стоит _хорошая _погода, outterm (X).
(1)
Кто _то:— мария
? любит (отдыхать, мария), стоит_ хорошая_ погода,
outterm (мария).
(2)
? стоит_хорошая _погода, outterm (мария).
(3)
outterm (мария). (встроенная процедура, выполняется всегда)
«мария».

Программа выполнилась и на этот раз, но в то же время X получило значение «мария», и это значение появилось в качестве печати. Следовательно, параметр цели «едет_отдыхать» вел себя как выходной параметр и принятое значение передал дальше следующей цели (в нашем случае процедуре «outterm»).

Теперь посмотрим, что произойдет, если какое-то условие не выполняется. Предположим, что погода плохая. В Прологе самый простой способ отрицания выражается опущением утверждения, т. е. о чем мы открыто не утверждаем, что это истина, это не истина. Измененный вариант нашей программы имеет вид:

```
(1) есть (отпуск, мария).
(2) любит (отдыхать, мария).
(3) едет_отдыхать (Y):— есть (отпуск, Y),
                         любит (отдыхать, Y),
                         стоит_хорошая_погода.
(4) ? едет_отдыхать (мария).
    ? едет_отдыхать (мария).
```

Диаграмма работы программы имеет следующий вид:

```
(3) Y:— мария
    ? есть (отпуск, мария), любит (отдыхать, мария),
                            стоит_хорошая_погода.
(1)
    ? любит (отдыхать, мария), стоит_хорошая_погода.
(2)
    ? стоит_хорошая_погода. (это условие не выполняется)
```

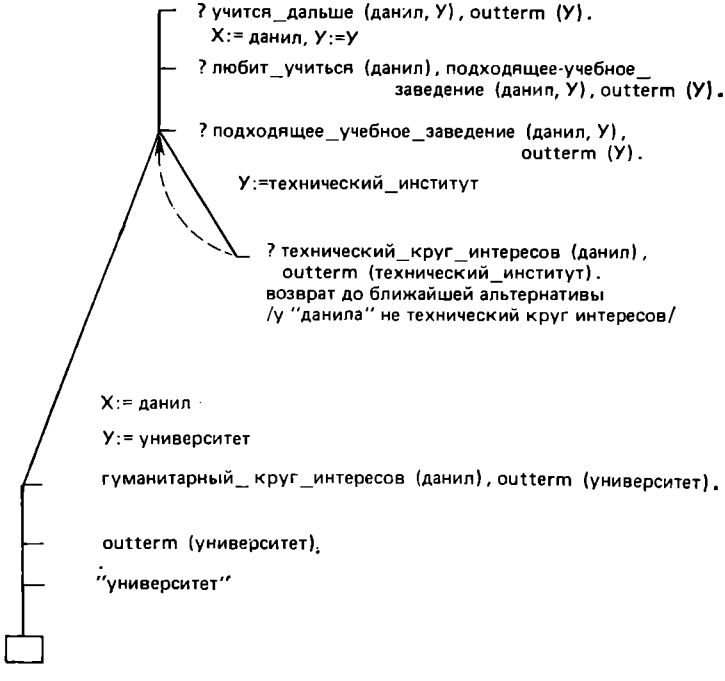
Так как в базе данных нет ссылки на хорошую погоду, Пролог предполагает, что это утверждение не истинно, а значит, оно не выполняется. В связи с этим каждое условие отдыха Марии «возвращается» назад, и программа останавливается появлением сообщения NO. Возврат может быть полезен при выборе альтернативы. Когда с помощью программы на Прологе задается множество проблем, имеется возможность и для перечисления альтернатив. Пролог по очереди обходит эти альтернативы до тех пор, пока не находит первое возможное решение. Такую ситуацию иллюстрирует следующая программа:

```
(1) учится_дальше (X, Y):— любит учиться (X),
                            подходящее_учебное_заведение (X, Y).
(2) подходящее_учебное_заведение (X, технический институт):—
                            технический_круг_интересов (X).
(3) подходящее_учебное_заведение (X, университет):—
                            гуманитарный_круг_интересов (X).
(4) любит_учиться (данил).
(5) гуманитарный_круг_интересов (данил).
(6) ? учится_дальше (данил, Y).
```

Отдельные элементы программы с диалогом выглядят следующим образом:

(1) X учится в институте Y, если X любит учиться и X подходит институт Y; (2) для X подходит технический институт, если у X технический круг интересов, или (3) X подходит университет, если у него гуманитарный круг интересов, и (5) и у Данила гуманитарный круг интересов. Вопрос (6), пойдет ли учиться дальше Данил в какое-нибудь учебное заведение Y.

Рассмотрим выполнение этой программы:



Как видно, программа автоматически возвращается на такую ближайшую точку, где может быть выбрана новая альтернатива, а затем оттуда продолжает поиск решения.

В процессе возврата Пролог стирает все присвоения значений, происшедшие до того, как программа была вынуждена возвратиться назад, до той точки, где она может выбрать новую альтернативу, и, если это необходимо, присваивает переменным новые значения. В нашем примере переменная Y сначала получила значение «технический институт», затем после возврата — значение «университет». Если новая альтернатива не оказалась успешной, в программе происходит возврат до тех пор, пока либо будет найдено возможное решение, либо, после безуспешного опробования всех возможных альтернатив, это означает, что задача не имеет решения. При этом выполнение программ остановится и будет выдано сообщение NO.

В рассматриваемых примерах использовались только очень простые структуры данных: постоянные и переменные (Надо от-

метить, что в Прологе им не нужна отдельная декларация!) Однако предусмотрена возможность использовать сложные структуры данных — так называемые выражения.

Рассмотрим самую простую из сложных структур — *список*. Список в Прологе обычно определяется с помощью оператора «.» (точка), например с помощью декларации оператора

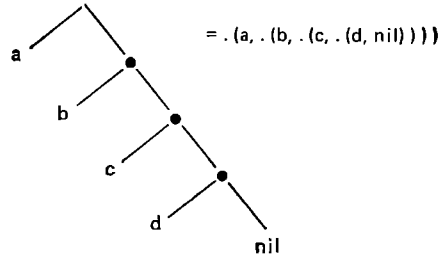
```
operator (., rl, 5)
```

что означает оператор «.» с направлением справа налево

```
(rl — tight to left)
```

и приоритетом 5. Последний элемент списка — выражение nil (пустой список).

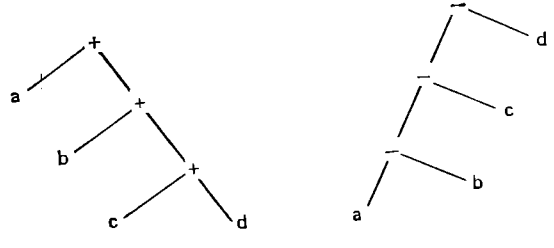
Таким образом, a. b. c. d. nil обозначает в Прологе список, состоящий из элементов a. b. c. d. Его внутренний вид:



Это важно знать потому, что при сравнении по образцу важно, с каким направлением определен оператор. Пусть будет два оператора, определение которых

```
operator (+, rl , 8).
operator (—, rl , 6).
```

При этом выражения «a+b+c+d» и «a—b—c—d» имеют следующий внутренний вид:

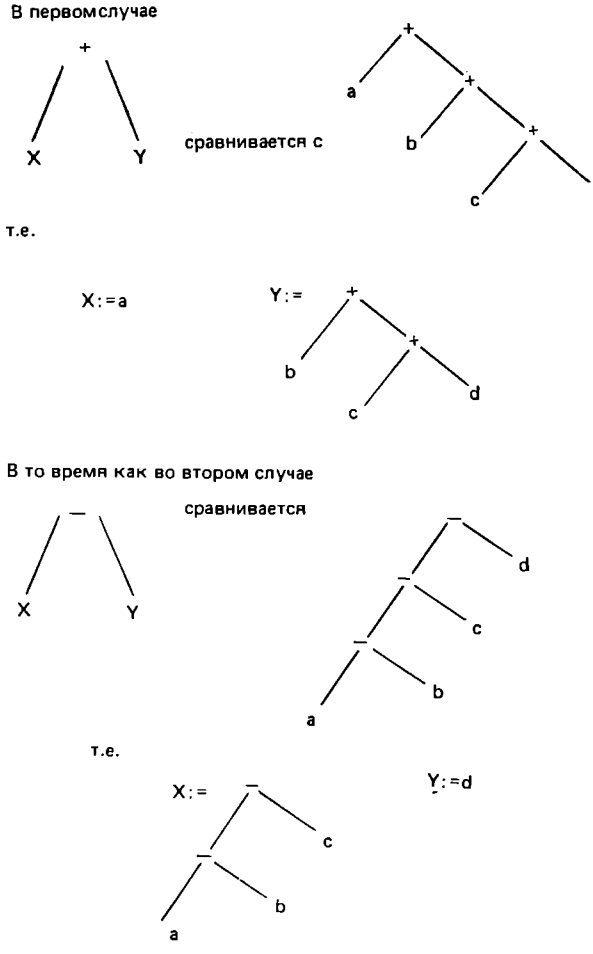


Предположим, что программа состоит из двух фактов:

(1) выражение (a+b+c+d).
(2) выражение (a—b—c—d).

и ставятся следующие вопросы:

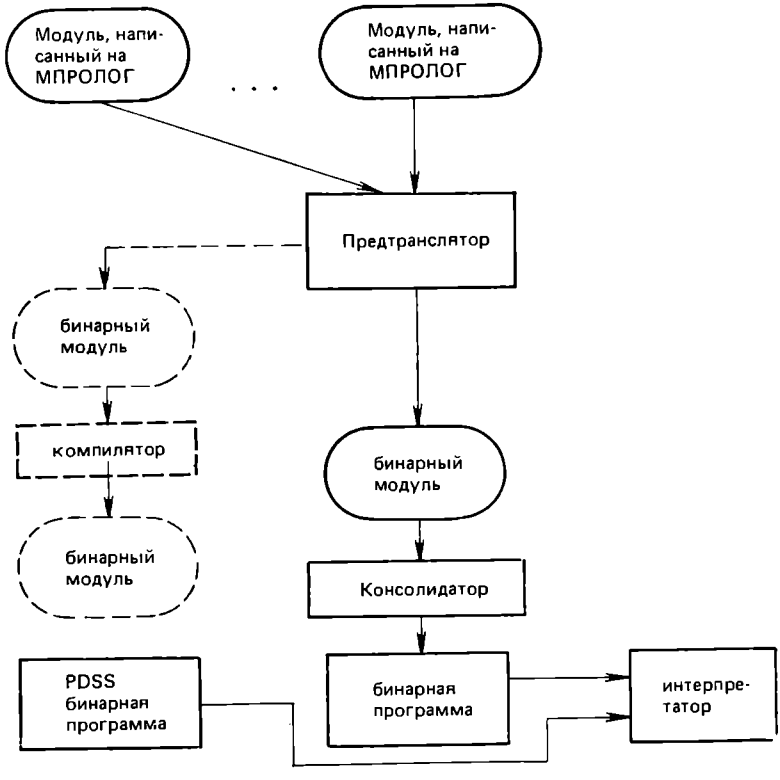
(3) ? выражение (X+Y), outterm (X).
(4) ? выражение (X—Y), outterm (X).



Мы рассмотрели самые важные свойства Пролога. Для эффективного программирования, а также для формулирования отдельных понятий необходимы также специальные процедуры управления. Одной из важнейших процедур является процедура «/» (slash либо в новой терминологии cut). Более сильное действие оказывает процедура /(A) (slash ancestor), которая обеспечивает большой скачок в процессе возврата. Объем статьи не позволяет рассмотреть действие этих процедур. Пролог обладает еще одной интересной, хотя, по мнению многих, «нечестной» возможностью, которая заключается в том, что в процессе выполнения программы может быть изменена первоначальная программа — с помощью стирания либо записи новых фактов и правил вывода.

Рассмотрим системы программирования Пролог венгерской разработки.

Система МПролог (см. рисунок) состоит из следующих элементов [3]: интерпретатора; предтранслятора; консолидатора; компилятора; среды для развития программы на МПрологе (PDSS — Program Development SubSystem).



Структура системы программирования МПролог

Предтранслятор считывает модуль, написанный на МПрологе, осуществляет лексическую и синтактическую проверку, статическую семантическую проверку, оптимизирует модуль и составляет компактную внутреннюю структуру.

Консолидатор — редактор связей на уровне МПролога. Он из бинарных модулей составляет бинарную программу, которая может быть передана интерпретатору.

Интерпретатор выполняет последовательность целей, заданную в главном модуле.

Компилятор — специальное усовершенствование транслятора. Он изготовляет из бинарных модулей бинарные модули с прежней структурой и внешним интерфейсом, а интерпретируемую внутреннюю структуру внутри модуля заменяет на машинные коман-

ды. Этот подход обеспечивает редактирование связей интерпретируемых и компилируемых модулей.

Среда PDSS написана на самом языке МПролог. Это означает, что использование PDSS означает в то же время и применение интерпретатора, где эта среда рассматривается как бинарная программа.

МПролог в настоящее время обладает 250 встроенными процедурами, обеспечивающими функции ввода-вывода, вычислительные функции и т. д., которые были сформулированы на основании опыта практического применения.

T/TC-Пролог. Для моделирования на вычислительных машинах оказалась полезной комбинация концепции времени и процесса с непроцедурной философией программирования на Прологе. В результате был создан T/TC-Пролог, усовершенствованный вариант МПролога.

Основные элементы систем программирования T/TC-Пролог:

процесс (компонент). Каждому компоненту модели принадлежит параллельно выполняющийся во времени процесс. Каждому процессу принадлежит цель, которая выполняется с помощью виртуальной системы МПролог;

симулируемое время. К определенным шагам программы независимо от действительного времени их выполнения может быть вызван (подключен) промежуток времени. Внутренние часы с помощью диспетчера в порядке очереди осуществляют выполнение акции;

сообщения и условия. Выполнение процессов может быть прервано в то время, когда они ожидают сообщений либо выполнения какого-то условия. Для формулирования условий может быть использован весь МПролог;

возврат во времени. Вся система полностью, включая и время, может возвратиться в одно из предыдущих состояний для того, чтобы, выбрав другую альтернативу, попытаться найти решение поставленных целей. Возврат может быть вызван недостатком какой-либо информации, состоянием dead-lock или же превышением времени, предписанного для решения задачи (в симулируемой единице времени).

Литература

1. Colmereaer A., Kanoui H., Pasero R., Roussel P. Un Systéme de Communication Homme-nechine en Francais.— Groupe d'Intelligence Artificielle Université d'Aix Marseille.— 1973.

2. Darvas F., Futó I., Szeredi P. A logic-based programming system for predicting drug interactions International Journal of Biomedical Computing. — 1978. — V. 9. — P. 259—271.

3. Dömölki B., Szeredi P. PROLOG in Practice.— Information Processing 83 (IFIP) Elsevier Science Publishers B. V. North Holland.— 1983.— P. 629—636.

4. Futó I., Darvas F., Cholnoky E. Practical Applications of an AI Language (PROLOG).— Second Hungarian Computer Science Conf. — 1977. — P. 388—399.

5. Futó I., Darvas F., Szeredi P. The Applications of PROLOG to the Development of QA and DBM Systems Gallaire H., Minker J. Logic and Data Bases Plenum Press New York. — 1978. — P. 347—376.
6. Futó I., Gergely T. A Logical Approach to Simulation (TS—PROLOG) Wedde H.— Adequate Nodeling Modeling of Systems. — Berlin: Spriner Verlag, 1983. — P. 27—52.
7. Futó I, Combined Discrete/Continuous Modeling and Problem Solving 1985 SCS Multiconference on AI, Graphics and Simulation, San Diego California.— 1985.— P. 23—28.
8. Futó I., Szeredi P. A Discrete Simulation System Based on Artificial Intelligence Methods//Jávor A. Discrete Simulation and Related Fields.— North Holland.— 1982.— P. 135—156.
9. Márkusz Es. How to design variants of flats using programming language PROLOG, based on mathematical logic// IFIP'77.— Toronto.— 1977.— P. 885—890.
10. Szeredi P., Santhané-Toth E. PROLOG Applications in Hungary//Clari K. I., Tarnlund S. A. Logic Programming.— Academic Press-London.— 1988.— P. 19—31.
11. Szeredi P. PROLOG a Very High Level Language Based on Predicate Logic//Second Hungarian Computer Science Conference. — Budapest. — 1977. — P. 853—866.

УДК 681.3.06

КАЧЕСТВО ПРОГРАММНЫХ СРЕДСТВ, ПУТИ И МЕТОДЫ РЕШЕНИЯ

В. П. КУПРИЯНОВ,
канд. экон. наук (СССР),
С. Л. КОТОВ, инженер (СССР),
А. В. СМАГИН, инженер (СССР)

Развитие представлений о программных средствах (ПС) как о продукте производственно-технического назначения вызвало необходимость определения их качественных характеристик. С этой целью в Совете по применению средств вычислительной техники (СП СВТ) в начале 1986 г. был образован временный коллектив специалистов по разработке общей методики оценки качества программных средств. Этому предшествовало изучение состояния проблемы. Был организован специальный семинар по управлению качеством программного обеспечения. На основе экспертных методов оценки качества программных средств был подготовлен анализ основных результатов [1].

В конце 1986 г. была разработана Общая методика оценки качества программных средств, а в начале 1987 г. она принята 11-м заседанием СП СВТ как временная для получения опыта ее использования при оценке качества в конкретных программных средствах.

Большой вклад в подготовку материалов, разработку разделов методики и ее обсуждение внесли руководители авторских коллективов С. Лиловски (НРБ), К. Геберт (ГДР), Л. Ярабек (ПНР), В. Куприянов (СССР), И. Рехтачек (ЧССР). Ценные замечания к методике были сделаны В. В. Шураковым и В. П. Морозовым.

Концепция разработки методики. Разработка методики базировалась на концепции технологии программирования, принятой в странах — участницах Соглашения в 1985 г. В ее основу положены этапы жизненного цикла программных средств [2]. Качество программного средства закладывается на ранних этапах жизненного цикла и формируется на всем его протяжении. Причем если распределение количества ошибок по этапам относительно равномерно (этап функционального специфицирования — 30%, этап проектирования — 30%, этап реализации — 40%), то трудоемкость исправления ошибок, возникших на ранних этапах жизненного цикла, в дальнейшем неизмеримо возрастает: например, исправление ошибок в спецификациях на порядок выше, чем в реализации [3]. Отсюда следует, что недостаточно фиксировать качество законченного ПС, а необходимо осуществлять контроль за качеством в процессе создания ПС. Система оценки качества должна строиться путем организации измерения конкретных оцениваемых элементов в контрольных точках, выбранных на каждом этапе жизненного цикла ПС. Причем для каждой контрольной точки выделяются элементы, отражающие существенные для данного этапа аспекты качества. По результатам оценки принимается решение о продолжении разработки или о возврате к предыдущему этапу для доработки.

Такая система оценки качества ПС должна опираться на унифицированные и стандартизованные методы и средства проектирования и программирования, применение современных организационных мероприятий, методов и средств оценки качества ПС.

Необходимо отметить, что некоторые элементы методики внедрялись в процессе ее разработки в программные проекты в ГДР (НП «Роботронпроект» в Дрездене) и в СССР (НПО «Центрпрограммсистем» в Калинине) и дали положительные результаты.

Основные положения методики. Предлагается представить взаимосвязь показателей качества ПС в виде иерархической структуры, состоящей из следующих уровней: фактор — критерий — метрика — оценочный элемент. Ниже приводятся лишь определения элементов без претензий на исчерпывающую полноту.

Под *фактором* качества понимаются некоторые свойства ПС, привлекательные для пользователя. В обширной литературе можно встретить до 80 таких показателей.

Из практики работы с пользователями можно выделить важнейшие показатели для современного применения СВТ. Ниже приведены шесть отобранных с помощью статистической обработки пользовательских оценок факторов качества и дано их определение.

Фактор	Определение
Гибкость	Легкость адаптации к новым функциональным требованиям, например при изменении области применения или других условий его функционирования
Корректность	Степень соответствия реализуемому алгоритму и установленным требованиям к обработке данных, общесистемным требованиям, а также наличие ссылок на связанные с ними документы (программы)
Надежность ПС	Способность ПС выполнять заданные функции, предусмотренные программной документацией при отклонениях, возникающих в среде функционирования (аппаратные, программные отклонения или ошибки)
Сопровождаемость	Возможность устранения отклонений в процессе эксплуатации и поддержания ПС в актуальном состоянии
Удобство применения	Возможность освоения и эксплуатации ПС с минимальными трудозатратами
Эффективность	Полнота и скорость реализации специфицированных функций на заданных вычислительных ресурсах

Любой из указанных выше факторов определяется *критериями* качества, каждый из которых в свою очередь определяется несколькими *метриками.* Критерии в отличие от факторов являются по своим свойствам уже программно-ориентированными. В методике выделены 23 критерия и дано их определение.

Ниже приведены фрагмент методики (из раздела, подготовленного специалистами НРБ), показывающий критерии фактора корректности, и их определение.

Критерии	Определение
Полнота	Полнота описания и реализации всех функций ПС, предусмотренных ТЗ, и соответствие программной документации и текстам программ
Непротиворечивость	Однозначное, непротиворечивое описание объектов и функций в различных частях программной документации и текстах программ
Согласованность	Однозначное использование терминалов, определений, символов и т. д.
Соответствие нормативно-техническим документам	Соответствие ПС и его документации нормативно-техническим документам и технологии программирования
Тестируемость	Возможность проверки соответствия всех компонентов ПС требованиям спецификации
Логическая правильность	Последовательность реализации заданных функций и режимов работы при эксплуатации ПС в условиях, указанных в спецификациях

Специалистами других социалистических стран подготовлены определения других критериев-факторов — «гибкость», «надежность» и т. п. Обобщение и согласование определений, структуры и взаимосвязи показателей осуществлялось авторским коллективом совместно.

Ниже показано соответствие всех согласованных друг с другом критериев принятым факторам.

Фактор	Критерии
Гибкость	Широта охвата функций
	Гибкость структуры
	Мобильность
	Модифицируемость
Корректность	Полнота
	Согласованность
	Соответствие
	Тестируемость
	Логическая правильность
Надежность	Помехоустойчивость
	Устойчивость функционирования
	Работоспособность
Сопровождаемость	Структурность
	Простота конструкции
	Наглядность
Удобство применения	Легкость освоения
	Пригодность документации пользователя
	Удобство эксплуатации и обслуживания
Эффективность	Функциональность (выполнение функций)
	Точность вычисления
	Затраты времени
	Используемые ресурсы

Термин «метрика» (измерение) широко используется в литературе по вопросам оценки качества ПС. В методике под метрикой понимается регламентированная экспертиза программных документов, например, описания системной архитектуры, спецификации на модули, описания тестовых примеров и т. д. Результатом измерения является протокол экспертизы, заполняемый экспертом в ходе оценки качества.

На рисунке приведена в качестве примера структура фактора «сопровождаемость» на этапе реализации.

Каждая методика определяется набором *оценочных элементов.* В нашей методике представлено свыше 250 оценочных элементов, сведенных в словарь-справочник. Разделы словаря соответствуют названию фактора, внутри раздела все оценочные элементы классифицируются по метрикам. Фрагмент словаря-справочника оценочных элементов из методики для сопровождаемости на этапе «реализация ПС» представлен в табл. 1.

Методика оценки. Оценка качества ПС осуществляется исходя из следующих принципов квалиметрии [4]. Исходя из первого принципа квалиметрии оценка качества ПС производится по уровням иерархии системы начиная с нижнего уровня (оценочный элемент). Характеристики качества каждого вышестоящего уровня определяются характеристиками нижестоящего. Для характеристик ка-

Структура фактора «сопровождаемость» на этапах реализации, тестирования и изготовления

чества ПС на всех уровнях иерархической структуры принимается единая шкала оценки от 0 до 1. При оценке качества на каждом уровне приводятся вычисления характеристик качества ПС, т. е. определение абсолютных показателей P_{ij} (j — порядковый номер характеристик данного уровня для i-й характеристики вышестоящего уровня).

Для критериев определяется (в соответствии со вторым принципом квалиметрии) относительный показатель K_{ij} по формуле

$$K_{ij} = \frac{P_{ij}}{P_{ij}^{\text{баз}}},$$

где $P_{ij}^{\text{баз}}$ — базовое значение показателя.

Таблица 1. **Фрагмент справочника оценочных элементов**

Код элемента	Оценочный элемент	Метод оценки	Оценка
С0803	Наличие комментариев в точках входа и выхода программы	Экспертный	0—1
С0302	Общее количество точек входа в программу (Д)	Расчетный	$\frac{1}{(Д+1)} \cdot \frac{1}{(F+1)}$
С0303	Общее количество точек выхода из программы (F)	То же	
С1002	Общее количество переходов по условию (А)	»	
С1003	Общее количество исполняемых операторов (В)	»	$1-\frac{A}{B}$
С0305	Наличие контроля за правильностью данных, поступающих в вызывающий модуль от вызываемого	Экспертный	0—1
С0802	Наличие комментария к машинно-зависимым операторам программ	То же	0—1
С0101	Наличие схемы иерархии модулей программы	»	0—1
С0602	Соблюдение принципа структурного программирования	»	0—1

Каждая характеристика качества 2-го и 3-го уровней (критерий и метрика) характеризуется двумя числовыми параметрами: абсолютным показателем P_{ij} (для критерия — относительным показателем) и весомостью этой характеристики (в соответствии с пятым принципом квалиметрии). Сумма весомостей характеристик качества ПС одного уровня — величина постоянная, равная определенному числу (согласно шестому принципу квалиметрии).

Оценка качества ПС заканчивается получением конкретных значений, которые нецелесообразно сводить к интегрированному показателю, поскольку программное средство может оцениваться с самых различных сторон, а оценки часто противоречивы.

Общая оценка качества ПС определяется по наименьшему из полученных значений фактора для данного класса ПС.

Организационные аспекты экспертизы. Целесообразно определить следующие основные этапы проведения экспертизы, последовательность которых может изменяться в зависимости от реальных условий и ограничений:

определение цели экспертизы и разработка метрик оценки;

формирование группы испытателей;

отбор и формирование группы экспертов;

анализ и обработка информации, полученной от испытателей;

проведение экспертизы;

синтез статистической информации и информации, полученной в результате экспертизы, с целью приведения ее в форму, удобную для принятия решения.

В общем случае процесс оценки осуществляется в следующей последовательности:

1) эксперты на основании ТЗ на конкретное ПС заполняют анкету, в которой указывают перечень факторов качества, выбранных для исследования;

2) заполненные анкеты служат основанием для составления плана и методики испытаний соответствующего программного средства в группе испытаний; результаты испытаний оформляются в виде структурной схемы;

3) структурная схема поступает к экспертам для последующей оценки конечных результатов.

Области применения. Эффективность практического использования методики во многом зависит от правильного выбора метрик и оценочных элементов, которые должны отразить суть качественного состояния данного программного средства. Поскольку номенклатура программных средств широка, а области их применения разнообразны, поэтому, чтобы определить универсальный набор метрик и оценочных элементов, эффективных при оценке качества различных типов программных средств, целесообразно ввести двухуровневую классификацию «класс программного обеспечения — объект автоматизации», представленную табл. 2. В дальнейшем с учетом этой классификации необходимо пополнять и вести словарь-справочник оценочных элементов.

Подход, предложенный в методике, не только позволяет проводить оценку качества программных средств в процессе разработки, но и обеспечивает возможность формирования требований (со стороны областей применения) к необходимому и реально достижимому научно-техническому уровню различных типов программных средств.

Введя в методику базовые значения показателей для различных уровней оценки (с учетом предложенной классификации), мы тем самым в явном виде формулируем требования к качеству программных средств, которое необходимо обеспечить в процессе разработки. Обеспечение данных требований является целью современных технологий программирования. Если брать во внимание затраты на достижение этих целей, то можно на их основе проводить оценку эффективности и самих технологий.

Первоначально предполагалось, что Общая методика оценки качества программных средств будет использована в основном как методическое, а в перспективе — и как инструментальное средство технологии программирования, поэтому выполнение работ было предусмотрено детализированной программой сотрудничества по проблеме 1.1.6 «Развитие технологии разработки и промышленного производства программных СВТ» КП НТП СЭВ до 2000 г.

В процессе проработки материалов были выявлены универсальный характер методики, ее пригодность для оценки качества программных средств, разрабатываемых в рамках проблемы 1.1.7 «Разработка программных и технических средств систем управления базами данных и знаний», проблемы 1.1.8 «Создание совмест-

Таблица 2. **Учет оценочных элементов и их базовых значений**

Объект автоматизации	Класс программного обеспечения											
	ПС расширения ОС				ПС, производство и исследование ПО				ПС общего назначения			
	ОС	телеобработка	сети ЭВМ	организация вычислительного процесса	системы программирования и трансляторы	технология программирования	метрологические ПС по оценке качества ПО	СУБД и ИПС	методоориентированные ПС	для решения экономико-организационных задач		
АСОУ												
АСУ ГАП												
АСУ ТП												
АСНИ												
САПР пром. изд.												
ИПС												
АСО												
АЭС												
АС ОВП												
САПР ПО												

Обозначения:

АСОУ	— автоматизированная система организационного управления
АСУ ГАП	— автоматизированная система управления гибкими производствами
АСУ ТП	— автоматизированная система управления технологическими процессами
АСНИ	— автоматизированная система научных исследований
САПР пром. изд.	— система автоматизированного проектирования промышленных изделий
ИПС	— информационно-поисковая система
АСО	— автоматизированная система в обучении
АЭС	— автоматизированные экспертные системы
АС ОВП	— автоматизированная система организации вычислительного процесса
САПР ПО	— система автоматизированного проектирования программного обеспечения

ного фонда алгоритмов и программ (ИнтерФАП)» и некоторых других проблем.

Заключение. Практической и теоретической основой развития данной методики должно стать ведение и постоянное расширение словаря-справочника оценочных элементов, а также методов их измерения и оценки и, как следствие, уточнение базовых значений факторов, критериев, метрик.

В перспективе предусматривается создание на основе сбора и систематизации самой разнообразной информации о качестве программных средств автоматизированной экспертной системы. Результатом совершенствования организационных форм обеспечения качества в странах может быть создание государственных испы-

тательных центров. Опыт функционирования государственного испытательного центра в СССР убедительно свидетельствует о правомерности и практической ценности подхода к оценке качества программных средств, сформулированных временным коллективом специалистов по вопросам оценки качества программ в рассматриваемой методике.

Литература

1. Котов С. Л., Третьяков В. В. Экспертные методы в оценке качества программных средств: Методические рекомендации.— Калинин: НПО «Центрпрограммсистем», 1986.— 25 с.
2. Куприянов В. П., Кальниш Г. И., Шарков В. П. Концепция технологии программирования, принятая в странах сотрудничества//Вычисл. техника соц. стран.— 1985.— Вып. 18.— С. 7—14.
3. Гантер Р. Методы управления проектированием программного обеспечения.— М.: Мир, 1981.— 322 с.
4. Азгальдов Г. Г., Райхман Э. П. О квалиметрии.— М.: Стандарты, 1973.— 171 с.

УДК 681.3.068+519.685

ИНВАРИАНТНОЕ ЯДРО ПАКЕТОВ ПРИКЛАДНЫХ ПРОГРАММ

В. В. ПОДБЕЛЬСКИЙ,
канд. техн. наук (СССР)

Естественный путь для снижения затрат на создание программных комплексов — стандартизация и унификация проектных решений при программировании. Один из подходов к решению этой проблемы состоит в том, чтобы выделить из существующих ППП для решения определенного круга задач типовые компоненты и создать на основе анализа их функций такой комплекс, который был бы достаточно инвариантен к смене предметных областей и допускал бы настройку на каждую из них. Программный комплекс с указанными возможностями будем называть *инвариантным ядром* пакетов прикладных программ определенной области применения.

Рассмотрим создание инвариантного ядра ППП для случая, когда это ядро должно служить основой для специализированных программных комплексов, позволяющих выполнять многоэтапные научно-технические расчеты. Анализ применения ЭВМ для научно-технических расчетов показывает, что унификации соответствующих алгоритмов и программных средств можно достичь, если положить в ее основу схему «объект исследования — метод исследования», где объектом служит математическая модель или таблично-заданная функциональная зависимость. Количество типов при-

меняемых на практике методов исследования таких объектов сравнительно невелико, и в научном плане эти методы достаточно хорошо разработаны. В связи с этим, если ввести стандартные представления математических моделей и функциональных зависимостей и настроить на эти стандартные представления программы, реализующие типовые методы, можно получить программный комплекс, пригодный для решения задач из разных предметных областей. При этом целесообразно учитывать, что научно-технический расчет на ЭВМ достаточно часто является многоэтапным, причем этапы одного расчета информационно связаны друг с другом и при переходе к очередному этапу может меняться как объект, так и метод его исследования. Необходимую преемственность этапов должна обеспечить обоснованно выбранная операционная среда [1] модулей комплекса. Определяя особенности построения инвариантного ядра ППП, следует обратить внимание и на организацию многосеансных расчетов, для чего в ядре ППП должна быть предусмотрена специализированная база данных для контрольных точек расчетов.

В качестве стандартного представления математической модели в инвариантном ядре ППП выбран (рис. 1) абстрактный, ориентированный неантисипативный объект с конечным числом входов, выходов и конечной памятью [2]:

$$M: \{|R(\bar{X}) \times |R[\bar{U}(t)]\} \to \{|R(\bar{Y}) \times |R[\bar{Z}(t)]\}, \qquad (1)$$

где M — отображение;

$|R(\bar{X}) = |R^n$ — пространство параметров;

$\bar{X}$ — вектор параметров, $\bar{X} \in |R(\bar{X})$, $\dim \bar{X} = n$;

$|R[\bar{U}(t)] = |R(\bar{U})$ — пространство входных воздействий, $\bar{U}(t) \in |R[\bar{U}(t)]$, $\dim \bar{U}(t) = k$;

$|R(\bar{Y}) = |R^m$ — пространство показателей;

$\bar{Y}$ — вектор показателей, $\bar{Y} \in |R(\bar{Y})$, $\dim \bar{Y} = m$;

$|R[\bar{Z}(t)] = |R(\bar{Z})$ — пространство реакций;

$\bar{Z}(t)$ — вектор реакций, $\bar{Z}(t) \in |R[\bar{Z}(t)]$, $\dim \bar{Z}(t) = l$;

t — аргумент (независимая переменная), изменяющаяся при экспериментах с абстрактной моделью на интервале $(t_н, t_в)$;

$\times$ — операция декартово произведения;

n, k, l, m — натуральные числа.

