МИКРОПРОЦЕССОРНЫЕ СРЕДСТВА И СИСТЕМЫ

3 | 1986

ISSN 0233-4844

Представление знаний — новая ступень в использовании вычислительных средств

Микропроцессорный комплект БИС серии К1815 для конвейерных систем цкфровой обработки сигналов: быстродействующее АЛУ, преобразователь кодов последовательных чисел, микросхема ортогональной регистровой памяти

Компьютер «Ириша»: клавиатура, телевизионный монитор, источник питания, программа работы с кассетным магнитофоном

Кросс-средства для автоматизации разработки, тестирования и отладки программ однокристальных микроЭВМ серии КМ1816: цикл статей

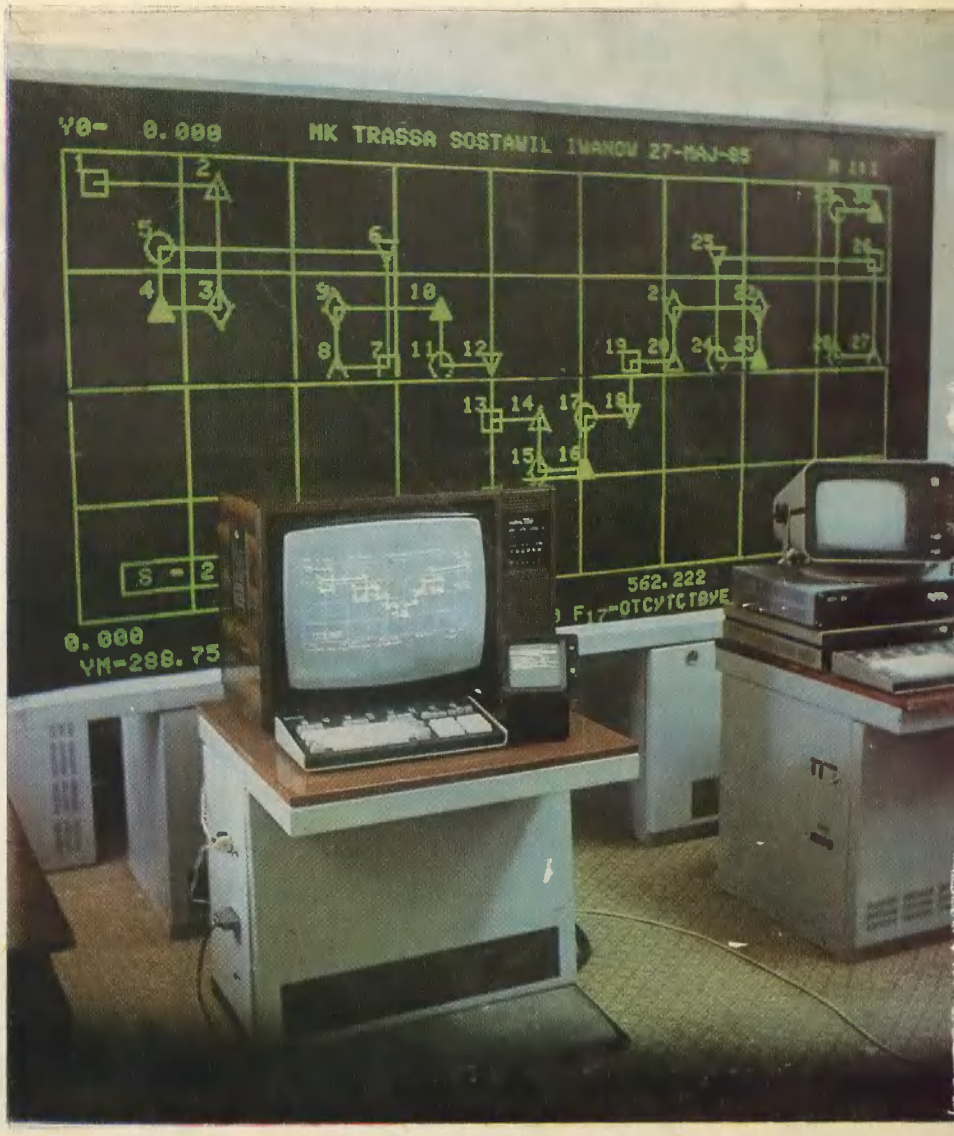
Контроллер накопителя на гибком магнитном диске КР1818ВГ93: технические характеристики, система команд, служебные коды и форматы записи, временные диаграммы режимов работы, особенности применения

Спецпроцессор на базе микропроцессора К580ИК80А позволяет использовать практически любую ЭВМ для разработки программного обеспечения микропроцессорных устройств, имеющих в основе БИС серии К580

Модуль «Регенератор функционирования программ» поддерживает механизм восстановления работоспособности микроЭВМ в условиях помех

Адаптер магистралей на основе БИС серии КР1802 для связи СМ ЭВМ и микроЭВМ «Электроника 60М»

Автоформализация профессиональных знаний — вторая информационная революция



ПЕРСОНАЛЬНАЯ АВТОНОМНАЯ ГРАФИЧЕСКАЯ СТАНЦИЯ

Отличительные особенности графической станции ПЕГАС — универсальность назначения, возможность автономной работы, простота конструкции.

Автономность обеспечивается микроЭВМ «Электроника МС 1201.01», встроенной в конструктив алфавитно-цифрового дисплея 15ИЭ-00-013.

Черно-белое изображение на растре 400×275 строится непосредственно в памяти микроЭВМ.

Системный канал микроЭВМ доступен для подключения дополнительных устройств, которые могут размещаться в корпусе графической станции.

ПЕГАС используется для автоматизации труда технологов-программистов в гибкой интегрированной производственной системе КАПРИ.

Персональкая графическая станция ПЕГАС
Функциональные блоки ПЕГАС
(см. ст. М. М. Ляпунова и Б. И. Белякова)

Пролетарии всех стран, соединяйтесь!

ОРГАН ГОСУДАРСТВЕННОГО КОМИТЕТА СССР ПО НАУКЕ И ТЕХНИКЕ

Издается с 1984 года

МП МИКРОПРОЦЕССОРНЫЕ СРЕДСТВА И СИСТЕМЫ

ВЫХОДИТ ШЕСТЬ РАЗ В ГОД НАУЧНО-ТЕХНИЧЕСКИЙ И ПРОИЗВОДСТВЕННЫЙ ЖУРНАЛ 3/1986 МОСКВА

СОДЕРЖАНИЕ

**Адрес редакции: 101820, проезд Серова, 5, редакция журнала «Микропроцессорные средства и системы»
Телефоны 228-18-88; 221-99-26**

Сдано в набор 30.04.86. Т 10457
Подписано к печати 23.06.86.
Формат 84×108¹/₁₆. Бумага № 1.
Высокая печать. Усл. печ. л. 10,08
Уч.-изд. л. 14,4. Тираж 54 300.
Зак. № 107. Цена 1 р. 10 к.

Орган Государственного комитета СССР по науке и технике

Московская типография № 13
ПО «Периодика» ВО «Союзполиграфпром» Государственного комитета СССР по делам издательств, полиграфии и книжной торговли.
107005, Москва, Б-5, Денисовский пер., д. 30

На 1-й странице обложки — Многотерминальная система отображения информации (см. статью Н. Н. Насруллаева и др.)

ОБРАБОТКА ИНФОРМАЦИИ: ОТ ДАННЫХ К ЗНАНИЯМ

Происходящее у нас на глазах становление информатики как науки и как сферы человеческой деятельности вызывает брожение умов, которое при всей своей разнонаправленности и кажущейся противоречивости неуклонно действует в двух важнейших направлениях: формирует новые понятия и процессы, а также показывает в новом свете и переосмысливает многие компоненты науки и практики, бывшие достоянием человеческой цивилизации в течение тысячелетий.

Сказанное в полной мере относится к таким понятиям, как «представление знаний», «обработка знаний» и «базы знаний». На первый взгляд они воспринимаются как экзотические признаки 5-го поколения ЭВМ. Однако уже следующий шаг вглубь показывает, что мы не сможем продвинуться в будущее ни на шаг, не подвергнув анализу всю многовековую практику добычи, накопления и использования знаний.

Мы привычно смотрим на ЭВМ как на «систему обработки данных». Само слово «данные», пришедшее из статистики и научного эксперимента, вызывает в нашем воображении строй немых цифр, пробуждаемых к жизни волшебной палочкой программиста.

Знание всегда считалось священной прерогативой человека. И хотя общее знание зачастую отличается от единичного факта лишь нюансами грамматики (сравните: «Поезд пришел в 9 утра» и «Поезд приходит в 9 утра»), их употребление всегда размещалось на разных этапах мыслительной деятельности.

Первый уровень отчуждения знания от его носителя был достигнут при появлении письменности, без которой был бы невозможен переход от диалогов Сократа к логике Аристотеля. Книгопечатание резко усилило объективный компонент в накоплении знания, но по-прежнему сохраняя его своеобразный межличностный характер.

Новая эпоха началась с появления первой библиотеки стандартных программ. Программа — это принципиально важный пример знания, реализуемого без участия человека. Потребовалось меньше двадцати лет, чтобы осознать, что программное обеспечение стало, по выражению акад. Г. И. Марчука, «овеществленным интеллектом человечества».

В какой мере программа является воплощением знания, в такой же степени трансляторы, системы программирования, а затем и системы разработки программ по спецификациям стали первыми системами обработки знаний.

Аналогичное развитие претерпели информационные системы. Начав со складирования элементарных и неизменных данных, эти системы постепенно обрастали возможностями обобщенной обработки информации, привлекая логический вывод в помощь сортировке.

Постепенно информатика приближается к решению наиболее важной задачи: извлечение знания из данных и модификация имеющегося знания на основе накопления новых фактов. И опять программирование является источником важных моделей: синтез программ по примерам и конкретизация универсальных программ на основе дополнительного знания об области их применения.

Очень важную прикладную и научную роль должны сыграть экспертные системы. В их разработке центральное место занимает очень трудная проблема объединения эмпирического и формального знания.

В повышении уровня машинной обработки информации мы сталкиваемся с интересной диалектикой: для того, чтобы перейти от данных к знаниям, нам нужно научиться обрабатывать знания как данные — другими словами, создать вычислительные методы обработки знаний. Обработка данных стала возможной благодаря таким мощным процедурам, как обращение матриц, нахождение собственных значений, численное интегрирование, симплекс-метод, сортировка, транзитивное замыкание и т. п. И хотя в обработке знаний мы уже можем назвать некоторые процедуры: метод резолюций, алгоритм Кнута — Бендикса, смешанные вычисления — основная работа по вычленению и обоснованию подобных методов еще впереди.

А. Ершов

УДК 681.327.8.06

В. А. Коваленко, А. В. Олейник, Л. П. Пархоменко, Л. М. Солдатенко

БИС КОНТРОЛЛЕРА КР1818ВГ93 ДЛЯ НАКОПИТЕЛЯ НА ГИБКОМ ДИСКЕ

Микросхема КР1818ВГ93 представляет собой однокристальное программируемое устройство, предназначенное для управления выводом информации из ЭВМ на гибкие магнитные диски и вводом информации из НГМД в ЭВМ. БИС обеспечивает программирование номеров дорожки, сектора и стороны диска, а также длины сектора, режимов поиска дорожки и установки магнитной головки (МГ) в исходное положение, режимов чтения или записи информации, скорости перемещения МГ.

Контроллер КР1818ВГ93 позволяет организовать автоматический контроль считываемой и записываемой информации по контрольному коду (КК), записанному в конце индексного и информационного массивов. Индексный массив включает адресную метку, номер сектора, длину сектора, номер дорожки и номер стороны диска. Информационный массив содержит метку и непосредственно данные.

В режиме записи микросхема обеспечивает выдачу сигналов предкомпенсации записи в зависимости от кодов, представляющих информацию. Вывод информации из ЭВМ выполняется по сигналу Запрос данных, формируемому микросхемой, а считывание определяется сигналами Готовность и Индексный импульс, выдаваемыми аппаратурной логикой НГМД. Технические данные и характеристики микросхемы КР1818ВГ93 при $T=-10...70$ °С, $U_{cc}=5$ В±5 % и 12 В±5 % и токах потребления 60 и 20 мА приведены ниже.

Электрические параметры микросхемы

Входное напряжение высокого уровня, В, не менее	2,6
Входное напряжение низкого уровня, В, не более	0,8
Выходное напряжение высокого уровня, В, не менее	2,8
Выходное напряжение низкого уровня, В, не более	0,45
Выходной ток высокого уровня, мА, не более	—0,15
Выходной ток низкого уровня, мА, не более	1,9
Емкость нагрузки по выходам пФ, не более	100
Максимальная потребляемая мощность, мВт, не более	500

Обмен информацией с ЭВМ происходит по 8-разрядной двунаправленной шине данных. Запись информации на ГМД осуществляется с одинарной или удвоенной плотностью. Под одинарной плотностью подразумевается запись информации с частотной модуляцией (рис. 1), а под

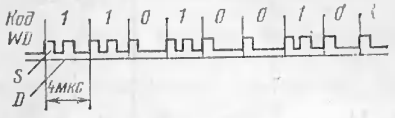


Рис. 1. Режим записи с частотной модуляцией при $f_{CLC}=2$ МГц

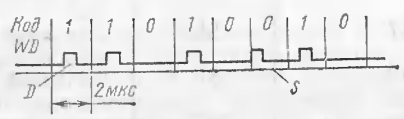


Рис. 2. Режим записи с модифицированной частотной модуляцией при $f_{CLC}=2$ МГц: S — синхросигнал, D — импульс данных

удвоенной — запись информации с модифицированной модуляцией (рис. 2). Микросхема обеспечивает работу с ГМД размером 133 или 203 мм (по длине стороны конверта). Максимальное программируемое число дорожек на ГМД — 256. Максимальная скорость обмена информацией при одинарной плотности записи составляет 250 Кбит/с, при удвоенной плотности записи — 500 Кбит/с. Тактовая частота внешнего генератора равна 1 МГц для ГМД размером 133 мм и 2 МГц для ГМД размером 203 мм. Микросхема выполнена в 40-выводном корпусе типа 2.123.40-2. Назначение выводов БИС КР1818ВГ93 показано в табл. 1.

Выводы 33, 38, 39 микросхемы — выходы с открытым истоком, требующие подключения к источнику питания U_{cc1} через резисторы номиналом 10 кОм±10 %. Назначение внутренних регистров БИС, выбираемых с помощью адресных сигналов A0, A1, следующее:

регистр данных (РгД) и регистр сдвиговый (РгСдв) — для приема, хранения и преобразования данных;

регистр сектора (РгСект) — для хранения информации о номере считываемого (записываемого) сектора;

регистр (дорожки (РгДор) — для записи номера требуемой дорожки или хранения информации о номере дорожки, на которой находится МГ;

регистр команд (РгКом) — для записи текущей выполняемой команды;

регистр состояния (РгСост) — для определения текущего состояния различных функциональных узлов микросхемы и НГМД.

Функциональное назначение каждого разряда РгСост при выполнении соответствующих команд приведено в табл. 2 (указанным признакам соответствует «Лог. 1» в РгСост).

Микросхема обеспечивает прием и выполнение 11 команд. Все команды условно разделены на четыре типа: *вспомогательные, записи и чтения информации, поиска и чтения индексного поля на ГМД, принудительного прерывания* (табл. 3). Зависимость времени перемещения МГ от кодов $Ч_1$, $Ч_0$ и состояния входного сигнала $\overline{TEST}$ показана в табл. 4.

Команда Восстановление обеспечивает переход МГ на нулевую дорожку ГМД. Если на входе $\overline{TR00}$ нет подтверждения о выходе на нулевую дорожку после выдачи 256 импульсов, выполнение команды прекращается.

Команда Поиск предполагает, что РгДор содержит информацию о текущем номере дорожки, а РгД — требуемой дорожки. Перемещение МГ выполняется до тех пор, пока содержимое РгДор не сравняется с содержимым РгД. Поиск выполняется при V=1.

Команда Шаг обеспечивает выдачу импульса на перемещение МГ на один шаг. Направление перемещения при этом не изменяется.

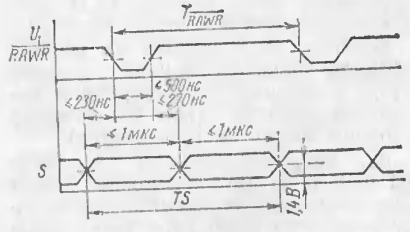


Рис. 3. Временные диаграммы сигналов считывания данных с НГМД и синхронизации

Команды Шаг вперед и Шаг назад обеспечивают выдачу сигнала DIRC (направление перемещения).

Команды типа 2 обеспечивают считывание информации с ГМД (рис. 3) и запись ее на ГМД. Перед вводом этих команд необходимо в РгСект установить номер требуемого сектора. Длина сектора задается кодом и записывается в индексной области при форматировании диска в соответствии с выводом, приведенным ниже.

Длина сектора	Число байт в секторе
00	128
01	256
02	512
03	1024

Временные параметры сигналов загрузки и готовности МГ показаны на рис. 4.

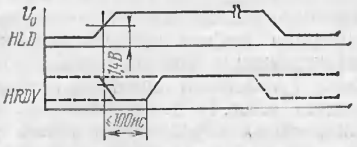


Рис. 4. Временные диаграммы сигналов загрузки и готовности МГ

По командам типа 2 выполняется запись или чтение требуемого сектора по заданным номерам стороны, сектора и дорожки ГМД с проверкой индексной адресной метки (ИАМ) и (КК). Признаки команд этого типа соответственно обозначают (см. табл. 3):

m — код, указывающий на обращение к одному (m=0) или нескольким (m=1) секторам. При m=0 после считывания (записи) одного сектора работа прекращается. При m=1 после окончания работы с первым сектором в РгСект прибавляется единица и начинается обработка следующего сектора. Эта операция продолжается до тех пор, пока не будет обработан самый последний сектор на данной дорожке;

S — код, определяющий номер стороны диска (0 или 1);

E — код, указывающий на выполнение задержки продолжительностью 15 мс для установки МГ в рабочее положение после сигнала HLD (при E=0 задержка не осуществляется);

C — код, указывающий на необходимость проверки номера стороны ГМД в процессе идентификации индексной области (при C=0 номер стороны диска не проверяется);

a_0 — код, используемый для выбора одного из двух возможных байт признака защиты данных для записи в области ИАМ (при a_0=1 записывается байт F8, указывая, что данные могут стираться; при a_0=0 записывается байт FB, указывая, что область данных сохраняется).

Таблица 1

Назначение выводов БИС КР1818ВГ93

Вывод	Обозначение	Выполняемая операция
1	2	3
1	BS	Вывод микросхемы не подключается. Предназначен для контроля уровня напряжения смещения подложки
2	$\overline{W}$	Разрешение записи информации с шины данных в выбранный регистр
3	$\overline{CS}$	Выбор микросхемы разрешает связь ЭВМ с микросхемой
4	$\overline{R}$	Разрешение чтения обеспечивает вывод информации из выбранного регистра на шину данных DB0 ... DB7
5, 6	A0, A1	Адресная шина. Код на этой шине определяет выбор соответствующего регистра для приема (передачи) информации с (на) шины данных, как показано ниже

A1	A0	Чтение	Запись
0	0	РгСост	РгКом
0	1	РгДор	РгДор
1	0	РгСект	РгСект
1	1	РгД	РгД

7 ... 14	DB0 ... DB7	8-разрядная двунаправленная шина данных
15	STEP	Выходной импульс для перемещения МГ на один шаг
16	DIRC	Сигнал, указывающий направление перемещения МГ: высокий — к центру ГМД; низкий — от центра
17	SL	Выходной сигнал, указывающий, что импульс данных WD должен быть сдвинут влево
18	SR	Выходной сигнал, указывающий, что импульс данных WD должен быть сдвинут вправо
19	$\overline{CLR}$	Сброс обеспечивает установку микросхемы в исходное состояние и запись кода 0000 0011 в регистр команд. На выходе 39 (INTRQ) устанавливается низкий уровень напряжения. По окончании действия сигнала CLR выполняется команда Восстановление независимо от готовности НГМД. Кроме того, записывается код 0000 0001 в регистр сектора
20	GND	Корпус
21	U_{cc1}	Напряжение питания 5 В±5 %
22	$\overline{TEST}$	При подаче на этот вход сигнала высокого уровня микросхема вырабатывает импульсы управления перемещением МГ (STEP) с повышенной частотой
23	HRDY	МГ в рабочем положении. Входной сигнал, указывающий, что МГ готова к работе
24	CLC	Сигналы тактовой частоты
25	RSTB	Строб чтения подтверждает прием данных от НГМД. На выходе устанавливается напряжение высокого уровня после приема двух байтов нулей при одинарной плотности записи и после приема четырех байтов нулей или единиц при удвоенной плотности записи
26		Синхронизирующий тактовый сигнал, вырабатываемый из сигналов $\overline{RAWR}$
27	$\overline{RAWR}$	Импульсный сигнал входных данных, считываемых с НГМД
28	HLD	Выходной сигнал, управляющий магнитной головкой
29	TR43	Выходной сигнал, указывающий, что МГ находится между дорожками 44 ... 76. Генерация сигнала происходит только в процессе выполнения команд Запись, Считывание
30	WSTB	Строб записи имеет высокий уровень на время записи информации на ГМД

Окончание табл. 1

Вывод	Обозначение	Выполняемая операция
1	2	3
31	WD	Сигналы записи данных на ГМД
32	CPRDY	Входной сигнал, указывающий на готовность НГМД выполнять команды Считывание или Запись. Если сигнал CPRDY низкого уровня, команды Считывание, Запись не выполняются и вырабатывается сигнал INTRQ. Вспомогательные команды, обеспечивающие подготовку НГМД к работе, выполняются независимо от состояния сигнала CPRDY
33	WF/DE	Двунаправленная шина, используемая для обозначения ошибки записи и размещения выбора данных, поступающих от ЭВМ. Если сигнал WSTB = =1, вывод WF/DE функционирует как WF-вход. Если сигнал WF=0, запись какой-либо команды будет немедленно прекращена. Если сигнал WSTB=0, вывод 33 функционирует как DE-выход. На выходе DE в процессе чтения после загрузки МГ и установки высокого уровня сигнала. HRDY будет напряжение низкого уровня
34	$\overline{\text{TROO}}$	Входной сигнал, указывающий микросхеме, что МГ установлена в исходное положение
35	$\overline{\text{JP}}$	Входной сигнал с НГМД, информирующий микросхему о том, что индексный импульс считан и ГМД начал очередной оборот
36	$\overline{\text{WPRT}}$	Входной сигнал запрещения записи на ГМД. Низкий уровень сигнала прекращает запись
37	$\overline{\text{DDEW}}$	Входной сигнал, указывающий микросхеме, с какой плотностью должны выполняться операции
38	DRQ	Выходной сигнал в режиме чтения указывает, что регистр данных содержит информацию для передачи. В режиме записи сигнал DRQ указывает на готовность приема информации с шины данных. Этот сигнал устанавливается в состояние низкого уровня, если данные считаны в ЭВМ или записаны из ЭВМ в регистр данных
39	JNTRQ	Готовность микросхемы. На этом выходе устанавливается напряжение высокого уровня, если выполнена какая-либо команда, и напряжение низкого уровня, если микросхема выполнила команду или считан регистр состояния
40	U_{cc2}	Источник питания 12±5 %

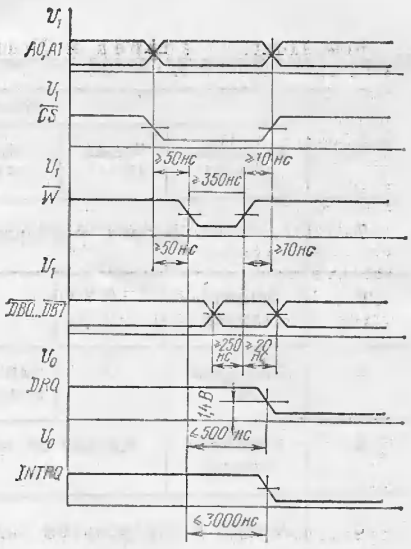


Рис. 6. Режим чтения

Контрольный код представлен в виде двух байт и вычисляется как циклическая сумма полинома

$$A = X^{15} + X^{12} + X^5 + 1.$$

Команда Чтение сектора выполняется, когда идентифицированы номера дорожки, сектора и КК (рис. 5). Адресная метка данных должна быть установлена через 30 байт для одинарной и через 43 байта для двойной плотности записи после КК индексной области. Если ИАМ не найдена, вырабатывается признак Массив чтения не найден, который выдается в РгСост. После прохождения адресной метки байты данных вводятся в РгСдв и передаются РгД. Каждый байт сопровождается сигналом DRQ Готовность данных. РгД должен быть считан до приема следующего байта. Если предыдущий байт не считан, записывается следующий, а в РгСост записывается признак Потеря данных.