Отображение (1) может представлять большое количество различных математических моделей, причем в конкретных случаях те или иные компоненты могут опускаться.

На модели (1) могут быть решены следующие задачи: получение реакций и показателей при заданных параметрах и входных воздействиях; построение зависимостей показателей от вариации в заданных пределах одного или двух параметров (графики изменений показателей, линии уровня показателей и т. п.); оценка чувствительности показателей и реакций к вариациям параметров; бе-

зусловная оптимизация показателей (одно- или многокритериальная); статистический анализ показателей и реакций при заданных законах вероятностного изменения параметров; статистический анализ реакций при известных характеристиках распределений входных воздействий.

Перечисленные задачи типичны для вычислительных экспериментов на математических моделях и широко используются в различных предметных областях. Однако предложенный набор методов недостаточен, например, для решения задач технического проектирования [3]. Здесь при исследовании математической модели обычно приходится задавать функциональные ограничения на параметры $\bar{X}$, показатели $\bar{Y}$, входные воздействия $\bar{U}(t)$, реакции $\bar{Z}(t)$ и на сочетания перечисленных компонентов. Функциональные связи между компонентами обычно известны при создании математической модели исследуемого объекта, и их целесообразно ввести в (1), не меняя внешней структуры отображения. Для этого необходимо расширить вектор $\bar{Y}$ показателей и считать показателями значения не зависимых от аргумента t ограничений $\bar{F}=\bar{F}[\bar{X}, \bar{Y}, \bar{U}(t), \bar{Z}(t)]$. Подобным образом можно увеличить размерность вектор-функции реакций $\bar{Z}(t)$, учтя с помощью дополнительных реакций зависимые от аргумента t функциональные связи между элементами исходной математической модели объекта:

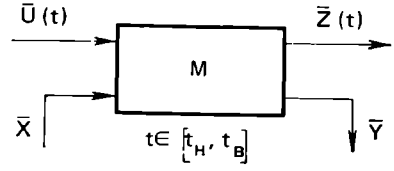


Рис. 1. Формальное представление математической модели

$$\bar{W}(t)=\bar{W}[\bar{X}, \bar{Y}, \bar{U}(t), \bar{Z}(t)].$$

Расширив модель за счет функциональных связей между ее компонентами, введя вектор $\bar{A}=\{a_1, a_2, ..., a_n\}$ масштабных множителей параметров и систему неравенств

$$\begin{aligned} \bar{X}_{min} \leqslant \bar{X} < \bar{X}_{max}, \\ \bar{Y}_{min} \leqslant \bar{Y} \leqslant \bar{Y}_{max}, \\ \bar{Z}_{min}(t) \leqslant \bar{Z}(t) \leqslant \bar{Z}_{max}(t), \end{aligned} \qquad (2)$$

можно формулировать и решать следующие задачи: условная параметрическая оптимизация показателей (решение общей задачи математического программирования); поиск допустимой области, выделяемой в пространстве параметров ограничениями (2); исследование конфигурации допустимой области (построение сечений, граничные испытания); синтез допусков на параметры; увеличение (оптимизация) допусков на параметры за счет направленного изменения их номинальных значений; статистический анализ, учитывающий требования (2) к показателям и реакциям.

В соответствии с (1) операционную среду математической модели при ее программной реализации образуют представления векторов $\bar{X}$, $\bar{Y}$ и вектор-функций $\bar{U}(t)$, $\bar{Z}(t)$. Для учета функциональных ограничений на параметры, показатели и реакции модели в операционную среду необходимо ввести векторы $\bar{X}_{min}$, $\bar{X}_{max}$, $\bar{Y}_{min}$, $\bar{Y}_{max}$ и вектор-функции $\bar{Z}_{min}(t)$, $\bar{Z}_{max}(t)$. Разную значимость вариаций параметров при параметрических исследованиях модели позволяет учесть вектор $\bar{A}$ масштабных множителей. Операционная среда в таком составе обеспечивает связь между моделью и методом, а также преемственность по данным этапов одного расчета (рис. 2).

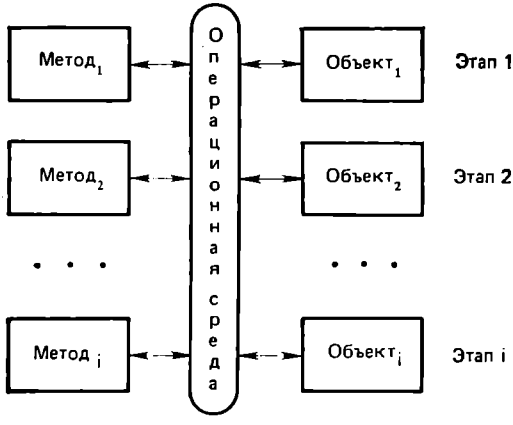


Рис. 2. Схема многоэтапной организации расчета

Во многих задачах различных предметных областей необходимо до выполнения моделирования соответствующим образом подготовить исходные данные, т. е. $\bar{X}$, $\bar{U}(t)$, а после моделирования обработать результаты, т. е. $\bar{Y}$, $\bar{Z}(t)$. Подобным образом до начала обработки может потребоваться формирование векторов $\bar{X}_{min}$, $\bar{X}_{max}$, $\bar{Y}_{min}$, $\bar{Y}_{max}$ и вектор-функций $\bar{Z}_{min}(t)$, $Z_{max}(t)$. Таким образом, векторы и вектор-функции операционной среды становятся объектами обработки. С точки зрения инструментальной поддержки одна из типовых операций над этими объектами — запоминание их значений в базе данных, т. е. создание контрольной точки сеанса. В дальнейшем состояние можно восстанавливать, т. е. значения векторов и вектор-функций операционной среды позволяют организовать многосеансное решение задач.

Как вектор, так и вектор-функция представляют собой заданную в виде таблиц функциональную зависимость. Такого рода объекты широко используются в научно-технических расчетах. Считая, как мы приняли функциональные зависимости объектом обработки, можно рассматривать ряд достаточно универсальных прикладных задач, решение которых целесообразно выполнять средствами инвариантного ядра ППП: генерация функциональной зависимости с заданными свойствами на выбранном интервале изменения аргумента; преобразование функциональной зависимости по типовому алгоритму (например, преобразование сигнала из временно́й области в частотную или формирование из белого шума сигнала с нужными статистическими характеристиками); анализ свойств функциональной зависимости (например, получение статистических характеристик случайной величины) и визуализация результатов

анализа в виде таблиц и/или графиков на терминальных устройствах; сравнительный анализ функциональных зависимостей (например, оценка уклонения одной функции от другой или получение корреляционной зависимости между двумя функциями).

Упоминаемые в перечисленных задачах функциональные зависимости могут быть получены в результате моделирования или исследования математических моделей. В свою очередь любая сформированная функциональная зависимость может быть подана как входное воздействие на модель при экспериментах с последней или использована как ограничение для реакции модели. Таким образом, этапы экспериментов с моделями и этапы обработки функциональных зависимостей могут чередоваться в любых последовательностях в зависимости от потребностей того или иного расчета.

Для достижения нужной универсальности инвариантного ядра ППП нужны дополнительные соглашения. Векторы $\bar{X}$, $\bar{X}_{min}$, $\bar{X}_{max}$, $\bar{Y}$, $\bar{Y}_{min}$, $\bar{Y}_{max}$, $\bar{A}$ естественно представлять массивами в основной памяти ЭВМ. Использование массивов для функциональных зависимостей приводит к следующим затруднениям. Во-первых, может неэкономно расходоваться основная память. Во-вторых, при табличном представлении функциональной зависимости для определения значения функции по заданному аргументу обычно приходится решать задачу интерполяции и программные средства для этого удобно подготовить заранее. В связи с этим каждый из компонентов вектор-функций будем представлять с помощью специального информационного объекта (ИО).

Информационный объект инвариантного ядра ППП определим как объединение динамической информационной структуры и программных средств поддержки этой структуры, позволяющих использовать ее для удобного представления таблично-заданной числовой функциональной зависимости. Программные средства ИО решают задачу интерполяции при обращении к нему и снабжают информационную структуру ИО механизмом виртуальной памяти. Информационные объекты создаются по явному запросу и уничтожаются при необходимости, что позволяет регулировать требования к памяти при работе инвариантного ядра ППП.

В отличие от массивов основной памяти, формирующих *статическую операционную среду,* информационные объекты образуют *операционную динамическую среду* модулей инвариантного ядра ППП. С учетом этого терминологического различия следует интерпретировать схему многоэтапной организации расчета (см. рис. 2). При создании инвариантного ядра ППП в нем выделяются следующие компоненты: управляющая программа; статическая операционная среда (СОС); программы обработки (ПО), реализующие методы обработки; программные модели (ПМ), каждая из которых соответствует отображению (1) или его редукции; операционная динамическая среда (ОДС), элементы которой взаимодействуют с ПМ или служат объектами обработки. В зависимости

от особенностей операционной системы, в среде которой функционирует инвариантное ядро, может быть принята та или иная схема взаимодействия программных компонентов: динамические связи модулей, оверлейная организация, многозадачный режим. Например, при оверлейной организации в корневом сегменте размещают (рис. 3) управляющую программу и СОС. Программы обработки перекрывают друг друга, возможно, в той же области. В другой области располагаются элементы ОДС, в следующей — программные модели. При такой структуре элементы СОС доступны всем модулям, а информационные объекты из ОДС могут взаимодействовать как с ПО, так и с ПМ.

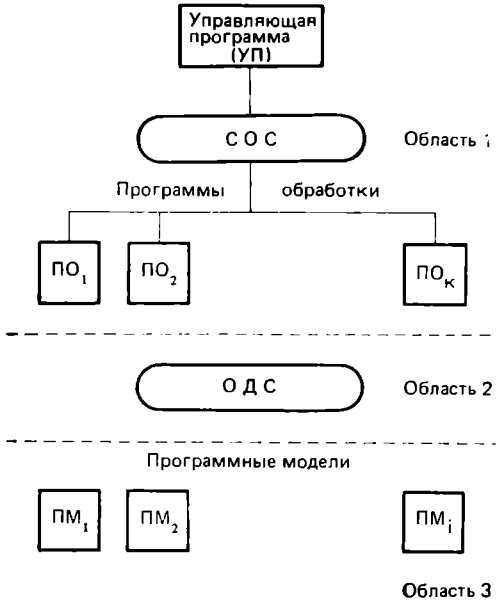


Рис. 3. Оверлейная структура реализации инвариантного ядра ППП

Инвариантное ядро предназначено для многоэтапных расчетов, поэтому для каждого этапа с помощью средств входного языка определяются: объект и метод обработки, параметры объекта, параметры метода. Между этапами предусмотрены преемственность по объекту обработки (ПМ и/или ИО сменяются только по явному указанию) и преемственность по данным. Преемственность по данным обеспечивается операционной средой (СОС и ОДС).

Для ведения базы данных многосеансных расчетов достаточно сохранять в виде контрольных точек состояния СОС и ОДС.

Инвариантное ядро ППП служит основой, на которой достаточно просто могут быть созданы специализированные проблемно-ориентированные программные комплексы для научно-исследовательских расчетов и учебных целей. Теоретически инвариантное ядро может одновременно настраиваться на любое количество предметных областей. Настройка на предметную область предполагает (рис. 4) подключение к ядру подсистемы формирования моделей. На входе подсистемы находится описание модели на языке предметной области. Выход подсистемы — модуль ПМ, структура которого не зависит от особенностей предметной области. После подключения модуля ПМ к ядру ППП возможно проведение описанных экспериментов с моделью.

На основе изложенных выше принципов построения инвариантного ядра ППП был разработан ППП ТРАП (типовые расчеты ав-

томатизированного проектирования). Он создан на ЕС ЭВМ стандартной конфигурации, ориентирован на ОС ЕС версии 6.1, а также на совместимые операционные системы. В качестве языков программирования использованы ПЛ/1, язык препроцессора ПЛ/1 и язык ассемблера ОС ЕС. Минимальный объем основной памяти желательно иметь не менее 1 Мбайта. Для библиотек на магнитных дисках необходимо выделять не менее 20 Мбайт. Требования к внешней памяти могут возрасти при увеличении количества поль-

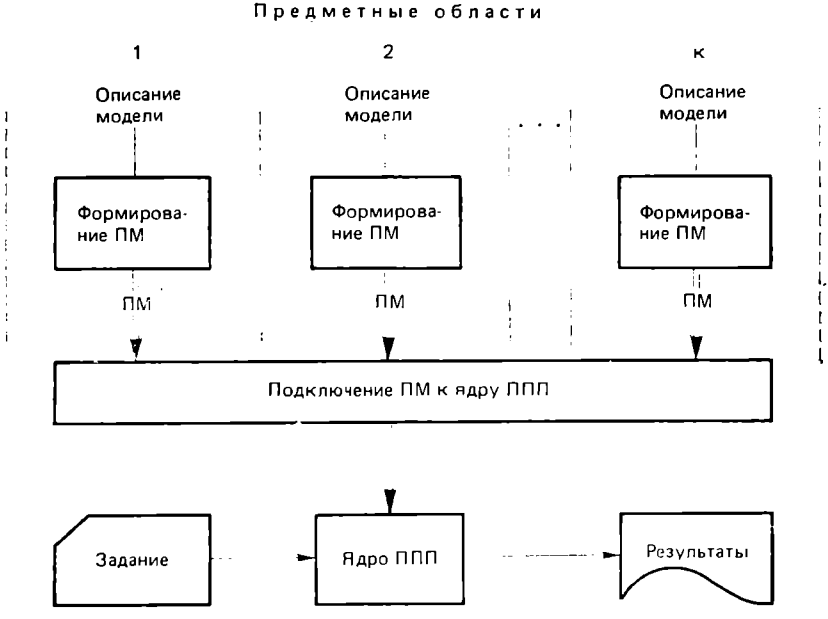


Рис. 4. Структура интегрированного программного комплекса на базе инвариантного ядра ППП

зователей и росте объема хранимой информации (исходные данные, контрольные точки).

Возможности одновременной настройки ППП ТРАП на различные предметные области и параллельного выполнения задач разными пользователями обеспечиваются следующими техническими решениями, принятыми при проектировании.

Во-первых, одновременно к ППП могут быть подключены до 99 моделей, как взаимосвязанных по данным, так и не зависимых друг от друга.

Во-вторых, одновременно могут обрабатываться до 50 функциональных зависимостей (ИО).

При эквидистантном размещении точек на оси аргумента в таблице одной функциональной зависимости (в одном ИО) может быть до 100 000 значений.

В-третьих, каждая из функциональных зависимостей может сохраняться в специализированной базе данных, объем такой базы определяется только возможностями внешних запоминающих устройств ЭВМ.

Параллельно с разработкой ППП ТРАП выполнена его настройка на решение задач из предметной области «автоматика»: созданы две специализированные подсистемы для автоматизированного построения программных моделей систем автоматического управления.

Одна из подсистем [6] ориентирована на представление исследуемого динамического объекта в виде структурной схемы, элементами которой являются типовые и нестандартные, линейные и нелинейные, динамические и безынерционные звенья. Подсистема включает библиотеку моделей звеньев, оформленных в виде макроопределений. Количество типов звеньев заранее не ограничено, и их набор можно расширять по мере появления новых объектов исследования.

Для описания динамического объекта, представленного структурной схемой, предложен специализированный язык описания моделей [6], построенный на основе препроцессорных средств языка ПЛ/1.

Вторая подсистема автоматизированного построения программных моделей формирует модель на основе математического описания в виде системы обыкновенных дифференциальных уравнений.

Язык описания моделей позволяет задавать правые части систем дифференциальных уравнений и определять процедуры оценки получаемых решений.

Национальные испытания инвариантного ядра ППП в виде комплекса ТРАП проведены в марте 1986 г.

Литература

1. Пратт Т. Языки программирования: разработка и реализация.— М.: Мир, 1979.— 514 с.
2. Заде Л., Дезоер Ч. Теория линейных систем (метод пространства состояний).— М.: Наука, 1970.— 704 с.
3. Системы автоматизированного проектирования в радиоэлектронике: Справочник/Е. В. Авдеев, А. Т. Еремин, И. П. Норенков, М. И. Песков; Под ред. И. П. Норенкова.— М.: Радио и связь, 1986. — 368 с.
4. Бендат Дж., Пирсол А. Применение корреляционного и спектрального анализа.— М.: Мир, 1983.— 312 с.
5. Пакет прикладных программ для типовых расчетов автоматизированного проектирования//Прикладное программное обеспечение Единой системы ЭВМ и системы мини-ЭВМ.— Вып. 10.— М.: МЦНТИ, 1986.— С. 66—68.
6. Подбельский В. В. Макрогенерация имитационных моделей//Электронное моделирование. — 1985. — № 4. — С. 35—39.

УДК 681.3.001.72

ОБ УНИФИКАЦИИ ТЕХНОЛОГИИ ПРОЕКТИРОВАНИЯ СОД

Э. Н. ХОТЯШОВ,
д-р экон. наук, профессор (СССР)

Комплексная программа научно-технического прогресса стран — членов СЭВ до 2000 г. предусматривает создание значительного количества АСУ различных классов. В связи с этим создание перспективной технологии проектирования систем обработки данных (СОД) должно стать одним из важных направлений сотрудничества, не уступающих по важности технологии программирования.

СОД является составной частью автоматизированной системы управления. В то же время СОД включает в себя совокупность программ, обеспечивающих непосредственную обработку данных. В научных публикациях зачастую отождествляются технология проектирования систем обработки и технология программирования. Однако эти понятия не идентичны. Рассмотрим основные отличия этих понятий.

Первое отличие состоит в том, что при проектировании СОД активно используются программные продукты, такие, как пакеты прикладных программ (ППП), системы автоматизированного проектирования, инструментальные средства проектирования, составляющие некоторую базу проектирования. При этом общая тенденция состоит в таком увеличении мощности базы проектирования, чтобы СОД создавались без участия профессиональных программистов. Такая тенденция приводит к универсализации специалистов по созданию СОД.

В конечном итоге процесс проектирования СОД должен сводиться к следующему. Пользователь СОД или аналитик по соответствующей предметной области формулирует информационные потребности, которые представляются в виде некоторой спецификации. Затем спецификация формализуется и с помощью средств базы проектирования превращается в программное обеспечение проектируемой СОД вместе с соответствующей документацией.

При создании СОД база проектирования играет более существенную роль, чем инструментальные средства, используемые при создании программного продукта.

Второе отличие состоит в том, что методы проектирования СОД существенно отличаются от методов создания программных продуктов. В некоторой степени это отличие является следствием первого.

Методы проектирования СОД разделяются на три класса:

ручное проектирование;

типовое проектирование, которое разделяется на элементный, подсистемный и объектный подклассы;

автоматизированное проектирование.

Каждый из названных классов и подклассов опирается на свою концепцию декомпозиции системы и на свою базу проектирования.

Отметим, что инструментальные средства, используемые при проектировании СОД и создании программных продуктов, пересекаются. Более того, как правило, инструментальные средства создания программ включаются в совокупность средств проектирования СОД.

При проектировании СОД используется целый ряд специфических средств. Так, в элементном подклассе используются типовые проектные решения (ТПР), в подсистемном — функциональные ППП, в объектном подклассе — типовые проекты СОД для соответствующего класса объектов управления.

Средства проектирования автоматизированного класса методов ориентированы на то, что на основании некоторого описания предметной области и ограничений на создаваемую СОД в человеко-машинном режиме осуществляется проектирование СОД.

Результатом проектирования СОД является проект, который включает документацию проектных решений, в том числе программы создаваемой СОД.

Технология проектирования СОД должна ориентироваться не на профессионального программиста; в ней должно учитываться то, что СОД проектируют главным образом аналитики и специалисты-пользователи.

Важным требованием к технологии проектирования является обеспечение устойчивости СОД в процессе функционирования. Известно, что любая организационно-экономическая система управления объективно подвержена изменениям. Меняются методики расчета тех или других показателей, меняются формы входных и выходных документов, изменяются требования к регламенту получения результатов и др. Естественным является требование к СОД — обеспечить адекватность СОД реальным информационным процессам на объекте управления за счет модификации СОД. Это вызывает необходимость предусматривать в СОД адаптивные свойства, с помощью которых обеспечивается автоматическая подстройка СОД к изменившимся на объекте условиям. Кроме того, проект СОД должен предоставить пользователю (без участия проектировщиков системы) возможность модификации проекта СОД в процессе функционирования.

Технология проектирования СОД должна быть формализованной в силу следующего. Средств проектирования СОД достаточно много. Нереально предположение, что проектировщик СОД может освоить все средства, тем более что их количество постоянно увеличивается. Поэтому надо ориентироваться на то, что каждое средство имеет некоторую технологию проектирования, построенную по формализованным канонам. В таком случае разработчик СОД бу-

дет в состоянии пользоваться практически любыми средствами проектирования с минимальными затратами на их освоение.

На основании изложенного можно сделать вывод о том, что создание формализованной технологии проектирования СОД актуально и принесет наибольший эффект в рамках международного сотрудничества.

Рассмотрим один из возможных вариантов формализованного описания процесса проектирования СОД. Его основу составляет понятие *технологической операции проектирования.*

Определение 1. Технологической операцией (ТО) проектирования будем называть кортеж ТО=<V, P, W, R, S>, где V — вход ТО; P — преобразователь; W — выход ТО; R — требуемые для выполнения преобразователя P ресурсы; S — используемые преобразователем P средства проектирования.

Технологическая операция ТО=<V, P, W, R, S> определяет некоторый процесс, в котором на основании заданного входа V выполнением преобразователя P получается требуемый выход W. При этом необходимы ресурсы R (например, труд, машинное время) и будут использованы средства проектирования S.

Вход V и выход W технологической операции составляют компоненты проектирования, которые разделим на четыре класса: Д — документы, П — параметры, U — универсумы (универсальные множества), G — программы.

Приведем определения структурных составляющих ТО.

Определение 2. Документ Д — это описатель некоторых фактов, условий, требований, количественных и качественных параметров. По функциональному назначению документы ТО разделим на два типа: конечные документы, в полном объеме входящие в проект разрабатываемой системы, и промежуточные, используемые в качестве входных компонентов для других технологических операций. Отметим, что некоторые документы могут одновременно принадлежать и к промежуточному, и к конечному типам.

Определение 3. Параметр П — это количественная характеристика, некоторое условие или некоторое ограничение на проектируемую систему, заданные в явном виде.

Примерами параметров ТО могут служить: объем финансирования, выделяемый на разработку системы; календарные сроки разработки; площадь, отводимая под вычислительный центр; число работающих на объекте управления и т. п. Из определения 3 следует, что параметры могут рассматриваться как подкласс документов. Однако для лучшей формализации процесса проектирования их удобнее выделять в самостоятельный класс.

Определение 4. Универсум U — это полный перечень возможных значений некоторого компонента ТО или полный объем знаний о нем.

Универсум характеризуется многообразием элементов, которые исчерпывающе на определенный момент времени отражают состояние некоторого компонента проектирования или задают перечень его возможных состояний. Будем различать универсумы двух ти-

пов: проектные и инструментальные. *Проектные универсумы* являются результатом проектирования СОД, они полностью включаются в соответствующий проект. Используемое в специальной литературе понятие «номенклатура постоянной информации» является частным случаем введенного понятия проектного универсума. Примерами универсумов могут служить различные классификаторы: материалов и покупных изделий, работ и услуг, управленческой документации и т. п.

Инструментальные универсумы включают некоторые классы инструментальных средств. Например, перечень систем управления базами данных (СУБД), применение которых допустимо в рамках определенной технологии проектирования, может быть представлен в виде универсума. Элементами такого универсума будут выступать описания соответствующих СУБД.

Определение 5. Программа G — это некоторое проектное решение по реализации заданной функции управления объектом или обработки данных для управления, представленное одним из возможных способов записи программ.

В процессе создания СОД программы могут иметь различные состояния и, следовательно, различные формы документального отображения этих состояний: функциональные спецификации; программные спецификации; HIPO-диаграммы; блок-схемы алгоритмов; алгоритмы, записанные на одном из алгоритмических языков на специальных бланках, и др. Документально зафиксированные состояния программ призваны обеспечивать корреспонденцию между различными технологическими операциями проектирования при создании программного обеспечения СОД.

Определение 6. Преобразователь P — это некоторая методика, формализованный алгоритм или машинный алгоритм преобразования входа технологической операции в ее выход.

Исходя из определения можно выделить три типа преобразователей: ручные, человеко-машинные и машинные. В качестве примера ручного преобразователя может выступать методика проектирования форм документов. Примером человеко-машинного преобразователя может служить сценарий интерактивной отладки программ с помощью соответствующих инструментальных средств. Транслятор с алгоритмического языка является представителем машинно-реализуемых преобразователей.

Определение 7. Ресурсы R — это нормированные значения трудовых, материальных и технических (машинных) ресурсов, необходимых для выполнения преобразователя P с помощью средств проектирования S.

Определение 8. Средства проектирования S — это программно-алгоритмические системы, предназначенные для автоматизации процесса проектирования и используемые при выполнении преобразователя P.

С помощью технологических операций проектирования процесс разработки СОД может быть однозначно описан технологической сетью проектирования, являющейся результатом взаимоувязки опе-

раций по их входам и выходам. В такой интерпретации технологическая сеть представления является одним из вариантов сетей Петри [1]. Используя технологическую сеть проектирования, можно практически в любой точке процесса разработки сделать «срез» технологической сети и по содержанию компонентов проектирования однозначно идентифицировать состояние проекта.

Всю совокупность преобразователей, определяющих содержание соответствующих технологических операций по созданию СОД, разделим на несколько больших классов.

Поиск и выборка. В процессе проектирования, особенно на начальных его этапах, необходимо постоянно вести целенаправленный поиск концепций, аналогов, методических материалов, универсумов, которые могут оказаться полезными и которые необходимо учитывать при разработке СОД. Поиск необходимой информации в проектных универсумах (например, выбор СУБД из универсума доступных СУБД) будем называть *выборкой.* Выборка осуществляется на основе поискового образа, который может задаваться в виде дескрипторов, совокупности параметров или в виде логических выражений. При этом множество, удовлетворяющее поисковому образу, может быть пустым, состоять из одного элемента или включать некоторый перечень элементов. В последнем случае возможна постановка экстремальной задачи для выбора оптимального проектного решения.

Поисковые операции могут выполняться как вручную, так и с помощью машинных информационно-поисковых систем и вестись как среди документированной, так и среди фактографической информации. Естественно, что при этом правила формулирования поисковых образов и правила выполнения поиска будут различаться.

Формализация расчета показателей. Исходными данными (исходными компонентами проектирования) для этого класса преобразователей выступают действующие ручные методики расчета показателей, которые, как правило, не удовлетворяют требованиям машинной обработки данных. Наиболее перспективным направлением разработки формализованных методик расчета показателей является техника построения элементарных двудольных графов для каждого рассчитываемого показателя с заданием всех операций (арифметических и логических), которые необходимо осуществлять над первичными показателями для получения значений рассчитываемого показателя [2].

Создание инструментальных универсумов. Смысл этого класса преобразователей заключается в отслеживании теоретических концепций и практического опыта создания СОД и формирования на этой основе универсумов по методам проектирования, пакетам прикладных программ, системам классификации и кодирования, формам документов для различных классов объектов, системам ввода-вывода и др.

При формировании структуры универсумов кроме кода, наименования, аннотации и документации по каждому элементу универ-

сального множества необходимо задать некоторые параметры, которые позволили бы сравнивать при реализации операции поиска и выборки.

Управление метаданными. При проектировании СОД необходимо описывать данные (реквизиты, показатели, составные единицы информации (СЕИ), массивы, базы данных), задавать их структуру, назначение, связи и способы представления. Требуется также описывать формы входных и выходных документов и формы терминальных сообщений. Для обеспечения единообразия и непротиворечивости использования названных информационных образований используется централизованное отображение в виде словаря данных (метаданных). Словари данных организуются по принципу сетевых баз данных. Словари данных являются мощными вспомогательными средствами при разработке СОД. Для создания, ведения и использования словарей данных требуется определенный набор ручных, человеко-машинных и машинных процедур.

Параметризация компонентов СОД. Такая параметризация проводится с целью обеспечения возможности сравнения различных вариантов компонентов проектируемой системы, а также с целью формирования качественных проектных универсумов.

Выбор общесистемных проектных решений. Общесистемные решения определяют основные принципы любой создаваемой СОД и включают технические решения:

по комплексу технических средств, в том числе выбор центрального процессора, средств сбора и передачи информации;

по программному обеспечению, в том числе выбор версии операционной системы, систем программирования, средств телеобработки;

по информационному обеспечению, в том числе выбор систем классификации и кодирования и способа организации данных, определение целесообразности использования интегрированных баз данных;

по методам и средствам проектирования, в том числе выбор инструментальных средств, пакетов прикладных программ и систем автоматизированного проектирования.

Общесистемные проектные решения выбираются на основании соответствующих инструментальных универсумов. Алгоритм выбора, как правило, сводится к построению и использованию экономико-математических моделей. При построении моделей необходимо учитывать взаимные влияния между проектными решениями по различным аспектам проектирования СОД.

Использование инструментальных средств проектирования. В данный класс преобразователей включаются операции, которые определяются используемыми инструментальными средствами проектирования, в том числе по документированию проектных решений, по использованию словаря данных, по управлению процессом проектирования и др.

Преобразование программ. К этому классу относятся традиционные операции по трансляции, синтаксису, семантическому контролю, сборке, каталогизации программ.

Преобразование алгоритмов. К этому классу относятся операции по машинной ориентации алгоритмов, включая алгоритмы прямого счета и экономико-математических моделей.

Проведение контроля. В процессе проектирования СОД проводится контроль правильности проектных решений, с тем чтобы ошибки, допущенные на промежуточных этапах, не привели к некорректности проекта в целом. Различают два типа контроля: смысловой (семантический) и организационный. Отметим, что детальный контроль, например, после каждой технологической операции существенно увеличивает затраты ресурсов на контрольные операции. С другой стороны, проведение контроля на небольшом числе технологических операций может привести к заметному увеличению затрат ресурсов на переделку проектных решений. Следовательно, при проектировании СОД необходимо вырабатывать такую стратегию контрольных операций, чтобы минимизировать математическое ожидание затрат на контроль и переделки. Особое место в контрольных операциях занимает тестовая проверка законченного проекта.

Декомпозиция процесса разработки СОД на технологические операции и концепция технологической сети могут быть с успехом использованы при разработке технологических документов, описывающих применение инструментальных средств проектирования, в частности при описании технологических сетей применения ППП. Сущность применения ППП заключается в формировании параметрического потока, настраиваемость на который предусмотрена в соответствующем пакете, и идентификации действий проектировщиков при возникновении недопустимых ситуаций в процессе генерации ППП. Целесообразно описывать порядок формирования каждой позиции параметрического потока и порядок устранения каждой нежелательной ситуации в процессе генерации ППП с помощью технологических операций проектирования. В результате объединения таких операций по входам и выходам создается технологическая сеть, которая исчерпывающе и просто описывает технологию применения соответствующего ППП в процессе проектирования СОД. Отметим, что такое представление заметно упрощает изучение и освоение пакетов проектировщиками.

Изложенный подход формализованного описания совокупности действий по созданию СОД в виде технологических операций и концепция технологической сети проектирования создают хорошие предпосылки для эффективного управления процессом разработки по цепочке «планирование — учет — контроль — анализ — регулирование». Действительно, технологическая сеть проектирования, содержащая все компоненты проекта, включая промежуточные, все преобразователи, а также ресурсы, требуемые для выполнения операций, и используемые средства проектирования, после привязки к конкретной организационной структуре проектной организации

представляет собой модель процесса проектирования. Такая модель позволяет планировать процесс, отслеживать траекторию развития проекта и вырабатывать корректирующие воздействия.

Таким образом, предложенный подход к формализованному описанию процессов разработки обеспечивает необходимые условия для создания хорошо структурированной технологии проектирования систем машинной обработки данных.

Некоторые элементы описанного подхода нашли применение в конкретных разработках. Так, в ряде организаций СССР, например в Ивановском ГПКИ АСУ, достаточно давно используются специальные карты технологических операций, содержание каждой из которых соответствует одной технологической операции. Концептуально такие карты представляют собой реализацию некоторых элементов рассмотренного подхода к проектированию СОД без применения ЭВМ. Отметим, что в Институте проблем информатики АН СССР в настоящее время разрабатывается машинная система проектирования СОД, построенная в соответствии с изложенными положениями. Система ориентирована на использование ППП, описанных технологическими сетями проектирования СОД.

Литература

1. Питерсон Д. Теория сетей Петри и моделирование систем.— М.: Мир, 1984.— 263 с.
2. Королев М. А., Мишенин А. И., Хотяшов Э. Н. Теория экономических информационных систем.— М.: Финансы и статистика, 1984.— 233 с.

УДК 681.3.068

СИСТЕМА УПРАВЛЕНИЯ БАЗОЙ ДАННЫХ MIMER

Д. ШРАЙТЕР, д-р техн. наук (ГДР)

Система управления базами данных (СУБД) MIMER[1] включает следующие компоненты (концепция «семейства»): DB — Data Base Handler, QL — Query Language, SH — Screen Handler, PG — Program and Application Generator, RG — Report Generator, ST — Statistic Package, IR — Information Retrieval. Помимо этого для работы с базой данных в распоряжении пользователя имеется графический интерфейс. Система дополняется различными программами для автоматического восстановления в случае отказов системы, для контроля (и повторной установки) бан-

[1] Реляционная СУБД MIMER разработана вычислительным центром университета г. Упсала. После образования в 1984 г. фирмы MIMER Information System AB (Упсала) за дальнейшее развитие этой СУБД и ее сбыт отвечает эта фирма. На СУБД выдана лицензия, так что ее адаптация и применение возможны только при согласии разработчика.

ков данных и для приема/передачи структур и содержимого банка данных.