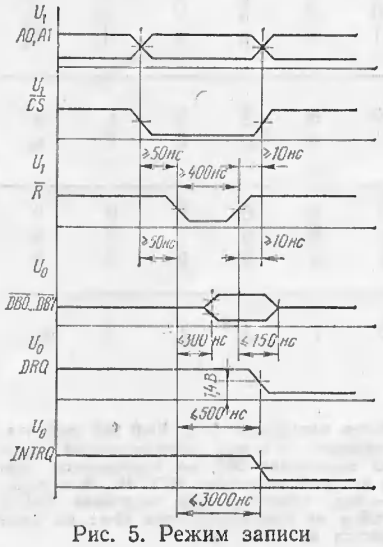


Рис. 5. Режим записи

В конце считывания массива данных КК должен совпадать с генерируемым в микросхеме. Если они не совпадают, выставляется бит Ошибка КК в РгСост и прекращается выполнение команды даже при п=1.

Команда Запись сектора выполняется подобно команде Чтение сектора в части анализа индексного массива, определения номера дорожки, стороны диска, длины сектора и вычисления КК (рис. 6). Сигнал DRQ генерируется, запрашивая первый байт данных, который должен быть записан на ГМД. Затем микросхема вычисляет 11 байт при одинарной (или 22 при двойной) плотности записи для обеспечения пробела между индексной областью и данными. С момента прохождения 11 или 22 байт (если первый запрос сигнала DRQ обслужен и данные записаны в РгД) выдается строб записи WSTB и 6 байт нулей для одинарной (или 12 байт для двойной) плотности записываются на диск. Это соответствует записи пробела, а затем записывается ИАМ. Байт признака данных может быть или FB (без стирания данных) или F8 (со стиранием) в соответствии с кодом a_0.

При записи данных на ГМД каждый байт заносится в РгД, передается в РгСдв и затем на диск. Сигнал DRQ вырабатывается для ЭВМ на каждый последующий байт данных. Если DRQ не обслужен, вырабатывается сигнал Потеря данных в разряде S1 РгСост, а на диске записывается байт нулей. После записи данных записывается КК в виде двух байт, генерируемых микросхемой, затем один байт FF, и устанавливается низкий уровень сигнала WSTB.

Таблица 2

Назначение битов регистра состояния

Разряд	Выполняемая команда					
	Вспомогательная	Чтение адреса	Чтение сектора	Чтение дорожки	Запись сектора	Запись дорожки
7	Разряд, указывающий на готовность НГМД					
6	Защита записи	0	0	0	Защита записи	
5	Загрузка МГ	0	Запись со стиранием	0	Ошибка записи	
4	Ошибка поиска	Массив не найден		0	Массив не найден	0
3	Ошибка в контрольном коде			0	Ошибка в контрольном коде	0
2	МГ в исходном состоянии	Потеря данных				
1	Индексный импульс	Запрос данных				
0	Занято (идет выполнение команды)					

Таблица 3

Структура команд контроллера КР1818ВГ93

	Команда	Структура кода, бит							
		7	6	5	4	3	2	1	0
Тип 1	Восстановление	0	0	0	0	h	V	$Ч_1$	$Ч_0$
	Поиск	0	0	0	1	h	V	$Ч_1$	$Ч_0$
	Шаг	0	0	1	И	h	V	$Ч_1$	$Ч_0$
	Шаг вперед	0	1	0	И	h	V	$Ч_1$	$Ч_0$
	Шаг назад	0	1	1	И	h	V	$Ч_1$	$Ч_0$
Тип 2	Чтение сектора	1	0	0	m	S	E	C	0
	Запись сектора	1	0	1	m	S	E	C	a_0
Тип 3	Чтение адреса	1	1	0	0	0	E	0	0
	Чтение дорожки	1	1	1	0	0	E	0	0
	Запись дорожки	1	1	1	1	0	E	0	0
Тип 4	Принудительное прерывание	1	1	0	1	J_3	J_2	J_1	J_0

Примечание. h — код установки МГ в рабочее положение (при h=0 МГ поднята, при h=1 МГ устанавливается в рабочее положение); V — код, определяющий необходимость проверки положения МГ (при V=0 положение МГ не проверяется, при V=1 читается и проверяется номер дорожки, на которой находится МГ); $Ч_1$, $Ч_0$ — коды, определяющие скорость перемещения МГ; И — код, определяющий состояние РгДор при перемещении ММГ (при И=0 состояние РгДор не изменяется, при И=1 на каждом шаговом импульсе состояние РгДор изменяется на один бит).

Таблица 4

Коды установки времени перемещения магнитной головки

TEST	$Ч_1$	$Ч_0$	Время перемещения на шаг, мс	
			CLC=1 МГц	CLC=2 МГц
1	0	0	6	3
1	0	1	12	6
1	1	0	20	10
1	1	1	30	15
0	—	—	400	200

Команды типа 3 предназначены для поиска информации на диске или записи информации (форматирование диска). Структура кода содержит один бит признака, определяющего необходимость включения задержки 15 мс после сигнала HLD, как и при выполнении команд типа 2.

Команда Чтение адреса выполняется при установке МГ в рабочее положение (HLD=1). В бит состояния Занято записывается единица. Последовательно считываются 6 байт индексной области, включая КК, и передаются на шину данных в сопровождении сигнала DPQ. КК считывается и передается на шину данных, микросхема проверяет его, если КК не совпадает, выдается бит состояния Ошибка КК и продолжается выполнение команды чтения. При выполнении этой команды содержимое РгДор пересылается в РгСект и запоминается. По окончании выполнения команды генерируется сигнал INTRQ и очищается бит состояния Занято.

Команда Чтение дорожки обеспечивает чтение всей информации, включая индексный массив, контрольные коды, пробелы и массив данных, и передачу ее в ЭВМ. В процессе чтения не выдается строб чтения и не выполняется проверка КК, что позволяет использовать данную команду в диагностических целях.

Команда Запись дорожки предназначена для разметки ГМД. Информация в ЭВМ для этой процедуры должна содержать все пробелы и индексные метки. Любая последовательность данных, имеющаяся в ЭВМ, записывается. Если появляются байты F5...FE, то они интерпретируются как адресные метки данных. Контрольный код генерируется в момент передачи байтов F8...FE из РгД в РгСдв в режиме ЧМ или при появлении байта F5 в режиме МЧМ. При появлении кода F7 КК записывается двумя байтами. Таким образом, байты F5...FE не должны записываться в местах пробелов, области данных или индексных массивах.

Таблица 5

Назначение байтов информации БИС КР1818ВГ93

Байт данных	Назначение	
	режим частотной модуляции	режим модифицированной частотной модуляции (МЧМ)
00 ... F4	Запись 00 ... F4 CLK=FF	Запись 00 ... F4
F5	Не допускается	Запись A1, инициализация КК
F6	Не допускается	Запись C2
F7	Запись двух байтов КК	Запись двух байтов КК
F8 ... FB	Запись F8 ... FB CLK=C7, инициализация КК	Запись F8 ... FB
FC	Запись FC с CLK=D7	Запись FC
F	Запись FD с CLK=FF	Запись FD
FE	Запись FE CLK=C7, инициализация КК	Запись FE
FF	Запись FF с CLK=FF	Запись FF

Команда типа 4 Принудительное прерывание задается для завершения какой-либо выполняемой команды. В отличие от других команд она может быть записана в РгКом в любой момент времени.

Однако исполнение команды может определяться состоянием младших битов J_0...J_3. Если биты J_0...J_3 находятся в состоянии 0, прекращается выполнение текущей команды и сигнал INTRQ не вырабатывается. При

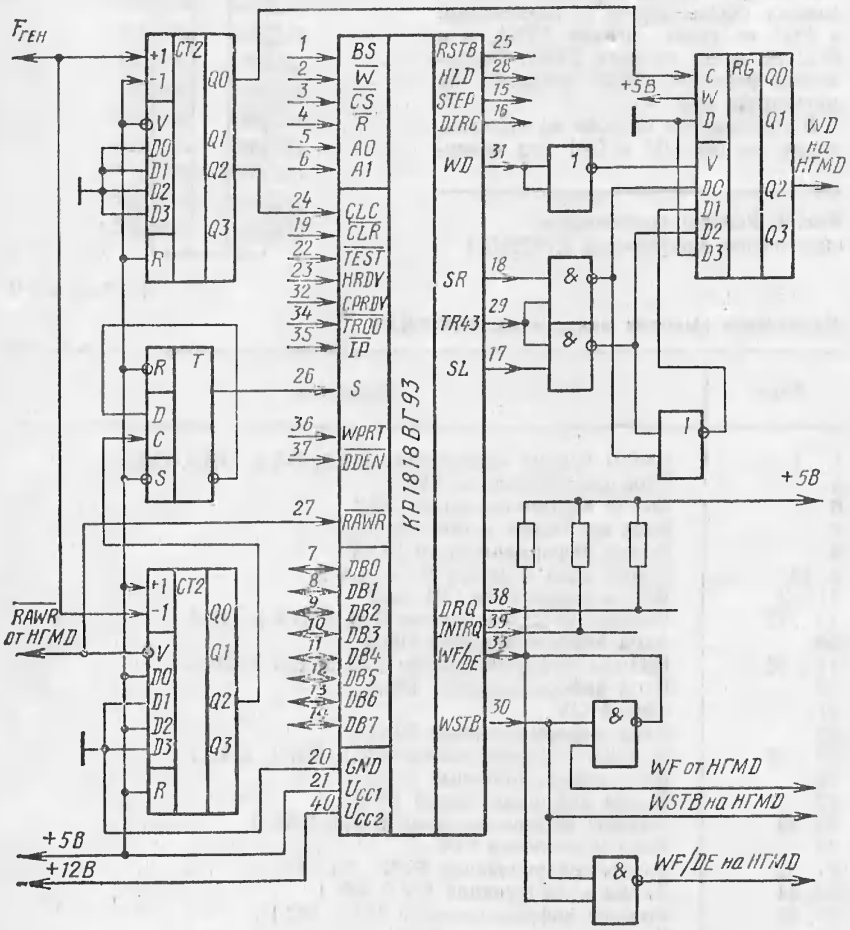


Рис. 7. Схема включения БИС КР1818ВГ93 с формирователями импульсов синхронизации S и обеспечения предкомпенсации сигналов WD

Таблица 6

Форматы массивов данных в режиме записи с частотной модуляцией

Число байтов	Кол	Назначение
40	FF (00)	Пятый пробел (от начала индексного импульса)
6	00	
1	FC	Индексная метка
26	FF (00)	Первый пробел
6	00	
1	FE	Адресная метка индексных данных
1	XX	Номер дорожки
1	XX	Номер стороны диска (00 или 01)
1	XX	Номер сектора (1 ... 1A)
1	XX	Длина сектора (00)
1	F7	Запись двух байтов КК
11	FF (00)	Второй пробел
6	00	
1	FB	Адресная метка данных
128	XX	Данные
1	F7	Запись двух байтов КК
27	FF (00)	Третий пробел
247	FF (00)	Продолжение записи до выдачи прерывания (четвертый пробел до начала индексного импульса)

Один сектор

J_0=1 прерывание выполняется после перехода сигнала CPRDY из низкого уровня в высокий. J_1=1 определяет прерывание при переходе CPRDY из высокого уровня в низкий. J_2=1 — прерывание по приходу индексного импульса JP. J_3=1 обеспечивает немедленное прерывание выполняемой команды. После выполнения этих условий вырабатывается сигнал INTRQ.

Каждый служебный байт (табл. 5) может быть размещен в индексной области в соответствии с форматом массива. Байт FC определяет индексную метку, которая ставится перед первым индексным массивом. FE — адресную метку индексных данных, которая записывается в начале индексного массива. F7 — код, который указывает на необходимость записи результата вычислений двух байтов КК.

В табл. 6 и 7 приведены примерные форматы массивов данных, записываемых на ГМД соответственно с одинарной и удвоенной плотностью. При записи отдельных служебных кодов с ЧМ часть синхросигналов опускается. При этом наличие сигналов S определяется кодом CLK, приведенным в табл. 6.

Таблица 7

Форматы массивов данных в режиме записи с модифицированной частотной модуляцией

Число байтов		Код	Назначение
80		4E	Пятый пробел (от начала индексного импульса)
12		00	
3		F6	Запись С2
1		F6	Индексная метка
50	Один сектор	4E	Первый пробел
12		00	
3		F5	Запись А1
1		FE	Адресная метка индексных данных
1		XX	Номер дорожки (0—4С)
1		XX	Номер стороны (0 или 1)
1		XX	Номер сектора (1 ... 1А)
1		XX	Длина сектора (01)
1		F7	Запись двух байтов КК
22		4E	Второй пробел
12		00	
3		F5	Запись А1
1		FB	Адресная метка данных
256		XX	Данные
1		F7	Запись двух байтов КК
54		4E	Третий пробел
598		4E	Продолжение записи до выдачи прерывания (четвертый пробел до начала индексного импульса)

Представленный контроллер нашел практическое применение для управления мини-дисками (рис. 7) в профессиональной персональной ЭВМ «Электроника МС 0585».

Статья поступила 13 января 1986 г.

НА КНИЖНОЙ ПОЛКЕ

Ершов А. П. Три урока по программированию.— М.: Наука. Гл. ред. физ.-мат. лит., 1987 (IV кв.).— 5 л.— 20 к.

Содержит изложение основ программирования, достаточно строгое, но живое и иллюстративное. Схема рассказа такова: даются общие основы программирования, затем привлекается внимание к некоторым тонкостям; наконец, основные понятия и приемы иллюстрируются примерами.

Для широкого круга читателей — всех, кто хотел бы быстро ознакомиться с основами программирования.

В последующих трех статьях раздела завершается рассмотрение комплекта микропроцессорных БИС для цифровой обработки сигналов. Первые четыре статьи этого цикла с описанием универсального процессорного элемента К1815ВФ1, сумматора последовательных чисел К1815ИМ1, накапливающего сумматора с интерфейсом К1815ВФ2, микропроцессора быстрого преобразования Фурье К1815ВФ3 были опубликованы в «МП», 1986 г., № 2, с. 14.

УДК 681.325.5-181.48

В. В. Горовой, В. А. Евдокимов, В. И. Медведев, А. М. Сахаров

БИС СПЕЦИАЛИЗИРОВАННОГО АЛУ К1815ИА1

БИС специализированного АЛУ К1815ИА1 предназначена для построения процессоров с последовательно-параллельной обработкой информации. БИС содержит четыре одноразрядных АЛУ с общим управлением. Условное графическое обозначение БИС приведено на рис. 1, назначение выводов в табл. 1.

В состав каждого разряда входят два буферных регистра PG1, PG2, одноразрядное АЛУ ALU, регистр «маски» PG3, регистр переноса PG4, три коммутатора S1...S3 (рис. 2).

В буферные регистры PG1 и PG2 информация подается с магистралей DA и DB соответственно. Запись данных осуществляется независимо: в PG1 по срезу сигнала SYN1, а в PG2 по срезу сигнала SYN2. Содержимое регистра PG2 выдается на магистраль DQ.

В зависимости от кода на управляющих входах DU и C01 над содер-

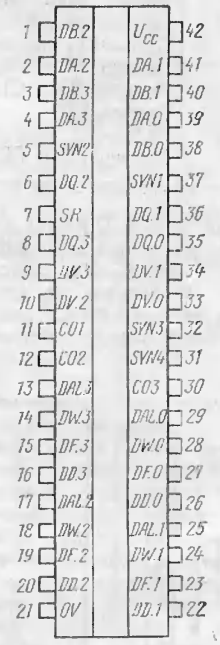


Рис. 1. Условно-графическое обозначение микросхемы К1815ИА1

Таблица 1

Назначение выводов микросхемы К1815ИА1

Вывод	Назначение
1 ... 4	Входы приема информации DB.2, DA.2, DB.3, DA.3
5	Вход синхронизации SYN2
6	Выход информационный DQ.2
7	Вход начальной установки SR
8	Выход информационный DQ.3
9, 10	Входы кода функции DV.3, DV.2
11, 12	Входы управления C01, C02
13 ... 15	Выходы информационные DAL.3, DW.3, DF.3
16	Вход информационный DD.3
17 ... 19	Выходы информационные DAL.2, DW.2, DF.2
20	Вход информационный DD.2
21	Общий OV
22	Вход информационный DD.1
23 ... 25	Выходы информационные DF.1, DW.1, DAL.1
26	Вход информационный DD.0
27	Выход информационный DF.0
28, 29	Выходы информационные DW.0, DAL.0
30	Вход управления C03
21, 32	Входы синхронизации SYN4, SYN3
33, 34	Входы кода функции DV.0, DV.1
35, 36	Выходы информационные DQ.0, DQ.1
37	Вход синхронизации SYN1
38 ... 41	Входы информационные DB.0, DA.0, DB.1, DA.1
42	Напряжение питания U_{CC}

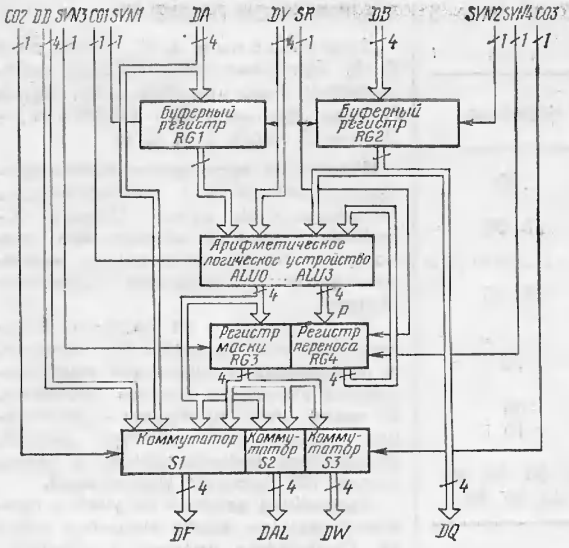


Рис. 2. Структурная схема БИС К1815ИА1

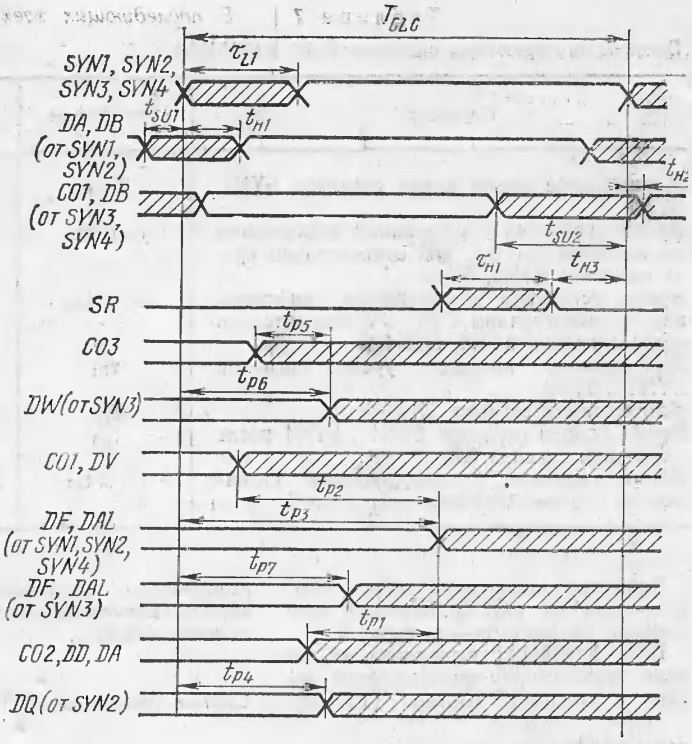


Рис. 3. Временная диаграмма работы БИС К1815ИА1

жимым регистров PG1 и PG2 в ALU выполняются соответствующие операции (табл. 2). В регистр PG3 по срезу сигнала SYN3 записывается результат операции, выполняемой ALU.

Возникающий при выполнении арифметических операций перенос может быть записан в PG4 по срезу сигнала SYN4.

Четырехразрядный коммутатор S1 обеспечивает передачу на выходы DF0...3 сигналов с трех направлений:

с выходов АЛУ при состоянии «0» триггеров «маски» PG3.0...3 и значении «1» сигнала C02;

со входов DA.0...3 при значении «1» триггеров «маски» PG3.0...3 и значении «1» сигнала C02;

со входов DD.0...3 при значении «0» сигнала C02.

Четырехразрядный коммутатор S2 обеспечивает передачу на выход БИС DAL.0...3 информации с выходов АЛУ при значении «0» триггеров «маски» PG3.0...3. Четырехразрядный коммутатор S3 служит для выдачи на выходную магистраль DW состояния регистра «маски» PC3 при значении «0» сигнала C03. В БИС предусмотрена установка в «0» регистров PG1, PG2, PG4 при состоянии «1» на входе SR.

Основные электрические параметры БИС в диапазоне рабочих температур —10...+85 °С представлены ниже.

Таблица функций БИС К1815ИА1

Таблица

№ п/п	Набор функций DV3	DV2	DV1	DV0	C0I = 1 Логическая функция	C0I = Арифметическая функция $\overline{P}O = 1$	$\overline{P}O = 0$
1	0	0	0	0	$\overline{A}$	A	A + 1
2	0	0	0	1	$\overline{A \vee B}$	A + B	(A + B) плюс 1
3	0	0	1	0	$\overline{A} \vee B$	A + $\overline{B}$	(A + $\overline{B}$) плюс 1
4	0	0	1	1	0	минус 1	0
5	0	1	0	0	$A\overline{B}$	A плюс $A\overline{B}$	A плюс $A\overline{B}$ плюс 1
6	0	1	0	1	$\overline{B}$	(A + B) плюс $A\overline{B}$	(A + B) плюс $A\overline{B}$ плюс 1
7	0	1	1	0	$A \oplus B$	A минус B минус 1	A минус B
8	0	1	1	1	$A \wedge \overline{B}$	$A\overline{B}$ минус 1	$A\overline{B}$
9	1	0	0	0	$\overline{A} \vee B$	A плюс AB	A плюс AB плюс 1
10	1	0	0	1	$\overline{A \oplus B}$	A плюс B	A плюс B плюс 1
11	1	0	1	0	B	(A плюс $\overline{B}$) плюс AB	(A + $\overline{B}_1$) плюс AB плюс 1
12	1	0	1	1	$A \wedge B$	AB минус 1	AB
13	1	1	0	0	1	A плюс A*	A плюс A плюс 1
14	1	1	0	1	$A \vee \overline{B}$	(A + B) плюс A	(A + $\overline{B}_1$) плюс A плюс 1
15	1	1	1	0	$A \vee B$	(A + $\overline{B}$) плюс A	(A + $\overline{B}$) плюс A плюс 1
16	1	1	1	1	A	A минус 1	A

Электрические параметры БИС 1815ИА1

Выходное напряжение низкого уровня U_{OL}, В, не более . . .	0,5
Выходное напряжение высокого уровня U_{OH}, В, не менее	2,4
Входной ток низкого уровня I_{IL}, мА, не более	0,2
Входной ток высокого уровня I_{IH}, мкА, не более	40
Ток потребления I_{cc}, при U_{cc}=5В±10 %, мА, не более	150
Максимальная рабочая частота, МГц, не менее	8,3

Таблица 3

Временные параметры сигналов БИС К1815ИА1

Параметр	Обозначение	Норма, нс
Минимальное время цикла сигналов SYN1, SYN4	T_{CLC}	120
Время установки и удержания информации по магистралям DA, DB относительно среза сигналов SYN1, SYN2	t_{SU1}, t_{H1}	25, 25
Время установки и удержания информации по магистралям CO1, DV относительно среза сигналов SYN3, SYN4	t_{SU2}, t_{H2}	35, 15
Длительность низкого уровня сигналов SYN1 ... SYN4	τ_{L1}	75
Длительность сигнала SR	τ_{H1}	35
Время подачи сигналов SYN1 ... SYN4 после окончания сигнала SR	t_{H3}	15
Время задержки распространения сигналов до соответствующих магистралей	t_{p1} ... t_{p7}	35, 50, 50, 40, 20, 50, 40

Временная диаграмма работы БИС приведена на рис. 3, значения временных параметров — в табл. 3.

БИС К1815ИА1 выполнена на основе транзисторно-транзисторной логики с диодами Шотки (ТТЛШ). Микросхема выпускается в 42-выводном металлокерамическом корпусе типа 429.42-1.

Статья поступила 30 августа 1985 г.

НА КНИЖНОЙ ПОЛКЕ

Кушниренко А. Г., Лебедев Г. В. **Программирование для математиков:** Учеб. пособие.— М.: Наука. Гл. ред. физ.-мат. лит. 1987 (IV кв.)— 24 л.— (В пер.): 1 р. 10 к.

Излагается курс практического программирования для математических специальностей вузов. Основу составляют понятия исполнителя, технология программирования «сверху — вниз» и развитие структуры данных.

Курс рассчитан на решение большого количества задач по изучению и модификации тщательно подготовленных эталонных текстов программ. В числе этих программ — реализация различных структур данных, реализации мини-проектов, а также около 100 простых упражнений.

Изложение ведется на учебно-производственном языке высокого уровня. Приводятся правила и примеры перекодировки программ с этого языка на язык Фортран.

Для студентов математических факультетов университетов и факультетов прикладной математики вузов.

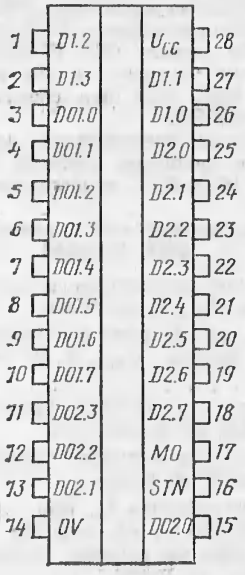
УДК 621.382.33

В. И. Кулешов, А. В. Прибыльский, В. С. Сякерский, Ю. В. Яковлев

БИС ОРТОГОНАЛЬНОЙ МАТРИЦЫ РЕГИСТРОВ СДВИГА К1815ИР1

Одним из основных требований, предъявляемых к многопроцессорным вычислительным системам, можно считать необходимость создания быстродействующих устройств для преобразования параллельного кода в последовательный и наоборот с одновременным перераспределением информации между рядами передатчиков и приемников. Эти функции могут выполнять последовательно-параллельные регистры сдвига, организованные в виде ортогональных матриц статической регистровой памяти. К ним можно отнести микросхему быстродействующей ортогональной матрицы регистров сдвига (ОМРС) с перестраиваемой структурой К1815ИР1 информационной емкостью 8×4 бит. На кристалле размером 3,7×3,35 мм² содержится ~2100 элементов. Условное графическое обозначение БИС К1815ИР1 представлено на рис. 1. Назначение выводов показано в таблице. Структурная схема БИС приведена на рис. 2.

В качестве элементов матрицы регистров сдвига ис-



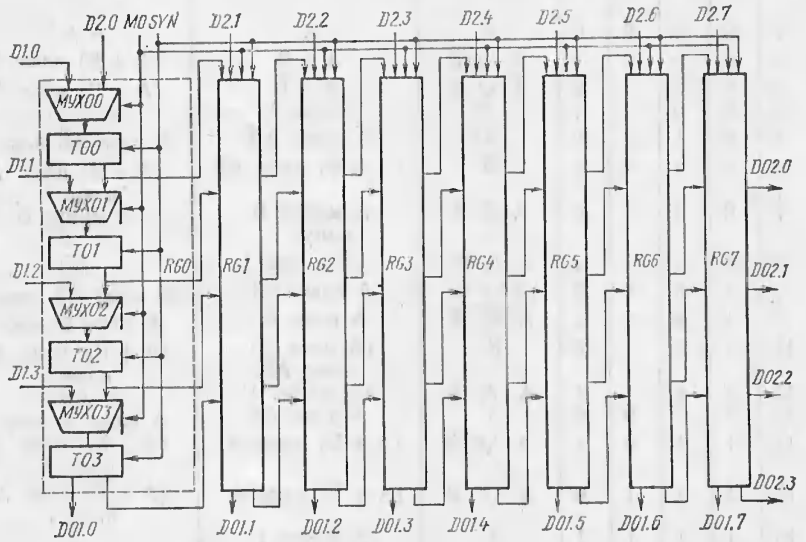


Рис. 1. Условное графическое обозначение БИС К1815ИР1
Рис. 2. Структурная схема БИС К1815ИР1

Вывод	Обозначение	Назначение
1	D1.2	Входная магистраль данных D1
2	D1.3	»
3 ... 10	D01.0 ... D01.7	Выходная магистраль данных D01
11 ... 13	D02.3 ... D02.1	Выходная магистраль данных D02
14	0V	Общий
15	D02.0	Выходная магистраль данных D02
16	SYN	Вход синхронизации
17	MO	Вход управления приемом информации
18 ... 25	D2.7 ... D2.0	Входная магистраль данных D2
26	D1.0	Входная магистраль данных D1
27	D1.1	»
28	U_{cc}	Напряжение питания 5 В

пользуется D-триггер с динамическим синхровходом (рис. 3). На входе каждого триггера установлен мультиплексор МУХ, управляемый шиной МО, которая обеспечивает две конфигурации БИС:

1. При подаче на шину МО высокого уровня матрица принимает конфигурацию «восемь 4-разрядных сдвиговых регистра» (сдвиг по столбцам). Ввод информации может осуществляться только через входную шину D2.0 ... D2.7.

2. При подаче на шину МО низкого уровня матрица принимает конфигурацию «четыре 8-разрядных сдвиговых регистра» (сдвиг по строкам). Ввод информации осуществляется только через входную шину D1.0 ... D1.3.

Выдача информации начинается с нижней строки и

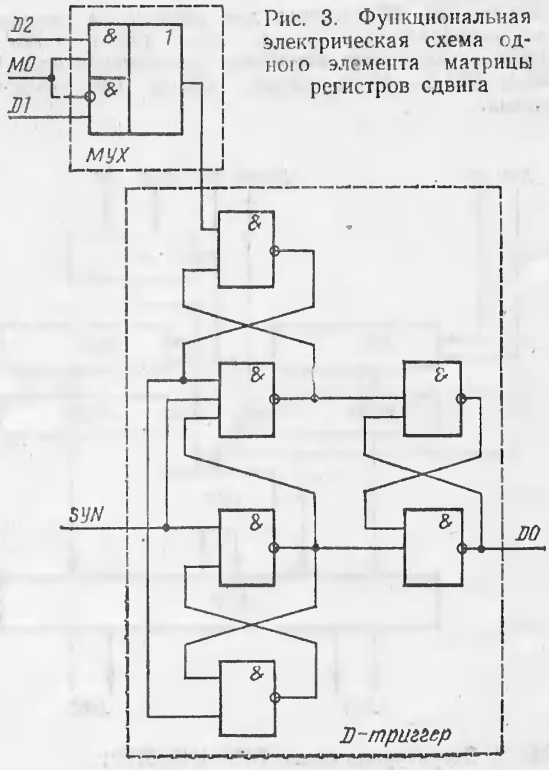


Рис. 3. Функциональная электрическая схема одного элемента матрицы регистров сдвига

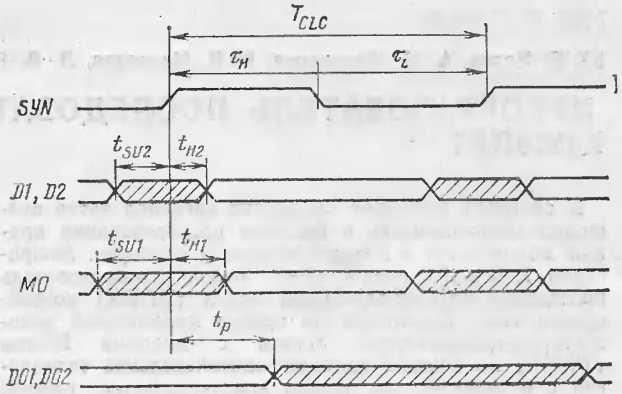


Рис. 4. Временная диаграмма работы микросхемы БИС К1815ИР1

крайнего правого столбца соответственно на шины D01.0 ... D01.7 и D02.0 ... D02.3. БИС позволяет наращивать разрядность обрабатываемой информации по четыре или (и) восемь разрядов. В микросхеме К1815ИР1 используется синхронный способ организации внутреннего управления, т. е. запись информации во все регистры осуществляется положительным фронтом сигнала синхронизации SYN (рис. 4). Период следования импульсов синхросигнала SYN составляет $T_{CLC} \geqslant 118$ нс. Основные динамические и электрические параметры БИС К1815ИР1 представлены ниже.

Динамические параметры БИС К1815ИР1
(в наносекундах)

Время цикла T_{CLC}	118
Длительность низкой полки синхросигнала SYN, τ_L	42
Длительность высокой полки синхросигнала SYN, τ_H	30
Время опережения информации, подаваемой на входы D1, D2 относительно входа SYN, t_{SU2}	25
Время опережения сигнала, подаваемого по входу МО относительно входа SYN, t_{SU1}	35
Время удержания информации, подаваемой на входы D1, D2 относительно входа SYN, t_{H2}	10
Время удержания сигнала, подаваемого по входу МО относительно входа SYN, t_{H1}	15
Время распространения информации при управлении по входу SYN, t_p	40

Электрические параметры БИС К1815ИР1

Напряжение питания, U_{cc}, В	$+5 \pm 10$ %
Ток потребления I_{cc}, мА, не более	150
Входное напряжение высокого уровня, U_{IH}, В, не менее	2
Входное напряжение низкого уровня, U_{IL}, В, не более	0,8
Выходное напряжение высокого уровня, U_{OH}, В, не менее	2,4
Выходное напряжение низкого уровня, U_{OL}, не более	0,5
Входной ток низкого уровня, I_{IL}, мА, не более	—0,2
Входной ток высокого уровня, I_{IH}, мА, не более	40

Конструктивно БИС К1815ИР1 выполнена в металлокерамическом 28-выводном корпусе типа 4119.28-1 с планарным расположением выводов.

Статья поступила 30 августа 1985 г.

УДК 621.3.049.77

Ю. П. Попов, А. И. Милованов, В. И. Медведев, Л. В. Васильев

ПРЕОБРАЗОВАТЕЛЬ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНЫХ КОДОВ К1815ПР1

В системах цифровой обработки сигналов часто возникает необходимость в быстром преобразовании прямых кодов чисел в дополнительные и наоборот. Микросхема К1815ПР1 представляет собой преобразователь последовательно-параллельных кодов (ППК) конвейерного типа. Выполнена на основе маломощной транзисторно-транзисторной логики с диодами Шотки (ТТЛШ) по базовой планарно-эпитаксиальной технологии с использованием тонких эпитаксиальных пленок, окисной изоляции и двухуровневой металлизации. Содержит около 3000 элементов на кристалле. Выпускается в 16-выводном корпусе (рис. 1). Назначение выводов показано в табл. 1. Основные электрические и динамические параметры БИС К1815ПР1 в диапазоне температур —10...+85 °С приведены в табл. 2.

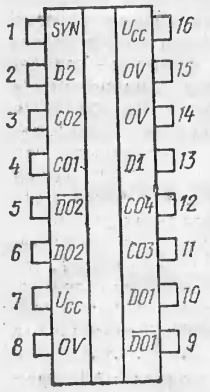


Рис. 1. Условное графическое обозначение микросхемы К1815ПР1

Микросхема предназначена для обработки n-разрядных (n=8, 16, 32) последовательно-параллельных либо n/2-разрядных последовательных двоичных кодов и выполняет следующие операции.

Для последовательно-параллельных и последовательных кодов:

преобразование прямого кода в дополнительный и дополнительного в прямой;

изменение знака числа с последующим его преобразованием в дополнительный или прямой код либо без преобразования;

транзитная передача числа с задержкой на 6, 10, 18 тактов синхросигнала.

Для последовательных кодов:

преобразование прямого кода в обратный и обратного в прямой;

инвертирование кода.

Коды поступают младшими разрядами вперед, последний разряд — знак числа. Входные данные представлены в отрицательной, а выходные — в отрицательной и положительной логике.

Микросхема К1815ПР1 имеет (рис. 2):

входную 2-разрядную (D1, D2) и выходную 4-разрядную (D01, D02, $\overline{D01}$, $\overline{D02}$) магистрали данных;

вход признака сопровождения знака C01;

вход управления инвертированием знака C02;

два входа управления разрядностью преобразуемых чисел C03, C04;

вход синхронизации SYN;

два последовательных 16-разрядных сдвиговых регистра RG1, RG2;

блок преобразования знака SID;

два мультиплексора MUX1, MUX2;

преобразователь прямого кода в дополнительный X/Y;

блок управления СОД.

Вход C02 управляет инвертированием знака числа. При C02-0 (низкий уровень) знак преобразуемого числа не меняется, при C02-1 (высокий уровень) знак числа в блоке преобразования знака изменяется на противоположный.

Входы C03, C04 служат для управления разрядностью преобразуемых чисел. Если C03-0, C04-0, то ППК осуществляет обработку 8-разрядных чисел, при C03-1, C04-0 — 16-разрядных, C03-0, C04-1 — 32-разрядных.

Таблица 1

Назначение выводов МС К1815ПР1

Вывод	Назначение
1	Вход синхросигнала SYN
2	Вход информационный четных разрядов кода D2
3	Вход управления инвертированием знака C02
4	Вход признака сопровождения знака C01
5	Выход инверсный четных разрядов результата $\overline{D02}$
6	Выход четных разрядов результата D02
7	Напряжение питания U_{cc}
8	Общий 0V
9	Выход инверсный нечетных разрядов результата $\overline{D01}$
10	Выход нечетных разрядов результата D01
11	Вход задания режима работы по разрядности C03
12	Вход задания режима работы по разрядности C04
13	Вход информационный нечетных разрядов кода D1
14	Общий 0V
15	»
16	Напряжение питания U_{cc}

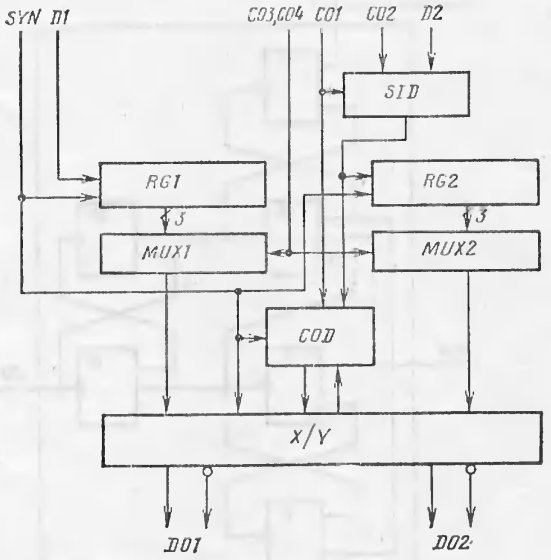


Рис. 2. Структурная схема БИС К1815ПР1

В режиме преобразования последовательно-параллельных кодов на вход D1 подаются нечетные разряды числа, начиная с первого, а на входы D2 — четные, начиная со второго и кончая знаком числа. На выходы $\overline{D01}$, $\overline{D02}$ выдаются соответственно нечетные и четные разряды результата в отрицательной, на выходы D01, D02 — в положительной логике. Разряды преобразуемых последовательных чисел поступают на вход D2, а результат преобразования выдается на выход $\overline{D02}$ в отрицательной, на выход D02 — в положительной логике. Одновременно со знаком числа на вход C02 поступает сигнал признака сопровождения знака «Лог. 1». Если на входе C02 постоянно будет присутствовать уровень сигнала, соответствующий «Лог. 1», то информация пройдет через ППЛК без изменения.

В режиме преобразования последовательных n/2-разрядных чисел достаточно на вход D1 подать постоянный высокий уровень сигнала, соответствующий «Лог. 0» в отрицательной логике. Если на вход D1 подать низкий уровень, соответствующий «Лог. 1», то будет осуществляться преобразование последовательных чисел из прямого кода в обратный и из обратного в прямой. При C01-1, C02-1 информация, поступающая на вход D2, будет инвертироваться.

В связи с необходимостью обеспечения высокого темпа выдачи результата на выходную магистраль структурная схема ППЛК реализована по конвейерному методу обработки информации. Способ организации внутреннего управления ППЛК — синхронный. Работа микросхемы синхронизируется по входу SYN внешним управляющим сигналом с периодом следования тактовых импульсов 120 нс. Рабочим является передний фронт синхросигнала. Упрощенная временная диаграмма работы микросхемы приведена на рис. 3. Временные соотношения показаны в табл. 2.

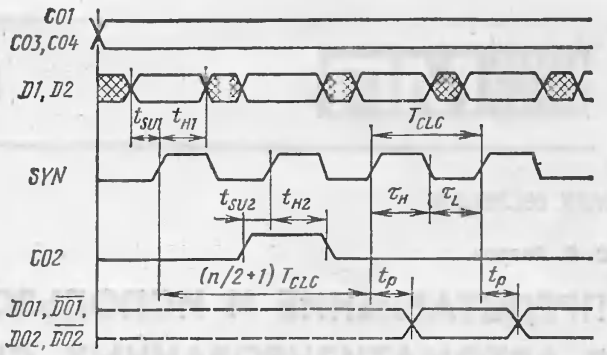


Рис. 3. Временная диаграмма работы БИС К1815ПР1

Таблица 2

Электрические и динамические параметры микросхемы К1815ПР1

Параметр	Значение параметра	
	мин.	макс.
Напряжение источника питания U_{CC}, В	5,5	4,5
Ток потребления I_{CC}, мА	100	—
Входное напряжение низкого уровня U_{IL}, В	0,8	—
Входное напряжение высокого уровня U_{IH}, В	—	2
Входной ток низкого уровня I_{IL}, мА	—0,2	
Входной ток высокого уровня I_{IH}, мкА	40	
Выходное напряжение низкого уровня U_{OL}, В	0,5	
Выходное напряжение высокого уровня U_{OH}, В		2,4
Выходной ток высокого уровня I_{OH}, мА		—0,4
Выходной ток низкого уровня I_{OL}, мА		4
Время установки информации по информационным входам t_{SU1}, нс		15
Время установки информации по входу C02 t_{SU2}, нс		15
Время удержания информации по информационным входам t_{H1}, нс		20
Время удержания информации по входу C02 t_{H2}, нс		45
Время задержки выдачи результата t_p, нс	40	
Длительность высокого уровня синхросигнала τ_H, нс		45
Длительность низкого уровня синхросигнала τ_L, нс		45
Период следования синхроимпульсов T_{CLC}, нс		120

Базовый элемент микросхемы К1815ПР1 — элемент ТТЛШ, выполняющий операцию И-ИЛИ-НЕ и обеспечивающий быстродействие не хуже 4 нс при рассеиваемой мощности не более 1 мВт. Базовый элемент памяти микросхемы — R-S-триггер, тактируемый фронтом синхросигнала и выполненный по схеме M-S. Использование современной элементной базы позволило при типовой мощности рассеяния 500 мВт получить время задержки выдачи информации на выходную магистраль не более 20 нс. По энергетическим параметрам преобразователь К1815ПР1 полностью совместим с микросхемами К133, К155, К533, К584, К589 и обладает высоким коэффициентом объединения по входу, что обеспечивается применением во входных каскадах быстродействующих транзисторов p-n-p типа. Для обеспечения высокой крутизны фронтов выходных сигналов и высокой нагрузочной способности выходы микросхемы выполнены по схеме с «активной» нагрузкой.