Версия 3.2 написана на языке программирования Ада и частично на Лисп, что одинаково удобно как для пользователя, так и для системного администратора. В результате работы предтранслятора генерируется код языка Фортран (Фортран IV или Фортран 77). Этим обеспечивается исключительно удобная переносимость системы на множество ЭВМ и операционных систем: от типа ЭВМ зависит лишь менее 5% программ.

Основной вариант СУБД MIMER включает, как правило, компоненты DB, QL, SH и RG, поэтому его можно применять также на 16-битовых персональных ЭВМ, работающих под управлением MS DOS и UNIX. СУБД MIMER с полным объемом функций имеется для следующих вычислительных систем: IBM (370, 43XX, 30XX), UNIVAC (1107, PRIME 250/750), DEC (VAX 730, 750, 780; DEC 20, 10), Control Data (CYBER), Burroughs (6800, 7800), Hewlett-Packard (9000 UNIX), Data General (MV 4000, 6000, 8000, 10000) и др.

Перенос на модели ЕС ЭВМ и СМ ЭВМ проходит относительно легко. Для производимых в настоящее время на НП «Роботрон» 32-битовых ЭВМ (Robotron K 1840) и 16-битовых персональных ЭВМ (ЕС1834, СМ1910) трудностей при переносе СУБД не встречается. СУБД MIMER соответствует современному мировому уровню и включает функции, которые у других СУБД еще не являются стандартными. Ниже приведены ее важнейшие свойства [1].

Реляционная база данных. СУБД MIMER базируется на реляционной модели Е. Кодда [2], исключительно удобной во многих отношениях. Все наборы данных объединяются в системе в базу данных. База данных состоит из системных банков данных (SYSDB, SYSQL, TRANSDB, LOGDB), сколь угодно большого количества банков данных пользователя (включая PROCDB) и последовательных файлов (файлов для передачи данных, файлов печати). Каждый банк данных с точки зрения операционной системы образует файл (массив). Во время одного сеанса одновременно может открываться произвольное количество банков данных. Они в свою очередь состоят из произвольного количества таблиц (реляций, отношений). Каждая таблица имеет один первичный ключ (столбцы для ключа), по которому производится запись с сортировкой в порядке возрастания ключа. В каждый столбец, не занятый ключом, в любой момент может быть занесен вторичный индекс, который может быть стерт. Доступ к таблицам, которые содержатся в различных банках, возможен посредством одной команды.

Длина одной строки (в байтах) и количество описателей машинно-независимы. Архитектура системы, примененная в СУБД для базы данных, исключительно гибкая и высокоэффективная.

---

[1] Применение СУБД MIMER в здравоохранении ГДР описано в [1].

СУБД MIMER — полностью и равномерно реляционная система. В соответствии с определением по Е. Кодда [3] СУБД тогда полностью реляционна, когда она поддерживает базу данных с табличной структурой и позволяет синтаксически представлять реляционные операции (Projection, Selection, Foin), а также операции теории множеств (Union, Intersection, Difference). Е. Кодд однородно реляционной определяет такую СУБД, которая поддерживает как конечного пользователя посредством языка высокого уровня, так и прикладного программиста путем предоставления ему соответствующих интерфейсов с методо-ориентированными языками. При таком подходе СУБД MIMER можно считать как полностью, так и однородно реляционной. Язык высокого уровня для конечного пользователя в компоненте MIMER/QL содержит требуемые команды с удобными для пользователя синтаксисом и семантикой. Начиная с версии 4.1 реализован также псевдостандарт языка SQL. Языковой интерфейс для подключения реляционных базовых программ СУБД MIMER/DB к программам пользователя существует для языков Фортран, Кобол, Паскаль, ПЛ/1, Лисп и Пролог.

Словарь. Важнейшая управляющая информация для базы данных и СУБД хранится в активном словаре данных (обычном банке данных с именем SYSDB). Это касается всех сведений для банков данных, таблиц, столбцов, пользователей, а также их общих и специальных прав на доступ. Начиная с версии 4.1 в SYSDB содержится также информация, касающаяся новых возможностей View, Domain и Report.

Много- и однопользовательский режимы. Стандартно работа ведется в многопользовательском режиме. Для того чтобы расширить базу данных, не мешая процессу вычислений, можно работать и в режиме одного пользователя. При этом пользователь является одновременно и администратором своего банка данных. Отдельные базы данных можно использовать без ограничений как в том, так и в другом режимах.

Защита данных. Она ориентирована на пользователей и на банки данных (начиная с версии 4.1 она касается таблиц и различных точек зрения). В качестве пользователя может выступать пользователь X (администратор банка данных) и пользователь S (обычный пользователь). Администратор банка данных является «владельцем» базы данных и имеет все права. Для банков данных предусмотрены следующие общие права, касающиеся доступа: R — чтение; S — чтение и запись; X — расширенный доступ, включая право на определение таблиц; P — приватный банк данных — общий доступ к нему отсутствует; B — банк данных. Кроме того, отдельным пользователям могут предоставляться специальные права на доступ, которые заменяют общие права на доступ.

Каждый допущенный к системе пользователь «представляется» ей посредством своего имени и пароля (Password). Пароль (код защиты) закодирован необратимо.

Целостность/непротиворечивость. Целостность (интегрированность) и непротиворечивость (согласованность) базы данных обес-

печивается главным образом за счет управления транзакциями и за счет восстанавливаемости (Recovery). Семантические условия целостности можно сформулировать по-разному. Управление транзакциями (которое может и «отключаться») использует метод Optimistic Concurrency Control [4], в котором высокая защищенность соединяется с высокой эффективностью, а также системный банк данных TRANSDB. Управление транзакциями предусмотрено для всех языковых уровней. При отказах системы защита данных обеспечивается посредством автоматического восстановления. Системная программа DBBRU использует для этого копию Backup непротиворечивого состояния банка данных и информацию обо всех изменениях в банке данных после последнего восстановления, заносимую в системные банки данных LOGDB и TRANSDB.

Физическое хранение данных. Физически все записи хранятся в виде дерева B$*$. Это означает, что область памяти разделяется на страницы. Требуемое количество уровней для индексных страниц устанавливается автоматически. Сами записи данных располагаются полностью на «листьях» дерева. Данный метод весьма эффективен, поскольку страницы загружены в среднем на $^5/_6$. При этом создается побочный положительный эффект — автоматическая реорганизация базы данных при операциях «включение» и «стирание» записей данных. В обычной среде при помощи только четырех индексов можно обслуживать примерно 100 млн. записей.

Распределенная база данных. СУБД для поддержки распределенной базы данных в чистом виде (единый Data Dictionary) пока существует только в виде лабораторных образцов. Однако система MIMER уже обеспечивает удобную пересылку содержимого баз данных в другие ЭВМ (распределение или объединение). Хорошей поддержкой служат при этом команды системного компонента QL COPY ... TO или COPY ... FROM и особенно вспомогательная программа DBEXIM, которая осуществляет «экспорт» структуры и содержимого банка данных из СУБД и ЭВМ и «импорт» этой информации в другую систему банков данных и другую ЭВМ.

Системная поддержка. В системном банке данных SYSQL в основном хранится информация функции Help, касающаяся синтаксиса и семантики команд компонента QL; сообщения об ошибках; системные процедуры. Благодаря этим компонентам использование СУБД MIMER на базе языка высокого уровня QL широко поддерживается комментариями, и поэтому при работе с дисплеем можно обойтись без письменных руководств.

Ниже описываются функции компонентов СУБД MIMER.

MIMER/DB. Data Base Hardler MIMER/DB — ядро системы. Оно представляет собой управляемую словарем данных реляционную многопользовательскую СУБД, которая реализует физическую структуру базы данных, логическое определение файлов (массивов), размещение и актуализацию данных. Все остальные компоненты СУБД основаны на ядре системы. Готовые подпрограммы MIMER/DB при помощи унифицированного интерфейса (обычные FORTRAN-Calls) могут компоноваться в прикладные программы,

написанные на языках Фортран, Кобол, Паскаль, ПЛ/1, Лисп и т. д. В библиотеке модулей хранятся как все необходимые подпрограммы банка данных, так и машинно-зависимые подпрограммы (MDR). Архитектура системы способствует гибкости ее применения (например, специальные адаптации САПР)и создает удобства для прикладного программиста. Начиная с версии 4.1 предоставляется еще один интерфейс высокого уровня (программный интерфейс PT и MIMER Data Language-MDL).

MIMER/QL. В интерактивной среде этот компонент служит для определения и загрузки баз данных, а также для манипулирования с их содержимым. Имеются следующие группы команд:

для определения среды (5 команд);

для описания среды (3 команды);

для индикации метаданных, типа данных о структуре банка данных и структуре таблиц (2 команды);

для манипулирования с метаданными (6 команд):

DBA — для определения/повторного определения/стирания/ /индикации банков данных, пользователей и прав на доступ;

DEFINE/REMOVE TABLE — для определения/удаления таблиц;

DEFINE/REMOVE INDEX — для определения/удаления вторичного индекса;

для манипулирования с данными:

INSERT — для включения новых строк в таблицу;

UPDATE — для замены значений по столбцам в одной или нескольких строках таблицы;

DELETE — для стирания одной или нескольких строк в одной таблице;

COPY ... FROM/TO — для считывания/записи записей в последовательном массиве и наоборот (при необходимости с управлением выборкой и управлением форматом);

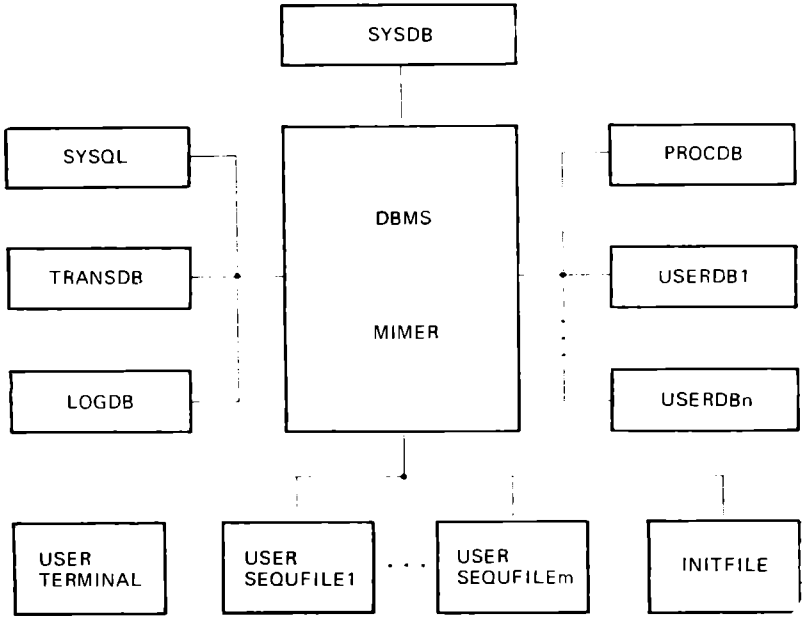
для поиска данных в базе данных: GET — для поиска данных в таблицах (даже если они расположены в различных банках данных); содержит операции Selection, Projection и Join; вывод результата на терминал и/или на печать или в рабочую таблицу.

Процедурная концепция в СУБД MIMER/QL. Это мощное дополнение к языку QL. Благодаря ему можно заранее определять последовательности команд языка QL и затем комбинировать их на терминале с готовыми наборами команд меню (Promptings). Дополнительные операторы процедур позволяют реализовать возможности, близкие к возможностям языка Бейсик. Процедуры запоминаются в PROCDB (см. рисунок). Интегрированный редактор процедур позволяет без труда писать процедуры, актуализировать их, копировать, стирать и т. д. даже конечному пользователю с небольшими знаниями в области электронной обработки данных.

MIMER/SH. Screen Handler (SH) служит для разработки прикладных решений в форме «неподвижных» изображений для сбора, контроля и индикации данных в комбинации с базой данных. Поскольку эти «изображения» имеются в форме загрузочных модулей,

которые скомпонованы с соответствующими модулями для манипулирования и опроса базы данных, время ответа очень мало. SH — вспомогательное средство для прикладного программиста, для поддержки которого имеются редактор SH и массивы команд для компилирования и компоновки программ.

MIMER/PG — это система для разработки прикладных решений при использовании прототипов и для создания прикладных прог-



Система управления базами данных MIMER

рамм и форм отчетов (протоколов) на языках Кобол или Фортран. Она может использоваться для реализации функций ввода в реальном масштабе времени (включая проверку на достоверность — семантические условия интегрированности), расчетов, актуализации таблиц, пакетной обработки, определения отчетов, управления опросом и администрирования банком данных. Система работает интерактивно, по сравнению с обычными методами она проста в обслуживании и позволяет значительно экономить время и затраты на разработку, тестирование и обслуживание. Эта система обеспечивает следующие дополнительные по отношению к MIMER/QL функции:

определение общих задач на языке высокого уровня;

немедленное выполнение специфических прикладных решений и компилирование их (при необходимости) после проведенного тестирования;

определение отчетов для генерирования очень сложных отчетов, включая расчеты;

генерирование программ на языке Кобол или Фортран исходя из «специфицированных прикладных решений». Эти программы могут генерироваться для ЭВМ, которая не обязательно должна быть идентична той ЭВМ, на которой работает компонент PG (например, генерирование прикладной программы на ЭВМ ЕС1056 для СМ-4);

полностью реляционный язык, включая кванторы FORALL и EXIST, заранее определенные функции и функции, определяемые пользователем;

всевозможные расчеты, касающиеся даты;

простой интерфейс с MIMER/SH;

функции редактора и библиотечные функции для всех сгенерированных прикладных решений на всех фазах программ.

MIMER/IR—это информационно-поисковая система (ИПС). Запросы формулируются на языке CCL (Common Command Language), который является стандартом в Западной Европе. Данная система имеет все функции ИПС (включая тезаурус) и работает полностью интерактивно. Однако ввод документов может осуществляться также и в пакетном режиме (децентрализованный сбор, например, при помощи персональной ЭВМ).

Условия применения. В ГДР СУБД MIMER используется на ЭВМ СМ-4-10 и СМ-4-20 (емкость основной памяти 256 Кбайт, ЗУ с прямым доступом на кассетах, иногда ЗУ на сменных дисках емкостью 29 Мбайт; операционная система ДОС РВ или ОС РВ). Установка системы на указанных ЭВМ очень проста. При хорошей подготовке опытный системный программист проводит установку, как правило, за 2—4 часа (DB+QL+SH). Для поддержки хода установки имеются файлы неявных команд (файлы готовых наборов команд), которые быстро выполняются в режиме диалога.

Для создания (построения) базы данных вначале требуется выполнение системной программы SYSDBGEN, которая генерирует SYSDB (продолжительность — 3 мин). После этого можно работать с MIMER/QL. Описание базы данных (банков данных, таблиц и т. д.) выполняется за несколько минут.

Требуемые ресурсы ЭВМ зависят от количества банков данных, таблиц и столбцов.

Для актуальной работы в главной памяти должны быть выделены два раздела (для многопользовательского режима). Эта процедура и некоторые глобальные назначения сведены в один файл готовых команд, так что работа с СУБД MIMER каждый раз может начинаться без существенной подготовки, если ЭВМ не полностью используется как ЭВМ банка данных.

При наличии вышеназванной конфигурации с СМ-4 могут одновременно обслуживаться 3 терминала с приемлемым временем ответа в случае, если используется компонент QL. Значительно быстрее идет работа с программами, поддерживаемыми компонентами SH и DB. ЗУ рекомендуется использовать на сменных дисках ем-

костью 29 или 100 Мбайт в связи с более высокой скоростью передачи данных. Опыт работы на устройстве СМ ЭВМ с основной памятью емкостью 1 или 2 Мбайта дал примерно 15-кратное ускорение выполнения при обслуживании до 8 терминалов.

СУБД MIMER работает в ГДР на ЭВМ ЕС1022, ЕС1040, ЕС1055 и ЕС1056. На моделях ЕС ЭВМ Ряда-1 для нее требуется приблизительно 250 Кбайт основной памяти (или меньше), на моделях ЕС ЭВМ Ряда-2 должен предусматриваться раздел размером 700 Кбайт. В качестве поддержки режима диалога используется преимущественно TSO (ЕС ЭВМ-2), а также BTAM. Для ЕС1057 (и аналогичных ЭВМ) может быть реализован полный объем СУБД MIMER. Установка системы в этом случае, как и ее применение, так же проста, как и для моделей СМ ЭВМ. Шаг выполнения SYSDBGEN отпадает, поскольку системные банки данных SYSDB, TRANSDB и LOGDB поставляются в сгенерированном виде, что обусловлено операционной системой. Время ответа на ЕС1056 под управлением TSO невелико, несмотря на наличие других пользователей в режиме TSO и фоновую пакетную обработку.

Оценка и перспективы. По имеющимся международным отзывам, СУБД MIMER называют в одном ряду с INGRES, ORACLE, RAPPORT. Тесты Berchmark и Scoring дали положительную оценку для системы MIMER (без учета возможностей MIMER/PG). Коллектив разработчиков считает, что СУБД MIMER — перспективная система.

Особо важным представляется тот факт, что эта система может работать на всех наших ЭВМ: ЕС ЭВМ, СМ ЭВМ и даже на 16-битовых персональных ЭВМ. В результате распределенные базы данных с единым словарем данных становятся реальной возможностью, как и интерфейсы с системами графической обработки и другими системами пользователя.

Литература

1. Шрайтер Д., Донат К.-Д., Штраубе Р. Применение банка данных в области здравоохранения ГДР//Вычисл. техника соц. стран.— Вып. 19.— М.: Финансы и статистика, 1986.— С. 139—145.

2. Codd E. F. A Relational Model for Large Shared Data Banks//Comm. of the ACM 13. — 1970. — N6. — S. 377—387.

3. Codd E. F. Relational Database Systems: Analysis and Comparision. — Berlin/Heidelberg/New York/Tokyo: Springer-Verlag, 1983.

4. Kunt H. T. and Robinson J. R. An Optimistic Method for Concurrency Control; Carnegie-Mellon University//ACM Transactions on Data Base Systems 6. — 1981. — N 2. — S. 213—226.

УДК 681.3.06

СИСТЕМА ПРОГРАММИРОВАНИЯ ФОРТРАН 77 ДЛЯ ЕС ЭВМ И ПЕРСПЕКТИВЫ ЕЕ РАЗВИТИЯ

Г. Д. СМИРНОВ,
канд. техн. наук (СССР),
В. И. ЦАГЕЛЬСКИЙ,
канд. физ.-мат. наук (СССР),
Э. В. КОВАЛЕВИЧ, инженер (СССР),
З. С. БРИЧ, инженер (СССР)

Ни один язык программирования не имеет такой долгой жизни, как Фортран. 30 лет назад он появился как инструмент для решения задач инженеров и научных работников, и до сих пор эта категория пользователей ему не изменяет. С годами Фортран проник и в другие области, например в обработку текстов, работу с файлами. Живучесть этого языка объясняется тем, что он полностью удовлетворяет тех пользователей, для которых он предназначен, а также его относительной простотой и высокой эффективностью создаваемых трансляторами программ. За свою длительную историю развития язык Фортран дважды подвергался стандартизации. Первый стандарт имеет обозначение ИСО 1539—1966, его теперь называют Фортран 66, второй — ИСО 1539—1980, или Фортран 77 [1].

В Фортране 77 по сравнению с Фортраном 66 существенно увеличены возможности структурирования программы, введены средства обработки текста, расширены свойства ряда операторов (в частности, операторов ввода-вывода, оператора цикла), введены другие усовершенствования.

Фортран ЕС ЭВМ. В системе программных средств ЕС ЭВМ имеются трансляторы, базирующиеся как на Фортране 66, так и на Фортране 77.

На основе Фортрана 66 разработаны трансляторы Фортран ST, SE, CC, OP, OE [2]. В этих трансляторах реализована расширенная версия языка Фортран 66. Часть расширений содержит элементы Фортрана 77, часть — учитывает архитектуру ЕС ЭВМ. Наличие такого количества трансляторов с Фортрана 66 объясняется их этапным развитием. Трансляторы Фортран ST и OP составляли неотъемлемую часть операционной системы ОС 6.1. Трансляторы Фортран CC, SE и OE являются их дальнейшим развитием в части диалоговых и отладочных средств, заменяют их и представляют собой самостоятельный программный продукт по отношению к операционным системам ЕС ЭВМ (ОС 6.1 и ОС 7.1).

Переход от Фортрана 66 к Фортрану 77 в ЕС ЭВМ. На основе языка Фортран 77 создана соответствующая система программирования [3]. В ней стандарт языка Фортран 77 реализован в полном объеме, причем возможности стандарта усилены полезными свойствами, более полно учитывающими особенности предыдущих версий Фортрана в ЕС ЭВМ. К существенным расширениям Фортрана 77 для ЕС ЭВМ относятся:

операторы и стандартные функции из предыдущих версий Фортрана ЕС ЭВМ, отсутствующие в стандарте Фортрана 77. Это упрощает проблему совместимости со всеми трансляторами предыдущих версий с Фортрана ЕС ЭВМ на уровне входного языка;

оператор, позволяющий во время трансляции подключать ранее разработанные фрагменты исходной программы;

средства для обработки данных повышенной точности;

использование выражений с операндами смешанных типов в операторах, не оговоренных стандартом;

средства для работы с данными в шестнадцатеричной системе счисления;

сочетание текстовых и арифметических данных в операторах.

Для пользователей ЕС ЭВМ, которые после применения Фортрана 66 переходят к Фортрану 77, новые свойства Фортрана 77 условно можно разделить на две группы.

К первой группе относятся возможности, расширяющие язык:

наличие данных текстового типа и операций для их обработки;

новые операторы (блочные операторы IF, ELSE IF, ELSE, END IF, оператор объявления типа CHARACTER, вспомогательные операторы ввода-вывода OPEN, CLOSE, INQUIRE, оператор объявления стандартных функций INTRINSIC, оператор объявления имени константы PARAMETER, оператор объявления имени главной программы PROGRAM, оператор включения фрагмента программы INCLUDE, оператор сохранения значений объектов из подпрограмм SAVE);

отрицательные и нулевые значения нижней границы измерения массива;

задание границы измерения массива в виде выражения целого типа в объявлении массива;

выражение в списке вывода;

ввод-вывод во внутренние файлы;

именованный модуль BLOCK DATA;

введение обобщенных имен для стандартных функций;

введение логических операций .EQV и .NEQV.;

применение операций отношения .EQ и .NE. к данным комплексного типа;

использование элементов массива в определении внутренней функции.

Вторую группу составляют новые свойства, расширяющие возможности имеющихся операторов и обеспечивающие:

задание параметров цикла DO в виде выражений целого или вещественного типа;

введение параметров UNIT, FMT, REC, ERR и IOSTAT для операторов ввода-вывода;

введение новых видов преобразования данных в операторе COMMON;

использование в операторе COMMON списка с циклом;

задание идентификатора формата в операторах ввода-вывода в виде текстовой константы;

упрощение синтаксиса некоторых операторов (например, запятую в операторе COMMON перед именем общего блока можно опускать).

Тем не менее между версиями Фортрана 66 и Фортрана 77, реализованными в ЕС ЭВМ, как и в соответствующих стандартах, имеются противоречия. Некоторые противоречия характерны только для Фортрана ЕС ЭВМ и заключаются в различии формы записи операторов ввода-вывода прямого доступа. Другие несовместимости определены требованиями стандарта Фортрана 77. Например, в Фортране 77 в отличие от Фортрана 66 тело цикла может не выполняться, список вывода не должен заключаться в скобки, цикл не должен иметь расширенную область.

Несмотря на указанные противоречия, система программирования Фортран 77 поддерживает совместимость с предыдущими версиями Фортрана ЕС ЭВМ. Концепция совместимости обеспечивается по двум направлениям. Во-первых, введен специальный режим трансляции, по которому выполняется обработка программ на Фортране 66. Во-вторых, объектные модули, полученные трансляторами предыдущих версий Фортрана, допускается объединить при редактировании с объектными модулями, полученными транслятором системы программирования Фортран 77, в один загрузочный модуль. Такой подход дает возможность пользователям принять без переделок исходные и объектные модули, созданные на Фортране 66, в системе программирования Фортран 77.

Кроме того, система программирования (СП) Фортран 77 обеспечивает совместимость с транслятором FORTRAN VS снизу вверх на уровне исходных и объектных модулей. Это означает, что модули, подготовленные для обработки СП Фортран 77, могут без изменений быть обработаны транслятором FORTRAN VS, а объектные модули, полученные транслятором Фортран 77, можно соединять при редактировании с объектными модулями, полученными транслятором FORTRAN VS, в один объектный модуль.

Характеристика системы программирования Фортран 77. Эта система выдержала два издания: первое содержало транслятор и библиотеки для работы в пакетном режиме, второе представляет собой совокупность программных компонентов. Они обеспечивают трансляцию и отладку исходных программ на Фортране 77 как в пакетном, так и диалоговом режиме (ПДО СВМ и СРВ ОС 6 и ОС 7). Особенно развитыми являются средства отладки в диалоговом режиме, которые обеспечиваются диалоговым отладчиком.

Транслятор функционирует в пакетном и диалоговом режимах, вызов транслятора в диалоговом режиме производится с помощью

команд, которые являются посредниками ПДО и СРВ. Транслятор осуществляет ввод исходной программы и ее преобразование в объектную программу. Транслятор написан на языке системного программирования и состоит из 4 фаз: анализа, распределения памяти, генерации объектного кода и вывода. Анализ базируется на методе рекурсивного спуска и объединяет в себе лексический и синтаксический контроль исходной программы. Анализ программы сопровождается построением промежуточного текста в форме обратной польской записи. Проверка контекстных условий выполняется на фазах анализа и распределения памяти. При генерации объектного кода учитываются условия, допускающие локальную оптимизацию объектного кода. Режимы трансляции позволяют управлять выводом результатов трансляции. Результатами вывода могут быть объектная программа, распечатки исходной и объектной программы, таблица распределения памяти элементов исходной программы, таблица перекрестных ссылок, сообщения об ошибках в программе. Информация об ошибке достаточно полно характеризует суть нарушений правил языка, допущенных пользователем.

Библиотека программ обеспечивает реализацию элементов входного языка, относящихся к операциям ввода-вывода, стандартных функций языка, а также диагностику и обработку ошибочных ситуаций, возникающих во время выполнения рабочей программы. Каждое сообщение об ошибке содержит информацию, достаточную для определения местоположения ошибки в исходной программе. Библиотека включает развитые средства обработки ошибок, которые позволяют:

вводить в программах пользователя обработку новых ошибочных ситуаций, не обнаруживаемых программами библиотеки;

управлять выводом сообщений об ошибках и выводом списка вызываемых подпрограмм, получивших управление к моменту возникновения ошибки, а также выполнением рабочей программы после обнаружения ошибки;

управлять корректирующими действиями после возникновения ошибки. Различаются два типа корректирующих действий: стандартное, автоматически обеспечиваемое библиотекой Фортрана, и нестандартное, которое пользователь планирует при кодировании исходной программы.

Подключение программ библиотеки к объектным программам производится на этапе редактирования средствами операционной системы.

Диалоговый отладчик позволяет выполнять отладку программ на уровне и в терминах входного языка. Для задания отладочных действий используется язык отладки в виде подкоманд специальной команды ПДО или СРВ. Широкий набор подкоманд дает возможность задавать разнообразные действия, позволяющие осуществлять динамический контроль за выполнением рабочей программы на уровне объектов исходной программы. В зависимости от получаемых результатов пользователь может изменять последовательность выполнения операторов исходной программы, распеча-

тывать и модифицировать данные, а также запрашивать выполнение различных сервисных функций, упрощающих проведение отладки. Отладочные действия задаются в определенных абонентом точках прерывания и распространяются на такие объекты исходной программы, как переменные, массивы и операторы. Для указания некоторых отладочных действий могут использоваться арифметические, логические и текстовые выражения. Программы, предназначенные для диалоговой отладки, должны быть протранслированы с указанием специального режима, при котором в объектном модуле для каждого оператора устанавливается связь с диалоговым отладчиком, а также строятся дополнительные таблицы (таблица символических имен и таблица операторов), позволяющие в процессе отладки программы обращаться к переменным и массивам по их именам, а к операторам — по их меткам или номерам строк.

Развитие системы программирования Фортран 77 продолжается. Система будет наращиваться компонентами, которые предоставят пользователям возможность получать высокоэффективные оптимизированные программы, а также реентерабельные программы для экономии памяти. Для этой цели в трансляторе предусматриваются 3 уровня оптимизации. Первый уровень обеспечивает наилучшее использование ограниченного количества регистров и более быстрых команд перехода. Второй и третий уровни оптимизации, кроме возможностей первого уровня, включают оптимизацию текста программы. Причем в отличие от третьего уровня на втором уровне оптимизации исключаются те перемещения текста программы, которые могут привести к сбойным ситуациям. Для получения рабочих реентерабельных программ обеспечивается соответствующая поддержка в трансляторе, библиотеке и включается в систему дополнительный компонент — программа разделения. Расширяются возможности ввода-вывода на уровне языка для использования средств виртуального метода доступа, который обеспечивает высокую эффективность обработки данных. Все последующие издания СП Фортран 77 будут также обеспечивать полную совместимость с транслятором FORTRAN VS на уровне исходных и объектных модулей.

Планируется, чтобы система Фортран 77 стала единственной и универсальной системой для пользователей Фортрана в базовых программных средствах ЕС ЭВМ.

Литература

1. Колдербэнк В. Дж. Программирование на Фортране. Фортран 66 и Фортран 77: Пер. с англ.— М.: Радио и связь, 1986.— 176 с.

2. Брич З. С. и др. Фортран ЕС ЭВМ/З. С. Брич, Д. В. Капилевич, С. Ю. Котик, В. И. Цагельский. — 2-е изд. — М.: Финансы и статистика, 1985. — 264 с.

3. Брич З. С. и др. Основные принципы системы программирования Фортран 77 ЕС ЭВМ//Вопросы радиоэлектроники.— Сер. ЭВТ.— Вып. 15.— 1984.— С. 12—14.

УДК 681.3.181.5

ПРОБЛЕМНО-НЕЗАВИСИМЫЕ ЭКСПЕРТНЫЕ СИСТЕМЫ ДЛЯ МИКРОЭВМ — ПРОЕКТИРОВАНИЕ, ПРИМЕНЕНИЕ, АНАЛИЗ

Л. ДАКОВСКИ, д-р техн. наук (НРБ),
Н. КАСАБОВ, канд. техн. наук (НРБ)

Проблемно-независимые экспертные системы (ЭС) позволяют создавать базы знаний (БЗн) в различных областях — обучении, медицине, военном деле, сельском хозяйстве. С помощью ЭС можно проектировать множество БЗн в зависимости от их назначения (например, консультации по выбору компьютера, лечение определенного заболевания и др.). К ЭС для микроЭВМ предъявляются требования, которые необходимо соблюдать при проектировании и использовании данных машин. Эти требования обычно связаны с ограниченной емкостью оперативной памяти и сравнительно невысоким быстродействием микроЭВМ.

Проблемно-независимые ЭС используют представление знаний в виде, подходящем для различных проблемных областей, хотя и с ограниченными возможностями [1]. Например, представление знаний возможно в виде фактов и продукций (правил) типа

if условие1
условие2
. . . . . .
then
заключение

Рассмотрим основные принципы проектирования продукционных ЭС [2, 3] для микроЭВМ с акцентом на внутреннем представлении БЗн. Первый принцип — **модульность архитектуры.** ЭС состоит из физически самостоятельных, но логически и функционально связанных модулей (рис. 1). Этот принцип позволяет создавать ЭС в виде сравнительно небольших программных модулей, работающих самостоятельно и располагаемых в оперативной памяти с перекрытием. Каждый из модулей может также иметь модульную структуру, что определяется особенностями конкретной реализации ЭС.

Основным модулем является доказывающая программа. Она использует созданную экспертом БЗн посредством модуля-редактора. БЗн размещается в оперативной памяти. Доказывающая программа обрабатывает созданную БЗн и доказывает в ней правила, проверяя верность условий в них, до конечного решения или до установления, что оно не может быть доказано.

Важным модулем является также модуль интерфейса с пользователем, который осуществляет диалог, необходимый доказываю-

щей программе. Модуль может состоять из нескольких подмодулей: для текстового диалога, для диалога на естественном языке, для речевого диалога. В общем случае модуль интерфейса имеет программно-схемную реализацию. Два дополнительных модуля для генерации ситуаций и статистической обработки при обращении к ним служат для генерации по заданному в БЗн случайному закону некоторой ситуации (вопроса, программы) или для накопления

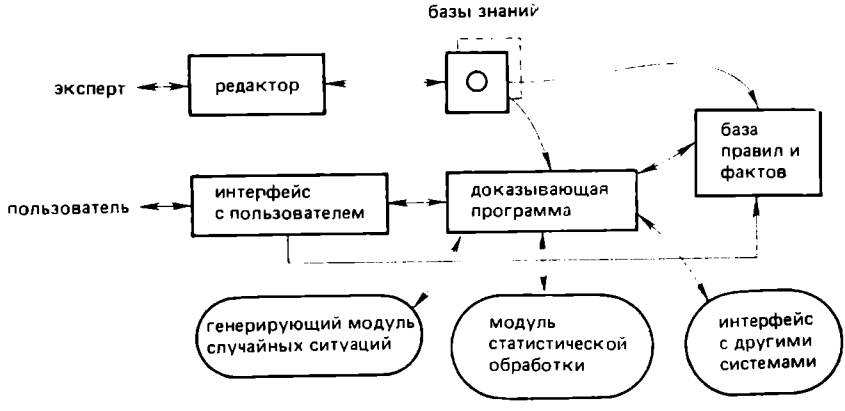


Рис. 1. Модульно-расширенная инструментальная среда для экспертных систем

статистики о работе с конкретной БЗн, необходимой эксперту для ее совершенствования.

Цикл работы с ЭС завершается усовершенствованием БЗн посредством вмешательства эксперта или посредством самообучения. Модуль-редактор можно надстраивать дополнительными подмодулями для извлечения экспертом необходимых ему сведений. Модуль интерфейса с другими системами встроен в основной модуль — доказывающую программу. Этот модуль «открывает» ЭС для других программных систем в операционной среде.