Статья поступила 30 августа 1985 г.

КРАТКИЕ СООБЩЕНИЯ

УДК 681.3-181.4

Кучинский А. Г., Кукель И. Н., Якушев А. К. **Устройство сопряжения микроЭВМ «Электроника 60» с контроллером крейта КАМАК УВК СМ-3 (СМ-4).**

Устройство сопряжения обеспечивает программное управление крейтом или ветвью КАМАК с помощью микроЭВМ «Электроника 60» посредством серийно выпускаемого контроллера крейта. Указанный контроллер имеет входной интерфейс «Общая шина» и предназначен для сопряжения мини-ЭВМ СМ-3 (СМ-4) с аппаратурой в стандарте КАМАК. Таким образом, предлагаемое устройство позволяет существенно упростить процедуру изменения пространственной конфигурации многопроцессорных систем автоматизации, а также обеспечивает применение микроЭВМ «Электроника 60» для работы в программном режиме с периферийным оборудованием, имеющим интерфейс «Общая шина». Устройство выполнено в виде однократного ТЭЗа микроЭВМ «Электроника 60» и питается от источника постоянного тока напряжением 5±0,25 В.

За справками обращаться по адресу: 220106, г. Минск, ул. Курчатова, 7, НИИ прикладных физических проблем им. А. Н. Севченко, лаборатория многоканальных измерительных систем.

УДК 681.327.8.06

С. С. Лавров

ПРЕДСТАВЛЕНИЕ И ИСПОЛЬЗОВАНИЕ ЗНАНИЙ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

Автоматизированные системы решения задач резко различаются по масштабам и сложности исследуемого объекта, по требованиям к быстродействию (от долей секунды до нескольких месяцев), по степени ответственности принимаемого решения, зависящей от ресурсов, вовлекаемых в исполнение решения, и от возможности корректировки этого решения, наконец, по структуре и принципам работы самой системы. Такое разнообразие очень осложняет задачу — дать общую характеристику как самих систем, так и проблем их создания, в частности, проблемы представления и применения знаний в подобных системах. Однако ясно, что эта проблема занимает одно из центральных мест в общей проблеме создания автоматизированных систем, поскольку эффективность любой подобной системы и ее соответствие целям, поставленным при ее разработке, решающим образом зависят от того, насколько полно и точно в системе отражен опыт, накопленный человеком при решении таких задач, которые призвана решать система.

1. ВИДЫ ЗНАНИЙ

Сами знания, которыми располагает и на которые опирается человек, решая ту или иную задачу, весьма разнородны.

Это прежде всего *понятийные* знания — тот набор понятий, которым пользуется человек, работающий в некоторой области, а также свойства и взаимосвязи этих понятий. Эта категория знаний вырабатывается преимущественно в сфере фундаментальных наук, а также в теоретических разделах многих прикладных и инженерных наук.

Близко к понятийным знаниям стоят *конструктивные* знания — знания о возможной структуре и взаимодействии частей различных объектов, с которыми сталкивается человек в той или иной сфере деятельности. Знания этой категории составляют основное содержание большинства технических, прикладных наук. Они формулируются в терминах понятий, сформировавшихся на предыдущем, понятийном уровне и служат постоянным источником пополнения знаний этого понятийного уровня.

Если в качестве примера взять программирование, то понятийное знание — это знание о структуре выражений, операторов, данных и описаний языка программирования или даже совокупности всех таких языков, а конструктивное знание — это знание об устройстве конкретных программ, о типичных алгоритмах.

Далее можно выделить *процедурные* знания — методы, алгоритмы и программы решения различных задач, с которыми человек уже сталкивался раньше и научился их решать. В производственной сфере аналогом процедурных знаний являются технологические знания о способах организации и осуществления разнообразных производственных процессов. В научно-технической практике роль и объем процедурных знаний особенно возросли, а их структура усложнилась с появлением электронных вычислительных машин. Эти знания стали использоваться в наиболее естественной для них алгоритмической форме.

Наконец, есть категория знаний, которые можно назвать *фактографическими*, — количественные и качественные характеристики конкретных объектов, явлений и их элементов. Традиционно эти знания накапливались в виде разнообразных таблиц и справочников, а с появлением и внедрением ЭВМ приобрели форму информационных массивов (файлов) и баз данных.

Можно было бы назвать и некоторые другие виды знаний, в частности, знания о человеке как необходимом элементе многих систем, о функционировании общественных, социальных систем различного уровня. Но эти знания, если они достаточно надежны и точны, могут быть отнесены к одной из названных выше категорий.

Различение этих категорий знаний позволяет лучше понять и полнее охарактеризовать особенности многих систем — их структуру, поведение, характер использования и т. д.

Все названные виды знаний обычно оказываются тесно взаимосвязанными, без чего их использование было бы невозможным. Однако четкое представление о том, на каком виде знаний преимущественно основываются те или иные действия, помогает лучше их спланировать, организовать и осуществить.

2. ЗАДАЧИ И ИХ РЕШЕНИЯ

Всякую человеческую деятельность, а с момента появления автоматизированных и полностью автоматических систем — и функционирование подобных систем, можно охарактеризовать как решение задач.

Решение задач проходит через ряд этапов, результаты многих из этих этапов могут рассматриваться как частичное решение задачи. Нередко промежуточные шаги оказываются важнее завершающих, так как позволяют по-новому взглянуть на задачу и на весь класс подобных задач, изменить постановку задачи или вообще отказаться от ее решения.

Решение задачи всегда начинается на понятийном уровне, когда выясняется и анализируется обстановка, в которой возникает задача, ставится цель, для достижения которой необходимо решить задачу, формулируются критерии достижения этой цели, анализируются последствия (положительные и отрицательные), к которым может повести достижение этой цели. Ошибки на этом этапе особенно опасны, они приводят к тому, что цель вообще не достигается, ресурсы на решение задачи расходуются впустую, а в худшем случае последствия оказываются совсем нежелательными и даже катастрофическими.

Обычно этот этап решения выполняется человеком, но для повышения его надежности в автоматизированные системы решения задач следует по возможности закладывать средства анализа целей и последствий любых планируемых действий.

На следующем этапе уточняется общая структура того объекта, который должен быть создан или исследован в результате решения задачи, определяются его самые общие характеристики и возможные варианты, намечаются средства решения зада-

чи. Этот этап базируется на знаниях, охарактеризованных выше как конструктивные. Просчеты на этом этапе могут привести к потере некоторых возможных путей решения задачи (среди которых при грубых ошибках могут оказаться и самые эффективные и даже единственно осуществимые при доступных ресурсах). Этап также выполняется в основном человеком, но если объект проектирования или исследования достаточно типичен, то знания, заложенные в систему, могут оказаться весьма полезными, поскольку система не имеет тенденции к выбору субъективных, односторонних решений, часто свойственных человеку (разумеется, если она хорошо построена и наполнена действительно объективными и всесторонними знаниями).

Все последующие этапы решения задачи, а точнее — этапы, еще не включающие в себя выполнение каких-либо физических действий, создание или уничтожение материальных объектов (рассматривается класс задач, где все этапы имеют именно такой характер), можно рассматривать как уточнение характеристик исследуемого объекта, т. е. как пополнение фактографического знания. Эти этапы в значительно большей степени, чем предыдущие, поддаются автоматизации. Это обеспечивается широким использованием ранее накопленных процедурных знаний (библиотек или пакетов алгоритмов и программ), а также математических методов, позволяющих расширить и реорганизовать запас процедурных знаний. На этих этапах также сравниваются и анализируются возможные варианты решения и из них выбирается вариант (или — на промежуточных этапах — несколько вариантов), удовлетворяющий намеченным критериям, а если среди критериев выбран главный, то оптимизирующий значение этого критерия при соблюдении всех остальных.

Каждый из упомянутых выше этапов можно рассматривать как самостоятельную задачу, подчиненную главной. Более того, на каждом из них, в особенности на последних, возникают и другие промежуточные или вспомогательные задачи (почему об этих этапах и говорится во множественном числе). Хорошая система решения задач должна автоматизировать, насколько это возможно, процесс порождения иерархии связанных задач, обращаясь к человеку лишь в принципиальных или критических ситуациях. Разумеется, понятие и признаки принципиальной или критической ситуации относятся к сфере понятийного знания, они должны быть сформированы и заложены в систему (в ту совокупность знаний, которой она оперирует) на самом первом этапе.

3. КЛАССИФИКАЦИЯ СИСТЕМ ОБРАБОТКИ ЗНАНИЙ

3.1. Классификация по степени использования различных видов знаний

Одним из важнейших критериев классификации систем представления, хранения, обработки и использования знаний (в дальнейшем для краткости будем говорить лишь об обработке знаний) является степень использования различных категорий знаний и их место в таких системах. В докомпьютерную эпоху были созданы весьма мощные средства представления знаний — естественные и искусственные языки и другие знаковые, в частности табличные и графические, системы. На этой основе появились средства и системы хранения знаний всех категорий — книги, журналы, другие виды литературы и формы документации, картотеки, дела, архивы, библиотеки и т. п. Обработка и использование знаний были монополией человека, за исключением простейших процессов механизации вычислений (отдельных операций обработки фактографического знания).

С появлением электронных вычислительных машин положение в корне изменилось. Принципиальная возможность использования ЭВМ для обработки знаний любой категории следует из того, что любая задача (по крайней мере, такая, решение которой один человек может поручить другому или другим) может быть сформулирована словесно, т. е. в текстовой форме, любые знания также выражаемы в этой форме, а на ЭВМ можно выполнять весьма сложные процессы анализа и обработки текстов. Разумеется, эта возможность имеет лишь принципиальный характер — далеко не каждая словесно сформулированная и даже формально специфицированная задача может быть фактически решена машиной, она может оказаться непосильной человеку и даже большим коллективам людей.

Единая форма представления программ и данных позволила в первую очередь сделать доступным машине процедурное знание наряду с фактографическим. Трансляторы с алгоритмических языков, библиотеки и пакеты прикладных программ с их системным наполнением сделали процедурное знание объектом не только использования, но и обработки.

Элементы представления понятийного знания в очень примитивной форме — «содержательные» обозначения, имена, идентификаторы — начали проникать в сферу общения человека с ЭВМ очень давно. Дальнейшим этапом в представлении и использовании понятийного знания стали системы управления базами данных. Схемы и подсхемы баз данных — это типичные примеры представления понятийного знания, хотя и в ограниченной форме и с довольно скромными целями.

Параллельно и независимо развивались значительно более мощные (по замыслу) средства обработки понятийного знания — системы *искусственного интеллекта* [5, 6, 13, 25], в частности решатели задач, программы автоматического доказательства теорем, вопросно-ответные системы. Создавались, и развивались языки, ориентированные на построение подобных систем (а косвенно — на обработку понятийного знания): Лисп [15, 17], Рефал [4], Пролог [33]. Последний ориентирован на представление и обработку понятийного знания в наиболее чистом виде, что, впрочем, не делает его самым удобным для использования, потому что на практике использование понятийного знания всегда должно сопровождаться привлечением фактографического и процедурного знания, плохо выразимого на этом языке.

3.2. Классификация по форме представления знаний

Следующий признак классификации систем обработки знаний — это форма представления знаний в системе. Из сказанного выше следует, что разные категории знаний стимулировали разные формы своего представления: фактографическое знание — представление в форме *баз данных* [7, 16, 29], процедурное знание — в форме *алгоритмических языков* [14], *библиотек и пакетов программных модулей*, понятийное знание — в форме упомянутых выше *языков искусственного интеллекта* (Лисп и др.) и *логического программирования* (Пролог).

Когда же была осознана (или прочувствована) необходимость совместного представления и использования всех категорий знания, начали возникать новые языки и средства, мало похожие на все существовавшее ранее [25]. Разработчикам подобных средств нужно было найти единый аппарат для работы со всеми видами знаний. Так появились концепции фреймов, семантических сетей, систем продукций, абстрактных типов данных. Под влиянием этих концепций развивались и создавались вновь универсальные языки программирования: Алгол-68 [23], Эль-76 [22], Ада [32]. Правда, это развитие было медленным, перепительным и противоречивым. Половинчатость и эклектичность многих решений ставит, в частности, под серьезное сомнение перспективы использования языка Ада несмотря на оказываемую ему мощную поддержку (Пентагон и другие ведомства и учреждения).

Сходным образом эволюционировали и языки искусственного интеллекта. Возникший на базе Лиспа язык Planner (Плэнер [24]) помимо всех

лисповских возможностей содержит такие, естественные для языков подобного класса, средства, как аппарат поиска по образцу, механизм возвратов, средства организации базы данных (хотя и примитивной) и метод косвенного обращения к процедурам по результату сопоставления двух образцов. С другой стороны, необходимость представления процедурного знания во всех его формах вызвала проникновение в эти языки чуждых им по духу (первоначальному) понятий и операций, например операций вещественной арифметики, операторов присваивания и перехода и т. п.

По поводу специализированных средств представления знаний можно сказать следующее.

Идея *фрейма* [18, 26] с большим количеством позиций (или слотов) как средства всеобъемлющего представления понятия со всеми его возможными связями уже себя изжила. Каждый объект может, в принципе, иметь такое количество связей с другими объектами, что фрейм для описания типа этого объекта, его возможных ролевых функций становится недопустимо громоздким. У каждого конкретного объекта обычно реализуется лишь малая доля его потенциальных связей и ролей.

Все большее значение приобретают описания не типов объектов, а типов связей (отношений [6, 10, 12]) между ними в качестве первичных понятий. Количество свободных позиций (мест) у таких отношений фиксировано и невелико. Во многих системах используются лишь двух-, трехместные отношения. Широкое распространение (не только в базах данных) получили в последнее время *реляционные языки* [9—12].

Идея *семантической сети* [21, 25, 30] как средства представления знаний продолжает существовать и развиваться. Но из статических или квазистатических (медленно и локально изменяющихся) объектов сети постепенно превратились в весьма динамичные объекты, быстро реагирующие на изменение обстановки, которую они призваны отображать.

Далеко продвинутую идею подобной сети развивает и реализует Г. С. Цейтин [21, 30]. В его системе представления знаний важную роль, в частности, играет операция приписывания класса объекту (узлу сети). Класс — это шаблон связей объекта с другими объектами, соответствующий лишь одной ролевой функции этого объекта. Раз начавшись, эта операция инициирует цепную реакцию приписывания классов объектам, с которыми исходный объект прямо или косвенно связан в сети. Это может сопровождаться порождением новых объектов и отождествлением уже существующих. Этим и обеспечивается высокий динамизм знаний, представленных сетью, что дает Г. С. Цейтину основание считать сеть оперативной, а не долговременной памятью системы. В терминальных узлах сети размещаются конкретные значения, которыми могут быть, в частности, константы или процедуры. Благодаря этому в сети естественно может быть представлено фактографическое и процедурное знание.

Идея *систем продукций* [9, 20, 31] нашла чрезвычайно обширное применение в *экспертных системах* [12, 25, 31]. Типичное назначение такой системы — высказать суждение по поводу предъявленного ей объекта или дать рекомендацию по способу действий в предъявленной ей ситуации. Подобный запрос определяет как начальное состояние системы, так и требования к конечному состоянию, в которое она должна прийти (конечную цель). *Продукция* или *правило* имеет структуру

условие → действие.

Условие любого правила может быть проверено в любом состоянии системы. Среди правил, условия которых удовлетворяются, отбирается одно (случайным образом или в результате неявного использования других дополнительных условий) и выполняется соответствующее действие, переводящее систему в следующее состояние. В этом состоянии, если цель еще не достигнута, все повторяется снова. В конечном состоянии, а возможно и в промежуточных, система информирует пользователя о результатах своей работы.

Состояние характеризуется прежде всего набором значений различных внутренних переменных системы (таким образом, действие меняет некоторые из этих значений). Но в зависимости от уровня и степени сложности системы в состояние может входить (учитываться в нем) еще ряд компонент: набор промежуточных целей, оценка близости состояния к требуемому (конечному), суждение пользователя о текущем состоянии и о прогрессе в работе системы, самооценка системы, сам набор правил (который система оказывается способной изменять по ходу работы), а в самом общем случае — весь багаж знаний системы, включая и понятийные (что позволяет системе образовывать новые понятия и связи между ними). Развитые экспертные системы обладают подсистемой объяснения, благодаря чему они не только отвечают на запрос пользователя, но и мотивируют свой ответ. В составе системы может присутствовать и подсистема доверия, которая по особому запросу пользователя сообщает ему, что она знает о том или ином предмете (объекте или абстрактном понятии), демонстрируя тем самым свою квалификацию.

Абстрактные типы [1, 19, 26] как средство представления знаний используются следующим образом. Описание абстрактного типа говорит пользователю, что каждый объект этого типа состоит из определенного набора компонент с определенными именами. Компонентами могут быть константы, переменные, значения других типов, процедуры и функции, ссылки на другие объекты и т. п. К любой из этих компонент пользователь может обратиться в соответствии с ее природой — взять значение константы, изменить значение переменной, вызвать процедуру, получить доступ к другому объекту и т. д. Кроме этих явных компонент объект абстрактного типа может содержать некоторое число других (скрытых от пользователя) компонент, обеспечивающих определенное поведение такого объекта, например работу включенных в него процедур (пользователь видит только заголовки, но не тела доступных ему процедур).

Современная тенденция заключается в том, чтобы поведение таких объектов было возможно более активным. Например, изменение значения одной компоненты может вызвать (в соответствии со связями, известными или неизвестными пользователю) изменение значений других компонент, запуск некоторых процедур и даже аналогичные процессы в других объектах, связанных с данным. Подобная идея была высказана почти одновременно А. С. Нариньяни [19, 20], А. М. Степановым [26, 27], сходным образом ведут себя объекты в уже упоминавшихся сетях Г. С. Цейтина.

Разумеется, не всякое поведение и соответственно не всякая попытка пользователя воздействовать на объект допустимы. Простейший пример — если для переменной указан (или неявно определен) диапазон допустимых значений, то попытка присвоить ей значение, выходящее за этот диапазон, вызывает немедленную реакцию системы с отказом выполнить такое действие и с объяснением причин отказа. Если незаконность действий пользователя выявляется на более глубоком уровне, то и цепочка реакций оказывается более длинной, а реакция — более замедленной.

3.3. Классификация по виду решения задачи

Следующий критерий классификации систем обработки знаний — это вид результата работы системы, иными словами, вид решения задачи. Многие из систем и классов систем, перечисленных выше, ориентированы на получение результата в фактографической форме, т. е. на получение решения одного варианта задачи. Получить следующий вариант можно, только решая задачу заново с другими исходными данными (речь идет о вариантах задачи, а не о вариантах решения при одних и тех же исходных данных, например о вариантах квартирного обмена, когда достаточно лишь продолжить поиск возможных решений).

Если требуется решить несколько вариантов задачи, то, как правило, выгоднее получить (синтезировать) решение задачи в процедурной форме — на языке, достаточно близком к машинному, а затем многократно запускать эту программу с разными исходными данными. Если программу предполагается использовать в сочетании с другими программами, написанными вручную или также синтезированными, то более приемлемой (но также процедурной) формой решения задачи будет программа на языке более высокого уровня. Подобные системы естественно называть системами с *автоматическим синтезом программ* [13, 25, 28].

Наконец, возможны случаи, когда результат решения задачи целесообразно иметь на понятийном уровне. При этом строится не столько программа, сколько схема решения задачи, непосредственно не выполнимая на ЭВМ,— требуемая последовательность действий описывается в терминах наиболее общих, абстрактных понятий. Такую схему будем называть *абстрактной программой,* хотя термины «абстрактная схема» или «абстрактный алгоритм» были бы, вероятно, точнее.

Проводя аналогию между системами обработки знаний (включая понятийное) и традиционными системами программирования, можно сказать, что экспертные, вопросно-ответные системы и им подобные работают по принципу интерпретации описания задачи, тогда как системы синтеза программ — по принципу компиляции. Известно, что интерпретаторы языков программирования писать, как правило, легче, чем компиляторы, но компилированные программы работают обычно значительно быстрее интерпретируемых. Это наблюдение, по-видимому, останется в силе и для систем обработки знаний.

Первая из известных нам систем с автоматическим синтезом программ— это система ПРИЗ, созданная Э. Х. Тыугу и его сотрудниками [8, 28]. Эта работа, ведущаяся с начала 60-х гг., была и остается пионерской. На наш взгляд, она будет определять одно из важнейших направлений не только в отечественной, но и в мировой информатике на протяжении еще многих лет.

3.4. Классификация по степени универсальности

Довольно существенным критерием оценки системы обработки знаний является степень ее привязки к определенной предметной области. В информатике большинство новых идей возникали и первоначально воплощались в программах решения конкретных задач, трансляторах с определенного языка, операционной системе конкретной машины и т. п. И только позднее, когда более или менее общий характер найденных решений был осознан и признан, эти решения реализовались в виде более универсальных систем или подходов. Так происходило и в области обработки знаний. Однако этот процесс был далеко не равномерным.

Так, идея универсальной системы представления и использования фактографических знаний — системы управления базами данных [7, 16, 29] — возникла довольно рано. Эти системы быстро завоевали признание и получили достаточно широкое распространение. Возникли разные классы таких систем: иерархические, сетевые, реляционные. Нередко системы высокого уровня строятся на базе более элементарных систем, например реляционная — на базе иерархической и т. п.

В области организации процедурных знаний идея библиотеки подпрограмм была едва ли не первой идеей в системном программировании. Однако такие библиотеки создавались индивидуально для каждого типа машин, что, конечно, связано с индивидуальностью самих подпрограмм, не переносимых с машины на машину. Зато идея универсального языка программирования очень быстро завоевала всеобщее признание.