Вторым основным принципом построения проблемно-независимых продукционных ЭС для микроЭВМ является открытость баз знаний, что означает возможность использования в одной БЗн различных форм представления утверждений (условий и заключений). Некоторые формы представления утверждений и их значения после проверки (выполнения) даны ниже.

Формы утверждения	Логическое значение
Логическое умозаключение	Да/нет
Логическое выражение (включающее арифметические вычисления)	Истинное/ложное
Действие (выполнение оператора, программы, программной системы, команды операционной системы и пр.)	Выполнено/не выполнено
Выполнение другой БЗн (в частности, повторение той же)	Доказано заданное решение/не доказано
Фреймовая структура	Заполнена/не заполнена

Второй принцип имеет существенное значение для расширения диапазона применимости ЭС. По своей сути продукционное представление знаний ограничено. Однако оно может быть расширено другими формами, причем продукция и продукционное дерево — это лишь скелет БЗн (рис. 2). Возможность вызова одной БЗн из другой (в частном случае она может быть той же самой) приводит к рекурсивности представления знаний (рис. 3) независимо от рекурсивности продукционного дерева как структуры.

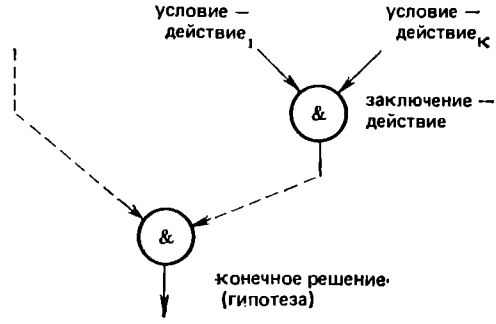


Рис. 2. Примерное продукционное дерево

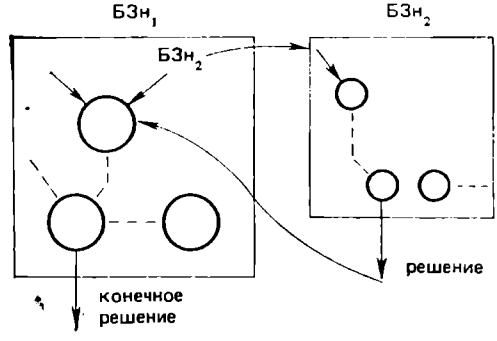


Рис. 3. Рекурсивное обращение баз знаний

Третий принцип — числовое представление правил. Представление правил в виде чисел в оперативной памяти более компактно, чем прямое их представление, и в большинстве случаев обеспечивает более быструю их проверку при доказательстве правил заданной БЗн. Одно из математических представлений БЗн, обеспечивающих числовое представление правил, имеет следующий вид:

$$\text{БЗн} = (X, Y, H, R),$$

где X — множество входных условий (воздействий);
Y — множество заключений;
H — множество гипотез ($H \subseteq Y$);
R — множество правил.

Пусть во множества $Z = XUY, R$ введена упорядоченность и соответствующие ряды обозначены соответственно $\overline{Z}$ и $\overline{R}$. Тогда каждому $r_i \in R$ соответствует одно-единственное правило $\overline{r}_i$ в чистом виде так, что

$$\overline{r}_i = (m,\ n, \ldots, p),$$

где

$$r_i = (x_{i_1},\ x_{i_2}, \ldots, x_{i_{n-1}},\ y_n),$$

$$x_{i_1} = z_m,\ x_{i_2} = z_n, \ldots, y_n = z_p.$$

Каждое правило $\overline{r}$ представляется числовой сборкой n, причем числа (n—1) представляют условия, а последнее число — заключение, с их номерами ряда $\overline{z}$.

Данное математическое описание БЗн позволяет представить ее следующим образом. Упорядоченное множество z, а также ряды $\overline{R}$ и $\overline{H'}$, где $\overline{H'}$ — последовательность номеров правил $\overline{R}$, заканчивающихся гипотезами, хранятся во внешнем запоминающем устройстве. В оперативной памяти поддерживаются следующие структуры:

$$\overline{R},\ D,\ P,\ V,\ M,$$

где

$$[D]=[Z]$$

$$d_i=\begin{cases}-1, & z_i \text{ опровергнуто}\\ 0, & z_i \text{ неизвестно}\\ 1, & z_i \text{ подтверждено}\end{cases}$$

$[P]=[Z]$, где P_i — вероятность (достоверность) для утверждения Z_i (если в БЗн предусмотрено вычисление вероятностей-достоверностей).

$[M]=[\overline{R}]$, где M_i — структура, содержащая управляющую информацию и информацию о состоянии правила $r_i \in \overline{R}$ (например, доказано или нет, приоритет для доказательства и т. д.). Здесь [X] обозначает мощность соответствующего множества или ряда. V — массив с реальными переменными, необходимыми для арифметических вычислений в БЗн. Структура M задает и стратегию работы ЭС, которая может изменяться посредством введения новых значений в структуре M перед началом доказательств правил определенной БЗн. Таким образом можно реализовать основные стратегии — «прямой ход», «обратный ход» — или некоторые смешанные стратегии, в том числе динамически изменяющиеся в процессе работы с БЗн.

Проблемно-независимая продукционная ЭС для микрокомпьютеров ЕС1839 написана на языке Паскаль. Объем основных модулей-редакторов и доказывающей программы — около 64 Кбайт каждый. Максимальное число правил $[\overline{R}]_{max}=1500$, а утверждений — $[Z]_{max}=3000$. В реализации предусмотрена возможность обмена данными между БЗн и внешними программами посредством файла данных (реальных чисел). Это необходимо для работы с БЗн, для которых часть данных уже накоплена или накапливается в реальном времени, а не задается оператором (пользователем).

Из анализа результатов эксперимента с некоторыми конкретными БЗн можно сделать следующие выводы:

быстродействие ЭС при конкретной БЗн зависит от того, насколько заложенная стратегия (в случае стратегии «обратный ход») подходит для конкретной БЗн. Повышения быстродействия можно достичь при стратегии, динамически изменяющейся в ходе работы с одной БЗн;

принцип открытости БЗн значительно расширяет применимость ЭС рассматриваемого типа в различных областях (обучении, САПР и др.);

подобные ЭС обладают хорошей «переносимостью» с одних микроЭВМ на другие. С увеличением быстродействия и оперативной памяти новых поколений микрокомпьютеров становится возможным надстраивание ЭС интеллектуальными и интерфейсными модулями.

Литература

1. Клещев А. С., Черняховская М. Ю. Системы представления проблемно-ориентированных знаний//Техническая кибернетика.— 1982.— № 5.
2. Buchanan B. G. and Duda R. O. Principles of Rule—Based Expert Systems//Advances in Computers.— Academic Press, 1983.— V. 22.
3. Davis R., King J. An overwew of productions systems// Machine Intelligence.— Wiley, 1987.— P. 300—332.

УДК 681.324.019.3

ЭКСПЕРТНАЯ СИСТЕМА ЭСИНА

И. П. ПОПЧЕВ, д-р техн. наук (НРБ),
Н. П. ЗЛАТАРЕВА,
канд. техн. наук (НРБ)

Существуют разные представления о том, какой должна быть экспертная система (ЭС), и разные подходы к каждой конкретной ее реализации. Однако в конечном счете все они сводятся к следующему: система, располагая определенным объемом знаний и «зная», как ими манипулировать, должна выводить новые сведения или раскрывать новые взаимодействия, которые первоначально не были описаны в ее базе знаний (БЗн) в явном виде. Очевидно, что если ЭС обладает верными и полными (с логической точки зрения) знаниями и если задача, которую система должна решить, сформулирована корректно, то мы не должны сомневаться в правильности ее решения.

Но ЭС могут обладать неполными или ненадежными знаниями, а условия задач, которые система должна решать, и первоначальные данные о них могут быть неточными и даже противоречивыми. При таких обстоятельствах ЭС не должна «рассуждать» единообразно, так как возможно, что на некотором следующем этапе могут появиться новые данные, отвергающие полученные заключения, или можно прийти к заключению, противоречащему предварительным предположениям. Чтобы система была в состоянии выбрать, какому из предположений (при возникновении такого конфликта) отдать предпочтение, она должна иметь «дневник» возникновений каждого из них — когда и на каком основании оно возникло, существуют ли данные, которые его подтверждают или, наоборот, существовала какая-то причина, которая вызвала данное предположение, но данные против него (сами данные, однако, могут быть неточными или противоречивыми), какими правилами оно подтверждается или, наоборот, отбрасывается. Следовательно, для работы системы важен не столько факт, что какое-то предположение существует, сколько до какой степени можно ему верить. В статье на базе системы ЭСИНА описан механизм вывода, ориен-

тированный на «правдоподобные» схемы вывода. Он работает благодаря использованию расширенных качественных описаний декларативных и процедурных знаний. Для того чтобы объяснить, почему именно такой подход (несмотря на многие проблемы, которые возникают при его практической реализации) является самым подходящим для данной проблемной области, связанной с надежностью, необходимо перечислить некоторые ее особенности.

Теория надежности оперирует большим объемом «граничных» знаний, для которых нет количественного эквивалента, и они трудно формализуются как продукционные правила, не вызывая проблем квантования.

Существуют сложные ситуации, в которых возможно несколько решений. Предоставляя пользователю одно из них, система не сможет удовлетворить специалиста полностью, если не будет в состоянии объяснить, на каком основании предпочла именно это решение.

Трудно ожидать, что у пользователя имеется полное представление об объекте, который он хочет исследовать, особенно на начальном этапе — когда он обладает некоторым гипотетическим описанием этого объекта, полученным на базе изделий-аналогов.

Общее описание системы. ЭСИНА обеспечивает:

консультации по широкому кругу вопросов, связанных с выбором номенклатуры показателей надежности, с нормированием выбранных показателей, выбором гипотезы для распределения наработки и планированием испытаний надежности;

обучение пользователя по проблемам надежности, причем для этой цели предлагается использование специализированных уроков, терминологической базы данных и механизма объяснения;

помощь пользователю своими вычислительными возможностями, которые включают в себя программы проверки гипотез распределения наработки, планирования контрольных испытаний и оценки показателей надежности.

В соответствии с назначением ЭСИНА состоит из трех подсистем (рис. 1 и 2) — консультирующей, обучающей и вычислительной. Взаимодействие между ними определяется внутрисистемным интерфейсом. Его задача усложнена из-за того, что он должен координировать работу совершенно разных по своим функциям и возможностям подсистем — вместе с «чистой» обработкой данных задачи в области надежности выполняются и традиционные процедуры численного программирования.

Консультирующая подсистема включает в себя базу знаний, механизм вывода, механизм объяснения, а также специализированную терминологическую базу данных, которая является частью базы знаний.

База знаний (рис. 3) состоит из двух частей — динамически изменяющейся (левой) части, выполняющей роль кратковременной памяти и содержащей все возможные суждения с их характеристиками, и неизменяющейся (правой) части, выполняющей роль постоянной памяти и содержащей правила вывода. Декларативные

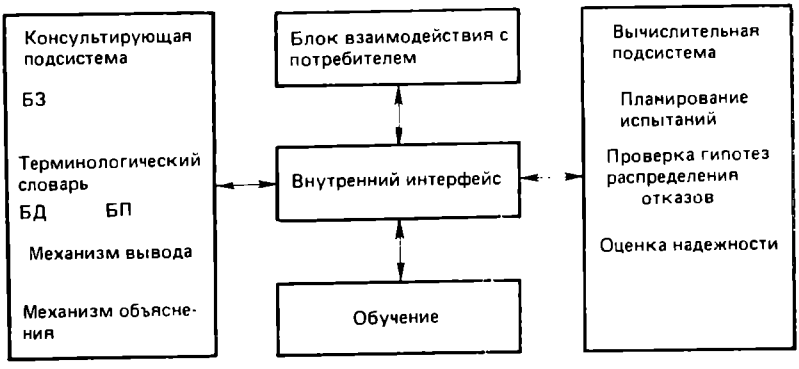


Рис. 1. Общая структурная схема ЭСИНА

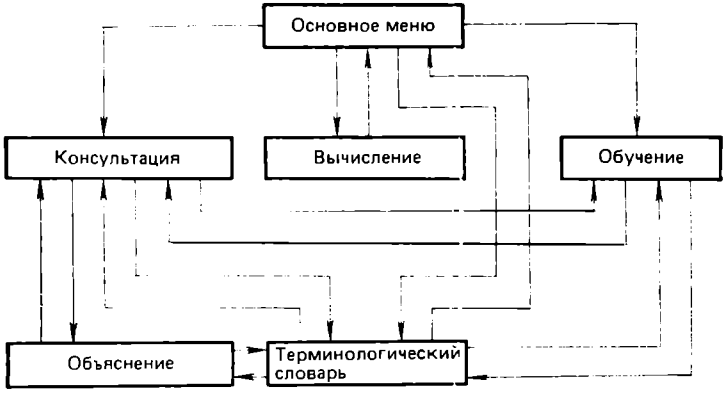


Рис. 2. Схема функциональных взаимодействий между отдельными подсистемами

знания описываются посредством суждений со следующей структурой:

<суждение : : = (<подсписок 1> <подсписок 2> <подсписок 3>),

где подсписок 1 — это предикат, описывающий суждение;

подсписок 2 состоит из идентификаторов, показывающих в результате чего это утверждение введено в БЗн. Они могут иметь разные значения. Например, T показывает, что суждение введено потребителем и определено как истина, F — суждение определено потребителем как неправда, N — суждение введено, но не уточнена его правдоподобность, I — суждение является исключением какого-то правила для вывода, D — суждение получено в результате проведенных вычислений, A — суждение является рекомендацией, полученной из метауправления, и т. д.;

подсписок 3 содержит формализованную информацию о достоверности данного суждения. Он состоит из двух частей (подсписков). Первая содержит информацию, которая говорит «в пользу» достоверности суждения, а вторая — информацию «против» его достоверности. Эта информация актуализируется в процессе вывода, в результате чего соответствующим образом меняется содержание подсписка 2 и подсписка 3, которые выступают в роли «дневника» суждения. С его

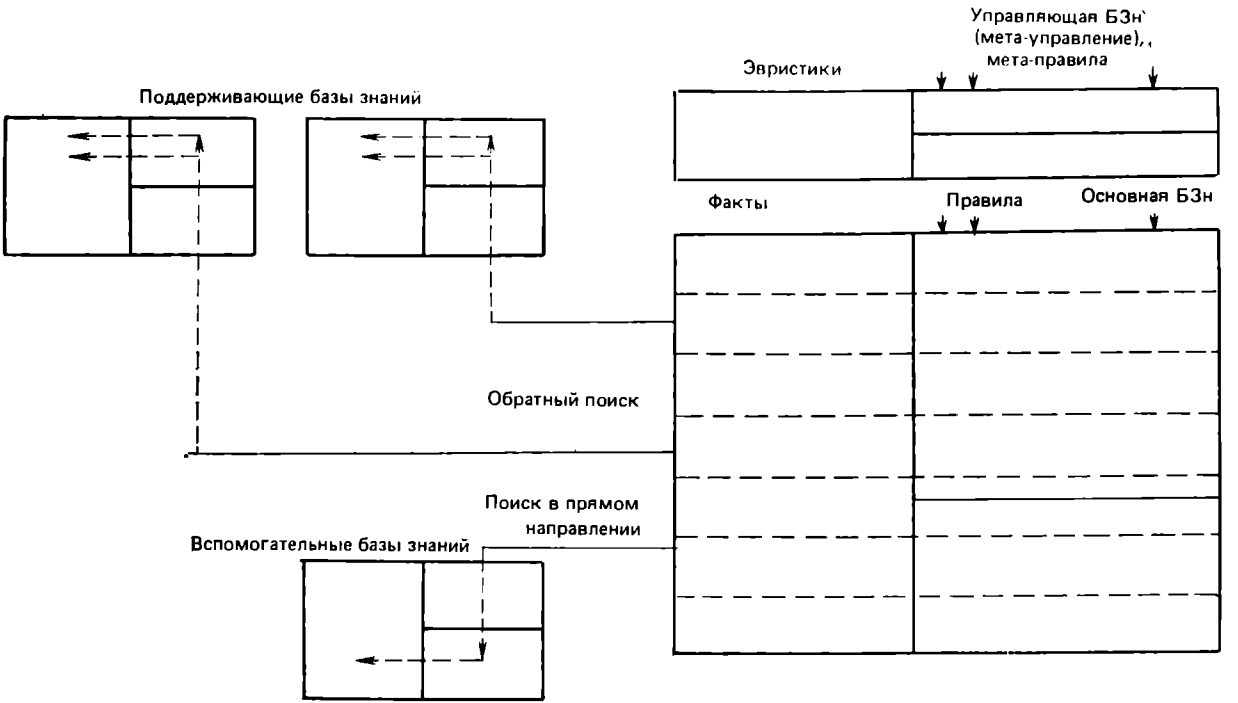


Рис. 3. Организация базы знаний

помощью можно «проследить», как менялась достоверность суждения в процессе вывода, т. е. в процессе накопления новых знаний. Необходимо подчеркнуть, что само утверждение (подсписок 1) не изменяется, изменяется лишь его описание.

База правил содержит экспертные знания, представленные в виде продукционных правил, названных П-правилами. Они объединены в отдельные группы на основе принципа «черной доски». П-правила имеют следующую структуру:

правило : : = (PUTQQ <имя> <условие>)
(PUTQQ <имя> <заключение>)
(PUTQQ <имя> <пояснение>)

где имя — это атом, посредством которого определяется место правила в соответствующей группе П-правил. Он оценивается в процессе решения на базе предварительно сформулированных эвристик в соответствии с предпочтениями эксперта.

Условная часть правил может быть представлена следующим образом:

<условие> : : = IFPAPT (<клауза>{(клауза)})

<клауза> : : = (<распознающая функция> <суждение> <тип>)

За каждой распознающей функцией стоит определенная процедура проверки конкретного условия. Это дает возможность гибко обрабатывать отдельные условия в зависимости от текущего содержания их описания. Например, если условие предполагает унификацию с утверждением, которое отбрасывает данную возможность, то обрабатывающая его распознающая функция основывается на принципе «замкнутого мира», при традиционном «сопоставлении по образцу» этот принцип привел бы к генерированию противоречивых суждений.

Механизм вывода работает с тремя видами распознающих функций:

функцией для обработки условий-фактов, которая проверяет тип условия и активизирует соответствующую процедуру исследования его достоверности;

функцией для обработки условий-отрицаний, которая, если данное условие определено как условие-отрицание, генерирует его согласно слабому закону выключенного третьего;

функцией для обработки условий-вопросов; если условия-вопросы унифицируются утверждениями, которые в данный момент могут и не фигурировать в БЗн, то они задаются в виде вопросов к работающему с системой.

Сами условия являются предикатами, которые унифицируются конкретными утверждениями из БЗн. Предусмотренный в системе механизм унификации позволяет осуществлять «гибкую» унификацию, т. е. в зависимости от типа условия данное правило может быть активизировано даже тогда, когда некоторое его условие не удовлетворяется текущим содержанием БЗн.

Существуют три типа условий: обязательные, необходимые и информативные.

Проверка обязательных условий проводится по классической схеме доказательства или по ее модификации:

$$\frac{A \rightarrow B,\ A}{B}$$

$A \rightarrow B$, A сильно поддерживается в БЗн и ничто не указывает на то, что оно может оказаться неверным

B очень правдоподобно

Активизирование правила, содержащего обязательные условия, возможно только, если эти условия унифицируются конкретным утверждением в БЗн, определяются как истинные (в подсписке 2 их описания записан идентификатор T) или достаточно сильно поддерживаются информацией о своей достоверности (первая часть подсписка 3 содержит достаточно причин, которые подтверждают возможность, что утверждение истинно, в то время как вторая часть подсписка 3 пуста).

Если в выводе отсутствуют обязательные условия, правило оценивается механизмом вывода как «слабое» с информативной точки зрения и характеризуется как нововыведенный факт с относительной достоверностью. Это означает, что если условия вывода являются верными, то правило не будет интерпретироваться как истина, а только как правдоподобное утверждение. Но если это утверждение уже фигурирует в БЗн, его достоверность возрастает. В первой части подсписка 3 записывается новая информация. Вывод может быть «слабым» из-за того, что описывает очень общее знание о факте, или потому, что характеризует достаточно хорошо рассматриваемый случай. Введение «слабых» правил позволяет формализовать ту часть экспертных знаний, которые в большей степени отражают интуицию и предположения, чем дают ясное представление о том, что точно нужно сделать. Таким образом, можно «безболезненно» формализовать некоторые из так называемых «общих знаний».

Если правило содержит обязательные условия, удовлетворяющиеся после их проверки, то существует некоторое априорное представление о достоверности заключения. Это представление оформляется (если даже оно может быть отброшено) в зависимости от результатов проверки необходимых и информативных условий. Если необходимое условие преобразуется в суждение, определенное как истина, достоверность заключения возрастает. Для удовлетворения условиям достаточно, однако, чтобы оно унифицировалось до правдоподобного суждения.

Проверка необходимых условий проводится согласно схеме

$A \rightarrow B$, A правдоподобно

B правдоподобно

Если в БЗн нет суждения, с которым необходимое условие может быть унифицировано, и не содержится также такой информации, которая указывала бы на то, что такое суждение невозможно, правило может быть активизировано. В описании вызванного им

суждения будет записано, что оно выведено, не имея на это всех оснований. Следовательно, дальше в процессе вывода достоверность этого суждения может изменяться в том или ином направлении. Правило не учитывается (отброшена возможность, когда заключение оказывается истиной после первоначального представления об этом, которое вызвано проверкой его обязательных условий), если необходимое условие унифицируется с суждением, которое является неправдоподобным (или ложным), т. е. первая часть подсписка 3 пуста, в то время как вторая содержит достаточно причин, объясняющих, почему это суждение неправдоподобно.

Третий тип условий, названных информативными, дает возможность к П-правилам добавить и некоторые принципиально несущественные компоненты, которые иногда могут создать дополнительное представление о заключении. Если информативное условие выполняется, достоверность заключения возрастает. Наряду с этим, если БЗн не содержит утверждения, с которым условие может быть унифицировано, достоверность заключения не изменяется. Если условие не является истинным или оно очень маловероятно, правдоподобность заключения лишь уменьшается. Иными словами, если остальные условия правила удовлетворяются, заключение задействовано даже тогда, когда информативное условие не выполняется, но это должно отразиться соответствующим образом на достоверности заключения. Заключение П-правила имеет следующую структуру:

<заключение> ::= THENPART(<распознаватель>(<блок>)

. . .

<распознаватель>(<блок>)

где распознаватель показывает, какой из блоков будет активным в зависимости от конечного результата проверки условий. Каждый блок содержит одно или несколько суждений, которые последовательно оцениваются. Если данное суждение генерируется в первый раз, то оно записывается в соответствующем виде в БЗн. Если, однако, существует априорная информация о достоверности этого суждения, то содержание характеризующих его списков изменяется. К ним добавляются новые компоненты, соответствующие конкретному выводу. Это может привести к существенному изменению достоверности суждения, что заставило бы механизм вывода пересмотреть свою работу после определенной «критической» точки (т. е. там, где это суждение использовано на базе определенных предположений, которые позднее отбрасываются).

У каждого правила может быть своя пояснительная часть:

<пояснение> ::= EXP(<текст>)

С ее помощью потребитель может получить неформальное пояснение, почему именно этому правилу отдано предпочтение или существуют ограничения, которые нужно учитывать, если соответствующее правило будет использовано. Эта информация используется механизмом объяснения.

Терминологическая база знаний содержит сведения о специфических терминах в области надежности, которые употребляет система в процессе диалога с пользователем. База знаний организо-

вана как линейный список записей. Каждая запись — это двухэлементный список, первый элемент которого

(термин), или
(OR (термин) (синоним 1) ... (синоним n)),

а второй — содержит определение термина.

Знания, которые содержатся в терминологической БЗн, помогают пользователю в его работе с системой. Здесь мы не будем подробно останавливаться на механизме поиска в терминологической БЗн. Подчеркнем только, что пользователю обеспечивается доступ к так называемому «терминологическому словарю» на ограниченном естественном языке. Ранее было упомянуто, что БЗн разделена отдельными группами П-правил и поиск в этих группах проводится в прямом направлении. Активизация определенной группы П-правил определяется на «метауровне» с помощью специализированных процедур-демонов, названных Д-правилами. Д-правила имеют следующую структуру:

$$\text{имя} :: = A_1 \rightarrow B_1$$
$$\cdots\cdots$$
$$A_l \rightarrow B_1$$

где A_l — условие для активизации соответствующей группы П-правил, а B_1 — имя этой группы.

Одно Д-правило может быть активизировано в разных ситуациях, причем при определении какой-нибудь из них оно активизирует ту группу П-правил, для которой существует максимальная возможность успешного продолжения поиска. Каждая ситуация зависит от конкретных условий, отражающих текущее состояние БЗн.

Важную часть консультирующей подсистемы составляет механизм объяснения. Его цель — показать ход и причины генерирования выведенных суждений. Механизм объяснения имеет две основные функции:

показать причины генерирования данного заключения. Это проводится по принятому способу представления суждений. При помощи подсписка становятся доступными правила, подтверждающие суждение. Объединение части THENPART этих правил дает список суждений, которые являются причинами генерирования конкретного заключения;

объяснить последовательность логических действий, которые приводят к генерированию выведенных заключений. Эта часть объяснения показывает пользователю содержание правил, которые задействованы в процессе вывода. При помощи части EXP этих правил можно получить полное представление о ходе рассуждений.

Обучающая подсистема включает в себя экспертные знания, организованные в виде уроков, касающихся основных вопросов компетентности системы ЭСИНА. Они предназначены в основном для пользователей-неспециалистов по проблемам надежности. Их основная цель — ознакомить пользователя с сущностью проблемы, для решения которой он ищет помощи у ЭСИНА, и помочь ему лучше понять рекомендации и действия системы.

Надежность — это проблемная область, которая характеризуется большим объемом вычислений. Одна часть из них необходима для работы консультирующей подсистемы, а другая — осуществляет на практике предлагаемые консультирующей подсистемой советы и рекомендации. Вычислительная подсистема ЭСИНА содержит библиотеку вычислительных модулей, каждый из которых кроме своих вычислительных возможностей предлагает пользователю различные рекомендации и статистические данные, облегчающие его работу с конкретным модулем.

Взаимодействие с пользователем. Чтобы описание системы ЭСИНА было полным (хотя бы в отношении блоков, показанных на рис. 1), необходимо упомянуть о том, как осуществляется взаимодействие между пользователем и системой.

Обращаясь к системе за консультацией, пользователь должен уметь ответить на ряд вопросов, цель которых создать «модель» объекта, который будет исследоваться. Диалог, в котором ведущей стороной является система, является адаптивным (он зависит от конкретного содержания БЗн и осуществляется на основе развитой сети меню) (рис. 4). Прежде чем ответить на вопросы системы, пользователь может обратиться за помощью к терминологическому словарю или к обучающей системе.

На рис. 5 и 6 показаны два экрана, которые представляют собой ответы системы. Первый показывает конкретную рекомендацию, которую можно интерпретировать как промежуточное решение, а второй — объяснение, которое система предлагает потребителю после его вопроса: почему принято данное решение? Это одна из трех возможностей механизма объяснения системы ЭСИНА.

Опыт эксплуатации. Экспертная система ЭСИНА создана коллективом Института технической кибернетики и робототехники по заказу Комитета качества Совета Министров НРБ. В роли экспертов выступали специалисты по проблемам надежности из различных отраслей: электроники, вычислительной техники, машиностроения и автоматизации производства.

ЭСИНА работает на персональной ЭВМ Правец-16.

В качестве базового программного средства для создания системы ЭСИНА был выбран язык Лисп. Как дополнительные программные языки для реализации отдельных модулей использованы: ассемблер, Бейсик, Турбо-Паскаль и Си. Все программное обеспечение работает в среде МС ДОС.

Система ЭСИНА — законченный программный продукт, который успешно используется в нескольких болгарских институтах и организациях. Практика показывает, что он полезен различным категориям пользователей. Тем, кто заказывает изделия, система компетентно помогает формулировать свои требования к заказываемому изделию. Разработчикам изделий ЭСИНА предлагает консультации по проблемам, связанным с надежностью будущего изделия, оказывает конкретную помощь в их непосредственной работе. Производителям система помогает советами по организации

```
ОТКАЗ ОБЪЕКТА МОЖЕТ ПРИВЕСТИ К
  А – УГРОЗЕ БЕЗОПАСНОСТИ ЛЮДЕЙ ИЛИ К ЗНАЧИТЕЛЬНЫМ МА-
      ТЕРИАЛЬНЫМ И МОРАЛЬНЫМ ЗАТРАТАМ, ЧТО ТРЕБУЕТ ПРИ-
      НЯТИЯ МЕР ДЛЯ ИЗБЕЖАНИЯ ОТКАЗА
  Б – МАТЕРИАЛЬНЫМ ЗАТРАТАМ ВСЛЕДСТВИЕ НЕВЫПОЛНЕНИЯ ЗА-
      ДАЧИ ИЛИ ПРОСТОЯ ОБЪЕКТА, КОТОРЫЕ СОИЗМЕРИМЫ С ЕГО
      СТОИМОСТЬЮ
  В – МАТЕРИАЛЬНЫМ ЗАТРАТАМ, СВЯЗАННЫМ В ОСНОВНОМ С ПОТЕ-
      РЕЙ ОБЪЕКТА ИЛИ С ЕГО ВОССТАНОВЛЕНИЕМ
  Г – К ТОМУ, ЧТО НЕ МОЖЕТ БЫТЬ ДАНА ОЦЕНКА ПОСЛЕДСТВИЯМ
      ОТКАЗА
НАЖМИТЕ А Б В Г
F1 – КОНСУЛЬТАЦИЯ F2 – ВЫЧИСЛЕНИЕ F3 – ОБУЧЕНИЕ F4 – ТЕРМИНЫ F5 – ВЫХОД В DOS
```

Рис. 4. Пример меню в ходе консультации

```
ЭСИНА РЕКОМЕНДУЕТ ВАМ ВЫБРАТЬ СЛЕДУЮЩУЮ НОМЕНКЛАТУРУ
ПОКАЗАТЕЛЕЙ НАДЕЖНОСТИ:

ПОКАЗАТЕЛИ БЕЗОТКАЗНОСТИ  T  TO
КОРРИГИРОВАННЫЙ ПОКАЗАТЕЛЬ БЕЗОТКАЗНОСТИ  T
ПОКАЗАТЕЛИ ДОЛГОВЕЧНОСТИ  TR
КОНКРЕТИЗИРОВАННЫЕ ПОКАЗАТЕЛИ ДОЛГОВЕЧНОСТИ TRSR TROSN TRBR
ПОКАЗАТЕЛИ СОХРАНЯЕМОСТИ  TCXG
ПОКАЗАТЕЛИ РЕМОНТОПРИГОДНОСТИ  TTO TPSTO  C  TT1
НАЖМИТЕ < RETURN > ИЛИ ПОЧЕМУ
F1 – КОНСУЛЬТАЦИЯ F2 – ВЫЧИСЛЕНИЕ F3 – ОБУЧЕНИЕ F4 – ТЕРМИНЫ F5 – ВЫХОД В DOS
```

Рис. 5. Рекомендация системы

```
УКАЗАННОЕ СУЖДЕНИЕ ВЫВЕДЕНО НА ОСНОВЕ СЛЕДУЮЩИХ
ФАКТОВ:

ДЛЯ ОБЪЕКТА ПРЕДУСМАТРИВАЕТСЯ ТЕХНИЧЕСКОЕ ОБСЛУЖИВАНИЕ
ТЕХНИЧЕСКОЕ ОБСЛУЖИВАНИЕ ОБЪЕКТА ЗАВИСИТ КАК ОТ НАРА-
БОТКИ, ТАК И ОТ ЭКСПЛУАТАЦИОННОГО СРОКА
РЕЖИМ РАБОТЫ ОБЪЕКТА ЯВЛЯЕТСЯ НЕПРЕРЫВНЫМ
ГРУППА НАДЕЖНОСТИ ОБЪЕКТА ПЕРВАЯ
            НАЖМИТЕ < RETURN >
```

Рис. 6. Элемент объяснения системы

испытаний на надежность. Специалистам по надежности ЭСИНА в роли почти равноправного партнера помогает контролировать свои решения и дополнять свои знания в областях, где они не имеют достаточного практического опыта. Нормативному органу система обеспечивает контроль результатов, полученных на разных этапах создания и производства изделия.

Опыт показывает, что система ЭСИНА полезна и как база для проверки вариантов по разным нормативным документам, при создании и анализе новых концепций и экспериментальных решений по проблемам надежности.

УДК 681.3.06:519.68

ЯЗЫКОВОЙ ПРОЦЕССОР ПАСКАЛЬ ДЛЯ ОПЕРАЦИОННОЙ СИСТЕМЫ МОСВП

Л. МИШЕВ, инженер (НРБ),
Д. СТОЯНОВА, инженер (НРБ)

Язык Паскаль для МОСВП является реализацией языка Паскаль, компилятор с которого работает в среде операционной системы МОСВП, предназначенной для 32-разрядной ЭВМ ИЗОТ 1055С.

Язык Паскаль для МОСВП сконструирован так, чтобы использовать все преимущества операционной системы МОСВП, в среде которой работает.

Компилятор с языка Паскаль для МОСВП управляется необязательными ключами, которые можно задать в командной строке либо в виде атрибутов. Имеются следующие необязательные ключи:

CHECK — производит машинный код, который осуществляет проверки во время выполнения программы. Например, проверяет, находятся ли индексы данного массива в предварительно указанных границах, не превышает ли результат выполнения целочисленной операции диапазон целых чисел и т. д.;

CROSS REFERENCE — выводит полную информацию о всех идентификаторах, используемых в программе;

DEBUG — создает базу данных для символьного отладчика;

INVIRONMEN — создает файл среды;

ERROR LIMIT — прекращает компиляцию после обнаружения указанного числа ошибок;

GFLOATING — указывает, что в программе используются представление чисел с плавающей запятой в формате G и машинные команды для чисел типа DOUBLE;

LIST — указывает, что будет создан листинг скомпилированной программы;

MACHINE-CODE — включает в листинг код ассемблера скомпилированной программы;

OBJECT — указывает имя объектного файла;

OPTIMIZE — включает оптимизирующую секцию компилятора;

STANDART — указывает компилятору на вывод соответствующего общения при обнаружении конструкции, которая является расширением в языке Паскаль для МОСВП;

WARNING — выводит диагностические сообщения при обнаружении ошибок предупредительного характера.

Каждый из перечисленных ключей имеет отменяющий его действие альтернативный ключ, а также значение по умолчанию, которое принимается, когда соответствующий ключ не задан явно.

Компилятор с языка Паскаль для МОСВП выполняет оптими-

зацию исходной программы как в отношении времени выполнения, так и в отношении объема занимаемой памяти. При этом осуществляется:

вычисление выражений констант во время компиляции, включающих арифметические и логические операции, операции отношения, конкатенации, операции для работы со строками символов и множествами, арифметические функции, функции для преобразования типов, функции для работы с числами типа UNSIGNED, функции для выделения памяти, функции CARD, EXPO и ODD;

устранение подвыражений констант;

частичное удаление кода, который не подлежит выполнению;

оптимизация кода, который соответствует сложным операторам, включающим устранение инвариантных вычислений в циклах;

замена некоторых встроенных функций с их расширенным кодом;

преобразование выражений, содержащих униарный минус и операцию NOT с целью устранения униарных операций отрицания и дополнения;

частичное вычисление логических выражений;

вставка переменных в регистры для уменьшения количества обращений к памяти: если это возможно, компилятор заносит часто используемые переменные в регистры;

упорядочение вычисления выражения для сокращения количества необходимых временных значений;

локальная оптимизация последовательности инструкций — компилятор заменяет некоторые операции эквивалентными, которые работают быстрее первых и/или требуют меньше памяти.

С каждым элементом данных языка Паскаль связывается идентификатор такого типа, который определяет диапазон значений элементов, а также операции, которые можно совершать ими, и объем памяти, необходимый для представления каждого из возможных значений.

Язык Паскаль содержит идентификаторы встроенных типов (INTEGER, REAL, RECORD, SET, CHAR). Таким образом, в программе можно выполнять операции над целыми и действительными числами, двоичными и символьными данными, записями, множествами, массивами, файлами и указателями к динамическим переменным. Кроме того, предоставляется возможность определять пользовательские типы данных в соответствии с требованиями конкретного применения.

Выражение в языке Паскаль является комбинацией из констант, переменных, значений функций и операций. С каждым выражением связывается тип. В результате выполнения арифметических операций получается целое, целое без знака или действительное значение. Логические операции и операции отношения дают значение булевского типа. Предусмотрены также операции для работы со строками символов и с множествами.

Паскаль требует обязательного определения всех символиче-

ских констант и пользовательских типов и декларирования всех меток, переменных, процедур и функций, которые используются программой. Эти элементы описываются в декларативной секции программы, которая может содержать секции LABEL, CONSTANT, TYPE, VAR, PROCEDURE и FUNCTION. Все секции, за исключением LABEL, устанавливают идентификаторы и их определения. Секция LABEL задает числовую метку, соответствующую выполняемому оператору, который связан с оператором GOTO. Каждая выполняемая секция программы ограничивается ключевыми словами BEGIN и END и содержит условные операторы, операторы цикла, операторы присваивания и операторы управления.

Язык Паскаль дает возможность группировать дефиниции, декларации и выполняемые в подпрограмме операторы. Подпрограммы могут быть процедурами и функциями. Каждая подпрограмма состоит из заголовка и тела. Заголовок содержит имя подпрограммы и список формальных параметров, которые определяют внешние данные подпрограммы, а в случае функции — и тип результата. Тело состоит из необязательной декларативной секции и выполняемой секции.

Паскаль — блочно-структурированный язык, который допускает вложение блоков подпрограмм не только в основной программе, но и в самих подпрограммах. Каждая подпрограмма может иметь свои собственные дефиниции и декларации и, кроме того, может переопределять идентификаторы, которые определены во внешних блоках.

Подпрограмме, декларированной на внутреннем уровне, доступны все определения и декларации включающих ее блоков.

Областью действия каждого идентификатора является блок, в котором этот идентификатор описан, минус вложенные блоки, в которых идентификатор описывается повторно.

Язык Паскаль для МОСВП предоставляет компиляционные единицы «программа» и «модуль», каждую из которых можно скомпилировать отдельно. Программа, как и подпрограмма, состоит из заголовка и тела. Модуль содержит только заголовок и декларативную секцию. В заголовке указывается имя программы или модуля и, если необходимо, список идентификаторов, указывающих внешние файлы.

Элементы данных, декларированных в компиляционной единице, действительны для всех уровней компиляционной единицы и доступны для следующих скомпилированных программ или модулей, которые логически включают эти декларации.

С включением класса расширений языка, известных как атрибуты, Паскаль для МОСВП позволяет изменять многие из свойств программы, которые обычно определяются операционной системой: адресование границ, размер и тип памяти для переменных, способ компилирования программы, совместное использование данных. По сравнению с обычными возможностями язык Паскаль для МОСВП включает целый ряд расширений, которые можно сгруппировать в десять основных групп.

1. Лексические и синтаксические расширения:

зарезервированные слова MODULE, OTHERWISE, REM, VARYING, %STDESCR, %IMMED, %REF, %INCLUDE;

экспоненциальный оператор (**);

оператор для временной смены типа переменных и выражений (::);

двоичное, шестнадцатеричное и восьмеричное представление целых чисел;

представление чисел с двойной и четверной точностью;

денежный знак ($) и знак подчеркивания (_) — допустимые символы в идентификаторе;

идентификаторы можно начинать произвольным символом, отличным от цифры; идентификаторы могут иметь произвольную длину, 31 первый символ которой считается уникальным;

расширенный синтаксис для включения символов, у которых нет печатного изображения;

выражения, состоящие только из констант, которые обрабатываются во время компиляции, допускаются везде, где возможна константа;

конструкции структурированных типов можно использовать вместо константы соответствующего структурированного типа;

атрибуты, которые можно использовать для дополнительного сцецифицирования данных, подпрограмм и компиляционных единиц;

большая возможность свободы в правилах, по которым осуществляется совместимость двух частей при присваивании;

структурная совместимость между формальными и фактическими параметрами.

2. Встроенные типы данных:

типы UNSIGNED, SINGLE, DOUBLE;

структурированный тип VARYING OF CHAR и оператор конкатенации строк символов типа VARYING.

3. Встроенные процедуры: CLOSE, DATA, DELETE, ESTABLISH, FIND, FINDK, HALT, LINELIMIT, LOCATE, OPEN, READY, RESETK, REVERT, TIME, TRUNCATE, UNLOCK, UPDATE, WRITEV.

4. Встроенные функции:

булевские функции UFB и UNDEFIND;

функции преобразования типа DBLE, INT, QUAD, TRUNC, UINT, UROUND, UTRUNG;

функция ADDRESS, возвращающая указатель к указанному объекту, который может быть произвольным типом, за исключением элемента пакетированного структурированного типа;

функции для обработки строк символов — BIN, HEX, INDEX, LENGTH, OCT, PAD, SUBSTR;

функции UAND, UNOT, UOR, UXOR, которые выполняют поразрядные логические операции конъюнкции, отрицания, дизъюнкции и исключающего ИЛИ над операндами типа UNSIGNED;

функции SIZE, NEXT, BINEXT, BITSIZE, которые возвращают информацию о количестве памяти, назначенной для переменных и компонентов различных типов;

функции ADD_INTERLOCKED, SET_INTERLOCKED и CLEAR_INTERLOCKED, которые предоставляют возможность параллельным процессам и асинхронным подпрограммам работать в реальном масштабе времени или в мультизадачной среде;

функция CARD, которая возвращает число элементов выражения типа SET, функция CLOCK, возвращающая период времени, в котором центральный процессор занят текущим процессом, функция EXPO, которая возвращает порядок числа, представленного с плавающей запятой.

5. Расширение функций READ, READLN, WRITE, WRITELN. Эти расширения относятся к вводу-выводу строк символов и параметров типа SET и к форматированию текстовых файлов.

6. Расширенные возможности ввода-вывода:

прямой доступ и относительная организация файлов;

доступ по ключу и индексная организация файлов.

7. Декларации:

секции декларирования и дефинирования могут встретиться больше одного раза и в произвольном порядке;

инициализация статических переменных в секции VAR программного или модульного уровня.

8. Операторы:

допускается клауза OTHERWISE в операторе CASE.

9. Процедуры и функции:

функции, которые возвращают значения структурированного типа;

вызов функций как процедуры;

декларации внешних процедур и функций;

подразумеваются значения формальных параметров;

непозиционная передача параметров;

расширенные механизмы передачи параметров к внешним процедурам и функциям.

10. Расширения, относящиеся к раздельной компиляции:

возможность объединения деклараций и дефиниций в отдельных модулях, которые компилируются независимо от основной программы;

атрибуты INVIRONTMENT и INHERIT для управления раздельной компиляцией.

Благодаря перечисленным расширениям язык Паскаль для МОСВП становится мощным и удобным средством для программирования сложных научно-технических задач в среде операционной системы МОСВП, в том числе в реальном масштабе времени. Имея в виду, что 32-разрядная ЭВМ ИЗОТ 1055С комплектуется графической станцией ИЗОТ 1040С, язык Паскаль для МОСВП можно с успехом использовать для создания прикладного программного обеспечения для этой станции.

III

Применение средств вычислительной техники

УДК 681.3

ОСОБЕННОСТИ ОРГАНИЗАЦИИ ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ В СИСТЕМЕ КОМПЛЕКСНОГО ЦЕНТРАЛИЗОВАННОГО ОБСЛУЖИВАНИЯ И ИСПОЛЬЗОВАНИЯ СВТ

М. Е. КОБРАНОВ,
канд. техн. наук (СССР),
О. Н. КВАШНИН, инженер (СССР)

Интенсивное развитие системы комплексного централизованного обслуживания и использования (СКЦОИ) средств вычислительной техники (СВТ) началось в СССР более 10 лет назад. На ранних стадиях развития СКЦОИ обеспечивала проведение совокупности научно-технических и организационных мероприятий, главным образом по обслуживанию технических средств вычислительной техники, фондированию, поставкам и сопровождению программных средств, подготовке специалистов по эксплуатации СВТ. В дальнейшем особое значение стало уделяться решению вопросов, связанных с рациональным использованием машинного времени, обеспечением высокого уровня загрузки СВТ у пользователей, повышением эффективности использования СВТ в народном хозяйстве страны.

СКЦОИ — сложная система, включающая целый ряд производственных, научно-исследовательских, научно-учебных и других предприятий и организаций, деятельность которых охватывает всю территорию страны. В связи с необходимостью решения большого количества многоаспектных проблем важнейшую роль в СКЦОИ

призвана играть система информационного обеспечения (СИО). Особую остроту проблема повышения эффективности информационного обеспечения приобрела на современном этапе, когда в целях коренного улучшения координации работ по созданию, использованию и обслуживанию СВТ, повышения их технического уровня, качества и надежности, развития работ в области информатики, а также обеспечения технического перевооружения отраслей народного хозяйства СССР и ускорения научно-технического прогресса на основе применения СВТ и автоматизированных систем образован Государственный комитет СССР по вычислительной технике и информатике (ГКВТИ СССР). СКЦОИ, ориентированная на решение важнейших задач, поставленных перед ГКВТИ СССР, обладает следующей важной особенностью, влияющей на организацию СИО. Предприятия и организации, оснащающие свои вычислительные центры техническими и программными средствами ВТ, отдельные граждане, приобретающие персональные ЭВМ, становятся пользователями СВТ. Учитывая, что Основными направлениями экономического и социального развития СССР на 1986—1990 годы и на период до 2000 года предусмотрена организация массового выпуска персональных компьютеров, обеспечение роста объема производства вычислительной техники в 2—2,3 раза, коллективными и индивидуальными пользователями СВТ становятся широкие слои населения, включая учащихся средних школ на всей территории страны. С вопросами по различным аспектам обслуживания и использования СВТ пользователи обращаются в СКЦОИ. Таким образом, если отраслевые и подотраслевые информационные системы традиционно работают в основном на специалистов отраслей, разработчиков научно-исследовательских и опытно-конструкторских работ, организаторов производства и управления, то СИО СКЦОИ должна быть рассчитана на представление информационных услуг всем гражданам страны — потенциальным пользователям СВТ. Такой информационный спрос не может быть полностью удовлетворен, если СИО СКЦОИ не будет развиваться в соответствии с требованиями времени.

В настоящее время в регионах страны накоплены определенные массивы информации по тематике СКЦОИ. Эти массивы постоянно пополняются новыми данными, ими регулярно пользуются как специалисты СКЦОИ, так и пользователи СВТ регионов.

Однако в ведении данных массивов нет единообразия, применяются различные носители информации, не унифицированы форматы и т. д. Кроме того, эти массивы нельзя квалифицировать как базы данных, так как значительная их часть реализована на бумажных носителях, в виде картотек, досье, альбомов и т. п. Возможности межрегионального обмена информацией ограничены. В связи с этим первостепенным вопросом создания и функционирования СИО СКЦОИ на современном этапе является организация автоматизированных информационных баз данных на каждом предприятии, в каждой организации системы. Для этого необходимо обеспечить обработку получаемой информации по единой тех-

нологии, ее запись в унифицированном формате на машинном носителе, разработку единого дескриптора, использование единого языка программирования (для обеспечения совместимости с соответствующей государственной автоматизированной системой).

Далее требуется создать региональные информационные сети, подключив к ним как предприятия и организации СКЦОИ, так и пользователей регионов. Уже на этапе создания региональных баз данных руководители и специалисты предприятий и организаций СКЦОИ регионов получат в свое распоряжение полную информацию о состоянии дел в соответствующем регионе, о насыщенности его техникой, о всех аспектах обслуживания и использовании СВТ в конкретном регионе. Кроме того, в зависимости от проблемной ориентации региональной базы данных этим пользователям СКЦОИ будут передаваться оперативные и ретроспективные научно-информационные материалы о состоянии и перспективах развития СВТ в стране и за рубежом, которые подготавливаются традиционно службами научно-технической информации предприятий или организаций. На данном этапе целесообразно подключение персональных средств ВТ, используемых руководителями и ведущими специалистами СКЦОИ, к региональным базам данных.

Следующий этап — создание территориальных республиканских баз данных путем объединения в сеть региональных баз данных. Республиканские базы данных должны быть включены в единую информационную сеть СКЦОИ, а последняя — в государственную автоматизированную информационную систему. Программа разработки и внедрения СИО СКЦОИ будет активно взаимодействовать с программами внедрения автоматизированных систем управления, систем автоматизированного проектирования, гибких автоматизированных производств и в значительной степени использовать технические и программные средства этих систем.

Описанным этапам создания СИО СКЦОИ должна предшествовать глубокая системная научная проработка многих вопросов. Автоматизированные информационные системы состоят из технических средств, программного обеспечения и собственно информации. Если первые две составляющие в значительной степени интернациональны, то информация в основном носит ярко выраженный национальный характер. Она отражает специфику сложившихся в той или иной стране производственных отношений, социальной, экономической, технической и других инфраструктур. Поэтому если технические и программные средства автоматизированных информационых систем во многом можно заимствовать, то информацию нужно «делать» отдельно в каждой стране.

Следует отметить, что важным звеном СИО СКЦОИ (так же, впрочем, как и СИО государственной системы программного обеспечения и других систем ГКВТИ СССР) должна стать сеть центров информатики, формирование которой начато в СССР в 1987 г. Эта сеть в ближайшее время охватит все столицы союзных республик и крупные города страны. Оснащение центров информатики небольшими, но обладающими значительными возможностями пер-

сональными ЭВМ, предназначенными для индивидуального использования, и включение центров информатики в информационную сеть СКЦОИ предоставит любому пользователю возможность оперативно получить в полном объеме необходимую ему информацию и в конечном счете обеспечит высокоэффективное использование информационных ресурсов СКЦОИ.

УДК 681.322:62—192

О НЕКОТОРЫХ РЕЗУЛЬТАТАХ ФУНКЦИОНИРОВАНИЯ СИСТЕМЫ СБОРА И ОБРАБОТКИ ИНФОРМАЦИИ О НАДЕЖНОСТИ СВТ В НОТО СССР

Е. А. АНУФРИЕНКО,
канд. техн. наук (СССР),
Г. Р. ЮЗБАШЬЯНЦ, инженер (СССР)

Обеспечение высокой надежности СВТ — чрезвычайно актуальная задача. Для ее решения принимается комплекс мер на всех стадиях жизненного цикла СВТ: от разработки технических предложений до эксплуатации. Значительное место в этих работах занимает подтверждение нормативных показателей надежности технических средств вычислительной техники (ТСВТ) в процессе их эксплуатации и выявление критических по надежности составных частей.

Для обеспечения заинтересованных предприятий информацией о надежности СВТ на этапах ее ввода и эксплуатации в НОТО СССР создана и функционирует в течение 5 лет система сбора, обработки, распределения и реализации информации о надежности ТСВТ (СИНВТ НОТО). Она решает следующие задачи:

регулярный (один раз в квартал) сбор информации о результатах ввода и эксплуатации ТСВТ с точки зрения ее надежности;

проверка соответствия фактических показателей надежности ТСВТ требованиям технических условий (ТУ);

выявление элементов (ТЭЗов, микросхем), снижающих надежность СВТ;

анализ результатов деятельности предприятий-изготовителей (разработчиков) по повышению надежности технических средств;

обеспечение полученной информацией заинтересованных предприятий и организаций.

Первая и пятая задачи решаются путем разработки и представления отчетных документов, перечень и формы которых приведены в ОСТ 107. 460083.001—86. Схема связей в СИНВТ НОТО представлена на рисунке.

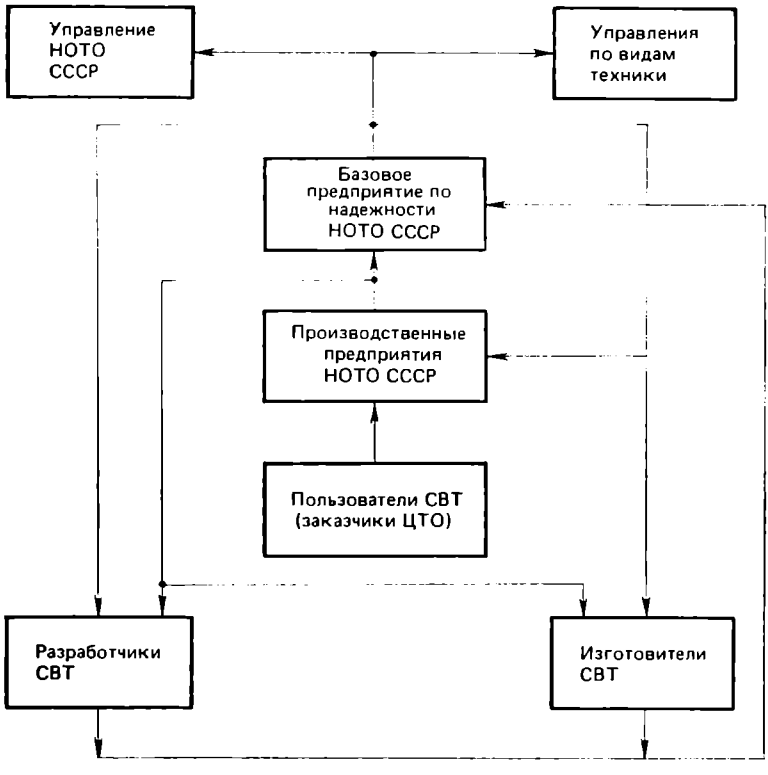


Схема связей в СИНВТ НОТО СССР

Решению второй задачи предшествует предварительная обработка информации, включающая проверку на полноту и достоверность ее представления.

Проверка на достоверность выполняется в следующей последовательности:

на основе данных банка ретроспективной информации строятся гистограммы для случайных величин, полученных в сообщениях с мест эксплуатации ТСВТ;

по результатам обработки выбираются теоретические распределения для анализируемых величин;

проверяется согласованность теоретических распределений со статистическими данными по критерию «хи-квадрат». В случае согласованности распределение принимается. Определяются значения параметров распределений, и эти значения принимаются за базовые;

данные сообщений анализируются, и выпадающие по критерию «трех сигм» наблюдения выбраковываются. Устройства, по которым были представлены «выпадающие» данные, берутся под осо-

бый контроль для выяснения причин появления отклоняющихся наблюдений.

Проверка соответствия фактических показателей надежности СВТ приведенным в ТУ (вторая задача) проводится как для отдельных устройств, так и для ЭВМ в целом. При этом в качестве исходной информации используется банк ретроспективной информации, накопленный в системе, а также данные, полученные за предыдущий квартал в виде сообщений. Проверка соответствия проводится по наработке на отказ (T_0), по среднему времени восстановления работоспособности ($T_в$) и коэффициенту технического использования (КТИ).

Для решения второй задачи выбирается нормативное значение доверительной вероятности (γ); обычно принимается $\gamma=0{,}9$. Для T_0 и КТИ вычисляются нижние доверительные границы, а для $T_в$ — верхняя доверительная граница, которые сравниваются с данными, приведенными в ТУ.

Результаты решения второй задачи служат исходными данными для выявления устройств, критичных в смысле надежности. Поскольку результаты решения второй задачи являются основным источником разногласий между предприятиями-изготовителями (разработчиками) СВТ, то особое внимание в работе СИНВТ НОТО СССР уделяется достоверности исходной информации. По результатам ранее описанных проверок информации проводятся периодические контрольные проверки органов комплексного централизованного обслуживания (КЦО); для повышения стабильности получаемых оценок определен постоянный подконтрольный парк ЭВМ и т. д.

Третья задача решается в следующей последовательности:

определяется перечень устройств, для которых не выполняются требования ТУ;

проводится анализ применяемости элементов в каждом таком устройстве на основе документации предприятия-разработчика (изготовителя);

строится гистограмма интенсивности отказов элементов с учетом их применяемости; эта гистограмма позволяет выявить элементы, определяющие критичность устройства по надежности; повторное применение этой процедуры дает возможность выявить критичные по надежности элементы любого уровня.

Результаты решения третьей задачи служат исходными данными для проведения предприятиями-изготовителями (разработчиками) ТСВТ комплекса работ по устранению причин повышенной интенсивности отказов некоторых устройств и элементов. В перечень этих работ входят:

проведение проверки режимов работы критичных элементов на испытательных стендах и в составе устройств с последующим сравнением реальных и допустимых условий применения;

анализ условий производства и отклонений технологического процесса при изготовлении конкретного устройства;

физико-химический анализ отказавших элементов и установление причин отказов;

разработка предложений по изменению конструкции устройства или замене критичного элемента на более надежный и т. д.

Решение четвертой задачи предполагает наличие достаточно обширного банка ретроспективной информации. В связи с этим в первые годы функционирования СИНВТ НОТО выводы о тенденциях изменения надежности СВТ делались на основе прямого сравнения текущих результатов с предшествующими оценками и носили в основном качественный характер. По мере накопления банка ретроспективной информации появилась возможность определять тенденции изменения надежности устройств и элементов ТСВТ методом статистической обработки. Это позволяет поднять анализ деятельности предприятий по повышению надежности ТСВТ на качественно новый уровень.

Алгоритмы, применяемые при анализе показателей надежности в СИНВТ НОТО СССР, являются универсальными и могут быть использованы любой НОТО страны — участницы Соглашения для любых видов СВТ.

Анализ опыта эксплуатации СИНВТ свидетельствует о ее высокой экономической эффективности. Так, на базовом предприятии по надежности НОТО СССР — головного органа СИНВТ — трудозатраты на составление сообщения об итогах ввода ЭВМ в эксплуатацию или о результатах эксплуатации ЭВМ за квартал в 3 раза ниже численности технического персонала. В то же время за 5 лет эксплуатации СИНВТ разработано около 200 предложений по повышению надежности СВТ, из которых значительная часть внедрена.

В 1985—1986 гг. наработка на отказ ЭВМ ЕС в среднем поднялась на 24%; этому способствовало также функционирование СИНВТ НОТО.

Эффективность работ по повышению надежности возрастет, если удастся устранить следующие недостатки в работе СИНВТ:

недостаточное качество исходной информации о надежности СВТ на этапе эксплуатации. Это вызвано прежде всего отсутствием специальных средств фиксации наработки СВТ, отказов, ожидания восстановления, моментов переключения режимов и т. д. (такие средства, по мнению авторов, должны быть автономными и не зависеть от работоспособности контролируемого оборудования), а также отсутствием системы подготовки специалистов, занятых на местах работами в интересах СИНВТ, и др.;

разнородность информации о надежности, поступающей с однотипных ЭВМ. Проблема однородности информации является особенно острой в НОТО СССР ввиду большого разнообразия климатических условий и территориальной удаленности районов эксплуатации ЭВМ. К возможным причинам, порождающим разнородность потока информации, можно отнести отсутствие системы инструментального контроля и фиксации реальных условий эксплуатации ЭВМ (температурно-влажностный режим, запыленность воз-

духа и др.) и отсутствие системы учета условий эксплуатации ЭВМ до ввода (транспортировка, хранение) и др.

Особенно тревожным является тот факт, что реальные условия эксплуатации ЭВМ не фиксируются даже для ограниченного подконтрольного парка СИНВТ. Это принципиально ограничивает достоверность выводов о надежности СВТ на этапе эксплуатации.

Для устранения указанных недостатков необходимо существенно расширить номенклатуру показателей, фиксируемых в первичных документах СИНВТ. Однако обоснованное решение этой проблемы возможно только после проведения предварительного широкомасштабного эксперимента по определению влияния факторов эксплуатации СВТ на ее надежность.

УДК 681.3.06

ОРГАНИЗАЦИЯ РАБОТЫ КОНСУЛЬТАЦИОННЫХ ПУНКТОВ ГДР ПО ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

К.-Х. МЮЛЛЕР, профессор (ГДР)

Ускоренное внедрение современных технологий, например, проектирования и подготовки производства при помощи ЭВМ, применения микроэлектроники, а также современной информационной технологии требует повышения эффективности разработки, производства, сопровождения и применения программного обеспечения. Одной из главных задач при этом является многократное применение программного обеспечения. С целью решения этой задачи в I квартале 1986 г. председателем Госплана ГДР и председателем Центрального государственного комитета по статистике было выпущено два директивных документа, опубликованных в законодательном информационном бюллетене ГДР. Первый документ [1] регламентирует планирование, пропорции и производство программного обеспечения, а второй [2] освещает принципы организации информации и проведения консультаций по вопросам разработки, производства и применения программного обеспечения.

В соответствии с назначением под *программным обеспечением* [1] понимается совокупность необходимых для эксплуатации ЭВМ средств в виде программ и документации. Здесь различают программные продукты и программные услуги.

Под *программным продуктом* мы понимаем программное обеспечение, выпущенное изготовителем технических средств с целью тиражирования и продаваемое третьим лицам. Таким образом, программные продукты являются составной частью товарной продукции промышленности. *Программные услуги* — это научно-техни-

ческие услуги, представляющие собой программное обеспечение специфического характера.

Если говорить о тиражировании программного обеспечения (ПО) в народном хозяйстве, то эта проблема связана с проверкой необходимости новой разработки программного обеспечения. Разработчик должен дать заявку на разработку ПО в соответствующий плановый орган соответствующей области народного хозяйства. Планирование выпуска ПО прежде всего поручается учреждениям, имеющим широкие специальные знания в этой области. Например, НП «Роботрон» является плановым органом в области операционных систем, языков программирования, компиляторов, СУБД, информационно-поисковых систем, программных инструментов и пакетов математических программ [3]. Планирование разработки специального пользовательского программного обеспечения должно осуществляться соответствующими министерствами.

Первый документ содержит номенклатуру важнейших направлений разработки ПО, но она не претендует на полноту и допускает возможное расширение. Программное обеспечение, требующее на разработку меньше 500 ч, не подлежит планированию, чтобы не задерживать разработку специальных пользовательских программ, в особенности для персональных ЭВМ массового применения.

Второй документ определяет создание пунктов по программному обеспечению, ориентированных на специализированные области консультационных, и центрального банка данных о программных средствах.

Работа консультационных пунктов сосредоточится на тиражируемом в народном хозяйстве программном обеспечении и на проведении консультаций в рамках соответствующей области. Основными задачами консультационных пунктов являются:

сбор информации о многократно применяемом и разрабатываемом программном обеспечении специализированной области;

оценка возможностей применения и качества программного обеспечения;

проведение консультаций с возможными пользователями с целью определения тиража программного обеспечения;

создание связей между разработчиками программного обеспечения одной прикладной области;

сбор отзывов о новых разработках и воздействие на параметры разрабатываемого программного обеспечения (ПО) с целью расширения его применения.

Центральный банк данных о программном обеспечении. Второй документ [2] определяет создание банка библиографических и технических данных о существующем и разрабатываемом в ГДР программном обеспечении НП «Комбинат-Датенферарбайтунг» с целью поддержки тиражирования этого ПО. Библиографические данные о программе идентичны с данными заявки на разработку. Данные собираются для этих целей с помощью одинаковой формы, утвержденной Центральным государственным комитетом по ста-

тистике ГДР. Таким образом определен единый метод сбора, запоминания и поиска программного обеспечения.

Важными данными являются:

данные об изготовителе программы (адрес предприятия-разработчика, номер его телефона, название программы);

дескрипторы, описывающие программу;

краткое описание задачи;

данные о технических средствах и системном программном обеспечении, например тип ЭВМ, потребность в оперативной памяти, язык программирования;

срок поставки, затраты на разработку, народнохозяйственная эффективность и стоимость применения.

Данные о программах собираются два раза: после утверждения технического задания и после разработки в начале эксплуатации при условии стабильного функционирования программ.

Центральный банк данных о программном обеспечении работает на базе разработанной НП «Роботрон» информационно-поисковой системы AIDOS. В качестве технической базы применяется ЭВМ типа ЕС с ОС ЕС или SVS ЕС. В рамках системы AIDOS с учетом единого интерфейса пользователю предоставлено ПО для 16- и 8-разрядных персональных ЭВМ [4].

Ядро системы AIDOS включает:

массив ссылок, в котором запоминаются все ссылки программ;

массив тезауруса, в котором запоминаются дескрипторы одной или нескольких областей;

поисковый язык для запросов в пакетном или диалоговом режимах.

Информация передается в центральный банк данных в виде заполненных формуляров или на мини-дискете. В результате создается возможность децентрализованной работы с помощью профессиональной персональной ЭВМ (ППЭВМ), например проведение поиска и распечатка их результатов. В информационно-поисковой системе AIDOS ссылки классифицируются при помощи дескрипторов, соединенных в тезаурусах по областям применений. Такие тезаурусы должны отслеживаться и совершенствоваться специалистами областей применения, так как они представляют собой базу процесса классификации программных средств. Строить и управлять тезаурусами можно также при помощи ППЭВМ. Эта задача в основном выполняется отраслевыми органами информационного обеспечения. При заполнении формуляров разработчик должен иметь тезаурус для выбора подходящих дескрипторов, а если нет тезауруса, разработчику следует обратиться за консультацией в центральный банк информации.

Второй документ определяет все условия использования банка данных о программном обеспечении. Прежде всего он предоставляет периодическую избирательную информацию о применяемом ПО. Обзорная информация выпускается периодически для министерств, комбинатов и консультационных пунктов. В ходе поиска логическим соединением дескрипторов составляется перечень нуж-

ных программ, который предоставляется предприятиям, организациям и научно-исследовательским институтам.

По состоянию на июнь 1987 г. в банке было зарегистрировано приблизительно 12 000 программ, за последние месяцы поступало приблизительно 100 новых программ в месяц, причем 75% из них относятся к ППЭВМ.

Количество поисков в месяц составляет приблизительно 60—100, причем некоторые из них очень сложны; в результате поисков нередко предлагается свыше 100 программ.

В связи с массовым применением ППЭВМ в ГДР значительно возрастает интерес к использованию центрального банка данных о программном обеспечении. Кроме выполнения поиска ежедневно выдается 10—20 справок по телефону. По нашему опыту, для действенного применения пользовательского ПО требуется подробное рассмотрение ПО будущими пользователями. В связи с этим важно, чтобы будущий пользователь мог связаться с коллективами разработчиков по схожим задачам для обмена опытом. Кроме того, необходимо вызвать интерес разработчика к повторному применению разработанных им программ и поддерживать его готовность создавать новые программы, в связи с чем необходимо предоставить пользователю такие услуги, как, например, обучение, помощь при доработке, сопровождении программ и устранении в них ошибок. При этом надо исходить из того, что задачей центрального банка данных является информирование пользователей, а вся остальная работа должна проводиться консультационными пунктами.

Дальнейшую работу нужно сосредоточить на следующих аспектах. Во-первых, необходимо полное выполнение обоих распоряжений, прежде всего по обеспечению обязательной регистрации существующих программ и заявок на разработку новых программ. Во-вторых, необходимо расширить функции консультационных пунктов путем проведения выставок программного обеспечения, рекламы и предоставления программного обеспечения на научных мероприятиях. Важно продолжить работу по классификации программного обеспечения, в особенности ПО, распространяющегося в нескольких отраслях, а также создать условия для непосредственного стимулирования разработчика программного обеспечения в зависимости от достигнутого тиража ПО.

Сотрудничество с ИнтерФАП. В рамках Комплексной программы СЭВ по научно-техническому прогрессу до 2000 года в 1986 г. была создана секция ИнтерФАП (Международный фонд алгоритмов и программ). Задачами секции являются планомерная информация о повторно применяемом ПО, расширение рекламы и торговли программами и оценка научно-технического уровня и качества ПО. Принятые в рамках ИнтерФАП меры скоординированы с учетом созданного в ГДР центрального банка данных о программном обеспечении. Странами-партнерами к 1987 г. были переданы в ГДР свыше 800 описаний программ, записанных в банк данных. Один из следующих шагов работы ИнтерФАП состоит в определе-

нии единой формы данных, приводимых в описании программы. Это позволит обмениваться данными на магнитной ленте или дискете и уменьшить ручную работу. Наряду с этим каждая страна — участница Соглашения имеет полный обзор существующего в ИнтерФАП программного обеспечения. Заинтересованный партнер на основе соответствующего поиска может получить необходимую информацию, организовать обмен опытом на двустороннем уровне и заключить контракты. Таким образом создаются возможности совместной разработки специальных систем программ. В связи с широким применением ППЭВМ и других средств вычислительной техники своевременное предоставление качественного пользовательского программного обеспечения приобретает огромное значение, потому что только единством технических и программных средств достигается высокий экономический эффект в народном хозяйстве.