К той же области относится идея пакета прикладных программ. Однако, возможно, под влиянием традиций, идущих от библиотек подпрограмм, или вследствие того, что на разработку пакета смотрели как на обычную задачу (отсюда эпитет «прикладные»), эти пакеты создавались более или менее независимо друг от друга. Каждый из них содержит две части, традиционно называемые функциональным и системным наполнением пакета. Универсальные системы управления пакетами получили (до признания значимости работ Э. Х. Тыугу) значительно менее широкое распространение, чем системы управления базами данных.

В области организации понятийных знаний прогресс был еще более медленным. Хотя элементы представления знаний этой категории появлялись в различных системах программирования уже давно (об этом было сказано выше), до сих пор господствует убеждение, что описание обстановки, в которой возникает задача, описание понятийной среды не может или не должно быть формализовано, и поэтому спецификации задач не могут быть формальными. Осознанно или неосознанно возникли и употребляются два термина: спецификация задачи и спецификация программы, которые часто смешиваются [2]. Но если и учитывается их различие (в спецификации задачи характеризуется только сама задача, без указания на способ ее решения, а в основе спецификации программы лежит конкретный метод, алгоритм или недетерминированный процесс), то возможность формализации обычно признается лишь за спецификацией программы.

Тем не менее идея возможности представления и обработки понятийного знания, идея (или даже мечта) о том, что когда-нибудь машине достаточно будет сказать, какую задачу надо решить, не уточняя, как это можно сделать, и машина найдет решение сама, жила в программистской среде и пути реализации такого подхода постепенно нащупывались. И здесь успех пришел в первую очередь там, где эту идею попытались реализовать не во всей общности, а применительно к конкретной предметной области,— именно так организованы в большинстве своем экспертные системы. Универсальные системы, которые можно было бы сделать экспертными в любой желаемой области, распространения пока не получили. Причина этого, по-видимому, в том, что наполнение системы знаниями о конкретной предметной области остается очень трудной задачей, за которую не хотят браться ни создатели универсальной системы, ни специалисты какой-либо области, особенно если в этой области уже есть готовые пакеты программных модулей и базы данных. Возникла даже проблема и соответствующая ей сфера деятельности — извлечение знаний из специалистов. На наш взгляд проблема эта в большой степени надумана — если специалисту дать в руки хороший, удобный и эффективный аппарат представления понятийных знаний, то он сам начнет им охотно пользоваться. Другая возможная причина заключается в том, что не только сами понятийные знания, но и способы, приемы работы с ними, методы их использования зависят от предметной области. Недаром прогресс любой отрасли науки, а в особенности техники, связан не только с открытием нового явления, но и с освоением конкретных способов использования этого явления в интересах практики.

Поэтому к универсальной системе обработки знаний необходимо предъявить требование, чтобы она была способна иметь дело не только с первичным знанием о предметной области, но и с *метазнанием,* т. е. со знанием, как следует обрабатывать и использовать первичное знание.

4. ОБЩАЯ ХАРАКТЕРИСТИКА СИСТЕМЫ ОБРАБОТКИ ЗНАНИЙ СПОРА

Автоматизированная система решения задач на основе баз знаний (система СПОРА [3, 13]) задумана как универсальная система, ориентированная на представление и использование всех категорий знаний, причем в форме, наиболее соответствующей каждой категории, и на синтез программ как на необходимый этап решения задачи.

Универсальность системы проявляется в том, что для ее использования в конкретной предметной области достаточно предварительно построить абстрактную понятийную модель этой предметной области, отразив в ней всю понятийную среду, необходимую для постановки конкретных задач. В системе, в нынешнем ее виде, не предусмотрены специальные средства или язык для описания объекта исследования. Для этого предлагается использовать те же средства, что и для описания моделей. Разумеется, описание объекта исследования является более частной задачей, чем описание понятийной модели, но реляционный язык описания одинаково хорошо работает в обоих случаях. Кроме того, в системе предусмотрена возможность ссылок из одной модели на другую. Эта возможность важна, когда создается модель предметной области (например, механики), опирающейся на понятия другой предметной области (допустим, геометрии или теории дифференциальных уравнений). Точно так же описание объекта исследования может рассматриваться как подмодель или подчиненная модель по отношению к модели той предметной области, которой принадлежит этот объект.

Абстракция от реализационных аспектов (от конкретных форм представления процедурного и фактографического знания) позволяет получить решение задачи на чисто понятийном уровне — в виде уже упоминавшейся абстрактной программы. Этот этап представляется достаточно важным, так как абстрактная программа может стать объектом самостоятельного анализа и оценки. В частности, специалист может решить, заслуживает ли найденный путь решения задачи того, чтобы воплотить его в программу на том или ином базовом языке программирования и заставить машину выполнить эту программу или следует продолжить поиски иного пути решения на понятийном уровне.

Система СПОРА содержит средства описания связи понятий, воплощенных в абстрактной модели предметной области, с компонентами *алгоритмической поддержки* этой модели — программными модулями, собранными в библиотеку или пакет, и ее *информационной поддержки* — данными и таблицами (отношениями), накопленными в базе данных. Это те формы, которые приняты в системе для представления процедурного и фактографического знания.

Универсальность системы заключается еще и в том, что она не предъявляет жестких требований к форме записи модулей, включаемых в библиотеки, и данных, помещаемых в базу данных. В этом же проявляется ориентация системы на представление знаний каждой категории в наиболее естественном для этой категории виде. Точнее, есть только два ограничения — языки программирования, на которых написаны модули, и язык запросов к базе данных должны быть базовыми языками системы, а сами модули и данные должны быть записаны в точном соответствии с правилами этих систем программирования и манипулирования данными. Второе требование вполне естественно, а первое не очень обременительно, так как подключение к системе новых базовых языков достигается сравнительно недорогой ценой.

Как уже было сказано, язык описания моделей построен на реляционной основе. Широко используется в нем и понятие абстрактного типа. Фактически вся понятийная модель предметной области — это система абстрактных типов и операций над ними, а ее алгоритмическая и информационная поддержка — это средство реализации этих типов и операций.

Продукционный подход также нашел в системе СПОРА свое отражение. В основе определения новых типов данных и функций лежит конструкция

кортеж<список компонент>
где<зависимости>**конец кортежа**

Здесь «зависимости» играют роль условия правила. Как только удалось найти или построить набор компонент, удовлетворяющий всем зависимостям, можно объявить этот набор значением кортежа, приписать ему соответствующий тип и, если требуется, имя. В этом и состоит выполняемое действие.

В существующем виде система не содержит средств для представления метазнаний. Не обладает система и высокой степенью динамизма модели предметной области и ее поддержки.

Спецификация конкретной задачи, возникающей по отношению к данному объекту исследования, может пониматься в широком и узком смысле. В узком понимании спецификация (запрос к модели) содержит только указание тех характеристик объекта исследования, которые считаются известными, и тех, которые требуется определить в результате решения задачи. Такая спецификация записывается на очень простом языке, которым пользователь, ставящий задачи, должен владеть активно. Но этой спецификации системе недостаточно для решения задачи. Достаточную (для первого этапа решения — получения абстрактной программы) информацию содержит спецификация в широком смысле, которая включает помимо запроса всю понятийную модель предметной области. Пользователь должен отчетливо это понимать и помнить, что обе части широкой спецификации — модель и запрос — должны быть полностью согласованы между собой. Прежде чем писать свой запрос, пользователь должен внимательно изучить описание модели предметной области и объекта исследования. Для этого он должен владеть (пусть пассивно) языком этого описания. Реляционный характер этого языка делает его достаточно простым и четким, хотя и требует определенной культуры. Эта культура может быть названа спецификационной [2] (в отличие от, скажем, математической или алгоритмической, хотя она и родственна им). Она в известной степени приобретается при изучении хорошей учебной и монографической литературы по данной предметной области (правда, только обладание такой культурой помогает отличить хорошую литературу от плохой), а также в процессе работы с некоторыми аналогичными формальными языками, например языками описания схем баз данных.

От разработчиков модели предметной области требуется активное владение языком описания моделей и соответственно гораздо более высокий уровень спецификационной культуры. Кроме того, они обязаны быть специалистами высокого класса в своей предметной области. Система понятий этой предметной области, которую они воплощают в модели, должна быть тщательно отобранной, достаточно полной и хорошо сбалансированной. Разумеется, она должна быть полностью обеспечена средствами алгоритмической и информационной поддержки.

Система СПОРА разрабатывалась с учетом опыта создания и использования системы ПРИЗ и восприняла многие концепции последней. Основные различия этих систем заключаются в следующем.

Систему СПОРА ее разработчики пытались сделать более модульной. Это проявилось, в частности, в расчленении входного языка системы на несколько более или менее независимых языков: языка описания предметной области (ее модели), языка описания внешности и реализации первичных (для описания связи модели со средствами ее алгоритмической и информационной поддержки), языка запросов.

Система ПРИЗ согласно описанию [8] базируется (т. е. способна синтезировать программы) лишь на одном языке — Ассемблере ЕС ЭВМ, тогда как в системе СПОРА предусматривается и обеспечивается использование нескольких базовых языков более высокого уровня (Паскаль, Фортран, Алгол и др.).

База данных рассматривается в системе ПРИЗ как одно из расширений входного языка системы. Так — в виде пакета прикладных программ, созданного средствами системы ПРИЗ,— она и была реализована. В системе СПОРА база данных с самого начала рассматривалась как органическая часть системы, без которой невозможно ее функционирование, а именно как средство информационной поддержки модели предмет-

ной области, носитель фактографического знания.

Однако несмотря на эти различия возможности обеих систем остаются очень близкими.

ЛИТЕРАТУРА

1. Агафонов В. Н. Типы и абстракция данных в языках программирования.— В кн.: Данные в языках программирования.— М.: Мир, 1982, с. 265—327.
2. Агафонов В. Н. Языки и средства спецификации программ.— В кн.: Требования и спецификации в разработке программ.— М.: Мир, 1984, с. 285—344.
3. Бабаев И. О., Новиков Ф. А., Петрушина Т. И. Язык Декарт — входной язык системы СПОРА.— В кн.: Прикладная информатика.— М.: Финансы и статистика, 1981, вып. 1, с. 35.
4. Базисный РЕФАЛ и его реализация на вычислительных машинах.— М.: ЦНИПИАСС, 1977.
5. Бухштаб Ю. А., Камынин С. С. Организация дедуктивной системы, отвечающей на вопросы.— Программирование, 1975, № 6, с. 83—90.
6. Бухштаб Ю. А., Камынин С. С., Любимский Э. З. Реализация немонотонных рассуждений.— Программирование, 1982, № 1, с. 84—88.
7. Дейт К. Введение в системы баз данных: Пер. с англ.— М.: Наука, 1980.
8. Кахро М. И. Калья А. П., Тыугу Э. Х. Инструментальная система программирования ЕС ЭВМ (ПРИЗ).— М.: Финансы и статистика, 1981.
9. Клещев А. С. Реляционная модель вычислений.— Программирование, 1980, № 4, с. 20—29.
10. Клещев А. С. Реляционный язык как программное средство для искусственного интеллекта.— Владивосток: ИАПУ ДВНЦ АН СССР (Препринт), 1980.
11. Клещев А. С. Реляционный язык программирования и принципы его реализации на последовательной ЭВМ.— Программирование, 1981, № 6, с. 45—53.
12. Клещев А. С. Представление знаний. Методология, формализмы, организация вычислений и программная поддержка.— В кн.: Прикладная информатика.— М.: Финансы и статистика, 1983, вып. 1, с. 49—93.
13. Лавров С. С. Использование вычислительной техники, программирование и искусственный интеллект (перспективы развития).— Микропроцессорные средства и системы, 1984, № 3, с. 3—9.
14. Лавров С. С. Основные понятия и конструкции языков программирования.— М.: Финансы и статистика, 1982.
15. Лавров С. С., Силагадзе Г. С. Автоматическая обработка данных. Язык Лисп и его реализация.— М.: Наука, 1978.
16. Мартин Дж. Организация баз данных в вычислительных системах. 2-е изд.: Пер. с англ.— М.: Мир, 1980.
17. Маурер У. Введение в программирование на языке Лисп: Пер. с англ.— М.: Мир, 1976.
18. Минский М. Фреймы для представления знаний.— М.: Энергия, 1979.
19. Нариньяни А. С. Недоопределенные модели и операции с недоопределенными значениями.— Новосибирск: ВЦ СОАН СССР (Препринт 400), 1982.
20. Нариньяни А. С. Система продукций как модульный программный комплекс.— В кн.: Прикладные и экспериментальные лингвистические процессоры.— Новосибирск: ВЦ СОАН СССР, 1982.
21. Округин М. Б. Реализация ассоциативной сети данных во внешней памяти.— В кн.: Прикладная информатика.— М.: Финансы и статистика, 1984, вып. 1(6).
22. Пентковский В. М. Автокод Эльбрус (Эль-76).— М.: Наука, 1982.
23. Пересмотренное сообщение об Алголе 68.— М.: Мир, 1979.
24. Пильщиков В. Н. Язык Плэнер.— М.: Наука, 1983.
25. Представление знаний в человеко-машинных и робототехнических системах. В 4-х т.— М.: ВЦ АН СССР, ВИНИТИ, 1984.
26. Степанов А. М. Фреймы и параллельные смешанные вычисления.— Новосибирск: ВЦ СОАН СССР (Препринт № 297), 1981.
27. Степанов А. М. Экспериментальная система программирования.— Новосибирск: ВЦ СОАН СССР (Препринт № 305), 1981.
28. Тыугу Э. Х. Концептуальное программирование.— М.: Наука, 1984.
29. Ульман Дж. Основы систем баз данных: Пер. с англ. М.: Финансы и статистика, 1983.
30. Цейтин Г. С. Программирование на ассоциативных сетях.— В кн.: ЭВМ в проектировании и производстве.— Л.: Машиностроение, 1985, вып. 2, с. 16—48.
31. Языки представления знаний и вопросы реализации экспертных систем.— Владивосток: ДВНЦ АН СССР, 1984.
32. Язык программирования Ада (предварительное описание): Пер. с англ.— М.: Финансы и статистика, 1981.
33. Clocksin W. F., Mellish C. S. Programming in Prolog.— Springer Verlag, 1981.

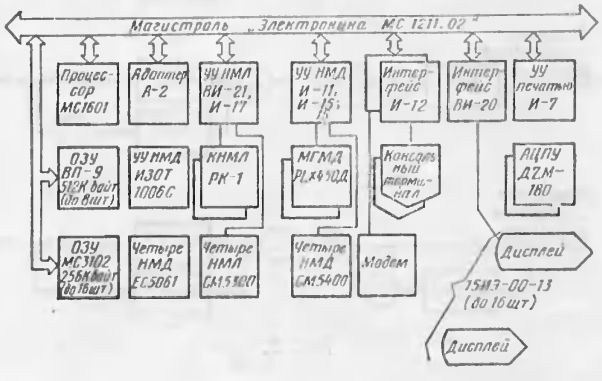
Статья поступила 18 февраля 1986 г.

КРАТКИЕ СООБЩЕНИЯ

УДК 681.326-681.3.06

Хацкевич Л. Д. **Программно-технический комплекс на базе микроЭВМ «Электроника МС 1211.02».**

Технической основой вычислительных средств, нашедших применение в отраслевом и территориальном управлении на региональном уровне, может служить микроЭВМ «Электроника МС 1211.02». На базе этой микроЭВМ был разработан программно-технический комплекс (ПТК), ставший основой сети ВЦКП г. Воронежа. ПТК состоит из технических средств, системного программного и технологического обеспечения и прикладных программ.



С использованием ПТК были созданы интеллектуальные абонентские пункты, на которых обрабатывается до 2/3 информации, циркулирующей в сети. Остальная информация обрабатывается на центральном вычислительном комплексе (на базе ЕС-1035), связанном с абонентскими пунктами каналами связи через буферную многомашинную систему. Схема проблемно-ориентированной комплектации ПТК обеспечивает функционирование автоматизированных систем обработки информации директивных органов, плановых и финансовых расчетов, государственной статистики и др. систем.

Основные проектные решения, разработанные при создании сети ВЦКП г. Воронежа, оформлены в виде типовых проектных решений и включены в состав типового рабочего проекта ВЦКП. Внедрение основных задач АСУ позволило получить экономический эффект в 2,9 млн. руб. со сроком окупаемости затрат 1,9 года.

За справками обращаться по адресу: 394088, г. Воронеж, ул. В. Невского, 9, кв. 90.

УДК 681.325.5 : 681.3.068

Г. П. Мозговой, С. С. Семенова, Е. И. Семин, О. В. Трещалин

МЕТА-АССЕМБЛЕР МЕАСС ДЛЯ МИКРОПРОЦЕССОРНЫХ СИСТЕМ С НАРАЩИВАЕМОЙ РАЗРЯДНОСТЬЮ

Разработан универсальный мета-микроассемблер МЕАСС, который можно использовать для любых секционированных микропроцессорных систем (МПС) независимо от их архитектуры, например для серий К582, К583, К584 (И²Л), К589, К1802, К1804 (ТТЛШ), К1800 (ЭСЛ) и К588 (К-МДП). Работа пользователя с мета-микроассемблером разбивается на два этапа: настройки и ассемблирования (рис. 1).

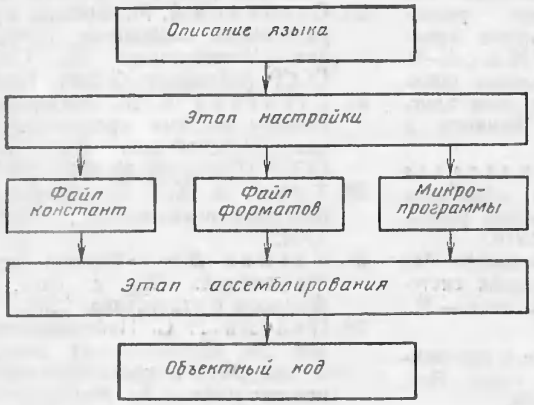


Рис. 1. Структура мета-микроассемблера

На этапе **настройки** создается своя версия символического языка, отвечающая особенностям конкретной МПС. При этом пользователем определяется длина микрокоманды (МК), имена и значения констант, имена и содержимое форматов полей МК. Программирование ведется на языке мета-микроассемблера.

На этапе **ассемблирования** транслируются микропрограммы, т. е. последовательности МК, записанных на созданном языке, в объектный код. В качестве языка микроассемблера использован язык AMDASM фирмы AMD [6, 7].

МЕАСС позволяет создавать языки микропрограммирования. Это обеспечивается следующими его особенностями:

— микрокоманда имеет произвольную, но одинаковую для всей микропрограммы длину, составляющую для предлагаемой версии транслятора от 1 до 128 разрядов (битов). Посредством перетрансляции МЕАСС можно создать версии с большим допустимым размером МК;

— этапы настройки (создание языка микропрограммирования) и ассемблирования микрокоманд на созданном языке в принципе независимы и осуществляются отдельными программными модулями с передачей информации через вспомогательный файл на магнитном диске. Однако на этапе ассемблирования можно доопределить язык с помощью основных языковых средств этапа настройки, за исключением длины МК, и переопределить уже заданные форматы, описывающие поля МК;

— микрокоманда на этапе настройки определяется как совокупность зависимых или независимых полей (форматов) с фиксированными размерами. Каждому полю можно поставить в соответствие набор мнемоник и соответствующих им численных значений разрядов поля. Кроме того, можно использовать переменные поля — поля постоянной длины, значения которым присваиваются на этапе ассемблирования (операндами или доопределением языка), и «безразличные» поля, состояние которых не влияет на работу системы при выполнении конкретной МК;

— константы записываются в явном виде или в виде выражения. В явном виде значение константы задается в одной из четырех систем счисления;

— наложение форматов на этапе ассемблирования позволяет определять все поля МК посредством их комбинирования;

— для облегчения процесса разработки и написания микропрограмм служат модификаторы, абсолютная и относительная адресации, директивы управления печатью и программным счетчиком.

Описание языка микроассемблера

Команды (операторы) языка делятся на шесть групп.

WORD — обязательный оператор на этапе настройки для определения длины МК (в пределах 1...128 для данной версии транслятора).

END — обязательный последний оператор в исходных файлах для обоих этапов.

Определяющие операторы (рис. 2).

EQU — оператор для определения констант. Имени константы присваивается определенное двоичное значение, записанное в явном виде или в виде выражения (рис. 3).

В явном виде значение константы задается в одной из четырех систем счисления, которая определяется соответствующим указателем (B # — двоичная, Q # — восьмеричная, D # — десятичная, H # — шестнадцатеричная). Для любого значения константы может быть задана явная длина. При отсутствии явной длины перед значением константы неявная длина определяется исхо-

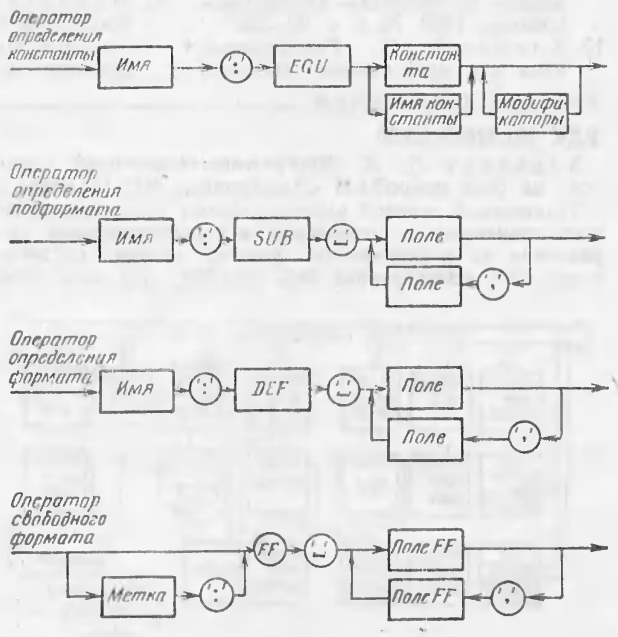


Рис. 2.