Литература

1. Anordnung über Planung, Bilanzierung und Abrechnung von Software vom 13.1.1986//Gesetzblatt der DDR.— 1986.— Teii 1.— No 4.— S. 33—40.—ISSN 0138—1644.
2. Anordnung über Informations — und Beratungsleistungen zur Entwicklung, Produktion und Mehrfachnutzung von Software in der DDR vom 26.2.1986//Gesetzblatt der DDR.— 1986.—Teil 1.— No 9.— S. 94—96.— ISSN 0138—1644.
3. Gräbler, R. Zielstellung und Arbeitsweise der sachgebiets — orientierten Informations — und Beratungseinrichtung des VEB Kombinat Robotron//Neue Technik im Büro. — 1987.— H. 3.— S. 76—77.
4. Hartmann H.— D. AIDOS/M 16 — Informationsrecherchesystem//Rechentechnik/Datenverarbeitung.— 1987.— H. 3.— S. 32—36.

IV
Новые средства СМ и ЕС ЭВМ

УДК 681.322—181.4.009.5

ЕС1834 — ППЭВМ КОМБИНАТА «РОБОТРОН»

Ф. КЕЛЕР, инженер (ГДР)

Персональная профессиональная ЭВМ (ППЭВМ) ЕС1834— первая ЭВМ нового ряда 16-разрядных ППЭВМ, выпускаемых НП «Роботрон». Этот ряд является вкладом ГДР в реализацию раздела «Электронизация народного хозяйства» Комплексной программы научно-технического прогресса стран—членов СЭВ до 2000 года. ППЭВМ ЕС1834 представляет собой базовую модель, модернизируемую в последующие годы выпуском более высокопроизводительных машин.

ЕС1834 предназначена для применения в народном хозяйстве ГДР и других стран — членов СЭВ. При разработке данной ППЭВМ к ней предъявлялись прежде всего такие требования, как минимальные затраты на применение ЕС1834, высокая надежность, хорошее прикладное программное обеспечение для пользователя, а также технологичность, которая обеспечит массовое производство и применение машины.

Основным принципом разработки было точное выполнение интерфейсов в соответствии с согласованными принципами операций ППЭВМ в ЕС ЭВМ. Это реализовано в техническом и в программном отношениях, так что кроме предложенных НП «Роботрон» технических и программных средств для комплектования системы могут быть использованы все технические и программные средства, точно удовлетворяющие требованиям согласованных принципов операций ППЭВМ Единой системы.

Технической основой ППЭВМ ЕС1834 является 16-разрядный микропроцессорный набор

К1810ВМ86. Ядро ЭВМ, т. е. сам микропроцессор, оперативная память емкостью 256 Кбайт и периферийное вычислительное устройство объемом 32 Кбайта, а также центральная часть логики ввода-вывода размещаются в многослойной печатной так называемой системной плате, смонтированной внизу на левой стороне устройства в горизонтальном положении. Для расширения конфигурации на системной плате находятся 8 разъемов для подключения дополнительных ТЭЗов, размещаемых в вертикальном положении. В устройстве находятся также один или два $5^1/_4$-дюймовых

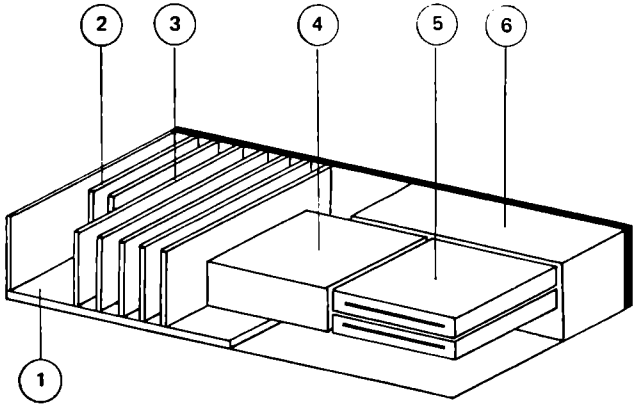


Рис. 1. Расположение блоков и ТЭЗов в корпусе устройства:
1 — системная плата; 2 — короткий ТЭЗ; 3 — нормальный ТЭЗ; 4 — накопитель типа «винчестер»; 5 — НГМД; 6 — блок питания

НГМД с емкостью каждой дискеты 360/720 Кбайт. При необходимости в устройство можно вставить накопитель типа «винчестер». За дисководами находится блок питания всей электронной части (мощностью 180 Вт) и вентилятор. На рис. 1 показано расположение блоков и ТЭЗов в корпусе устройства. К ЭВМ подключается клавиатура с алфавитом языка пользователя. На корпус ЭВМ ставится монитор. Печатающее устройство и второй монитор (его можно не ставить) размещаются рядом с основным блоком ЭВМ (рис. 2).

Системная плата содержит кроме микропроцессора К1810ВМ86 разъем и управляющую логику арифметического сопроцессора К1810ВМ87. На плате размещаются также тактовый генератор (4,915 мГц), устройство управления прерыванием и логика шины. К восьми разъемам для дополнительных ТЭЗов подключена 16-разрядная шина данных. Слева впереди на системной плате находятся управляющая логика (микропроцессор) и разъем для подключения клавиатуры. Через системную шину можно работать в режиме Multimaster.

Дополнительные ТЭЗы имеют такую же длину, как и системная плата, — 360 мм (иногда они бывают короче — две трети системной платы), и высоту 100 мм. На вторую и третью позицию слева мож-

но поставить лишь короткие ТЭЗы (см. рис. 1 и 2) из-за находящегося там разъема для подключения клавиатуры. На обратной стороне корпуса находятся разъемы для подключения периферийных устройств.

Дополнительные ТЭЗы представляют собой адаптеры, расширяющие возможности ППЭВМ:

адаптер расширения памяти (динамическая память емкостью 384 Кбайта) для повышения емкости памяти максимально до 640 Кбайт;

Рис. 2. Внешний вид ППЭВМ ЕС1834

адаптер для монохромного алфавитно-цифрового дисплея (25 строк по 80 символов в строке) и графического дисплея (640×480 точек);

адаптер для графического цветного дисплея (занимает две позиции) с разрешающей способностью в 640×480 точек и 16 цветами, для алфавитно-цифрового представления — 25 строк по 80 символов;

адаптер накопителя на несменных магнитных дисках типа «винчестер» (максимально 2 дисковода);

адаптер НГМД (максимально 4 дисковода);

адаптер подключения ППЭВМ к локальной сети;

адаптер последовательного обмена данными и синхронного режима передачи (BSC, SDLC). Для асинхронного режима (75—100 Бд) используются интерфейсы В24 и ИРПС, могут быть поставлены два адаптера, так что получаются 2 или 4 интерфейса; к этим интерфейсам могут быть подключены устройство вывода графической информации, второе печатающее устройство, устройство ввода графической информации и другие периферийные устройства;

адаптер обмена данными с ЭВМ типа ЕС. Подключение адаптера осуществляется при помощи интерфейса KIF терминала ЕС7920.М к любому контроллеру терминала ЕС7920.М;

адаптер расширения шины, который дает возможность пользователю разрабатывать и применять собственные адаптеры для решения специфических задач при помощи ППЭВМ;

адаптер печатающего устройства типа «центроникс».

В качестве операционной системы в ППЭВМ ЕС1834 используется DCP 3.20— современная дисковая операционная система. На

пользовательском уровне она совместима с применяемой на международном уровне ОС PC — DOS 3.20. Основной операционной системой является так называемый ROM-BIOS, содержащий физические драйверы клавиатуры, экрана, печатающего устройства, НГМД, управления датой и временем, а также программы тестирования технических средств, выполняемые после включения ППЭВМ. Этот комплекс размещен в ПЗУ системной платы.

На основе операционной системы BIOS работает базовая исполнительная программа DCPX 3.20, состоящая из следующих компонентов:

BIO.COM — логической части ROM-BIOS с логическими драйверами устройств и их управлением;

DOS.COM — логического ввода-вывода и интерфейса программиста;

COMMAND.COM — высокопроизводительного интерпретатора команд.

Базовая исполнительная программа характеризуется:

иерархически распределенной системой файлов;

совместимыми друг с другом процессами ввода-вывода файлов и устройств;

удобным языком команд с признаками языка программирования высокого уровня;

гибкой адаптацией операционной системы к разным конфигурациям технических средств;

возможностью включения в систему чужих периферийных устройств пользователем;

удобным тестовым и диагностическим программным обеспечением.

Пользователь имеет в своем распоряжении приблизительно 35 служебных программ ППЭВМ для решения пользовательских задач общего характера, например:

форматирования дискетов и дисков;

сравнения, копирования, переименования, стирания, индикации, контроля и распечатки файлов;

распечатки содержимого экрана на печатающее устройство;

записи ОС на дискеты с учетом набора символов национального языка страны пользователя;

настройки режимов работы печатающего устройства, цветного графического дисплея или монохромного дисплея и асинхронного режима передачи данных.

Система диагностики пользователя состоит из общей программы и некоторых специфических тестовых подпрограмм. Проверяются системная плата и стандартные блоки. Для написания программ имеются разнообразные инструменты. Первая группа из них — средства ассемблера. Сюда входят: макроассемблер для микропроцессора К1810ВМ86; редактор строки и экрана; символическая отладочная программа для эффективного тестирования с интегрированным реассемблером; редактор связей и библиотекарь.

Вторая, очень важная группа — языки программирования более высокого уровня. Предложенный спектр языков дает возможность пользователю применить самый удобный для решения его задачи язык — Бейсик, Турбо-Паскаль, Си, Фортран или Модула-2.

С целью упрощения применения ППЭВМ ЕС1834 пользователю предлагаются специальные пакеты программ, при помощи которых могут быть решены некоторые стандартные задачи, например:

пакет программ обработки текстов, который уже применяется на 8-разрядных ЭВМ;

усовершенствованная и производительная СУБД РЕДАБАС III;

программа составления таблиц MULTICALC;

пакет программ проектирования и конструирования для машиностроения, электроники и других отраслей народного хозяйства;

программы передачи данных на другие ЭВМ, в особенности на ЭВМ типа ЕС.

Большинство пакетов программ являются совместимыми снизу вверх с пакетами, работающими на 8-разрядных ЭВМ под управлением SCP. ЕС1834 благодаря ее высокой производительности можно широко использовать во многих областях применения, в том числе:

в составе САПР/ГАП, например в машиностроении, электронике и строительстве;

для решения экономических задач, в бухгалтерии, в области фактурирования, ведения контроля за складскими запасами, расчета заработной платы, графического изображения показателей и т. п.;

для решения научно-технических задач, например моделирование, задачи измерительной техники и небольшие задачи обработки знаний;

в поисково-информационных системах, для поддержки принятия решений и др.

В заключение можно еще раз подчеркнуть, что предложенная НП «Роботрон» ППЭВМ ЕС1834 отличается свойствами, облегчающими пользователю ее применение;

совместимостью со стандартизованными на международном уровне техническими и программными средствами в области ППЭВМ;

совместимостью данных с ОС SCP 8-разрядных ППЭВМ, полученной при помощи загружаемого драйвера DCP;

широкими возможностями применения в области САПР/ГАП и искусственного интеллекта;

хорошим исполнением.

УДК 681.327.22

ГРАФИЧЕСКАЯ ДИСПЛЕЙНАЯ СТАНЦИЯ СМ7411 (ИЗОТ 1040С)

Г. КУКУРЕШКОВ, инженер (НРБ),
А. ВЕЛИЧКОВ, инженер (НРБ),
Е. НИКОЛОВ, канд. техн. наук (НРБ)

Графическая дисплейная станция (ГДС) предназначена для автоматизации инженерного труда в различных областях народного хозяйства. Она создана на базе 16- и 32-разрядных мини-ЭВМ. ГДС обслуживает два рабочих места, включает цветной растровый монитор с высокой разрешающей способностью, планшет-кодировщик, интерактивный манипулятор, алфавитно-цифровую клавиатуру, алфавитно-цифровой видеотерминал и специализированную функциональную клавиатуру.

Основной частью ГДС является графический процессор. Он выполнен на модульном принципе, причем связь между отдельными модулями осуществляется с помощью одной 16-разрядной и одной 32-й разрядной двунаправленных магистралей. Выбранная современная архитектура процессора обеспечивает не только исключительное быстродействие и производительность ГДС, но и расширение ее возможностей путем добавления новых функциональных модулей и спецпроцессоров.

Графический процессор создан на базе биполярных МОП СИС и БИС бит-слайсовых микропроцессорных элементов и динамических МОП БИС ЗУ. Конструктивно процессор оформлен в виде стандартной для систем с мини-ЭВМ кассеты. Графический процессор состоит из интерфейсного модуля для связи с ЭВМ, модуля «векторная память», управляющего модуля, «интеллектуального» периферийного контроллера, модуля цифрового векторного генератора.

Интерфейсный модуль имеет 9 программно-доступных регистров со стороны ЭВМ и может работать с двумя видами обмена данными между станцией и ЭВМ — программируемым (обмен информационными и управляющими данными последовательно по словам под управлением специализированной программы) и прямым доступом — к оперативной памяти ЭВМ (осуществляется обмен информационными и управляющими данными без вмешательства программы).

Остальные блоки станции подключаются к модулю через два канала — графический и периферийный. По графическому каналу происходит обмен информацией между интерфейсным модулем, векторной памятью, графическим процессором, векторно-растровым преобразователем и видеовыходом.

По периферийному каналу осуществляется обмен между интерфейсным модулем и периферийными контроллерами, управля-

ющими работой станции с помощью функциональной и алфавитно-цифровой клавиатуры, интерактивным манипулятором и графическим планшетом-кодировщиком.

В интерфейсный модуль встроена аппаратная часть блока арбитража захвата графического канала и логики активирования системы прерывания по каналу «общая шина» в результате завершения обмена по графическому и/или периферийному каналу.

Со стороны ЭВМ имеются следующие программно-доступные регистры:

управления и состояния графического канала;

данных (16 старших разрядов из 32-разрядного слова, передаваемого по графическому каналу);

адресный регистр выбора ячейки оперативной памяти ЭВМ (используется только в режиме прямого доступа к ЭВМ);

счетчик слов (используется только в режиме прямого доступа к ЭВМ);

адресный регистр выбора адреса по графическому каналу;

регистр управления и состояния периферийного канала;

регистр данных, передаваемых по периферийному каналу станции;

регистр выбора периферийного устройства и считывания его состояния.

Интерфейсный модуль имеет аппаратную часть для начальной установки модулей станции, а также логику синхронизации работы вышеуказанных модулей. Интерфейсный модуль осуществляет связь и информационный обмен между ГДС и ЭВМ через интерфейс «общая шина».Обмен информацией можно реализовать путем прямого доступа к оперативной памяти ЭВМ или с помощью программного управления со стороны ЭВМ.

Модуль «векторная память» предназначен для хранения дисплейных программ и слов состояния и управления вышеуказанными модулями. Для представления инструкции графического процессора используется 32-разрядное слово. Один модуль памяти имеет емкость 64 Кбайта (16К 32-разрядных слова). В ГДС можно расширить «векторную память» до 192 Кбайт с помощью дополнительных модулей. В первом модуле находится и постоянная память емкостью 8 Кбайт для хранения программ автотестирования.

Дисплейные инструкции выполняются управляющим модулем. Он представляет собой 32-разрядный микропроцессор, использующий бит-слайсовую архитектуру. Использованные биполярные микропроцессорные секции обеспечивают исключительно большую скорость обработки и выполнения дисплейных инструкций. В этом модуле содержится и стандартная таблица ANSII 96 символов, которые можно изобразить в четырех основных направлениях (0°, 90°, 180° и 270°) и восьми различных масштабах.

В управляющем модуле предусмотрены два режима регенерации дисплейных изображений. Первый режим синхронизован с частотой сети, второй (непрерывной регенерации) используется в случаях, когда длина дисплейного файла превышает по времени

один период сетевой синхронизации. В этом случае дисплейный файл выполняется сразу после его окончания. В управляющем модуле предусмотрена возможность расширения таблицы символов еще на 128 символов. В данный момент в это расширение включен алфавит кириллицы.

Микропроцессор имеет также внутреннюю логику для запрещения или для установки интенсивности (цвета) элемента, которая активируется при записи абсолютных векторов в векторном генераторе.

К магистрали подключено и 32-разрядное ППЗУ для констант.

Управление подключенными к станции интерактивными периферийными устройствами осуществляется с помощью модулей интеллектуальных периферийных контроллеров (ПК). В стандартной конфигурации ПК обеспечивает связь с алфавитно-цифровой клавиатурой (ANSII и КОИ-8), одним устройством, выдающим координаты курсора как относительные величины (типа джойстик), одним устройством, определяющим координаты курсора как абсолютные величины (обычно планшет или устройство ввода графической информации).

Разработан также стандартный последовательный интерфейс, соответствующий требованиям RS232G (стык-2). Он спроектирован для работы с функциональной клавиатурой.

Периферийный контроллер позволяет:

передвигать метки по экрану в соответствии с перемещением курсора, джойстика и/или планшета;

ограничивать область перемещения границами, задаваемыми программным способом (для выбора в области экранного меню);

программировать задание режимов прерывания ЭВМ от интерактивных устройств;

самостоятельно проверять и обрабатывать текстовые буфера для визуализации символов, вводимых с клавиатуры или ЭВМ (с некоторыми возможностями редактирования);

организовывать обмен информацией с функциональной клавиатурой в режимах, задаваемых программным способом от ЭВМ.

Для управления этим модулем применяется микроЭВМ МН8035.

ПК предназначен для того, чтобы принять информацию от выбранного интерактивного устройства, обработать ее согласно заданным ЭВМ условиям и загрузить информацию в определенную область векторной памяти (блок управления) в виде, понятном для графического процессора.

Блок управления состоит из нескольких последовательных слов в векторной памяти, находящихся по заданному адресу. Блок организован как «почтовый ящик». ЭВМ производит запись в блок управления, указывая ПК способ обработки информации, а также считывает данные, записанные ПК.

ПК считывает управляющую информацию, указывающую ему режим работы, записывает принятые и обработанные данные, отражающие текущее состояние интерактивного устройства. МикроЭВМ

считывает данные, посылаемые ПК, и на их основе изменяет изображение. Имеются два способа модификации графического изображения в зависимости от перемен в состоянии интерактивного устройства. При первом ПК использует данные блока управления для построения изображений на экране при каждом сканировании дисплейного файла. При втором способе ЭВМ считывает данные блока управления и на их основе изменяет дисплейный файл, сканируемый ПК.

Второй режим работы отнимает много времени у ЭВМ, поэтому он используется только при выполнении некоторого условия (нажата клавиша, переход границы и др.), требующего вмешательства ЭВМ.

Описанный выше обмен информацией через блок управления в векторной памяти осуществляется по графическому каналу станции. Кроме того, ЭВМ и ПК производят обмен и по периферийному каналу. Последний используется ЭВМ для разрешения или запрещения работы ПК с некоторым из интерактивных устройств, а также для обмена алфавитно-цифровой информацией в случаях, когда графическая станция работает как терминал.

Преобразование информации из векторной формы в растровую осуществляется с помощью модуля цифрового векторного генератора.

Цифровой векторный генератор (ЦВГ) состоит из двух частей (ЦВГ-А и ЦВГ-Б). Он предназначен для преобразования входного потока в форме ряда абсолютных XY-координат крайних точек в адресах отдельных пикселей, которые впоследствии записываются в модулях растровой памяти. Кроме процесса векторного преобразования ЦВГ выполняет команды и специальные функции. ЦВГ осуществляет: генерацию пикселей с частотой 16,6 МГц (имеется 16 программируемых скоростей мерцания), генерацию прерывистых линий, кодированных 16 битами, двоичное масштабирование по каждой оси, смещение, маскировку отдельных растровых модулей и автоматическое стирание и управление типа «пинг-понг».

ЦВГ-А соединен с графической магистралью, от которой получает команды, содержащие XY-координаты крайних точек. ЦВГ-А интерпретирует входные данные, а также выполняет команды загрузки скорости мерцания, загрузки коэффициента масштабирования по оси X, загрузки коэффициента масштабирования по оси Y загрузки смещения по оси X, загрузки смещения по оси Y, загрузки слова о состоянии в режиме трансформации (разрешено-запрещено масштабирование, смещение) и в режиме конца дисплейного файла.

Растровое изображение хранится в блоке «растровая память», конструктивно оформленном в виде стандартной кассеты, что позволяет разместить в нем модуль синхронизации и управления растровой памятью, модуль растровой памяти и модуль видеоконтролера. В этом блоке можно использовать цветные растровые мониторы с прогрессивной разверткой и разрешающей способностью 1024×1024 пикселя. С его помощью обеспечивается возможность

изображения до 16 различных цветов из 4096 возможных. Нужный цвет получается посредством соединения трех основных цветов — красного, зеленого и синего, а цветные комбинации — путем изменения интенсивности основных цветов.

Модуль синхронизации и управления управляет работой растровой системы ГДС.

Управление видеоконтроллером осуществляет пара счетчиков — горизонтальный и вертикальный, чьи выходы подключены к ППЗУ. Временные соотношения видеосигналов определяются содержимым ППЗУ. Предусмотрены два набора ППЗУ для двух типов видеосинхронизации, выбираемых программным способом.

Адрес памяти для считывания определяют два дополнительных счетчика — горизонтальный и вертикальный. Эти счетчики запускаются импульсами видеосекции. Из четырех модулей памяти для одного разряда считываются 32-битовые слова, поэтому счетчик предусмотрен только 3-разрядный ($32\times8\times4=1024$). В системах с прогрессивной разверткой адресные счетчики идентичны счетчикам видеосинхронизации. Выходы счетчиков подключены к канальному передатчику для управления адресными шинами модулей растровой памяти. Вертикальный адрес (Y) инвертируется, потому что начало растровой координатной системы располагается в левом верхнем углу экрана.

В основу арбитража доступа к растровой памяти положен приоритетный принцип, причем наиболее высокий приоритет имеет видеоконтроллер. В системах с прогрессивной разверткой (1024×1024 точек) во время вычерчивания строки производится считывание пары слов из всех четырех модулей растровой памяти для каждого разряда кода интенсивности каждые 2,56 мкс четыре раза на одной линии. Все остальное время предоставляется цифровому векторному генератору для записи точек.

Перекрытие циклов записи осуществляется путем распределения записи последовательных точек в разных блоках памяти. Предусмотрена логика, которая следит за состоянием каждого блока и запускает цикл записи сразу после окончания предыдущей операции.

Управляющие сигналы генерируются регистрами сдвига отдельно для каждого блока памяти.

Вся растровая подсистема синхронизируется одним стробирующим импульсом с частотой 12,5 МГц. Кроме того, модуль синхронизации и управления генерирует и стробирующий импульс с частотой 25 МГц для управления видеоконтроллером в системах с прогрессивной разверткой.

Растровая память служит для хранения побитовой карты изображения. Изображение с разрешающей способностью 1024×1024 точек и двумя уровнями интенсивности помещается в четырех модулях памяти. Изображения с 8 или 16 уровнями интенсивности (цветов) хранятся соответственно в 12 и 16 модулях. При этом устройство памяти создано по принципу двойного буфериро-

вания, и в нем сохраняются два изображения с указанной разрешающей способностью и уровнями интенсивности (цветов).

Один модуль памяти содержит 32 динамические МОП БИС ЗУ по 16 Кбит и соответствующие функциональные узлы (буферы, регистры и сдвигающие регистры) и имеет емкость 512 Кбит. Адресный вход делит растровую память на два кадра, причем каждый из них располагается в отдельном буфере. Пока в одном буфере записывается новое изображение, другой буфер выводится на экран монитора (необходимо помнить, что запись и выборка двух буферов не могут осуществляться одновременно).

Один модуль растровой памяти состоит из четырех блоков памяти с соответствующими функциональными схемами, что позволяет осуществлять перекрытие циклов записи к отдельным блокам. Каждые 400 нс (цикл памяти) модуль растровой памяти может принять четыре запроса на обмен через 80 нс каждый. Этим обеспечивается минимальное время (100 нс) для записи одной точки изображения в тех интервалах времени, когда видеоконтроллер не считывает информацию для изображения на экране монитора.

Регенерация памяти осуществляется во время считывания информации. Период регенерации — 256 мкс.

Видеоконтроллер для стандартной прогрессивной развертки преобразует цифровые видеоданные модулей растровой памяти в аналоговые сигналы для управления растровым монитором. Видеоконтроллер содержит таблицу цветов (LVT) для расширения воспроизводимых цветов на экране монитора. Кроме того, он направляет управляющие сигналы для синхронизации строчной и кадровой развертки и для гашения луча. На ввод видеоконтроллера поступают два 16-разрядных потока данных с частотой 12,5 МГц, которые преобразуются в один 4-разрядный поток данных частотой 100 МГц. Другая функция видеоконтроллера — заполнить таблицу цветов стандартным содержанием по включению питания или программным путем. Видеоконтроллер вырабатывает для собственной синхронизации импульсную последовательность в 100 МГц.

УДК 681.3.068

ПРОГРАММНЫЕ СРЕДСТВА ДЛЯ РЕАЛИЗАЦИИ ИНТЕРФЕЙСА X.25 НА ОСНОВЕ СМ ЭВМ

П. ПАРИН, инженер (НРБ),
С. ЦОНЕВ, канд. техн. наук (НРБ),
П. СТОЙЧЕВА, инженер (НРБ)

В последние годы с учетом новейшей технологии средств телеобработки данных и активной разработки и утверж-

дения международных стандартов в социалистических странах активно развиваются и внедряются сети общего пользования с коммутацией пакетов (СОПКП), которые в определенной степени соответствуют архитектуре эталонной модели взаимодействия открытых систем. Как правило, эти сети обеспечивают интерфейсы доступа пользователей на основе стандартизированной рекомендации Х.25 МККТТ или соответствующего стандарта МОС ISO8208. Современная элементная база позволяет развивать и использовать СОПКП как в масштабе страны, так и в рамках отдельных организаций.

Разработанный в НРБ в Центральном институте вычислительной техники и технологии программный продукт (ПП) «Интерфейс Х.25 для ДОС РВ-Б» (Х.25/ДОС РВ-Б) обеспечивает подключение мини-ЭВМ класса СМ-4 (например, ИЗОТ 1016С, ИЗОТ 1054С, СМ1420) к СОПКП в соответствии с редакцией рекомендации Х.25 с 1980 г. Кроме того, на основе рекомендации Х.29 предоставляются средства для взаимодействия с удаленными сетевыми терминалами типа СМ1604. ПП работает под управлением операционной системы ДОС РВ-Б 03 (или версии 4) и дает возможность создавать оконечное оборудование данных (ООД) в терминах рекомендации Х.25. ЭВМ класса СМ-4 подключаются к СОПКП с помощью синхронных адаптеров и модемов. ПП может поддерживать две установки логически независимого ООД с различными сетевыми адресами, которые подключаются к сети при помощи собственных коммуникационных средств.

ПП «Х.25/ДОС РВ-Б» разработан по модульному принципу и представляет собой набор из коммуникационных и вспомогательных процессов, управляющих и обслуживающих программ и библиотек. Ядром ПП является коммуникационный супервизор, который размещается в адресном пространстве супервизора операционной системы ДОС РВ-Б и осуществляет планирование и управление коммуникационными и вспомогательными процессами.

Протоколы первых трех сетевых уровней, которые определены в рекомендации Х.25, реализованы в качестве самостоятельных коммуникационных процессов:

драйвера синхронного коммуникационного адаптера, который управляет функциями физического уровня. Процедуры физического уровня частично реализованы аппаратно;

процесса управления каналом передачи данных, который реализует сбалансированную процедуру доступа к каналу передачи данных АРВ;

процесса протокола пакетного уровня Х.25, который осуществляет специфические процедуры интерфейса Х.25: устанавливает и управляет виртуальными соединениями, передает данные (пакеты), управляет потоком, передает прерывания, сбрасывает виртуальные соединения и осуществляет рестарт ООД, назначает номера логических каналов для виртуальных соединений, обнаруживает протокольные ошибки пакетного уровня.

Функции доступа сетевых терминалов к ООД на основе рекомендации X.29 осуществляются псевдотерминальным драйвером и соответствующим вспомогательным процессом (X29ACP), которые управляют входящими запросами и виртуальными соединениями для взаимодействия с сетевыми терминалами.

Коммуникационный супервизор расширен вспомогательным процессом, который управляет буферами системного пула и осуществляет синхронизацию выполнения сетевых функций во времени. Сетевой псевдодрайвер совместно со вспомогательным управляющим процессом (X25ACP) обеспечивает доступ пользовательских программ к процессу пакетного уровня и осуществляет управление входящими запросами и сетевыми событиями. Системные и обслуживающие программы поддерживают постоянную и динамическую базы данных, которые определяют параметры и характеристики интерфейса X.25 и ПП. Библиотеки ПП содержат сетевые примитивы программных интерфейсов пользователей.

ПП «X.25/ДОС РВ-Б» предоставляет программные интерфейсы для разработки прикладных программ, которые осуществляют взаимодействие программ и обмен данными с программами, работающими в удаленных ООД через СОПКП. Эти интерфейсы представляют собой набор примитивов, созданных на основе средств пакетного уровня интерфейса X.25 в виде макровызовов для программ на языке макроассемблера или в виде обращений к подпрограммам для программ на Фортране. При выполнении прикладных программ сетевой псевдодрайвер преобразует на основе QIO-механизмов супервизора операционную систему системно-зависимых примитивов в определенный набор управляющих пакетов и пакетов данных в соответствии с процедурами интерфейса X.25. Это дает возможность устанавливать взаимодействие с удаленным ООД, работающим в различных операционных средах. Примитивы программных интерфейсов обеспечивают использование всех средств и услуг пакетного уровня интерфейса X.25:

установление, управление и разъединение виртуальных соединений, а также доступ и использование постоянных виртуальных цепей;

обмен данными, причем поддерживается возможность передачи квалифицированных данных двух уровней и передачи длинных сообщений в виде непрерывной последовательности пакетов с использованием Q и M битов в пакетах данных;

передачу прерываний и сброс виртуальных соединений.

Программные интерфейсы предоставляют также некоторые дополнительные средства для эффективного взаимодействия между удаленными программами: доступ к ресурсам сети и создание сетевой очереди программы для незатребованных данных, получение данных из сетевой очереди, преобразование имени ООД в адрес ООД и пересылку запроса на установление виртуального соединения другой программе. С другой стороны, примитивы интерфейсов пользователей дают возможность применять стандартные средства операционной системы, например обслуживание асинх-

ронных системных прерываний и флаги системных событий, устанавливаемые при завершении операций.

Интерфейсы пользователей предоставляют средства для дополнительных услуг интерфейса X.25 в соответствии с рекомендацией X.2: замкнутые группы пользователей (в частности, замкнутые группы двух пользователей и замкнутые группы с исходящим и входящим доступом), согласование классов пропускной способности, согласование параметров управления потоком, быструю выборку данных, реверсивную оплату услуг и др.

Управление одновременной работой нескольких прикладных программ осуществляется с учетом диапазона подадресов, системы приоритетов, назначения «объектов» обслуживания и маскирования входящих запросов.

Асинхронные терминалы типа СМ1604 могут работать с удаленными ООД под управлением ПП «X.25/ДОС РВ-Б», если они подключены к СОПКП через процедуру PAD в соответствии с рекомендациями X.3. и X.28. Взаимодействие процедуры PAD с ООД через сеть осуществляется X.29-драйвером и вспомогательным процессом в соответствии с рекомендацией X.29. Применяя эти средства, удаленные сетевые терминалы приобретают функциональные возможности, которые аналогичны возможностям локально подключенных терминалов. С помощью соответствующего интерпретатора команд (MCR или ДСЛ) с сетевых терминалов можно использовать все системные средства, которые предоставляются операционной системой ДОС РВ-Б. Взаимодействие прикладных программ с сетевыми и локальными терминалами осуществляется одинаковым способом, за исключением некоторых ограничений по отношению к использованию управляющих последовательностей. ПП предоставляет средства для установления и изменения параметров процедуры PAD, которые определяют функциональные характеристики сетевых терминалов. Эти параметры доступны как с терминала, при помощи команд типа SET/XXX, так и с прикладных программ на основе дополнительных подфункций директивы QIO. Параметры PAD поддерживаются в соответствии с редакциями рекомендации X.3 1978—1980 гг.

Генерация ПП «X.25/ДОС РВ-Б» осуществляется в диалоговом режиме и дает возможность определить состав компонентов, их характеристики, а также параметры интерфейса X.25 в зависимости от конфигурации аппаратных средств, специфических параметров и характеристик конкретной СОПКП и требований пользователя. Параметры и характеристики интерфейса X.25 записываются в постоянную базу данных, на основе которой во время загрузки ПП формируется динамическая база данных. Основные параметры этих баз данных могут изменяться обслуживающими программами в статическом и динамическом режимах. Постоянная база данных содержит параметры коммуникационных средств, системные параметры протоколов второго и третьего уровня интерфейса X.25 и описания процессов и сетевых компонентов.

Обслуживающая программа осуществляет загрузку ПП, управляет состоянием отдельных компонентов и выдает информацию о них. ПП включает в себя средства регистрации сетевых событий, накопления и обработки информации о функционировании интерфейса и о возникновении ошибок.

Предусмотрены также средства трассировки передаваемых кадров и пакетов для анализа правильности функционирования ПП и взаимодействия с сетью.

Требования к оперативной памяти (ОП) определяются в основном конфигурацией компонентов ПП и используемых средств, а также максимальным размером пакетов СОПКП и количеством поддерживаемых виртуальных соединений. Минимальная конфигурация ПП требует ОП не менее 29 Кслов, причем обслуживающие программы и процессы регистрации сетевых событий периодически занимают дополнительную память.

ПП «X.25/ДОС РВ-Б» успешно используется для работы мини-ЭВМ ИЗОТ 1016С и терминалов СМ1604.М1 через национальную сеть передачи данных с коммутацией пакетов БУЛПАК. ИЗОТ 1016С подключаются к узлам коммутации сети с помощью синхронных адаптеров СМ8507 и модемов ИЗОТ 8010 со скоростью передачи 2400 бит/с. Терминалы подключаются к сети с помощью модемов через PAD-концентраторы.

Основным этапом использования ПП является согласование значений параметров протоколов второго и третьего уровня интерфейса X.25, поддерживаемого сетью. Требуется также согласовывать способы конкретной реализации некоторых процедур пакетного уровня (например, адресацию и применение некоторых форматов пакетов установления и управления виртуальными соединениями) и возможности использования дополнительных услуг. Эти параметры и характеристики интерфейса X.25 могут настраиваться во время генерации ПП и с помощью средств системных обслуживающих программ. При подключении терминалов СМ1604 необходимо определить профиль терминала, состоящий из набора значений параметров PAD, для того чтобы функциональные возможности этих терминалов наиболее точно соответствовали локальным терминалам.

Во время работы пользователь может выбирать другие профили или изменять некоторые значения параметров PAD на основе средств ПП.

Опыт применения ПП «X.25/ДОС РВ-Б» показывает, что он позволяет эффективно использовать основные и дополнительные средства и услуги сети БУЛПАК. Интерфейсы пользователей предоставляют удобные средства для разработки прикладных систем, удовлетворяющих различным пользовательским запросам на информационное обслуживание. Развитие рассматриваемого ПП для старших моделей ЭВМ серии СМ создает предпосылки для построения на основе СОПКП распределенных информационных систем.

ОПЕРАЦИОННАЯ СИСТЕМА РЕАЛЬНОГО ВРЕМЕНИ BOS-1810

П. КЮЛЬБОРН, инженер (ГДР)

BOS-1810 — мультизадачная система реального времени, предназначенная для ППЭВМ СМ1910. Ее можно применять как специфическую для области применения операционную систему и как систему для разработки программного обеспечения.

Операционная система BOS-1810 требует, чтобы в конфигурации ППЭВМ как минимум были процессор К1810ВМ86, программируемый блок управления прерываниями К580WN59A, программируемый датчик интервалов времени К-580 WI53 и оперативная память емкостью 16 Кбайт.

Структура исполнительной программы BOS-1810 допускает ее расширение драйверами периферийных устройств, специфических для условий применения. Для ряда устройств, необходимых, в частности, для разработки программного обеспечения, BOS—1810 уже располагает соответствующими драйверами. Ниже приведены периферийные устройства ППЭВМ СМ1910 (Роботрон А 7100), с которыми может работать система BOS-1810.

Дисководы накопителя на гибком миниди-ске (макс. 4 дисковода, две стороны, 80 дорожек)	Роботрон К 5600.20 (СМ5640)
Накопитель на несменном диске с подвижными головками (диаметр 133 мм, емкость 30—50 Мбайт)	Роботрон К 5504 (СМ5505)
Блок управления алфавитно-цифровым монитором	Роботрон К 7071
Блок управления графическим монитором	Роботрон К 7075
Клавиатура	Роботрон К 7672
Последовательное печатающее устройство (через интерфейс «центроникс»)	Роботрон К 6313 (СМ6329.01М) или Роботрон К 6314 (СМ6329.02М)
Последовательное печатающее устройство (через ИРПС и ИРПР блока управления периферийными устройствами)	Роботрон К 1152 (СМ6317)
Терминал (монитор, клавиатура)	Роботрон К 8911 (СМ1608)

Альтернативно по отношению к дисплею и клавиатуре можно подключить один пульт управления СМ1608.

BOS-1810 представляет собой модульную операционную систему, состоящую из исполнительной программы, сервисных программ, трансляторов и/или интерпретаторов.

Исполнительная программа состоит из нескольких используемых по выбору подсистем. В рамках подсистем пользователь может выбирать необходимые ему услуги и задавать такую конфи-

гурацию исполнительной программе, которая оптимально соответствовала бы конкретным условиям применения.

BOS-1810 обеспечивает одновременный контроль независимых внешних событий и управление ими, дает возможность управлять большим количеством устройств ввода-вывода и внешних накопителей, готовит для оператора эффективные и гибкие средства наблюдения за поведением системы и ее изменением, позволяет применять трансляторы и другие вспомогательные средства разработки программ.

Рассмотрим системную архитектуру, организацию ввода-вывода, адаптацию к пользователю и вспомогательные средства BOS-1810.

Системная архитектура. Различают следующие типы объектов: задачи, задания, почтовые ящики, семафор, сегменты, соединения и др. (они могут быть образованы и определены пользователем).

BOS-1810 создает системные вызовы, с помощью которых пользователь может манипулировать в своей программе объектами, например CREATE, MAILBOX, DELETE MAILBOX. Объекты всех типов обладают специальными характеристиками, для обращения к которым существуют системные вызовы. BOS-1810 работает в мультизадачном режиме, закрепляя за каждым событием задачу по обработке. Это позволяет логично и просто реагировать на внешние асинхронные события.

Мультизадачный режим и обработка прерываний тесно связаны между собой. Способность технических средств к прерываниям и обработка прерываний позволяют эффективно управлять процессом в реальном времени. Способные к прерываниям системы можно легко расширять, закрепляя за новыми источниками прерываний задачи по обработке прерываний. BOS-1810 обеспечивает приоритетное управление выполнением задач, при котором каждой задаче сообщается ее приоритет. При одновременом возникновении нескольких прерываний активизируется задача, имеющая высший приоритет, а прерванная задача с более низким приоритетом ожидает окончания задачи, имеющей более высокий приоритет. Благодаря структуре BOS-1810, ориентированной на объекты, создаются условия для мультипрограммирования, а в независимых областях применения создается собственная программная среда с помощью объекта типа JOB. Система BOS-1810 обладает эффективными средствами для координации выполнения задач: обеспечивается обмен информацией между задачами, взаимное исключение задач при обращении за информацией, которая синхронизирует ход выполнения задач. Кроме того, структура BOS-1810 создает удобные возможности для расширения системы.

Организация ввода-вывода. В связи с тем, что различные области применения требуют разных характеристик ввода-вывода, BOS-1810 снабжена двумя системами ввода-вывода, которыми можно пользоваться альтернативно или в их сочетании.

Базисная система ввода-вывода образует низкий уровень организации ввода-вывода. Ее преимущество состоит в том, что пользователь может непосредственно обращаться к операциям ввода-

вывода, хотя это требует более глубоких знаний системы. Расширенной системой ввода-вывода пользоваться легче: она освобождает специалиста от необходимости формулировать подробные задания и указывать параметры. Эта система объективно обеспечивает (например, с помощью «опережающего» считывания и записи) последовательные операции ввода-вывода и реализует автоматическую буферизацию ввода-вывода. Не зависящие от устройств вызовы ввода-вывода READ и WRITE дают возможность пользовательским программам легко реализовать ввод-вывод. Устройство определяется одним из параметров вызова. Благодаря использованию переменной в качестве параметра устройства можно выполнять процедуры ввода-вывода, совершенно не зависящие от устройств.

BOS-1810 обеспечивает иерархическое поименование файлов в памяти большой емкости. Это упрощает организацию присвоения имен файлам и повышает гибкость системы. Расширять состав файлов для новых областей применения можно независимо от прежних имен файлов. Управление доступом к файлам позволяет работать с файлами, имеющими иерархическую организацию имен. Владелец файлов может определить права доступа к его файлам и реализовать таким образом их защиту. С помощью управления разделением файлов, т. е. определения размеров блоков каждого файла в памяти большой емкости, пользователь в состоянии управлять временем обращения к файлам и емкостью памяти.

При компоновке системы пользователь может выбрать драйверы устройств, предоставляемые BOS-1810, или составить собственные драйверы и включить их в систему.

С помощью вспомогательного кода терминала BOS-1810 можно создавать программируемый интерфейс между системой ввода-вывода и драйвером терминала. В результате, с одной стороны, обеспечивается определенная независимость терминала, позволяющая подключать практически терминалы всех типов, с другой стороны, пользователь получает возможность программировать для специальных символов и символов управления специальные виды обработки.

Адаптация к пользователю. Система BOS-1810 предоставляет пользователю широкие возможности для оптимальной адаптации системы к его условиям. Связь между человеком и машиной реализуется с помощью интерактивных команд пользователя. Каждая команда интерпретируется как имя программы; файл программ хранится в памяти большой емкости. Пользователь в состоянии без изменения системы модифицировать существующие команды и/или добавлять собственные. Кроме того, BOS-1810 предоставляет в распоряжение пользователя интерпретатор командных строк, помогающий пользователю при приеме параметров и их интерпретации. Под управлением BOS-1810 из внешних накопителей допускается загрузка прикладного программного обеспечения. Большие программы можно разбивать на сегменты. Загрузочная программа BOS-1810 обеспечивает загрузку этих программных сегментов (с

оверлейной структурой), в связи с чем в основной памяти не требуется хранить одновременно всю программу. Благодаря привязке к времени выполнения система BOS-1810 создает возможность динамической модификации системы. Предусмотрены три вида привязки ко времени выполнения: привязки объектов к задаче, файлов и устройств — к задаче, прикладного программного обеспечения — к BOS-1810.

Обработка ошибок обязательна для управления процессом. BOS-1810 предусматривает автоматическую обработку ошибок, в ходе которой предпринимается попытка устранить ошибку путем исключения из обработки задачи, содержащей ошибку, информирования оператора или игнорирования ошибки. Каждый системный вызов информирует о ходе своего выполнения в коде возврата, поэтому пользователь может реализовать в своей программе собственную обработку ошибок.

BOS-1810 обеспечивает динамическое распределение памяти, чем достигается эффективное использование ее емкости.

Вспомогательные средства. Вместе с системой BOS-1810 поставляется набор вспомогательных средств для разработки программ, присутствующий постоянно или только в случае разработки прикладного программного обеспечения.

Программа отладки Debugger, ориентированная на объекты, значительно упрощает процесс поиска ошибок в мультизадачной среде. Она обеспечивает отладку нескольких задач, в то время как остальная система продолжает работать в реальном времени, показывает, какие задачи или объекты находятся в очередях «почтовых ящиков» и «семафоров», и обеспечивает контроль за связью между задачами.

В комплект поставки BOS-1810 входит работоспособная базисная система, которой можно пользоваться в качестве начальной во многих случаях применения. Благодаря модификации файла, предназначенного для создания базисной системы, можно легко разработать специфические для областей применения системы.

Прикладное программное обеспечение можно разработать на ведущей ЭВМ (система разработки программ), после чего использовать на подчиненной ЭВМ. Благодаря средствам разработки программ, которыми располагает BOS-1810, подчиненная ЭВМ может также выполнять функции системы разработки программ. BOS-1810 включает интерактивную систему создания конфигураций, которая с помощью меню руководит пользователем в процессе создания им конфигураций с целью адаптации исполнительной программы к требованиям пользовательской системы. В результате интерактивного определения конфигурации получают файл конфигурации, с помощью которого можно повторить создания конфигурации системы. Этим файлом можно воспользоваться как базой для модификаций. Ядро, как основной элемент BOS-1810, должно быть составной частью всех пользовательских систем, разработанных на базе BOS-1810. К числу свободно выбираемых

компонентов, которые не обязательно должны входить в состав пользовательской системы, относятся система ввода-вывода, связь с оператором, прикладной загрузчик, программа отладки и блок управления терминалом. Для копирования и изменения имен файлов, получения и вывода файлов оглавлений, задания форматов НГМД и др. система BOS-1810 располагает рядом программ ведения файлов.

Сервисные программы. Кроме вспомогательных средств разработки программ, подчиненных исполнительной программе, в состав BOS-1810 входят следующие сервисные программы: редактор связей, библиотечная программа, редактор строк и редактор, ориентированный на применение дисплея, программа подготовки перекрестных ссылок, программа преобразования шестнадцатеричного кода, библиотеки объектных программ для выполнения арифметических операций.

Для работы в среде BOS-1810 разработаны системы программирования ассемблер ASM86, ПЛМ86, Фортран 77, служащие для системного и прикладного программирования решения научно-технических и экологических задач.

УДК 681.327.21/22

ГРАФИЧЕСКИЕ ПЕРИФЕРИЙНЫЕ УСТРОЙСТВА СМ ЭВМ ПРОИЗВОДСТВА ГДР

И. ГАНТЦ, канд. техн. наук (ГДР)

Среди периферийных устройств особенно важное значение имеют устройства, обладающие способностью обработки не только алфавитно-цифровой, но и графической информации.

Ниже представлены новые графические периферийные устройства производства НП «Роботрон», которые успешно прошли совместные испытания в рамках СМ ЭВМ.

Интеллектуальный графический терминал СМ1647 (Robotron K 8918) предназначен для применения в основном в САПР или АСУТП. На его основе создаются графические рабочие места, работающие в диалоговом режиме под управлением главной ЭВМ. В качестве управляющей ЭВМ могут применяться модели как СМ ЭВМ, так и ЕС ЭВМ. Ориентирование устройства СМ1647 на стандарт GKS дает возможность оптимально использовать свойства мобильности прикладных программ, разработанных на основе GKS. Простая передача программ и аппаратно-техническая независимость программирования дают большой экономический эффект при работе с этим устройством. Благодаря собственной «ин-

теллектуальности» СМ1647 диалоговый режим работы реализуется без дополнительной нагрузки на вышестоящую ЭВМ.

Технические параметры интеллектуального графического терминала СМ1647

Центральный процессор	К1810ВМ86
Разрядность, бит	16
Емкость ОП, Кбайт	256
Графический дисплей	Монохромный или цветной
разрешение, точек	640×480
емкость памяти изображения, бит	640×640, 4 уровня
частота повторения, Гц	60
количество полутонов/цветов	16 из 64
позиционирование	Алфавитно-цифровой курсор и визир
Клавиатура	С микропроцессорным управлением и самодиагностикой
набор знаков	Латинский, немецкий, кириллица
функциональные клавиши	2×15 программируемых клавиш,
дополнительные функции	41 клавиша
Интерфейсы	
с главной ЭВМ	ИРПС или С2, 9600 бод
для подключения печатающего устройства	ИРПС
для подключения графического планшета СМ6422	С2
Габаритные размеры (ширина×глубина×высота), мм	
основной корпус	445×440×170
дисплей	338×264×323
клавиатура	455×240×56

В СМ1647 предусмотрен блок управления на основе 16-разрядного и 8-разрядных микропроцессоров со свободно программируемой графической подсистемой. На растровом графическом дисплее представляется двумерное изображение с разрешением, достаточным для большинства случаев применения. Объекты изображения, так называемые сегменты, можно в диалоговом режиме определить, изменить, выделить или манипулировать ими. Для этих целей служат клавиатура и графический планшет, который подключается через стандартный интефейс. Для представления графических и алфавитно-цифровых данных на экране предусмотрены следующие режимы работы:

совмещенный, т. е. одновременное представление данных из алфавитно-цифровой и графической памяти;

режим разделения экрана на графическую и алфавитно-цифровую области с программируемой или управляемой клавишами границей;

альтернативный режим, т. е. представление содержимого только одной из этих памятей.

Графическое изображение из 640×480 точек на экране соответствует подвижному отрезку графической памяти емкостью

640×640 точек. Содержимое памяти изображения можно передвигать вверх или вниз по экрану с помощью клавиш. Для каждой точки можно определить цветовой признак. При этом имеется 16 цветов, которые выбираются из 64 возможных. Представление цветных изображений требует подключения цветного монитора. В стандартном варианте устройство СМ1647 оснащено монохроматическим монитором, на экране которого можно представить 16 полутонов.

Графическое изображение можно распечатать на мозаичном печатающем устройстве СМ6329 (Robotron K 6313/14), подключенном через стандартный интерфейс.

Графический планшет СМ6422 (Robotron K 6405) предназначен для интерактивной работы на графических рабочих местах с дисплеем. Он входит в состав СМ1647. Координатные значения курсора на активной площади планшета предоставляются прикладной программе для дальнейшей обработки информации в цифровом виде.

При совместной работе с графическим терминалом эта функция соответствует позиционированию курсора на экране дисплея. Кроме того, с помощью графического планшета производится ввод команд, символов и алфавитно-цифровых знаков, а также дигитализация.

Технические параметры графического планшета СМ6422

Активная площадь, мм²	210×297 (А4)
Принцип измерения	Индуктивный
Разрешение, мм	0,1
Точность (при 20°С), мм	±0,5 (с курсором) ±0,8 (с пером)
Объект дигитализации	Неметаллический толщиной до 2 мм
Интерфейс	С2
Скорость дигитализации, точек/с	120
Скорость обмена, бод	Переменная, до 19 200
Метод передачи	Асинхронный со стартстопным разрядом (7 разрядов данных, 1 разряд четности)
Код	КОИ-7
Габаритные размеры (ширина×глубина×высота), мм	490×410×35
Масса, кг	Около 3,5

Основными режимами работы планшета являются: POINT — однократное определение координат, RUN — непрерывное автоматическое определение координат, TRACK — непрерывное определение координат, требующее многократного нажатия переключателя курсора.

Поле меню устройства СМ6422 состоит из 48 алфавитно-цифровых знаков, 15 специальных функций и 24 программных функций.

Устройства вывода графической информации СМ6415 и СМ6416 предназначены для подключения к мини- и микроЭВМ. В частно-

сти, они являются составными частями САПР в рамках СМ и ЕС ЭВМ и применяются для изготовления конструкционных чертежей, представления данных в виде кривых и диаграмм, а также для записи данных при измерении и анализе технических процессов. Оба устройства позволяют в качестве носителя информации использовать бумагу, фольгу и кальку.

Технические параметры устройств вывода графической информации СМ6415/СМ6416

	СМ6415	СМ6416
Формат бумаги, мм²	297×420 (формат А3)	625×450 (формат А2)
Используемая площадь, мм	ось X : 370 ось Y : 270	ось X : 594 ось Y : 420
Крепление бумаги	Электростатическое	
Наименьший адресуемый шаг, мм	0,1	0,025
Точность вычерчивания, %	±0,2 (не менее 0,2 мм)	±0,15 (не менее 0,15 мм)
Максимальная скорость вычерчивания, мм/с	240	600 (в направлении оси)
Ускорение	1,0	2,0
Интерфейсы	ИРПР, ИРПС, С2	
Пишущий элемент	Фломастер	Фломастер, шарик, тушь
Количество пишущих элементов	1	8
Скорость вычерчивания текста, зн./с	2—3 (высота 3,5 мм)	
Передвижение пишущего элемента	Программно или от руки с помощью клавиш	
Запас знаков	2 набора	4 набора
Габаритные размеры (ширина× ×глубина×высота), мм	510×520×140	830×695×230
Масса, кг	Около 16	48

В устройстве СМ6415 предусмотрена возможность самостоятельного генерирования алфавитно-цифровых знаков (ASCII), а также возможность выполнения множества команд, реализованная на основе 8-разрядного микропроцессора. Высокая степень «интеллектуальности» устройства СМ6416 связана с использованием встроенных 16-разрядного и 8-разрядного микропроцессоров. Они управляют магазином пишущих элементов и всеми функциями вычерчивания, в том числе установлением окон и масштабными преобразованиями. В режиме дигитализации с помощью специальных клавиш можно передать координаты любых точек в вычислительную машину,

Последовательные печатающие устройства (ППУ) СМ6329.01М-Т2М являются модернизированными вариантами ППУ СМ6329.01/02. На основе программируемого растра до 1920 точек в строке возможно представление графических изображений с большим разрешением. Оба устройства имеют фактически одинаковые технические параметры; основным различием является количество знаков в строке.

Технические параметры последовательных печатающих устройств СМ6329.01М/СМ6329.02М (параметры СМ6329.02М даны в скобках)

Метод печати	Матричный, последовательный
Скорость печати, зн./с	100
Направление печати	Вперед-назад (в графическом режиме только вперед)
Количество иголок в головке	9
Матрица печати	Текстовый режим: 9×9 Графический режим: 480×8 (720×8, 960×8)
Набор знаков	96 знаков; 9 национальных наборов
Количество знаков в строке	Нормальная печать — 80 (136), вразрядку — 40 (68), уплотненная — 132 (233)
Красящая лента	Кассета, бобина
Расстояние между строками	1/6, 1/8 (переключаемое); 7/72″, 1/72″, 1/216 (программное)
Ширина бумаги, мм	Рулонная: 85—216 (85—375), отдельный лист: 85—216 (85—216), лепорелло: до 265 (410) в зависимости от транспортного механизма
Количество копий	2
Интерфейсы	Центроникс, С2, ИРПС
Габаритные размеры (ширина×глубина×высота), мм	370×280×130 (525×290×140)
Масса, кг	Около 7 (10)

Оба устройства отличаются высоким качеством печати, простым обслуживанием и высокой надежностью, возможностью печати текста и графики, а также низким уровнем создаваемого устройствами шума.

УДК 681.327.21/22—621.3.049.75

СРЕДСТВА ОБРАБОТКИ ГРАФИЧЕСКОЙ ИНФОРМАЦИИ И НОВЫЕ ПЕЧАТАЮЩИЕ УСТРОЙСТВА ПРОИЗВОДСТВА ГДР

Б. ЛОЙШЕЛЬ, инженер (ГДР)

Устройства ввода графической информации СМ6417 (К 6401) и СМ6418 (К 6402). Эти современные средства машинной графики применяются в системах CAD/CAM в составе графических рабочих мест. Описываемые средства обеспечивают ввод графических данных с помощью пера или курсора, который свободно перемещается оператором, предварительную обработку и уплотнение этих данных с помощью внутреннего процессора, а

также их передачу в двоичной форме в базовую ЭВМ рабочего места для дальнейшей обработки. Устройства можно также применять для проектно-конструкторских работ в таких областях, как электроника и электротехника, машиностроение, химия, строительство, архитектура и т. д.

Устройства СМ6417 и СМ6418 различаются конструктивным исполнением и размером рабочего поля.

СМ6417 выполнено в виде настольного устройства и состоит из измерительной платы с полем меню и индикацией, корпуса с устройствами управления, интерфейса и электропитания, пружинной опоры, датчика измеряемой величины (перо и курсор).

СМ6418 выпускается в виде стендового устройства. Оно состоит из измерительной платы с полем меню и датчиком измеряемой величины, логической панели с устройствами управления, интерфейса и электропитания, рамы для размещения логической панели и оптимальной установки измерительной платы в рабочее положение по желанию оператора, датчиков измеряемой величины (перо и курсор). Устройство управления (УУ) построено на базе микроЭВМ СМ50/40-2 (СМ1626).

Устройства СМ6417 и СМ6418 могут работать в автономном режиме (off-line) при проверке работоспособности с помощью микротеста и в оперативном режиме (on-line) при подключении к базовой ЭВМ рабочего места.

Работа устройств обеспечивается программными средствами DIGPRO (программа дигитализации) и HDGASS (ассемблер дигитализатора), которые работают под управлением операционной системы ДОС РВ базовой ЭВМ.

Технические характеристики устройств

	СМ6417	СМ6418
Принцип измерения	Индуктивный	
Разрешающая способность, мм	0,02	
Максимальная статическая погрешность, мм		
с пером	0,5	
с курсором	0,1	
Максимальная рабочая площадь (формат), мм²	420×594 (А2)	841×1189 (А0)
Датчик измеряемой величины	Перо Курсор	
Интерфейс	ИРПС С2	
Скорость обмена данными, бод	9600	9600 (при С2 возможна ступенчатая установка от 200 до 9600)
Расстояние передачи «перо/курсор — УУ», м		
Расстояние передачи «УУ — базовая ЭВМ», м	до 500	до 500 (без модема) до 12 000 (при использовании модема)

	СМ6417	СМ6418
Код обмена данными	КОИ-7	
Способ передачи данных	Асинхронный	
Скорость преобразования координат, точек/с	100	
Тип оригинала чертежа	Неметаллический	
Максимальная толщина оригинала, мм	2	
Потребляемая мощность, Вт	120	
Электропитание:		
напряжение сети, В	220±10%—15%	
частота, Гц	50/60	
Габаритные размеры, мм		
ширина	885	1630
высота:		
основное положение измерительной платы	220	нет определенного основного положения
максимальная	620	1960
глубина:		
основное положение измерительной платы	725	нет определенного основного положения
максимальная	725	1200
Масса, кг	45	110

Проблемно-ориентированный процессор обработки изображений СМ0512. Разработан на НП «Роботрон», является функциональным расширением проблемно-ориентированного комплекса цифровой обработки изображений BYS A 6470. Это современное устройство для быстрой обработки изображений в интерактивном режиме, которое по своим функциональным возможностям и техническим параметрам находится на уровне передовых мировых достижений в этой области. Оно с высокой эффективностью может применяться для подготовки и анализа информации, содержащейся в фотографиях, получаемых при дистанционном зондировании Земли (разведка природных ресурсов, своевременное обнаружение изменений окружающей среды, составление физических и специальных карт и др.), а также для анализа видеоинформации в таких областях, как медицина, биология, материаловедение и др.

Процессор СМ0512 содержит дисплейный процессор DIP K 2067.20, блоки памяти регенерации изображений BSP K 3567.30 и общей памяти изображений GSP K 3667.30, устройство управления графикой GST K 7067.21. Конструктивно дисплейный процессор, память регенерации изображений и общая память изображений выполнены в виде двух вставных автономных блоков высотой 6 U каждый, а устройство управления графикой — один вставной блок.

DIP K 2067.20 является специальным процессором для быстрой обработки видеоданных. Структура процессора обеспечивает как выполнение операций над отдельными точками мультиспектральных изображений, так и реализацию локальных, однородных по видеоданным функций. Обработка видеоданных производится синхронно в соответствии со стандартом. Видеоданные вводятся и выводятся из дисплейного процессора построчно. Дисплейный про-

цессор содержит три одинаковых процессорных канала, работающих параллельно, в которых обработка видеоданных происходит поточно. Программирование производится по принципу SIMD (каждая операция выполняется для всех точек изображения).

Блоки памяти предназначены для запоминания результатов обработки. Оба устройства функционально и конструктивно почти идентичны и различаются только интерфейсами и каналами данных. Блок BSP подключен к управляющей ЭВМ и процессору DIP. Управляющая ЭВМ может исполнять все функции PSP в четырех рабочих режимах: чтения, записи, в видеосинхронном режиме в соответствии со стандартом при внутренней синхронизации и в режиме асинхронной передачи с временным контролем от управляющей ЭВМ. Блок GSP соединен с диспетчерской ЭВМ, которая может координировать параллельную работу четырех проблемно-ориентированных процессоров всех управляющих на следующем уровне ЭВМ и DIP.

Память GSP может быть перепрограммирована через все подключенные к ней управляющие ЭВМ, обмениваться с ними данными и работать со всеми процессорами DIP в видеорежиме.

Устройство GST служит для построения по точкам изображений, находящихся в памяти. Кроме того, оно управляет на экране цветного монитора курсором маркировки произвольных точек изображения.

Синхронизация работы блоков DIP, BSP, GSP и GST между собой выполняется центральной системой синхронизации.

Техническая характеристика

Дисплейный процессор К 2067.20

Сопряжение ЭВМ	
интерфейс	параллельный интерфейс ИСС-1 16 бит СТ СЭВ 1610-79
режим	программируемый
Программирование обработки изображений	19 регистров, 18 таблиц
Обработка изображений	11 регистров, 3 таблицы
Канал изображений:	
принцип обработки	конвейерный
ширина обработки операций, байт	8
формат изображения, точек	512×512
формат данных на точку изображения, байт	1
количество уровней яркости	256
входные данные	8 потоков данных в байтовом формате через специальные каналы от устройств PSP и/или GSP; 5 потоков данных в однобайтовом формате через специальные каналы от устройства GST; 6 потоков данных в однобайтовом формате через специальные каналы на устройство BSP и/или GSP;

	3 аналоговых выхода через специальные каналы в цветной монитор
стандарт обработки	CIRT
Синхронизация, МГц	12,5; полутактовая
Потребляемая мощность, Вт	900
Обработка статистических данных изображения	Максимальный и минимальный уровень контраста, сумма и распределение уровней контраста

Блоки памяти BSP K 3567.30 и GSP K 3667.30

Емкость памяти, байт	4×512×512
Время цикла памяти, нс	640
Режим работы:	
видеосинхронный	в соответствии со стандартом
асинхронный	стартстопный
Время выборки:	
в видеорежиме, нс/точку (байт)	80
при асинхронной передаче данных, нс/точку (байт)	640 ... 2560
Входные данные:	
для программирования и передачи информации	1 ИРПР (макс. 5 ИРПР для подключения не более 5 управляющих ЭВМ)
для передачи информации в формат байта при синхронном видеорежиме	3 потока входных данных через специальные каналы от блока DIP (12 потоков входных данных через специальные каналы от 4 блоков DIP)
Выходные данные:	
для передачи информации	1 ИРПР (5 ИРПР для подключения не более 5 управляющих ЭВМ)
для передачи информации в формате байта при синхронном видеорежиме	4 потока выходной информации через специальные каналы в блок DIP (16 потоков выходной информации через специальные каналы в 4 блока DIP)
Потребляемая мощность, Вт	700 (макс. 800)

Устройство GST K 7067.21

Формат памяти изображений	Программируемый: 768×512 точек с 4 бит/точку или 1536×1024 точек с 1 бит/точку
Время цикла в видеорежиме, нс/точку	80
Продолжительность операции записи-чтения в режиме ввода-вывода, нс/точку	2,5 ... 4
Режим доступа	Матричный, точечный
Входные данные для программирования и передачи данных от блока трекбола	1 ИРПС последовательности импульсов (4 бит параллельно) через специальные каналы
Выходные данные	3 потока данных в однобайтовом формате через специальные каналы в блок DIP
Потребляемая мощность, Вт	85

Последовательные печатающие устройства. Устройства ЕС7083.07/СМ6330.07 (Robotron K 6327) и ЕС7083.08/СМ6330.08 (Robotron K 6328) предназначены для вывода информации в составе вычислительных комплексов ЕС ЭВМ Ряд-3 и СМ ЭВМ 3-й очереди и в составе персональных и конторских ЭВМ.

Устройства ЕС7083/СМ6330 отличаются широкими возможностями печати, различными шрифтами, а также графической информацией, высокой скоростью печати, низким уровнем акустических шумов и современным дизайном. Они работают по принципу ударной однорядной мозаичной печати. Печать производится матрицей 9×9 точек в прямом и обратном направлениях движения печатающей головки с оптимизацией пути головки. Кроме того, реализуется графическая печать и печать с уплотненным растром 18×36 точек.

Устройства имеют в своем составе печатающий механизм, управляющую электронику, модуль памяти, модуль интерфейса, блок питания, корпус.

Печатающий механизм включает печатающую головку с иголками и магнитами, печатающую каретку с направляющими и узлом горизонтального привода, узел установки и привода кассеты красящей ленты и бумагопротяжный валик.

Управляющая электроника состоит из микропроцессорного блока и функционального узла для управления печатающей головкой, подачей бумаги и печатающей кареткой. Модуль памяти имеет микропрограммную память и память знакового генератора. Вид интерфейса зависит от исполнения устройства.