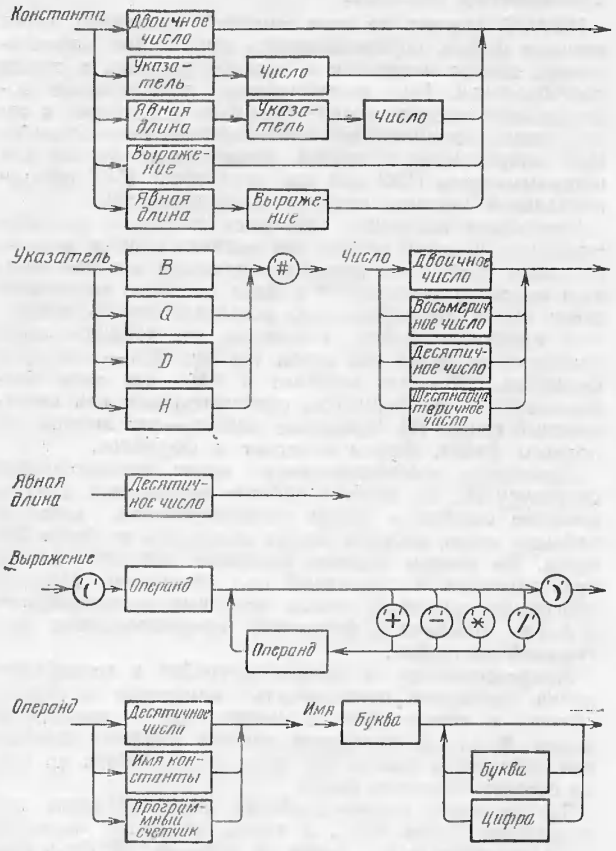


Рис. 3.

дя из заданной системы счисления и значения константы. Максимальное значение константы $2^{16}-1$.

В выражениях допускаются операции сложения, вычитания, умножения и деления. Операндами могут быть константы в явном виде, имена констант, текущее значение микропрограммного счетчика (для этапа ассемблирования). Выражение заключается в скобки.

Для облегчения задания констант служат модификаторы, предназначенные для изменения их значения. Модификаторы выполняют следующие операции: «$\times$» — инвертируют значение; «—» — дополняют значение до двойки; «:» — усекают значение слева, чтобы число оставшихся разрядов соответствовало явной длине этого значения; «%» — осуществляет установку значения в поле вправо, остающиеся слева биты принимаются нулевыми. К одному значению могут применяться несколько модификаторов.

Константы определяются на фазе настройки микроассемблера. Однако, если пользователь, перейдя от фазы настройки к фазе ассемблирования, обнаружит, что не определил какие-либо из используемых констант или хочет их изменить, он может это сделать на фазе ассемблирования, так как программа ассемблирования обрабатывает оператор EQU.

DEF — оператор для определения формата МК. В этом операторе описываются все поля МК, задается их длина и состояние. Поля могут быть числовыми, переменными, с «безразличным состоянием» и именем подформата.

Разделителем между полями является запятая. Для каждого поля задается явная длина. Максимальная длина поля, за исключением полей с «безразличным состоянием», равна 16.

В числовом поле задается константа в явном виде или выражение, или имя константы, определенной оператором EQU.

Поля с «безразличным состоянием» («X») указывают разряды МК, состояние которых не влияет на выполнение данной МК (рис. 4).

Рис. 4.

Определение поля как переменного позволяет присваивать ему значение на фазе ассемблирования. При этом пользователь может, описывая переменное поле на фазе настройки, задать для него значение по умолчанию, которое будет подставлено в данное переменное поле, если он не задаст другую подстановку. Язык микроассемблера позволяет распространить действие указателей системы счисления (B #, Q #, D #, H #) и модификаторов («x», «—», «:», «%») на все подстановки на фазе ассемблирования для данного переменного поля. В то же время допускается использовать модификаторы, относящиеся только к значению по умолчанию (рис. 5).

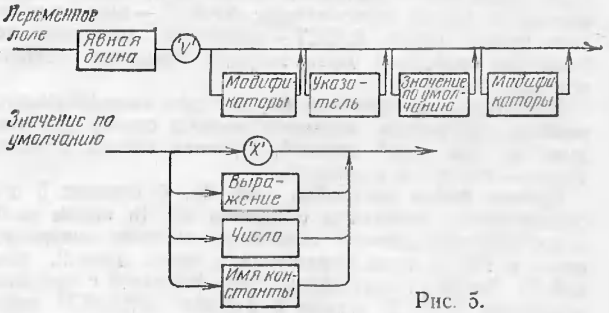


Рис. 5.

SUB — оператор определяет подформаты. Подформатом считается часть формата микрокоманды, длина которой меньше длины микрокоманды. Правила определения подформатов такие же, как и для форматов. Использование операторов SUB удобно в том случае, если несколько форматов имеют одинаковые смежные поля, которые и определяются как подформат.

Определяющие операторы DEF и SUB используются только на фазе настройки. Однако пользователю предоставлена возможность определить дополнительно микрокоманды на фазе ассемблирования с помощью оператора свободного формата FF. Правила определения полей микрокоманды в операторе FF те же, что и в операторах DEF и SUB, однако определять поля как переменные на этапе ассемблирования нет смысла.

Операторы управления микропрограммным счетчиком используются для изменения текущего значения микропрограммного счетчика в программе ассемблирования: ORG — устанавливает новое значение микропрограммного счетчика, равное аргументу оператора; RES — увеличивает текущее значение программного счетчика на заданную величину; ALIGN — устанавливает значения программного счетчика, являющегося ближайшим целочисленным значением кратным аргументу, и превышающем текущее значение программного счетчика.

Выполняемые операторы — совокупность имен форматов, участвующих в образовании одной микрокоманды (рис. 6). Эти операторы образуют тело программы этапа ассемблирования.

На этапе ассемблирования предусмотрена возможность наложения форматов, позволяющая определить все поля МК. В наложении участвуют только форматы, определенные с помощью оператора DEF. При наложении двух форматов биты с «безразличным состоя-

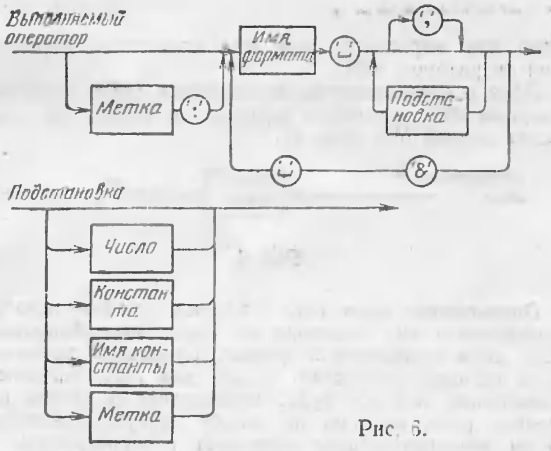


Рис. 6.

нием» первого формата устанавливаются в состояние соответствующих битов второго формата.

Операторы управления печатью предназначены для формирования желаемого вида листинга: TITLE — вызывает печать заголовка на каждой странице листинга; LIST (NOLIST) — вызывает (запрещает) печать строк исходного текста пользователя; SPACE — вызывает печать пустых строк; EJECT — вызывает заполнение оставшейся свободной части текущей страницы листинга пустыми строками.

Операторы комментария служат для самодокументирования программы, занимают полную строку или следуют за командной строкой. Признак начала комментария — это точка с запятой.

Пример файла настройки (рис. 7). В строках 5 и 6 определяются константы с именем R1 (в явном виде, в восьмеричной системе счисления, значение инвертируется) и R2 (в виде выражения с явной длиной, равной 5). Затем следует определение форматов с помощью оператора DEF. В строке 8 формат BRANCH имеет три поля: 10-разрядное с «безразличным состоянием», числовое с незаданной явной длиной и 14-разрядное переменное со значением по умолчанию. В формате NAFIELD в качестве значения по умолчанию для переменного поля (16 VX) используется «безразличное состояние». Файл настройки заканчивается обязательным оператором END.

```
a)
WORD 32                          ;ПРИМЕР ФАЙЛА НАСТРОЙКИ
TITLE(ФАЙЛ ОПРЕДЕЛЕНИЯ)          ;ДЛИНА МИКРОКОМАНДЫ 32 РАЗРЯДА
                                 ;ЗАГОЛОВОК
                                 ;ОПРЕДЕЛЕНИЕ КОНСТАНТ
R1: EQU O:35
R2: EQU 5(15+R1)
    .
    .
    .                            ;ОПРЕДЕЛЕНИЕ ФОРМАТОВ
BRANCH: DEF 10X,D:36,14VXR2
NAFIELD:DEF R1,16VX,10X
ALUFF:  DEF 7V,O:7,12X,10B:101Z
    .
    .
    .
END                              ;КОНЕЦ ФАЙЛА НАСТРОЙКИ

б)
                                 ;ПРИМЕР ФАЙЛА АССЕМБЛИРОВАНИЯ
ORG H:100                        ;НАЧАЛЬНАЯ УСТАНОВКА PC
BRANCH M1
M2: NAFIELD & ALUFF R2X
SP: EQU H:777A                   ;ДООПРЕДЕЛЕНИЕ ЯЗЫКА
    .
    .
    .
NAFIELD SP & ALUFF 7(X + 32)Z
M1: FF SP,8X,8D:511              ;ДООПРЕДЕЛЕНИЕ ЯЗЫКА
EJECT                            ;ПЕРЕХОД В ЛИСТИНГЕ НА НОВУЮ СТРАНИЦУ
NAFIELD M2-Z
    .
    .
    .
END                              ;КОНЕЦ ФАЙЛА АССЕМБЛИРОВАНИЯ
```

Рис. 7.

Транслятор МЕАСС

МЕАСС состоит из трех модулей, связанных через внешние файлы, осуществляющие выполнение вышеописанных этапов настройки и ассемблирования, а также постобработки. При постобработке производится доопределение «безразличных» разрядов (см. выше) в соответствии с техническими особенностями используемого ПЗУ микрокоманд и выдача управляющих файлов для программаторов ПЗУ или для эмуляторов ПЗУ инструментальной системы отладки микропрограмм.

Программа настройки проверяет синтаксис разрабатываемого языка и выдает при наличии ошибок диагностические сообщения. Входными данными для нее является описание записанного в файл входного символического языка, разработанного пользователем. В результате просмотра файла создаются две таблицы: имен размером не более 500 строк (из них 200 — для имен форматов, 200 — для констант и 100 — для имен подформатов) и подформатов, организованная как сверхплотный список [3]. Выходные данные — это листинг исходного файла, файлы констант и форматов.

Программа ассемблирования имеет двухпроходную структуру [4]. На первом проходе выявляются синтаксические ошибки в файле ассемблирования, строится таблица меток, которая может содержать не более 200 строк. На втором проходе исходные микропрограммы транслируются в объектный ход. Входными данными программы являются тексты исходных микропрограмм и файлы констант и форматов, сформированных программой настройки.

Микроассемблер на этапах настройки и ассемблирования проверяет правильность написания исходного текста в соответствии с синтаксисом и семантикой языка. В случае появления ошибки выдается сообщение. Обработка текста при этом продолжается до конца соответствующего файла.

Тестирование микроассемблера осуществлялось при разработке ряда МПС, а также на основе примеров микропрограмм из [1]. Время трансляции 200 строк программной настройки составило 1,5 мин, а 200 строк программой ассемблирования — около 5 мин. Программы разработаны на языке Паскаль, реализованы на микроЭВМ «Электроника 60» и требуют с используемой операционной системой ФОДОС 56К байт оперативной памяти.

В связи с применением при разработке транслятора методов структурного программирования и языка Паскаль возможно совершенствование транслятора: встраивание его в существующие системы и технологии программирования и модификация для расширения функциональных и количественных возможностей, включая планируемое введение условной трансляции, микрорасширений, переменной длины МК и других функций классических ассемблеров.

ЛИТЕРАТУРА

1. Мик Дж., Брик Дж. Проектирование микропроцессорных устройств с разрядно-модульной организацией. В 2-х книгах: Пер. с англ.— М.: Мир, 1984.
2. Злотник Е. М. Секционированные микропроцессоры.— Минск: Наука и техника, 1984.
3. Донован Дж. Системное программирование.—М.: Мир, 1971.
4. Флорес Р. Программное обеспечение.— М.: Мир, 1971.
5. Powers V. M. and Hernandez J. M. Microprogram assemblers for bit-slice microprocessors.— Computer, 1978, July, p. 108—119.
6. Ralph Th. and Blood W. Assembler Streamlines Microprogramming.— Computer Design, 1979, Desember, p. 78—89.
7. The AM2900 Famili With Related Support Circuits.— Advanced Micro Devices, Inc. 1978— p. 3:1—3:10, 4:0—4:35.

Статья поступила 5 января 1986 г.

КРОСС-СИСТЕМЫ — ЭФФЕКТИВНЫЕ СРЕДСТВА АВТОМАТИЗАЦИИ ПРОГРАММИРОВАНИЯ МИКРОПРОЦЕССОРНЫХ УСТРОЙСТВ

При проектировании микропроцессорных устройств (МПУ) наиболее трудоемкими и утомительными этапами являются проектирование, кодирование, тестирование и отладка целевых программ. Выполнение этих этапов без современных средств автоматизации программирования нецелесообразно и практически невозможно.

Средства автоматизации программирования МПУ представляют собой комплекс или пакет системных программных средств, ориентированных на конкретный тип микропроцессора (МП) или однокристальной микроЭВМ (ОЭВМ), функционирующих в среде определенного базового вычислительного комплекса (БВК) и операционной системы. В общем случае все программные средства автоматизации программирования разделяются на резидентные и кросс-средства. Резидентные средства функционируют в среде БВК, в качестве центрального процессора которого используется целевой МП. Для функционирования кросс-средств используется БВК, центральный процессор которого не совпадает с типом целевого МП.

Резидентные и кросс-средства автоматизации программирования по своим функциональным возможностям аналогичны и различаются только составом программных средств. С расширением сферы применения микропроцессоров и однокристальных микроЭВМ и ростом разнообразия их типов кросс-средства приобретают все большее значение и распространение по следующим основным причинам:

использование для автоматизации программирования массовых, недорогих и доступных пользователю моделей БВК, в первую очередь мини-ЭВМ и персональных ЭВМ;

необходимость выбора для функционирования средств программирования БВК с высокими характеристиками по быстродействию, объемам внутренней и внешней памяти, развитой периферией;

необходимость создания и применения универсальных систем программирования на базе единого БВК для автоматизации программирования многопроцессорных МПУ, имеющих в своем составе МП и ОЭВМ различных типов;

наличие значительной номенклатуры типов 4- и 8-разрядных ОЭВМ, не предназначенных для работы в качестве центральных процессоров развитых БВК, и, как следствие, отсутствие и практическая невозможность промышленного создания для некоторых типов МП и ОЭВМ соответствующих БВК с резидентным центральным процессором.

В качестве БВК и операционных систем для функционирования кросс-средств наиболее часто используются серийно вычислительные комплексы на базе микроЭВМ «Электроника 60», мини-ЭВМ «Электроника 100/25», «Электроника 79», СМ-4, СМ-1800, ДВК и операционные системы РАФОС, ОС-РВ, ОС ДВК, а также операционные системы, совместимые с СР/М.

В состав кросс-средств входит следующий типовой набор программных компонент: редактор текста, кросс-ассемблер (для систем программирования высокой производительности — компилятор), кросс-редактор связей, программно-логическая модель целевого МП или ОЭВМ и отладчик. Эти средства обеспечивают полную автоматизацию получения единого загрузочного модуля для МПУ из исходных текстов, его тестирование и отладку. Кросс-средства автоматизации программирования можно многократно использовать на всех основных этапах проектирования МПУ: проектирование, автономная и комплексная отладка и испытание МПУ в реальных условиях. Кросс-средства обеспечивают минимальные затраты времени на прохождение циклов проектирования МПУ и полностью исключают появление новых ошибок при преобразовании программ.

Цикл статей по кросс-системам для ОЭВМ КМ1816ВЕ48 открывает статья Кобылинского А. В. и др., содержащая описание языка ассемблера АСМ 48— основы для построения кросс-ассемблеров. Особенность кросс-системы, представленной в статье Антонова Б. В. и др., состоит в наиболее полном наборе программных компонент: кросс-ассемблер, редактор связей, программно-логическая модель и дизассемблер.

В статье Белова А. М. и др. описываются кросс-системы Микросс-580 и Микросс-048 для микропроцессора К580ВМ80 и ОЭВМ КМ1816ВЕ48. Данные кросс-системы сданы в ОФАП и переданы в эксплуатацию большому числу пользователей. Кросс-система, рассмотренная в статье Зобина Г. Я. и др., эксплуатируется на мини-ЭВМ СМ-4 в среде ОС РВ.

УДК 681.3.06

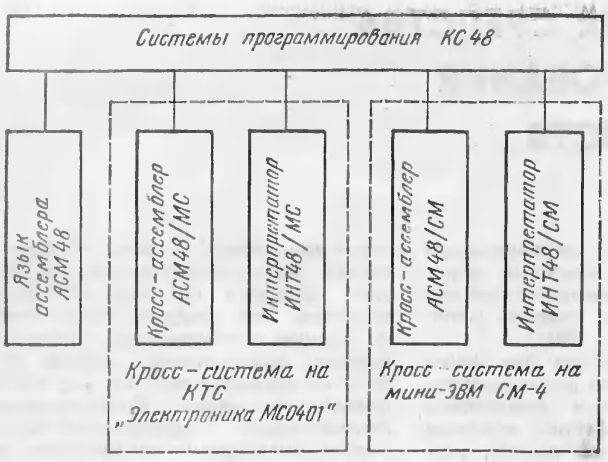
А. В. Кобылинский, Н. Г. Сабадаш, А. К. Тесленко

СИСТЕМА АВТОМАТИЗАЦИИ ПРОГРАММИРОВАНИЯ ОДНОКРИСТАЛЬНОЙ МИКРОЭВМ

Разработанная кросс-система КС48 реализована на широко распространенных инструментальных ЭВМ: комплексе технических средств (КТС) «Электроника МС0401» («Электроника К1-30») в операционной системе ДОС [2]; мини-ЭВМ СМ-4 в операционной системе ОС РВ.

В состав средств автоматизации подготовки программ кросс-системы (см. рисунок) входят редактор текстов инструментальной ЭВМ, язык ассемблера АСМ48 и кросс-ассемблер АСМ48.

Язык ассемблера АСМ48 для ОМЭВМ К1816ВЕ48 позволяет символически обозначать команды и адреса,



Структурная схема кросс-системы КС48

разнообразить способы задания данных, управлять процессом трансляции, в том числе распределением памяти, реализовать макрокоманды.

Программа на языке АСМ48 состоит из предложений, каждое из которых содержит не более 128 символов. Предложения строятся из символов алфавита языка, соответствующих наборам 0; 2 или 3 КОИ-7 ГОСТ 19767—74.

К лексемам языка относятся разделители, ключевые слова, идентификаторы, числа, символьные строки, комментарии. Ключевые слова подразделяются на следующие имена: машинных команд, программно-доступных устройств однокристальных микроЭВМ (ОМЭВМ), директив и операторов в выражении.

В идентификаторах прописные и строчные буквы латинского алфавита неразличимы. Вместо строчных латинских букв в соответствии с КОИ-7 могут использоваться буквы русского алфавита. Значащими в идентификаторах являются первые шесть символов.

Числа в языке АСМ48 можно представить в двоичной, восьмеричной, десятичной и шестнадцатеричной системе. Предложения языка делятся на инструкции, определяющие команды ОМЭВМ, и директивы (псевдокоманды). Инструкция и директива содержат поля метки (названия), мнемокода, аргументов и комментария. Выражения языка используются в поле аргументов и способствуют повышению выразительности программ.

Наименьшая и неделимая часть выражения (исключая обозначения операции) — это термы. В языке АСМ48 имеются следующие виды термов: число; символьная строка; имя метки; обозначение счетчика адреса; имя-синоним. Термы число, символьная строка и обозначение счетчика адреса — самоопределенные. Перемещаемые и внешние термы в языке отсутствуют. Это вызвано ограниченным объемом памяти ОМЭВМ КМ1816ВЕ48 и практической невозможностью побайтного перемещения программ, так как ссылки в командах условного перехода являются внутристраничными. Над термами выполняются операции сложения, вычитания, умножения, деления, логические поразрядные операции, операции линейного сдвига и выделения старшего и младшего байта. Арифметические операции выполняются по mod 65536.

Выражения языка ассемблера АСМ48 используются для определения длинного (11-битного) адреса ссылок внутри банка памяти команд, короткого (8-битного) адреса ссылок внутри страницы памяти команд и непосредственных 8-битных данных.

В первом случае выражение задается в командах безусловного перехода JMP и CALL. Значение выражения должно быть в пределах 0...4095 (т. е. 12-битное). Старший бит выражения при генерации команды не используется и служит для улучшения выразительности текста программы. Это вызвано тем, что хотя переходы из 2К-байтного банка памяти в банк осуществляются командами JMP и CALL, но задаются специальными командами выбора банка памяти.

Во втором случае выражение задается в командах условного перехода. При генерации кода используются младшие восемь битов, а старшие биты служат для контроля расположения точки назначения в требуемой странице памяти.

Значение выражения при задании непосредственных данных должно находиться в пределах 0...255, и ему должен предшествовать знак #. Кроме задания адресов и данных с помощью выражений, поле аргументов можно использовать также для определения операнда в программно-доступном устройстве ОМЭВМ и косвенного адреса операндов в памяти данных или косвенного адреса точки назначения в памяти команд. В этом случае значению выражения предшествует символ @, за которым следует имя программно-доступного элемента.

В состав псевдокоманд языка АСМ48 входят: ORG, DS, EQU, SET, DB, DW, IF, ELSE, ENDIF и END. Псевдокоманды ORG и DB обеспечивают установку или наращивание содержимого счетчика адреса ассемблера; EQU и SET служат для создания имен-синонимов; DB и DW используются для задания констант в памяти команд; IF, ELSE, ENDIF позволяют задавать условия ассемблирования отдельных участков программ; END задает признак конца программы. Выражение в поле аргументов псевдокоманды END позволяет указывать точку входа в программу; выражения псевдокоманд ORG, DS, EQU, SET и IF должны содержать только самоопределенные или предварительно определенные термы.

Заголовок макроопределений задается псевдокомандами MACRO, REPT, IRP и IRPC; конец макроопределений — ENDM. Псевдокоманда MACRO в поле названия содержит название макрокоманды, а в поле аргументов — формальные аргументы макрокоманды. Макрорасширение задается макрокомандой, содержащей имя в поле мнемокода, а в поле аргументов — фактические параметры (позиционные), которые могут передаваться как по форме, так и по содержанию. Точка расширения для REPT, IRP и IRPC располагается непосредственно за макроопределением. Эти макроопределения задают многократное дублирование тела макроопределения, причем REPT — без его изменения. Макроопределения IRP и IRPC позволяют при каждом дублировании их тела выбирать новые фактические параметры из поля аргументов макроопределения.