Технические характеристики

	K 6327	K 6328
Максимальное количество знаков в строке с шагом печати 1/10 дюйма	80	136
Максимальная скорость печати алфавитно-цифровой информации, зн./с:		
стандартный растр	167	167
уплотненный растр	27	27
Растр печати:		
стандартный	9×9 (полушаговая матрица)	
уплотненный	18×36 (1/3-шаговая матрица)	
Графический режим	Переменный	
Вывод графической информации, точек/дюйм	60 (полная скорость); 120 (половина скорости); 120 (полная скорость, полушаговая матрица); 240 (половина скорости, полушаговая матрица); 72 (графопостроитель); 80 (экранная графика); 90 (экранная графика)	
Плотность печати, зн./дюйм	10, 12, 17	
Подача бумаги, строк/дюйм	Управление форматом в шагах 1/216 дюйма или 1/72 дюйма, шаг строки 7/72 дюйма	
Виды шрифтов	Стандартный, жирный, двойной,	

	косой, пропорциональный, микрошрифт (индексы)	
Интерфейс	ИРПР-М (Centronics), С2, ИРПР	
Количество печатаемых экземпляров	1 оригинал, 2 копии	
Бумага:		
тип	Рулон, бумага лепорелло, бланк	
ширина, мм	240	420
Размеры, мм:		
глубина	370	370
ширина	445	590
высота	150	150
Масса, кг	10	12

Содержание

I

Технические средства ЕС ЭВМ и СМ ЭВМ

Фурнаджиев В., Найденов Г., Стоянов П., Петров В. Локальная сеть MICRONET-16 3
Глухов Ю. Н., Колосков М. С., Кожевников Ю. Б., Кузнецов А. Л. Станция локальной сети кольцевого типа СЛК-СМ 12
Прохоров Н. Л., Сергеев Б. Г. Архитектура высокопроизводительного комплекса моделирования цифровых схем 16
Рихтер И., Терпе Б. Технические и программные средства локальной сети ROLANET1 24
Копецкий В. Интерактивные графические системы ИГС-2 и ИГС-3 производства ЧССР 33
Зиберт Х., Пранге Х.-Д. Тестер печатных плат Р3040 40
Цонев Б., Ценкулов Б., Митев К. Накопители на магнитных дисках для малых и персональных ЭВМ. . . 49
Михальский А. Стандартизация интерфейсов микроЭВМ 56
Глушек Я., Грищ В. Технические и программные средства вычислительных сетей СМ ЭВМ в ЧССР . . . 59

II

Программное обеспечение ЭВМ

Фюле К. Система SOFTORG — интегрированная система вспомогательных средств для современной разработки прикладных систем 69
Штиллер Г. Неортогональность и ортогональность в современных языках программирования 75
Петерсон У., Пичке Ю. Применение языка Пролог для экспертных систем 82
Футо И. Венгерские системы Пролог 93
Куприянов В. П., Котов С. Л., Смагин А. В. Качество программных средств, пути и методы решения . . 103
Подбельский В. В. Инвариантное ядро пакетов прикладных программ 111
Хотяшов Э. Н. Об унификации технологии проектирования СОД 119
Шрайтер Д. Система управления базой данных MIMER 126
Смирнов Г. Д., Цагельский В. И., Ковалевич Э. В., Брич З. С. Система программирования Фортран 77 для ЕС ЭВМ и перспективы ее развития 134

Даковски Л., Касабов Н. Проблемно-независимые экспертные системы для микроЭВМ — проектирование, применение, анализ 139
Попчев И. П., Златарева Н. П. Экспертная система ЭСИНА 143
Мишев Л., Стоянова Д. Языковой процессор Паскаль для операционной системы МОСВП 153

III

Применение средств вычислительной техники

Кобранов М. Е., Квашнин О. Н. Особенности организации информационного обеспечения в системе комплексного централизованного обслуживания и использования СВТ 158
Ануфриенко Е. А., Юзбашьянц Г. Р. О некоторых результатах функционирования системы сбора и обработки информации о надежности СВТ в НОТО СССР 161
Мюллер К.-Х. Организация работы консультационных пунктов ГДР по программному обеспечению . . 165

IV

Новые средства СМ и ЕС ЭВМ

Кёлер Ф. ЕС1834 — ППЭВМ комбината «Роботрон» 170
Кукурешков Г., Величков А., Николов Е. Графическая дисплейная станция СМ7411 (ИЗОТ 1040С) 175
Парин П., Цонев С., Стойчева П. Программные средства для реализации интерфейса X.25 на основе СМ ЭВМ 180
Кюльборн П. Операционная система реального времени BOS-1810 185
Гантц И. Графические периферийные устройства СМ ЭВМ производства ГДР 189
Лойшель Б. Средства обработки графической информации и новые печатающие устройства производства ГДР 193

Contents

I

Computer hardware

Furnadjiev V., Naldenov G., Stoianov P., Petrov V. Local Network MICRONET-16 3
Glukhov J. N., Koloskov M. S., Kozschevnikov J. B., Kuznetsov A. L. The Local Ring Network Terminal Station SLK — SM 12
Prokhorov N. L., Sergeev B. G. Arhitecture of Highperformance System for Simulation of Digital Circuits 16
Richter J., Terpe B. Hardware and Software for the Local Network ROLANET-1 24
Kopecky V. Interactive Graphic Systems IGS-2 and IGS-3 Manufactired in the Czechoslovak Socialist Republic 33
Siebert H., Prange H.-D. Tester for the Circuit Board R 3040 40
Tsonev B., Tsenkulov B., Mitev K. Magnetic Disks for Mini and Personal Computers 49
Michalsky A. Micro Computer Interface Standardization 56
Hlušhek J., Hric V. Hardware and Software for SM Computer. Netwoorks in the Czechoslovak Socialist Republic 59

II

Computer software

Fule K. The SOFTORG System — Integrated Accessory System for Up-to-date Development of Application Systems 60
Stiller G. Nonortogonality and Ortogonality in the Modern Programming Languages 75
Peterson N., Pitschke J. Application of PROLOG Language for Expert System Development 82
Futo J. The Hungarian PROLOG System 93
Kuprianov V. P., Kotov S. L., Smagin A. V. Problems of Software Quality. Metods for their Solution . . . 103
Podbelski V. V. Invariant Kernel of Application Programms 111
Khotiashov E. N. On the Unification of Data Processing System Design Technology 119
Schreiter D. Data Base Management System MIMER. Functions and Application Conditions 126
Smirnov G. D., Tsagelski V. I., Kovalevitch E. V., Britch Z. S. Programming System FORTRAN 77 for

the ES Computers and Prospects of its Future Development 134
Dakovski L., Kasabov N. Problem—Independent Expert Systems for Micro Computer — Design, Application, Analysis 139
Popchev I. P., Zolotareva N. P. Expert System in the Field of Product Reliability ESINA 143
Mishev L., Stoianova D. The Language Processor PASCAL for the Operating System MOSVP 153

III

Operation and maintenance of computer systems

Kobranov M. E., Kvashin O. N. Organization of Information Facilities within the Frame of the Complete Centralised Computer Service and Computer Application 158
Anufrienko E. A., Juzbashiants G. R. On Some Results of the System for Collection and Processing of Computer Reliability Information in the National Maintenance Organization of the USSR 161
Mueller K. H. Formulation and Functioning of Software Consulting Sites in the German Democratic Republic 165

IV

Information on new facilities of the unified system of computers and unified system of mini computers

Köhler F. The Professional Personal Computer ES1834 Manufactured by the Kombinat ROBOTRON 170
Kukureshkov G., Velichkov A., Nikolov E. The Graphic Display Station SM7411 175
Parin P., Ionev S., Stoicheva P. Software for Realization of Interface X.25 Based on the SM Computers . . 180
Kühlborn P. The Real Time Operating System BOS-1810 185
Gantz J. The SM Graphic Peripherals Manufactured in the German Democratic Republic 189
Leuschel B. Graphic Information Processing Facilities. New Printers Manufactured in the German Democratic Republic 193

АННОТАЦИИ

УДК 681. 324

Локальная сеть MICRONET-16 / В. Фурнаджиев, Г. Найденов, П. Стоянов, В. Петров // ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 3—12.

Рассмотрены вопросы проектирования и реализации локальной сети 16-разрядных микроЭВМ. Описаны функциональные возможности и технические характеристики локальной сети, блок-схема транспортного контроллера. Предложен множественный метод доступа с детерминированным разрешением конфликтов. Определены основные виды сетевых операций и представлены реализующие их протоколы. Описаны способ и процедуры взаимодействия пользователей с сетью.

УДК 681.324

Станция локальной сети кольцевого типа СЛК-СМ/Ю. Н. Глухов, М. С. Колосков, Ю. Б. Кожевников, А. Л. Кузнецов//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 12—15.

Описана станция СЛК-СМ для построения сетей кольцевого типа, реализующая доступ к сети методом вставки регистра. Программное обеспечение станции позволяет отрабатывать систему протоколов, обеспечивающую подключение к сети устройств различного уровня «интеллектуальности». Станция имеет средства самодиагностики. Введение резервирования сегментов сетевого канала с автоматическим переключением и повышение помехозащищенности передачи дают возможность использовать сеть СЛК-СМ в учрежденческих и промышленных системах.

УДК 681.3.02.57

Архитектура высокопроизводительного комплекса моделирования цифровых схем/Н. Л. Прохоров, Б. Г. Сергеев//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 16—24.

Дан краткий обзор современных специализированных вычислительных средств моделирования цифровых схем. Описаны архитектура и принципы построения высокопроизводительного многопроцессорного комплекса для САПР СБИС, ускоряющего процесс моделирования по сравнению с обычными универсальными ЭВМ в 1000 и более раз. Приведены основные структурные особенности и технические характеристики входящих в состав комплекса процессоров.

УДК 681.324

Технические и программные средства локальной сети ROLANET1/И. Рихтер, Б. Терпе//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 24—33.

Рассмотрены состав и структура локальной сети ROLANET1, а также блок-схема контроллера, транссивера и способы подключения к сети ЭВМ Единой системы или СМ ЭВМ. Кратко описаны основные программные компоненты управления локальной сетью. Сеть может быть использована для создания информационной системы организационного управления предприятием и системы технологического проектирования.

УДК 681. 326

Интерактивные графические системы ИГС-2 и ИГС-3 производства ЧССР/ В. Копецкий//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 33—40.

Рассмотрены общая структура процесса обработки графической информации и архитектура графических станций первого, второго и третьего поколений. Описаны функциональные особенности и технические характеристики двух новых графических систем ИГС-2 и ИГС-3.

УДК 681.324

Тестер печатных плат Р 3040/Х. Зиберт, Х.-Д. Пранге//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 40—48.

Описан тестер, позволяющий выполнять внутрисхемный контроль дискретных и интегральных, аналоговых и цифровых элементов схем, а также обнаруживать короткие замыкания и обрывы проводников печатных плат после монтажа и пайки. Тестер построен на базе мини-ЭВМ типа «Электроника 60-1» в сочетании с двумя дисковыми накопителями емкостью 35 Мбайт каждый, двумя накопителями на гибких дисках и двумя терминалами. Охарактеризованы также основные модули программной библиотеки для подготовки цифровых тестов.

УДК 681.84

Накопители на магнитных дисках для малых и персональных ЭВМ/Б. Цонев, Б. Ценкулов, К. Митев//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 49—56.

Представлены разработки НРБ в области накопителей на магнитных дисках. Приведены характеристики интерфейсов, особенности схем позиционирования, схем привода основного двигателя, параметры типичных мини- и микроНМД, а также параметры надежности накопителей.

УДК 681.3.181.5

Стандартизация интерфейсов микроЭВМ/А. Михальский//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 56—59.

Проведено сравнение нескольких типов интерфейсов, наиболее широко используемых для микроЭВМ. Показано, что по сравниваемым параметрам самая высокая оценка связана с интерфейсами. По мнению автора, качество интерфейсов и особенно учет существующих стандартов во многом определяют коммерческий успех микроЭВМ.

УДК 681. 324

Технические и программные средства вычислительных сетей СМ ЭВМ в ЧССР/Я. Глушек, В. Грищ//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 59—68.

Представлены новые технические средства для создания вычислительных сетей, такие, как асинхронный адаптер, асинхронный мультиплексор, синхронный адаптер и процессор телеобработки, а также целый ряд модемов. Даны характеристики специальных программных средств управления сетью.

УДК 681.306

Система SOFTORG — интегрированная система вспомогательных средств для современной разработки прикладных систем/К. Фюле//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 69—75.

Рассмотрены особенности комплекса средств разработки программного обеспечения, обеспечивающего создание автоматизированных систем обработки данных, начиная с постановки задачи и определения потребностей и кончая выпуском документации и контролем качества разработки. Представлены все этапы проектирования систем и соответствующие им средства, помогающие разработчику или полностью автоматизирующие отдельные процедуры проектирования. Описаны функциональные свойства отдельных компонентов системы SOFTORG.

УДК 681.3.06 : 519.6

Неортогональность и ортогональность в современных языках программирования/Г. Штиллер//ВТ соц. стран.— М.: Финансы и статистика, 1988.— Вып. 24. — С. 75—82.

Дана концепция ортогональности с одновременным обсуждением разных языков программирования с позиции этой концепции в отношении типов данных. Акцент делается на взаимодействии принципов описания типов данных, эквивалентности типов данных и их контроля. Показано, как можно повысить ортогональность.

УДК 681.3.06

Применение языка Пролог для экспертных систем/У. Петерсон, Ю. Пичке//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 82—93.

Рассмотрены характеристики экспертных систем с функциональной и программной точек зрения и показаны их возможности, дающие новое качество в системах обработки знаний по сравнению с традиционными вычислительными системами. Описаны основные компоненты экспертных систем: база знаний, система логического вывода, блок объяснения работы системы и человеко-машинный интерфейс. Для конструирования экспертных систем применяются методы логического программирования на основе языка Пролог. Дан краткий обзор технологий разработки и областей применения экспертных систем.

УДК 681.3.06

Венгерские системы Пролог/И. Футо//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 93—103.

Кратко описана история создания языка Пролог. Рассмотрены основные элементы программ на языке. Проиллюстрировано основное правило, с помощью которого Пролог решает задачи. Описаны структуры данных, используемые в языке. Достаточно подробно рассмотрены структура и функции компонентов системы программирования Пролог венгерской разработки — МПролог.

УДК 681.3.06

Качество программных средств, пути и методы решения/В. П. Куприянов, С. Л. Котов, А. В. Смагин//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 103—111.

Описаны результаты работы по созданию методики оценки качества программных средств (ПС) в совете по применению СВТ. Методика основана на концепции жизненного цикла ПС. Приведены основные положения методики, рассмотрены организационные аспекты проведения экспертизы ПС и определена область применения методики.

УДК 681.3.068 : 519.685

Инвариантное ядро пакетов прикладных программ/В. В. Подбельский// ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 111—118.

Рассмотрен подход к снижению затрат на создание программных комплексов, состоящий в выделении из существующих ППП типовых компонентов с созданием на их основе ядра, достаточно инвариантного к смене предметных областей. Инвариантное ядро позволяет с минимальными затратами создавать специализированные проблемно-ориентированные программные комплексы для научно-технических расчетов и учебных целей.

УДК 681.3.001.72

Об унификации технологии проектирования СОД/Э. Н. Хотяшов//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 119—126.

Рассмотрен подход к созданию систем обработки данных через формализованное представление процесса проектирования в виде технологической сети проектирования. Проведена классификация технологических операций, которая является основой для машинного построения и анализа такой технологической сети. Показано, что технологическая сеть позволяет также обосновать структуру документации по пакетам прикладных программ, определяющей технологию применения такого пакета.

УДК 681.3.068

Система управления базой данных MIMER/Д. Шрайтер//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 126—133.

Описаны возможности и опыт использования переносимой СУБД MIMER реляционного типа для моделей ЕС и СМ ЭВМ. Основной вариант СУБД пред-

назначен для применения на персональных ЭВМ в многопользовательском режиме. Приведены функции базовых компонентов СУБД MIMER.

УДК 681.3.06

Система программирования Фортран 77 для ЕС ЭВМ и перспективы ее развития/Г. Д. Смирнов, В. И. Цагельский, Э. В. Ковалевич, З. С. Брич//ВТ соц. стран.— М.: Финансы и статистика, 1988.— Вып. 24.— С. 134—138.

Рассматриваются назначение и основные возможности системы программирования Фортран 77 для ЕС ЭВМ, базирующейся на актуальном состоянии международного языка Фортран. Система программирования содержит средства диалога и отладки, обеспечивающие разработку программ на разных этапах. Приводятся основные отличия языка Фортран 77 от языка Фортран 1966 года, а также концепции, положенные в основу реализаций изданий 01 и 02 системы программирования Фортран 77, перспективы ее развития.

УДК 681.3.181.5

Проблемно-независимые экспертные системы для микроЭВМ — проектирование, применение, анализ/Л. Даковски, Н. Касабов//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 139—143.

Показано, что одной из главных проблем, которые необходимо преодолеть при проектировании экспертных систем для микроЭВМ, являются ограничения по оперативной памяти и быстродействию. Предложена структура продукционной экспертной системы, состоящей из редактора базы знаний, доказывающей программы и модулей интерфейсного расширения. Обсуждены возможная стратегия доказательства и реализация этой стратегии. Рассмотрены результаты применения экспертной системы на персональных 16-разрядных ЭВМ в области обучения, медицины и сельского хозяйства.

УДК 681.324.019.3

Экспертная система ЭСИНА/И. П. Попчев, Н. П. Златарева//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 143—152.

Описана реализация экспертной системы для решения задач, связанных с надежностью новых изделий. Система предназначена для выдачи консультаций при определении надежности проектируемых или заказываемых изделий, консультаций по проектированию и оценке надежности для разработчика, а также консультаций по организации испытаний на надежность. Приведена общая структурная схема системы, описаны функции подсистем, организация базы знаний, механизм вывода. Главная особенность системы — возможность описания в базе знаний сложной проблемной области с неполной и ненадежной информацией.

УДК 681.3.06:519.68

Языковой процессор Паскаль для операционной системы МОСВП/Л. Мишев, Д. Стоянова//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 153—157.

Описан компилятор языка Паскаль для работы в среде операционной системы МОСВП для 32-разрядной ЭВМ ИЗОТ 1055С. Язык Паскаль для МОСВП сконструирован так, чтобы максимально использовать преимущества данной системы. Приведены особенности реализованной версии языка, классы расширений, типы данных и процедур.

УДК 681.3

Особенности организации информационного обеспечения в системе комплексного централизованного обслуживания и использования СВТ/М. Е. Кобранов, О. Н. Квашнин//ВТ соц. стран.— М.: Финансы и статистика, 1988.— Вып. 24. — С. 158—161.

Рассмотрены, исходя из целей и задач ГКВТИ, вопросы организации информационного обеспечения в системе КЦО, вопросы развития этой системы. Основное внимание уделено особенностям системы, связанным со спецификой СКЦОИ средств ВТ. Показана роль сотрудничества социалистических стран в решении проблем информационного обеспечения.

УДК 681.322:62—192

О некоторых результатах функционирования системы сбора и обработки информации о надежности СВТ в НОТО СССР/Е. А. Ануфриенко, Г. Р. Юзбашьянц // ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 161—165.

Описана работа информационной системы в НОТО СССР, позволяющей анализировать состояние и вырабатывать управляющие воздействия для процессов разработки, изготовления, эксплуатации и централизованного обслуживания СВТ. Накопленная исходная информация проверяется на достоверность и используется для проверки соответствия фактических показателей надежности заданным в ТУ, для выявления элементов, определяющих уровень надежности, анализа результатов деятельности изготовителей по повышению надежности ТС ЕС ЭВМ.

УДК 681.3.06

Организация работы консультационных пунктов ГДР по программному обеспечению/К.-Х Мюллер//ВТ соц. стран.— М.: Финансы и статистика, 1988.— Вып. 24. — С. 165—169.

В 1986 г. с целью повышения эффективности применения существующего в стране программного обеспечения (ПО) были определены основные принципы планирования работ в этой области и созданы консультационные пункты и центральный банк данных о ПО. Рассмотрены основные виды деятельности консультационных пунктов, содержание банка данных и использованные для его создания технические и программные средства, а также порядок взаимодействия новой для ГДР организации с международным фондом алгоритмов и программ ИНТЕРФАП.

УДК 681.322—181.4.009.5

ЕС1834 — ППЭВМ комбината «Роботрон»/Ф. Кёлер//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 170—174.

Описана функциональная структура и особенности конструкции ППЭВМ ЕС1834, построенной на базе 16-разрядного микропроцессора К1810ВМ86. Емкость оперативной памяти — 256—640 Кбайт. ППЭВМ работает в алфавитно-цифровом и графическом режимах. Возможно подключение накопителя типа «винчестер», цветного дисплея и печатающего устройства.

УДК 681.327.22

Графическая дисплейная станция СМ7411 (ИЗОТ 1040С)/Г. Кукурешков, А. Величков, Е. Николов//ВТ соц. стран.— М.: Финансы и статистика, 1988. — Вып. 24. — С. 175—180.

Рассмотрены структура, принцип работы и функциональные характеристики современной цветной графической станции для систем автоматизации инженерного труда на базе 16—32-разрядных мини-ЭВМ. Станция обслуживает до 1—2 рабочих мест и имеет в составе цветной растровый дисплей, планшет-кодировщик, интерактивный манипулятор, клавиатуру.

УДК 681.3.068

Программные средства для реализации интерфейса X.25 на основе СМ ЭВМ/ П. Парин, С. Цонев, П. Стойчева//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 180—184.

Описано программное средство, обеспечивающее подключение ЭВМ типа СМ-4 к сети общего пользования с коммутацией пакетов. Рассмотрены структура пакета и функции модулей. Приведен перечень поддерживаемых устройств.

УДК 681.3.06

Операционная система реального времени BOS-1810/П. Кюльборн//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 185—189.

Рассмотрены функциональные свойства мультизадачной операционной системы реального времени для ППЭВМ СМ1910. Даны сведения о требованиях к

техническим средствам и поддерживаемом конфигураторе. Описаны функции компонентов системы.

УДК 681.327.21/22

Графические периферийные устройства СМ ЭВМ производства ГДР/ И. Гантц//ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 189—193.

Приведены технические характеристики интеллектуального графического терминала СМ1647, графического планшета СМ6422, устройств вывода графической информации на бумагу СМ6415/6416 и устройств печати СМ6329.01М/6329.02М.

УДК 681. 327.21/22 — 621.3.049.75

Средства обработки графической информации и новые печатающие устройства производства ГДР / Б. Лойшель // ВТ соц. стран. — М.: Финансы и статистика, 1988. — Вып. 24. — С. 193—199.

Приведены технические параметры и описаны функциональные возможности устройств ввода графической информации СМ6417 и СМ1618 и проблемно-ориентированного процессора обработки изображений СМ0512. Представлены характеристики новых последовательных печатающих устройств. Описанные средства разработаны и выпускаются комбинатом «Роботрон» (ГДР).

Local Network MICRONET-16/V. Furnadjiev, G. Naidenov, P. Stoianov, V. Petrov//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 3—12.

Problems of design and realization of the 16-bit micro computer local network are considered. Functional possibilities and technical data of local network, functional diagram of transport controller are described. Multiaccess method with determined solution of conflicts is suggested. New types of network operations are defined and protocols realising them are presented. Method and user interaction procedures with network are described.

The Local Ring Network Terminal Station SLK-SM/J. N. Glukhov, M. S. Koloskov, J. B. Kozschevnikov, A. L. Kuznetsov//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 12—15.

The terminal SLK-SM for local ring network realizing access method of register insert is described. Terminal software permits to meet requirements of protocol system providing connection of devices with various levels of «intellegence» to the network. Terminal has its selfdiagnostics. Using of segment reservation for network channel with automatic switching and raising noise immunity of transmission permits to use the network SLK—SM in offices and commercial systems.

Architecture of Highperformance System for Simulation of Digital Circuts/ N. L. Prokhorov, B. G. Sergeev//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 16—24.

Short survey of up-to-date special computer hardware and software for simulation of digital circuits is given. Architecture and principles of design of highperformance multiprocessor system for CAD ELSI, speeding up the simulation process 1000 times comperatively with general purpose computers are described. Basic structural and technical characteristics of system processors are suggested.

Hardware and Software for the Local Network ROLANET 1/J. Richter, B. Terpe//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 24—33.

Configuration and structure of the network ROLANET 1, as well as block-diagram of controller, transceiver and methods of switching to the ES or SM network are considered. Short description of basic software components of local network control is given. Network could be used for development of information system for business management or computer aided design system.

Interactive Graphic Systems IGS-2 and IGS-3 Manufactured in the Czechoslovak Socialist Republic/V. Kopecký // In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 33—40.

General structure of graphic information processing and architecture of graphic terminal stations of the first, second, and third generations is described. Functional fetures and technical characteristics of the two new graphic systems IGS-2 and IGS-3 are described.

Tester for the Circuit Board R 3040/H. Siebert, H.-D. Prange//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988.— Issue 24. — P. 40—48.

Tester for intracircuit inspection of discret and integrated, analog and digital circuits, breaks in circuit boards after mounting and soldering is described. Tester is designed on the basis of mini computer «Electronika 60-1» and two disks of 35 M byte each, two floppy disks, and two terminals. Basic modules of program library used to prepare digital tests are characterized.

Magnetic Disks for Mini and Personal Computers/B. Tsonev, B. Tsenkulov, K. Mitev//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 49—56.

Developments in the field of magnetic disks in the Peoples Republic of Bulgaria are considered. Interface, characteristics, positioner design features, design of basic driver as well as technical data of typical mini and micro magnetic disks are given. Reliability parameters of disks are considered.

Micro Computer Interface Standardization/A. Michalsky//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 56—59.

Comparison of technical data for several types of interfaces widely used for micro computers manufactured in various countries of the world is provided. It is shown that by comparable parameters the highest grade is interrelated to interfaces. In authors opinion interface quality, and in particular taking into consideration the existing standards, defines commercial succes of micro computers.

Hardware and Software for SM Computer Networks in the Czechoslovak Socialist Republic/J. Hlušhek, V. Hric//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 59—68.

New hardware used in development of computer networks such as asynchronous adapter, asynchronous multiplexer, synchronous adapter and teleprocessor, as well as the whole line of modems are considered. Characteristics of special software for network control are given.

The SOFTORG System — Integrated Accessory System for Up-to-date Development of Application Systems/K. Fule//In coll. atricles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 69—75.

Features of software development system, providing data processing system design from formulation of a problem and requirements to documentation release and quality control are considered. All stages of the system development and related facilities for designer or fully automated certain design procedures are suggested. Functions of certain components in the SOFTORG system are described.

Nonortogonality and Ortogonality in the Modern Programming Languages/G. Stiller/In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 75—82.

Concept of ortogonality along with simultaenouse discussion of various programming languages from the point of view of this concept relating to the data types is given. Accented is interaction of principles of data type description, equivalence of data types and its control. It is shown how to raise ortogonality.

Application of PROLOG Language for Expert System Development/U. Peterson, J. Pitschke//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 82—93.

Specific features of expert systems are considered from the functional, programming points of view, and their possibilities giving new qualities to the knowledge processing systems in comparison to the traditional computer systems are shown. Basic components of expert systems: data base, logic input system, component for interpretation of system operation, and interface man—computer are described. Methods of logical programming based on the PROLOG language are used in expert system design. Short review of development technologies and application fields for expert systems is given.

The Hungarian PROLOG Systems/J. Futo//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 93—103.

The history of PROLOG language development is shortly described Basic programming components of the language are considered. Basic rule used by PROLOG for problem solution is illustrated. Data structures used in language are described. Structure and functions of the PROLOG programming system, developed in Hungary — MPROLOG is considered in detail.

Problems of Software Quality. Methods of their Solution/V. P. Kuprianov, S. L. Kotov, A. V. Smagin//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 103—111.

The results of an activity in developing of methods for quality evaluation of software within the frame of the Counsil for Application of Electronic Computer Technology are described. The methods are based on the concept of software life cycle. Basic aspects of methods are presented, managerial aspects for software expertizing are considered and field of application is defined.

Invariant Kernel of Application Programms/V. V. Podbelski//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 111—118.

An approach to the lowering of expences for development of programming systems is considered. Approach lies in separation from the existing application programms of typical components, and setting up from these components a kernel base sufficiently invariable for change of the objective fields. Invariant kernel permits with minimum expences to set up special problem—oriented program systems for technical and scientific computations and training purposes.

On the Unification of Data Processing System Design Technology/E. N. Khotiashov//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 119—126.

An approach to the formation of data processing system by formulation of design process as a technological design system is considered. Classification of technological operations — the basis for computer aided design and analysis of such technological system is given. It is shown that the technological system also permits to ground documentation structure of application programms, defining the technology of application of such application package.

Data Base Management System MIMER. Functions and Application Conditions/D. Schreiter//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 126—133.

Possibilities and application experience of the transferable DBMS MIMER for the ES and SM computers are described. The basic modification of DBMS is intended for application with personal computers in multiuser mode. Functions of DBMS MIMER components are presented.

Programming System FORTRAN 77 for the ES Computers and Prospects of Its Future Development./G. D. Smirnov, V. I. Tsagelski, E. V. Kovalevich, Z. S. Britch//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 134—138.

Destination and basic possibilities of the program system FORTRAN 77 for the ES computers based on actual state of the art of an international language FORTRAN are considered. The program system comprises interactive program debugging facilities providing program development at various stages. Basic distinctions of the language FORTRAN 66, as well as concepts used for realization of 01 and 02 releases of programming system FORTRAN 77, prospects of its future development are presented.

Problem-Independent Expert Systems for Micro Computer — Design, Application, Analysis/L. Dakovski, N Kasabov//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Lssue 24. — P. 139—143.

It is shown that one of the main problems to be solved in design of expert system for micro computer are limitations of main storage and performance of computer system. Structure of the productive expert system comprising knowledge base editor, proving programms, and modules for interface extention is suggested. Possible proving strategy and realization of this strategy are discussed. Results of expert system apllication with 16-bit personal computer in the fields of training, health, and agriculture are considered.

Expert System in the Field of Product Reliability ESINA/I. P. Popchev, N. P. Zolotareva // In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 143—152.

Realization of expert system for the solution of reliability problems for new products is described. The system is intended for providing consultations on defining of projected or ordered products, consultations on designed reliability evaluation for designer as well as consultations on organization of reliability tests. General system structure is given; subsystem functions, formation of knowledge base, output gear are described. The main feature of the system is possibility of description in knowledge base of complicated problem field with impcomplete and unreliable information.

The Language Processor PASCAL for the Operating System MOSVP/L. Mishev, D. Staianova//In coll articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 153—157.

Compiler from PASCAL for operation in the operating system environment MOSVP for the 32-bit computer ISOT 1055 C is described. The PASCAL language for MOSVP is so designed, that permits to use fully advantages of this system. The author describes features of the realized language version, extension classes, data types, and procedures.

Organization of Information Facilities within the Frame of the Complete Centralised Computer Service and Computer Application/M. E. Kobranov, O. N. Kvashin//In coll. articles: Vychisl. tekhnika sots. stran.— Finansy i statistika, 1988. — Issue 24. — P. 158—161.

Problems of organization of information facilities within the Complete Centralized Computer Service System of the State Committee on Computer Technology and Informatics, problems of the future development of this system are cosidered. Main attention is paid to the system features of the Complete Centralized Computer Service. The role of cooperation of the socialist countries in solution of information facility problems is shown.

On Some Results of the System for Collection and Processing of Computer Reliability Information in the National Maintenance Organization of the USSR/ E. A. Anufrienko, G. R. Juzbashiants//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 161—165.

Operating of information system in the USSR National Maintenance Organization, permitting to analyse status and develop control action for the process of development, manufacturing, operation and centralized maintenance of computers is described. Collected source information is checked for reliability and is used for comparison of actual reliability and that of specifications, for exposure of components defining reliability level, analisys of manufacturer activity in improvement of the ES computer hardware reliability.

Formulation and Functioning of Software Sonsulting Sites in the German Democratic Republic/K.-H. Mueller//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 165—169.

Having in mind raising of software application effectiveness in this country the basic principles for planning of activities in this field were defined as well as consulting sites and software central bank were set up in 1986. Basic features of consulting site activities, data bank content, as well as data bank hardware and software are considered. Interaction of this new for the GDR organization with international fund of algorithms and programms INTERFAP is described.

The Professional Personal Computer ES1834 Manufactured by the Kombinat ROBOTRON/F. Köhler/In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 170—174.

Functional structure and features of the PPC ES 1834 based on 16-bit microprocessor K 18010BM86 are described. The main memory capacity is 256—640 K Byte. PPC operates in alpha—numeric and graphic mode. It is possible to connect winchester disk, color display, and printer.

The Graphic Display Station SM7411/G. Kukureshkov, A. Velichkov, E. Nikolov//In coll. articles: Vychisl. tekhnika sots. stran. M.: Finansy i statistika, 1988. — Issue 24. — P. 175—180.

Structure, mode of operation, and functional features of up-to-date color graphic station used for computer aided engineering system based on 16-bit and 32-bit mini computers are cons idered. The station is intended for one or two working sites and includes color raster display, encoder—board, interactive operator, keyboard.

Software for Realization of Interface X.25 Based on the SM Computer/ P. Parin, S. Ionev, P. Stoicheva//In coll. articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 180—184.

Software, providing possibility of connecting SM-4 Computer to the network with commutation of application programms is described. Structure of application programms and functions of moduls are considered. List of Supported devices is given.

The Real Time Operating System BOS-1810/P. Kühlborn//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 185—189.

Functional features of multitask real-time operating system for the professional personal computer SM 1910 are considered. Information on technical requirements to hardware and supported configuration is given. System component functions are described.

The SM Graphic Peripherals Manufactured in the German Democratic Republic/ J. Gantz//In coll articles: Vychisl. tekhnika sots. stran.— M.: Finansy i statistika, 1988. — Issue 24. — P. 189—193.

Specifications for intellegent graphic terminal SM 1647, graphic plotter SM 6422, hard copy graphic output unit CM 6415/6416 and printer SM 6329.02M are presented.

Graphic Information Processing Fasilities. New Printers Manufactured in the German Democratic Republic/B. Leuschel//In coll. articles: Vychisl. tekhnika sots. stran. — M.: Finansy i statistika, 1988. — Issue 24. — P. 193—199.

Technical features are presented, functional possibilities of the graphic information input devices SM 6417 and SM 1618 as well as problem — oriented image processor SM 0512 are given. Features of the new serial printers are presented. Printers are developed and manufactured by the Kombinat ROBOTRON (the German Democratic Republic).

Научное издание

Вычислительная техника
социалистических стран

Выпуск 24

Науч. редактор *Ю. П. Селиванов*
Зав. редакцией *И. Г. Дмитриева*
Редактор *Л. А. Табакова*
Худож. редактор *Е. К. Самойлов*
Техн. редакторы *Л. Н. Фокина, И. В. Завгородняя*
Корректоры *Т. М. Колпакова, М. М. Виноградова, Е. Ю. Потапкина*
Обложка художника *Б. С. Вехтера*

ИБ № 2224

Сдано в набор 20.06.88. Подписано в печать 11.10.88. А 11282. Формат 60×90$^{1}/_{16}$. Бумага офс. № 2. Гарн. «Литературная». Печать высокая. Усл. печ. л. 13,5. Усл. кр.-отт. 13,75. Уч.-изд. л. 14,93. Тираж 18 000 экз. Заказ 481. Цена 95 коп.

Издательство «Финансы и статистика».
101000, Москва, ул. Чернышевского, 7.

Московская типография № 8 Союзполиграфпрома
при Государственном комитете СССР
по делам издательств, полиграфии и книжной торговли.
101898, Москва, Центр, Хохловский пер., 7.

ВНИМАНИЮ ЧИТАТЕЛЕЙ!

В издательстве «Финансы и статистика» в 1989 году выйдут следующие книги.

Многоуровневое структурное проектирование программ: Теоретические основы, инструментарий/ Е. Л. Ющенко, Г. Е. Цейтлин, В. П. Грицай, Т. К. Терзян: Науч. изд. — 14 л. — 2 р. 40 к.

Освещается круг вопросов, связанных с формализацией процесса проектирования программ на основе алгебры программ. Излагается сущность метода многоуровневого структурного проектирования программ; описывается семейство согласованных языков схем программ и ассоциированных с ним средств автоматизации программирования по данному методу. Изложение сопровождается иллюстрациями, поясняющими сущность понятийного аппарата, на котором основан предлагаемый метод.

Книга рассчитана на инженерно-технических работников, программистов, может быть полезна студентам вузов соответствующих специальностей.

Ванагс Э. Я. **АСОД в административном районе:** Науч. изд. — 12 л. — 2 р. 10 к.

Рассматриваются роль автоматизированной системы обработки данных (АСОД) в совершенствовании территориального управления, основные принципы создания АСОД административного района, использование регионального автоматизированного банка данных, развитие функциональных подсистем районной автоматизированной системы и ее экономическая эффективность.

Для экономистов, бухгалтеров, инженеров и других специалистов предприятий, организаций и районных органов управления, разработчиков и эксплуатационников территориальных АСОД.

Книги издательства можно приобрести в магазинах, распространяющих общественно-политическую литературу. Там же можно ознакомиться с тематическим планом выпуска литературы на 1989 год и сделать предварительные заказы.

95 к.

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА СОЦИАЛИСТИЧЕСКИХ СТРАН

Выпуск 24

Вычислительная техника социалистических стран, 1988, вып. 24, 1–214