Важное значение для широкого внедрения ОМЭВМ имеет реализация кросс-ассемблером стандартного (общепринятого) языка. Поскольку выпуск стандарта на язык представляет собой достаточно длительный процесс, то необходимо ориентироваться на общепринятый в мировой практике язык ассемблера. Для ОМЭВМ КМ1816ВЕ48 таким языком и является рассмотренный язык АСМ48.

Кросс-ассемблер АСМ48 в процессе выполнения основных функций — трансляции исходных программ на языке ассемблера АСМ48 в объектные программы и формирования листинга с указанием обнаруженных синтаксических ошибок — предоставляет пользователю следующие возможности:

включать в исходную программу макроопределения или подпрограммы, расположенные в других файлах. Это обеспечивает автоматическую сборку раздельно разрабатываемых программ;

управлять содержимым и форматом листинга;

включать в листинг таблицу символов и таблицу перекрестных ссылок, представленных в лексикографическом порядке;

определять устройства вывода для листинга [диск (файл), печать, экран];

управлять формированием объектного файла и определять устройство для его вывода [диск (файл), ленточный перфоратор].

Кросс-ассемблер АСМ48 создает объектный файл в шестнадцатеричном формате, распространенном в системах на основе микропроцессора КР580ИК80.

Особенность кросс-ассемблера АСМ48 – обнаружение не только синтаксических, но и семантических ошибок, таких как несоответствие страницы места расположения и точки назначения команд условной передачи управления и переход из первого банка памяти во второй без использования команд вызова подпрограмм или безусловной передачи управления.

Для автоматизации тестирования и отладки программ ОМЭВМ в кросс-системе КС48 служит **интерпретатор ИНТ48**, моделирующий все программно-доступные элементы ОМЭВМ, включая порты расширителя и внешнюю память команд и данных, а также обеспечивающий выполнение команд ОМЭВМ КМ1816ВЕ48 в моделируемой среде на инструментальной ЭВМ. По реализации значительная часть интерпретатора служит для обработки команд и директив на тестирование и отладку.

Наряду со стандартным набором возможностей [загрузка объектных файлов из внешних устройств инструментальной ЭВМ в моделируемую память; отображение и модификация содержимого программно-доступных элементов ОМЭВМ, а также счетчика циклов; формирование и вывод строк трассировки; организация точек разрывов и остановов по типичным событиям в программе (чтение и запись ячеек памяти, многократное выполнение некоторой команды, выполнение заданного количества команд); запуск отлаживаемой программы начиная с любой команды] интерпретатор ИНТ48 предоставляет пользователям дополнительные возможности:

оперативно изменять состав строки трассировки в зависимости от конкретных условий тестирования и отладки;

разделять точки разрыва на постоянные (удаляемые только специальной командой) и временные (удаляемые по их достижению);

использовать как интерактивный, так и предустановленный режимы реакции на обнаруживаемые события [3];

определять полноту тестовой задачи для тестируемой программы по количеству выполненных команд с указанием адресов невыполнявшихся команд;

отображать и модифицировать память команд с помощью встроенных дизассемблера и упрощенного ассемблера;

использовать моделирование команд ввода-вывода с помощью ячеек памяти или внешних устройств инструментальной ЭВМ;

оперативно и независимо изменять ввод команд и директив интерпретатора на тестирование и отладку с терминала или из дискового файла и вывод протокола отладки на дисплей, печать или диск, что существенно повышает удобства эксплуатации интерпретатора;

заданием специальных директив настраивать интерпретатор ИНТ48 на тестирование и отладку других ОМЭВМ, совместимых с КМ1816ВЕ48 по системе команд, но отличающихся объемом внешней и резидентной памяти команд и данных;

составлением специальных программ с последующей связью с одной из фаз интерпретатора ИНТ48 расширять его функциональные возможности при моделировании ввода-вывода и задании нетипичных событий в случае поиска сложных ошибок в программах.

Программы интерпретатора ИНТ48 составлялись на языке Фортран с учетом обеспечения их мобильности и отлажены на мини-ЭВМ СМ-4. При их переносе на КТС «Электроника МС0401» часть подпрограмм переписывалась на языке PL/M и ассемблере.

Кросс-система КС48 внедрена в состав программного обеспечения серийно выпускаемого КТС «Электроника МС0401». В эксплуатационную документацию входят: «Ассемблер АСМ48/МС. Описание языка»; «Ассемблер АСМ48/МС. Руководство программиста»; «Интерпретатор ИНТ48/МС. Руководство программиста».

В состав эксплуатационной документации кросс-системы КС48 на базе мини-ЭВМ СМ-4 входят: «Ассемблер АСМ48/МС. Руководство программиста»; «Интерпретатор ИНТ48/СМ. Руководство программиста».

ЛИТЕРАТУРА

1. Кобылинский А. В., Липовецкий Г. П. Однокристальные микроЭВМ серии К1816.— Микропроцессорные средства и системы, 1986, № 1, с. 10—19.
2. Добровольский Н. В., Кобылинский А. В., Быба В. И., Сабадаш Н. Г., Тесленко А. К. Средства автоматизации программирования для микропроцессорных комплектов БИС серий К580 и КР580.— Механизация и автоматизация управления, 1983, № 3, с. 22—25.
3. Парнюк О. А., Тесленко А. К. Диалоговый отладчик программ на языках высокого уровня.— Управляющие системы и машины, 1984, № 6, с. 70—74.

Статья поступила 26 ноября 1985 г.

УДК 681.3.06

Б. В. Антонов, С. Ф. Глазер, А. Г. Маликов, А. И. Шабалин

СИСТЕМА АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ОДНОКРИСТАЛЬНОЙ МИКРОЭВМ К1816ВЕ48

В состав кросс-системы автоматизации проектирования ПО для ОЭВМ К1816ВЕ48 (САПР-48), функционирующей под управлением ОС ДВК (ОС РАФОС) на ДВК-2М, микроЭВМ «Электроника-60», мини-ЭВМ «Электроника-100/25», «Электроника-79», входят кросс-макроассемблер, редактор связей, эмулятор-отладчик и дизассемблер. Запуск, задание входных и выходных файлов и режимов работы программ САПР-48 производится аналогично системным программам базовой ОС.

Кросс-макроассемблер преобразует исходную программу, написанную на языке ассемблера ОЭВМ К1816ВЕ48 и созданную с помощью редактора текста базовой ОС, в объектный модуль*. При необходимости кросс-макроассемблер выдает листинг программы, содержащий результаты трансляции исходного модуля (рис. 1).

* Синтаксис и мнемоника команд, директив и макросредств ассемблера полностью соответствуют описанным в работе MCS-48 Family of Single Chip Microcomputers. User's Manual Intel Corp. 1979. Дополнительно введены две директивы для организации межмодульных связей.

```
К1816ВЕ48 КРОСС-АССЕМБЛЕР, MAC48 VA2-01, 25-JUL-85 00:33:05 PAGE 1

LOC  OBJ          SEQ          SOURSE STATEMENT

                    01          RTMP    EQU     R5
                    02          COUNT   EQU     39
                    03          ORG     0
0000 04EA           04          JMP     INIT    ;ИНИЦИАЛИЗАЦИЯ
                    05          ORG     7       ;ОБРАБ.ПРЕРЫВАНИЯ
0007 35             06 TIMER:   DIS     TCNTI   ;ЗАПРЕТ ПРЕРЫВАНИЯ
0008 65             07          STOP    TCNT    ;ОСТАНОВ ТАЙМЕРА
0009 D5             08          SEL     RB1     ;1-ЫЙ БАНК РЕГИСТРОВ
000A AD             09          MOV     RTMP,A  ;СОХРАНИТЬ АККУМ.
000B 27             10          CLR     A       ;СБРОС АККУМУЛЯТОРА
000C 62             11          MOV     T,A     ;СБРОС ТАЙМЕРА
000D B827           12          MOV     R0,#COUNT ;АДР.СЧ.ПРЕРЫВАНИЙ
000F 23E0           13          MOV     A,#0E0H ;КОНТРОЛЬНОЕ ЧИСЛО
0011 60             14          ADD     A,@R0   ;НУЖНОЕ К-ВО ПРЕРЫВ.?
0012 E618           15          JNC     TIM1    ;НЕТ
0014 27             16          CLR     A       ;ДА, СБРОС АКК.
0015 A0             17          MOV     @R0,A   ;СБРОС СЧ-КА ПРЕРЫВ.
0016 041D           18          JMP     TIM2    ;ДАЛЬНЕЙШАЯ ОБРАБОТКА
0018 18             19 TIM1:    INC     @R0     ;УВЕЛИЧЕНИЕ СЧ. ПРЕР.
0019 FD             20          MOV     A,RTMP  ;ВОССТАНОВЛЕНИЕ АКК.
001A 25             21          EN      TCNTI   ;РАЗРЕШЕНИЕ ПРЕРЫВ.
001B 55             22          STRT    T       ;ПУСК ТАЙМЕРА
001C 93             23          RETR            ;ВОЗВРАТ
```

Рис. 1. Фрагмент листинга программы

Редактор связей настраивает адресные константы и объединяет отдельные объектные модули в один абсолютный загрузочный модуль, формирует карту загрузки (рис. 2).

```
        LINK48  VA1-01          03:27:25    15-JUN-84

MEMORY ALLOCATION MAP FOR DK :LIBR  .MAP
TRANSFER ADDRESS: 0002
**** MODULE .MAIN. FROM FILE  DK :LIBR0 .048
          LAB1    0100    LAB2    0106    LAB3    0120
          LAB4    0138
**** MODULE .MAIN. FROM FILE  DK :DDT   .048
          D.DDT   0002    D.END   0080
**** MODULE .MAIN. FROM FILE  DK :LIBR1 .048
          LAB5    0226    LAB6    0650    LAB7    0342
          LAB8    0208    LAB9    0308
**** MODULE .MAIN. FROM FILE  DK :LIBR2 .048
**** MODULE .MAIN. FROM FILE  DK :LIBR3 .048
          LAB10   0094
**** MODULE .MAIN. FROM FILE  DK :LIBR4 .048
          LAB11   0088    LAB12   0958    LAB13   0014
          LAB14   0839
**** MODULE .MAIN. FROM FILE  DK :LIBR5 .048
```

Рис. 2. Фрагмент карты памяти

Эмулятор-отладчик, содержащий логическую модель ОЭВМ К1816ВЕ48, предназначен для тестирования и отладки ПО в диалоговом режиме работы. Пользователь с помощью команд отладки имеет возможность:

запускать программу с любого адреса в различных режимах выполнения;

распечатывать и модифицировать содержимое областей памяти в адресном пространстве модели ОЭВМ;

осуществлять доступ к внутренним ресурсам ОЭВМ с целью их изменения, что на реальной аппаратуре практически неосуществимо;

задавать необходимый интервал времени работы микроЭВМ;

устанавливать до восьми точек контрольного останова по условиям адрес-данное и чтение-запись;

останавливать выполнение программы в любой момент времени;

определять для анализа путь, приведший к ошибке, распечатывая «историческую память», содержащую информацию о 32 последних выполненных командах;

сохранять отлаженную программу в файле.

В процессе работы ведется контроль реального времени выполнения программы, индицируются на экране дисплея входные и выходные данные, формируются и выдаются диагностические сообщения, дублируется протокол отладки в отдельном файле (рис. 3).

```
  #TIM
REAL TIME:    0 M  0.038630 S
  #TIM 237MS
  #TIM
INTR TIME:    0 M  0.237000 S
  #RUN
  **** INTR TIMER
  $BPT 2:C152/A
  #RUN 0
**** MASK 2 INTR OCCUR
PC=152, A=00, PSW=08,  RB0: 06 F3 D0 EA FE E5 38 D3
  #MOD A
A - 00/ FF
  #MOD CNT
CNT - 03/ CF
  #MOD PSW
PSW - 08/ FF
  #ACC
PC=300, A=FF, PSW=FF:  C=1, AC=1, F0=1, BS=1, SP=07
RB0: 06 F3 D0 EA FE E5 38 D3
RB1: 1E CC BB E4 DA 0A 24 E5
STK: 1111 2222 CC36 CBF8 EB02 E5B4 D2FE 84EC
F1=0, TF=0, CNT=CF
T0=0, T1=0, INT=1, BUS=FF, P1=FF, P2=FF
```

Рис. 3. Фрагмент протокола отладки

Дизассемблер преобразует программу в формате абсолютной загрузки в ассемблерный формат исходного модуля. Используется для определения сегментов перекрытия загрузочного модуля или для получения исходных модулей ПО ОЭВМ, на которое нет достаточной документации (рис. 4).

```
        ; НЕПОСРЕДСТВЕННЫЕ ДАННЫЕ
????7   EQU     027H
????E0  EQU     0E0H
        ORG     0000H
        ; НАЧАЛО СЕГМЕНТА КОМАНД
!!! <0000>  НЕТ КОМАНДЫ ПЕРЕХОДА ИЛИ ВЫЗОВА
        JMP     ????EA          ;;      DB      4,0EAH
        ORG     0007H
        ; НАЧАЛО СЕГМЕНТА КОМАНД
!!! <0007>  НЕТ КОМАНДЫ ПЕРЕХОДА ИЛИ ВЫЗОВА
        DIS     TCNTI           ;;      DB      35H
        STOP    TCNT            ;;      DB      65H
        SEL     RB1             ;;      DB      0D5H
        MOV     R5,A            ;;      DB      0ADH
        CLR     A               ;;      DB      27H
        MOV     T,A             ;;      DB      62H
        MOV     R0,#?????7      ;;      DB      0B8H,27H
        MOV     A,#????E0       ;;      DB      23H,0E0H
        ADD     A,@R0           ;;      DB      60H
        JNC     ?7018           ;;      DB      0E6H,18H
        CLR     A               ;;      DB      27H
        MOV     @R0,A           ;;      DB      0A0H
        JMP     ?701D           ;;      DB      4,1DH
```

Рис. 4. Фрагмент дизассемблированной программы

Загрузочный модуль, созданный и отлаженный с помощью САПР-48, можно записать в ППЗУ с помощью серийно выпускаемых программаторов МП-2 или «Пром-1».

САПР-48 позволяет эффективно разрабатывать и отлаживать системные и прикладные программы, сокращает сроки разработки ПО, повышает его качество. Система прошла функциональное тестирование (около 70 различ-

ных тестов и опытную эксплуатацию в течение одного года в трех проектных работах.

Система поставляется на МЛ или ГМД вместе с эксплуатационной документацией общим объемом около 200 страниц текста в составе:

описание применения;

описание программ;

ассемблер микроЭВМ К1816ВЕ48. Описание языка;

программа кросс-ассемблер, руководство программиста;

программа редактор связей, руководство программиста;

программа эмулятор, руководство программиста;

программа дизассемблер, руководство программиста.

Адрес для запроса документации: 290034, Львов, бюро заказов.

Статья поступила 11 ноября 1985 г.

УДК 681.3.06

А. М. Белов, Е. А. Иванов, Л. Л. Муренко

КОМПЛЕКСЫ КРОСС-ПРОГРАММ «ЭЛЕКТРОНИКА МИКРОСС»

Комплексы кросс-программ «Электроника Микросс» имеют в своем составе редактор текста, кросс-ассемблер, кросс-редактор связей, программно-логическую модель (отладчик), сервисные программы, библиотеку стандартных подпрограмм.

Перечисленные программы обеспечивают выполнение следующих функций:

формирование (редактирование) исходного текста прикладной программы и запись его на гибкий магнитный диск;

трансляция текстов исходных модулей с получением распечатки результатов трансляции на печатающем устройстве инструментальной ЭВМ и получение файлов перемещаемых объектных модулей на гибком магнитном диске;

компоновка отдельных объектных модулей (файлов) в единый загрузочный модуль (файл) на гибком магнитном диске и настройка его на заданное пользователем адресное пространство ПЗУ микропроцессора;

использование библиотеки стандартных подпрограмм на уровне кросс-редактора связей;

отладка разрабатываемой программы на программно-логической модели с получением полной информации о ходе выполнения отлаживаемой программы;

получение распечатки эксплуатационной документации на печатающем устройстве инструментальной ЭВМ;

реализация всех возможностей, предоставляемых аппаратными и программными средствами инструментальной ЭВМ.

Программно-логическая модель предоставляет пользователю возможность отладки прикладных программ: просмотр и изменение содержимого любой ячейки памяти отлаживаемой программы, получение твердой копии содержимого любого участка памяти отлаживаемой программы, прогон отлаживаемой программы или любой ее части, трассировка, имитация деления памяти отлаживаемой программы на ОЗУ и ПЗУ, имитация ввода-вывода и внешних прерываний, останов отлаживаемой программы в заданной точке, просмотр и изменение содержимого регистров моделируемого микропроцессора.

Комплексы кросс-программ «Электроника Микросс» поддерживают связь с пользователем как в режиме диалога, так и в режиме командных строк, обеспечивая максимальную простоту эксплуатации. Сопровождаются полным комплектом эксплуатационной документации, подготовленной в соответствии с ЕСПД. В отраслевой фонд алгоритмов и программ сданы комплексы «Электроника Микросс-580» для БИС серии К580 и «Электроника Микросс-048» для БИС микроЭВМ серии К1816.

Статья поступила 30 августа 1985 г.

УДК 681.3.06

Г. Я. Зобин, Е. С. Кривопальцев, А. Б. Минкович, А. И. Огнев, Г. Ф. Эпштейн

КРОСС-СИСТЕМА ДЛЯ ОДНОКРИСТАЛЬНОЙ МИКРОЭВМ КМ1816ВЕ48

В настоящее время отечественной промышленностью освоен выпуск однокристальной микроЭВМ (ОЭВМ) КМ1816ВЕ48 [1], отличающейся высоким быстродействием, наличием встроенной памяти программ, памяти данных, портов ввода-вывода, а также внутреннего таймера. Широкому распространению ОЭВМ КМ1816ВЕ48 должна способствовать оснащенность их кросс-средствами (на ЭВМ СМ-4 в операционной среде ОС РВ и РАФОС) для отладки прикладного программного обес-

печения (ПО), а также внутрисхемными эмуляторами.

В кросс-средства входят редактор текста, кросс-транслятор с ассемблера ОЭВМ, компоновщик, интерактивный отладчик.

Программы редактор, компоновщик, транслятор с языка «MACRO-11» обеспечиваются применяемой операционной системой, а кросс-транслятор с ассемблера и интерактивный отладчик разработаны.

Кросс-транслятор (20К слов) с ассемблера ОЭВМ КМ1816ВЕ48 реализован на базе языков «Фортран-4» и «MACRO-11». Кросс-транслятор из входного символьного файла с инструкциями и директивами ассемблера ОЭВМ создает два выходных файла: промежуточный символьный файл образа (содержащий директивы типа «.BYTE...») и файл листинга.

Спецификации файлов задаются стандартным для используемой операционной системы образом. Запись промежуточного файла в указанном виде позволяет применить стандартные средства операционной системы для создания образа задачи и дальнейшего использования его совместно с программой «Отладчик» для отработки программы.

Программа транслируется стандартно — в два прохода. На первом проходе распределяется память, составляется таблица используемых идентификаторов и проверяется правильность синтаксиса. На втором проходе транслятор строит коды команд, используя таблицу идентификаторов, полученную на 1-м проходе, а также осуществляет диагностику. Листинг трансляции содержит коды команд, соответствующие им адреса программной памяти и, кроме того, таблицу идентификаторов, а также обнаруженные ошибки с указанием их вида. Возможно использование арифметических выражений, представление чисел в восьмеричном, десятичном, шестнадцатеричном виде, управление счетчиком команд, использование директив условной трансляции, резервирование памяти в байтах и словах, запись констант и символьных строк.

На этапе трансляции производятся диагностика и выявление лексических, синтаксических ошибок, допустимости переходов, разрешение символических ссылок, проверка пересечения границ памяти и т. п. При обнаружении ошибок указывается их вид (рис. 1) и местоположение в листинге.

Полученный промежуточный файл транслируется компилятором «MACRO-11» в объектный файл и может быть использован непосредственно для программирования ПЗУ ОЭВМ. Кроме того, этот файл совместно с объектным файлом программы «Отладчик» обрабатывается построителем задач, выходным продуктом которого является отладочная модель, включающая в себя и программу «Отладчик».

```
1  - ОШИБКА ЧТЕНИЯ ФАЙЛА ИСХОДНОГО ТЕКСТА;
2  - НЕДОПУСТИМАЯ ВОСЬМЕРИЧНАЯ КОНСТАНТА;
3  - НЕДОПУСТИМЫЙ НОМЕР РЕГИСТРА ПРИ
     КОСВЕННОЙ АДРЕСАЦИИ;
4  - НЕДОПУСТИМЫЙ СИМВОЛ;
5  - ПОВТОРЕНИЕ МЕТКИ;
6  - НЕДОПУСТИМЫЙ АДРЕС ПЕРЕХОДА ВНУТРИ
     СТРАНИЦЫ;
7  - ОТСУТСВУЕТ ОПЕРАТОР END;
8  - ДЛИННАЯ СИМВОЛЬНАЯ СТРОКА;
9  - НЕОПРЕДЕЛЕННЫЙ ИДЕНТИФИКАТОР;
10 - НЕДОПУСТИМОЕ ЗНАЧЕНИЕ АРИФМЕТИЧЕСКОГО
     ВЫРАЖЕНИЯ;
11 - НЕДОПУСТИМЫЙ АДРЕС ПЕРЕХОДА В КОМАНДЕ
     JMP;
12 - ЗНАЧЕНИЕ СЧЕТЧИКА КОМАНД БОЛЬШЕ
     ДОПУСТИМОГО;
13 - ОШИБКА СИНТАКСИСА;
14 - ОПЕРАТОР ELSE ВНЕ ТЕЛА УСЛОВНОГО
     БЛОКА;
15 - НЕПРАВИЛЬНАЯ СТРУКТУРА УСЛОВНОГО
     БЛОКА;
```

Рис. 1. Виды ошибок, выявляемых кросс-транслятором

Программа «Отладчик» (24К слов) полностью имитирует внутреннюю структуру и полный набор команд ОЭВМ КМ1816ВЕ48 (рис. 2). «Отладчик» позволяет отлаживать программы пользователя (до 4К байт) и выполняется в режиме монитора и режиме исполнения.

В режиме монитора «Отладчик» позволяет:

просматривать и модифицировать содержимое любой ячейки микроЭВМ, всех флагов, портов ввода-вывода, таймера, тестируемых входов, ячеек внешних устройств ввода-вывода;

запускать программу с любого заданного или текущего адреса;

отрабатывать программу по шагам;

```
C/  - ПРОЧИТАТЬ СЧЕТЧИК КОМАНД;
A/  - ПРОЧИТАТЬ АККУМУЛЯТОР;
RK/ - ПРОЧИТАТЬ РЕГИСТР С НОМЕРОМ'K';
DK/ - ПРОЧИТАТЬ ЯЧЕЙКУ ПАМЯТИ ДАННЫХ С
      НОМЕРОМ'K';
W/  - ПРОЧИТАТЬ СЛОВО СОСТОЯНИЯ ПРОГРАММЫ;
BK/ - ПРОЧИТАТЬ ТОЧКУ ОСТАНОВА С НОМЕРОМ'K';
WS/ - ПРОЧИТАТЬ УКАЗАТЕЛЬ СТЕКА;
PK/ - ПРОЧИТАТЬ ПОРТ С НОМЕРОМ'K';
EK/ - ПРОЧИТАТЬ ЯЧЕЙКУ СТЕКА С НОМЕРОМ'K';
UK/ - ПРОЧИТАТЬ ВНЕШНЮЮ (ОТНОСИТЕЛЬНО ОМЭ)
      ЯЧЕЙКУ ПАМЯТИ ДАННЫХ С НОМЕРОМ'K';
KG  - ПУСТИТЬ ПРОГРАММУ С ЗАДАННОГО (K)
      ИЛИ ТЕКУЩЕГО АДРЕСА;
S   - ВЫПОЛНИТЬ ОДИН ШАГ ПРОГРАММЫ;
N   - ИМИТИРОВАТЬ ПЕРВОНАЧАЛЬНУЮ УСТАНОВКУ ОМЭ;
K/  - ПРОЧИТАТЬ ЯЧЕЙКУ ПРОГРАММЫ ПОЛЬЗОВАТЕЛЯ С
      НОМЕРОМ 'K';
K(BK)- ЗАПИСАТЬ В ЯЧЕЙКУ ЧИСЛО 'K';
K(ПС)- ЗАПИСАТЬ В ЯЧЕЙКУ ЧИСЛО 'K', ЗАКРЫТЬ И
       ПРОЧИТАТЬ СЛЕДУЮЩУЮ;
K(←) - ЗАПИСАТЬ В ЯЧЕЙКУ ЧИСЛО 'K', ЗАКРЫТЬ И
       ПРОЧИТАТЬ ПРЕДЫДУЩУЮ;
;KB- СНЯТЬ ТОЧКУ ОСТАНОВА С НОМЕРОМ 'K'.
```

Рис. 2. Список команд программы «Отладчик»

перестраивать счетчик команд ОЭВМ;

назначать до восьми точек останова программы;

модифицировать программу пользователя;

индицировать время исполнения программы в целом или ее фрагмента;

выводить файл трассировки на внешнее устройство (терминал, диск и т. п.).

В режим исполнения «Отладчик» переходит по специальной команде, задающей отработку отлаживаемой программы.

Допускается непрерывная отработка программы и в пошаговом режиме. В режиме исполнения «Отладчик» полностью имитирует работу ОЭВМ по программе пользователя (за исключением команды ENTO CLK, которая разрешает подачу внутренней синхронизирующей частоты на вывод 0).

В точках останова или после появления ошибочной ситуации в программе пользователя (недопустимый код команд, неправильное смещение в командах перехода и т. п.) из режима исполнения при непрерывной отработке программ ОЭВМ переходит в режим монитора.

Для режима исполнения характерны:

работа с внутренними ячейками ОЭВМ;

с портами ввода-вывода;

с внешними (до 256) относительно ОЭВМ устройствами ввода-вывода;

работа внутреннего таймера ОЭВМ в режиме счетчика внешних событий и в режиме интервального таймера;

выполнение режимов прерываний от внешнего события и от таймера;

```
МПС 1816

АДР.  КОД   ПОЗ.      ТЕКСТ ПРОГРАММЫ

                1  ;ПРОГРАММА ВЫЧИТАНИЯ 16-БИТОВЫХ ЧИСЕЛ
                2  ;R1,R0-СТ. И МЛ. ЧАСТИ УМЕНЬШАЕМОГО
                3  ;R3,R2-СТ. И МЛ. ЧАСТИ ВЫЧИТАЕМОГО
                4  ;    РЕЗУЛЬТАТ -- ПОРТ1;ПОРТ2
                5  ;
                6       AA=20H
                7       .=AA
0020  1430        8       CALL SUB16
0022  8628        9       JF0 S1          ;ПРОВЕРКА ОШИБКИ
0024  BC00       10       MOV R4,#0       ;  ОШИБКА=
0026  BD30       11       MOV R5,#0       ;= ОБНУЛЕНИЕ РЕЗУЛЬТАТА
0028  FC         12  S1:  MOV A,R4        ;
0029  39         13       OUTL P1,A       ;РЕЗ. МЛ. ВЫВЕСТИ В ПОРТ 1
002A  FD         14       MOV A,R5        ;
002B  3A         15       OUTL P2,A       ;РЕЗ. СТ. ВЫВЕСТИ В ПОРТ 2
                16                       ;
                17       .=AA+10H
0030  FA         18  SUB16:MOV A,R2       ;ВЗЯТЬ МЛ. ЧАСТЬ ВЫЧИТАЕМОГО
0031  37         19       CPL A           ;  ПОЛУЧИТЬ ДОПОЛНЕНИЕ
0032  0321       20       ADD A,#1        ;  МЛАДШЕЙ ЧАСТИ
0034  AC         21       MOV R4,A        ;  УМЕНЬШАЕМОГО
0035  FB         22       MOV A,R3
0036  37         23       CPL A           ;  ПОЛУЧИТЬ ДОПОЛНЕНИЕ
0037  1300       24       ADDC A,#0       ;  СТАРШЕЙ ЧАСТИ
0039  AD         25       MOV R5,A        ;  УМЕНЬШАЕМОГО
                26  ;  СЛОЖИТЬ
                27  ;  РЕЗУЛЬТАТ НА R4,R5
003A  FC         28       MOV A,R4
003B  68         29       ADD A,R0
003C  AC         30       MOV R4,A        ;МЛ. ЧАСТЬ РЕЗУЛЬТАТА
003D  FD         31       MOV A,R5
003E  79         32       ADDC A,R1
003F  AD         33       MOV R5,A        ;СТ. ЧАСТЬ РЕЗУЛЬТАТА
0040  85         34       CLR F0
0041  E644       35       JNC S2          ;ПРОВЕРКА ЗНАКА РЕЗУЛЬТАТА
0043  95         36       CPL F0          ;  РЕЗ.<0 , ЕСТЬ ОШИБКА (F0=1)
                37                       ;  РЕЗ.>0 , НЕТ ОШИБКИ (F0=0)
0044  83         38  S2:  RET
                39       END
 ОШИБОК НЕТ

      ***  ТАБЛИЦА СИМВОЛОВ  МПС 1816  ***
AA     = 0020  S1     = 0028  S2     = 0044  SUB16 = 0030

  SUB16.MAC,SUB16.LST=SUB16.MC8
```

Рис. 3. Листинг программы вычитания 16-битовых чисел

```
УМС   #A    #W    R0  R1  R2  R3  R4  R5  R6  R7   I  F0  F1  TF  T1   P0  P1  P2  T
0020  00  00001001  06  09  02  04  00  00  00  00   0   0   0   0   0   FF  00  00  00
0038  02  00001001  06  09  02  04  00  00  00  00   0   0   0   0   0   FF  00  00  00
0031  FD  00001001  06  09  02  04  00  00  00  00   0   0   0   0   0   FF  00  03  00
0032  FE  00001001  06  09  02  04  00  00  00  00   0   0   0   0   0   FF  00  00  00
0034  FE  00001001  06  09  02  04  FE  00  00  00   0   0   0   0   0   FF  00  00  00
0035  04  00001001  06  09  02  04  FE  00  00  00   0   0   0   0   0   FF  00  00  00
0036  FB  00001001  06  09  02  04  FE  00  00  00   0   0   0   0   0   FF  00  03  00
0037  FB  00001001  06  09  02  04  FE  00  00  00   0   0   0   0   0   FF  00  06  00
0039  FB  00001001  06  09  02  04  FE  FB  00  00   0   0   0   0   0   FF  00  00  00
003A  FE  00001001  06  09  02  04  FE  FB  00  00   0   0   0   0   0   FF  00  00  00
  C    A     W     R0  R1  R2  R3  R4  R5  R6  R7   I  F0  F1  TF  T1   P0  P1  P2  T
0038  04  11001001  06  09  02  04  FE  FB  00  00   0   0   0   0   0   FF  00  00  00
003C  04  11001001  06  09  02  04  04  FB  00  00   0   0   0   0   0   FF  00  00  00
003D  FB  11001001  06  09  02  04  04  FB  00  00   0   0   0   0   0   FF  00  00  00
003E  05  11001001  06  09  02  04  04  FB  00  00   0   0   0   0   0   FF  00  00  00
003F  05  11001001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0040  05  11001001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0041  05  11001001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0043  05  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0044  05  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0022  05  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
  C    A     W     R0  R1  R2  R3  R4  R5  R6  R7   I  F0  F1  TF  T1   P0  P1  P2  T
0028  04  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  00  00  00
0029  04  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  04  00  00
002A  05  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  04  00  00
002B  05  11101001  06  09  02  04  04  05  00  00   0   0   0   0   0   FF  04  05  00
```

Рис. 4. Файл трассировки программы вычитания 16-битовых чисел

создание файла трассировки, отражающего состояние счетчика команд, всех внутренних регистров, слова состояния программы, флагов, тестируемых входов, портов ввода-вывода;

подсчет времени исполнения программы и количества машинных циклов ОЭВМ.

В качестве примера показаны листинг программы пользователя (рис. 3), файл трассировки этой программы (рис. 4) и диагностика ошибок (рис. 5).

Таким образом, разработанный комплекс программных средств, выявляя ошибки на ранней стадии проектирования, позволяет отлаживать программы пользователя до изготовления реальных микропроцессорных систем на базе ОЭВМ КМ1816ВЕ48.

В ленинградском СКБ прецизионного станкостроения создан пакет программного обес-

```
МПС 1816

АДР.  КОД             ТЕКСТ ПРОГРАММЫ

                1  ;ПРОГРАММА ВЫЧИТАНИЯ 16-БИТОВЫХ ЦЕЛЫХ ЧИСЕЛ
                2  ;R1,R0-СТ. И МЛ. ЧАСТИ УМЕНЬШАЕМОГО
                3  ;R3,R2-СТ. И МЛ. ЧАСТИ ВЫЧИТАЕМОГО
                4  ;    РЕЗУЛЬТАТ -- ПОРТ1;ПОРТ2
                5  ;
                6       AA=20H
                7       .=AA
0020  1430        8       CALL SUB16
0022  8626        9       JF0 S1          ;ПРОВЕРКА ОШИБКИ
0024  BC00       10       MOV R4,#0       ;
0026  BD30       11       MOV R5,#0       ;ПРИ ОШИБКЕ = ОБНУЛЕНИЕ РЕЗУЛЬТАТА
**********  ОШИБКА 13 **********
0028  FC         12  S1:  MOV A,R4        ;
0029  39         13       OUTL P1,A       ;РЕЗ. МЛ. ВЫВЕСТИ В ПОРТ 1
002A  FD         14       MOV A,R5        ;
**********  ОШИБКА  3 **********
002B  3A         15       OUTL P2,A       ;РЕЗ. СТ. ВЫВЕСТИ В ПОРТ 2
                16                       ;
                17       .=AA+10H
0030  FA         18  SUB16:MOV A,R2       ;ВЗЯТЬ МЛАДШУЮ ЧАСТЬ ВЫЧИТАЕМОГО
0031  37         19       CPL A           ;  ПОЛУЧИТЬ ДОПОЛНЕНИЕ
0032  0321       20       ADD A,#1        ;  МЛАДШЕЙ ЧАСТИ
**********  ОШИБКА  2 **********
0033  AC         21       MOV R4,A        ;  УМЕНЬШАЕМОГО
0034  FB         22       MOV A,R3
0035  37         23       CPL A           ;  ПОЛУЧИТЬ ДОПОЛНЕНИЕ
0036  1300       24       ADDC A,#0       ;  СТАРШЕЙ ЧАСТИ
0038  AD         25       MOV R5,A        ;  УМЕНЬШАЕМОГО
                26  ;  СЛОЖИТЬ
                27  ;  РЕЗУЛЬТАТ НА R4,R5
0039  FC         28       MOV A,R4
003A  68         29       ADD A,R0
003B  AC         30       MOV R4,A        ;МЛ. ЧАСТЬ РЕЗУЛЬТАТА
003C  FD         31       MOV A,R5
003D  79         32       ADDC A,R1
003E  AD         33       MOV R5,A        ;СТ. ЧАСТЬ РЕЗУЛЬТАТА
003F  85         34       CLR F0
                35       JNC S2          ;ПРОВЕРКА ЗНАКА РЕЗУЛЬТАТА
**********  ОШИБКА  9 **********
0041  95         36       CPL F0          ;  РЕЗ.<0, ЕСТЬ ОШИБКА (F0=1)
                37                       ;  РЕЗ.>0, НЕТ ОШИБКИ (F0=0)
0042  83         38  S2:  RET
                39       END
 ОБНАРУЖЕНО 4 ОШИБКИ

      ***  ТАБЛИЦА СИМВОЛОВ  МПС 1816  ***
AA     = 0020  MV     = ****  S      = 0043  S1     = 0026  S2     = ****
SUB16 = 0030
```

Рис. 5. Пример диагностики ошибок при трансляции

В предлагаемых вниманию читателей двух статьях раскрываются некоторые особенности аппаратных и программных методов реализации средств отладки для микроконтроллеров.

УДК 681.326

В. И. Лобанов

АРХИТЕКТУРА ОТЛАДОЧНЫХ СРЕДСТВ ДЛЯ МИКРОКОНТРОЛЛЕРА

Средства отладки микроконтроллерного устройства (МКУ) делятся на три класса: отладочные устройства, отладочные системы на базе вычислительной техники класса персональных компьютеров и отладочное программное обеспечение, поставляемое отдельно.

Отладочные устройства (ОУ) характеризуются прежде всего автономностью, низкой стоимостью, малыми габаритными размерами. В связи с этим ОУ присущи следующие преимущества:

нет необходимости в дефицитных и дорогостоящих микроЭВМ класса персональных компьютеров (ПК);

как следствие, снимается проблема высококвалифицированного обслуживания вычислительной техники и периферийного оборудования;

при создании комплекса ОУ — вычислительная техника последняя используется только для трансляции программ пользователя и «перекачки» загрузочных модулей в ОЗУ программ ОУ, процесс отладки полностью автономен;

при обучении персонала программированию микроконтроллера (МК) экономится машинное время;

высокая надежность связана не только с простотой, но и с дополнительной возможностью резервирования.

Требования, предъявляемые к ОУ: обеспечение реального масштаба времени; относительная универсальность; наличие канала связи с ПК; обеспечение автоматического и пошагового режимов, а также режима контрольных точек; выведение на индикацию содержимого внутренних регистров, реализация на элементной базе с малым энергопотреблением. Этим требованиям удовлетворяет архитектура ОУ, представленная на рис. 1.

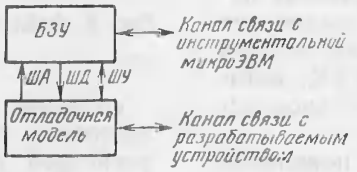


Рис. 1. Архитектура отладочного устройства:
БЗУ — блок загрузки и управления

Отладочная модель (ОМ) обеспечивает реальновременное выполнение программ, а также создает канал связи с устройством пользователя. Канал позволяет (если нет еще макета МК) набирать физическую модель МК из отдельных элементов: клавиатуры, индикаторов, датчиков, ЦАП и т. п. ОМ строится на том типе микропроцессора или микроЭВМ, на котором разрабатывается МК, поэтому ОМ является сменным блоком ОУ. ОМ [1] и БЗУ связаны между собой шинами адреса (ША), данных (ШД) и управления (ШУ).

Блок загрузки и управления осуществляет запись и чтение программ пользователя, обеспечивает вышеперечисленные режимы выполнения программ и индикацию содержимого внутренних регистров. Через БЗУ осуществляется связь с ПК.

В структурной схеме БЗУ (рис. 2) *блок ввода информации и установки режимов* (БВИУР) служит для ввода 16-ричных данных и адресов в буфер адресов и данных (БАД), а также для установки режимов РУЧН или АВТ и МП (работа с отладочной моделью) или ЗАГР (работа в режиме загрузки). Этот блок осуществляет очистку от дребезга сигналов от клавиатуры и тумблеров.

БАД преобразует последовательный 16-ричный код в 4-разрядный параллельный 16-ричный. При каждом нажатии цифровой клавиши вырабатывается импульс ИН, который продвигает и обновляет информацию в БАД. Выходы БАД образуют шину буфера (ШБ).

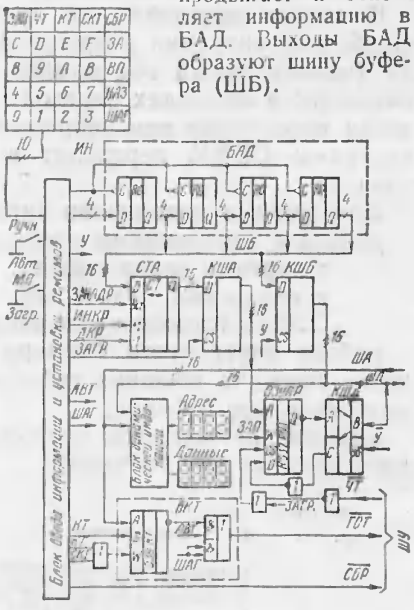


Рис. 2. Структурная схема БЗУ:
0....9, A...F — цифровые 16-ричные клавиши; ЗАП — запись в ОЗУ; ЧТ — чтение ОЗУ; КТ — установка и сброс КТ в ОЗУКТ; СБР — общий сброс для ОМ; ЗА — занесение адреса СТА; ВП и НАЗ — соответствующий просмотр ОЗУ программ (ОЗУПР); ШАГ — выполнение одной команды в режиме МП

печения разработки и отладки программ для ОЭВМ КМ1816ВЕ48. С его помощью отрабатываются программы управления реальными объектами, в частности для контроллера (на базе ОЭВМ) управления следящим приводом.

С запросами о приобретении технической документации обращаться по адресу: 197110, Ленинград, Большая Разночинная, 14; начальнику СКБ ПС.

ЛИТЕРАТУРА

1. Кобылинский А. В., Липовецкий Г. П. Однокристальные микроЭВМ серии КМ1816.— Микропроцессорные средства и системы, 1986, № 1, с. 10—19.
2. Иванов В. И., Лобанов В. И., Митрофанов А. В. Отладочные средства для малоразрядных однокристальных микроЭВМ.— Микропроцессорные средства и системы, 1984, № 2, с. 42—45.
3. Крылов Е. И. Однокристальные микроЭВМ серий К1814, К1820, К1816.— Микропроцессорные средства и системы, 1985, № 2, с. 3—7.

Статья поступила 23 сентября 1985 г.

Счетчик адреса (СТА) инкрементирует или декрементирует (в зависимости от состояния клавиш ВП и НАЗ) свое содержимое при каждом нажатии клавиш ЗАП, ЧТ, КТ, СКТ. По сигналу занесения адреса (ЗАП, АДР.) в СТА переписывается информация из БАД.

Коммутатор шины адреса (КША) подключает выходы СТА к ША только в режиме ЗАГР. *Коммутатор шины буфера* (КШБ) при наборе адреса или данных подключает БАД к ШД. Это позволяет контролировать вводимую информацию. Признаком обновления информации является сигнал У, вырабатываемый в БВИУР.

Блок динамической индикации [2] с 7-сегментными индикаторами контролирует состояние ША и ШД.

Операционное запоминающее устройство программ (ОЗУПР) хранит программу пользователя; в случае применения микросхем К537РУ3 или К537РУ5 отпадает необходимость в *коммутаторе шины данных* (КШД), осуществляющем переключение ШД на запись или чтение. ОЗУПР для ОМ доступно только в режиме чтения. Это повышает надежность в режиме МП.

Блок контрольных точек (БКТ) служит для записи по всему адресному пространству программы пользователя информации о наличии контрольных точек. В соответствии с этим вырабатывается сигнал готовности для ОМ. БКТ реализован на БИС ОЗУ с одноразрядной организацией. Все устройство реализовано на КМОП ИС.

Из рис. 2 видно, что аппаратный БЗУ не позволяет индицировать информацию о состоянии внутренних регистров. Для некоторых микропроцессоров, например КР580ИК80, с этим можно мириться. Достоинство аппаратной реализации БЗУ — простота и универсальность. Связь с инструментальной ЭВМ осуществляется по ША, ШД и ШУ через интерфейсную плату И2. Данная архитектура выполнена в отладочном устройстве «Электроника ОУ-04» для микроЭВМ К1814ВЕ2. БЗУ можно реализовать и на БИС К1011ВГ101. Разработаны сменные ОМ на базе КР580ИК80, К1816ВЕ48 и К1814ВЕ2.

Аппаратно-программная реализация БЗУ представлена на рис. 3. Достоинства устройства: практически неограниченные функциональные возможности; позволяет наращивать набор функций путем введения команды Пуск (адрес новой функции); добавлением еще одной БИС КР580ИК55 можно ввести функции программатора, накопителя на бытовом кассетном магнитофоне и т. д. Недостатки: три источника питания и высокое энергопотребление. Они могут быть устранены использованием БИС серии К588 или К1806.

Существует еще один тип архитектуры автономного аппаратно-программного отладочного устройства без разделения на БЗУ и СМ. Процессор выполняет поочередно функции монитора и физического моделирования. Эта архитектура компактна, но в ней нет универсальности: программное и аппаратное обеспечение для каждого типа МП создается заново; программы монитора и отладчика приходится разрабатывать на языках зачастую малоэффективных. По такому принципу разработано отладочное устройство «Электроника ОУ-48» для микроЭВМ КМ1816ВЕ48.

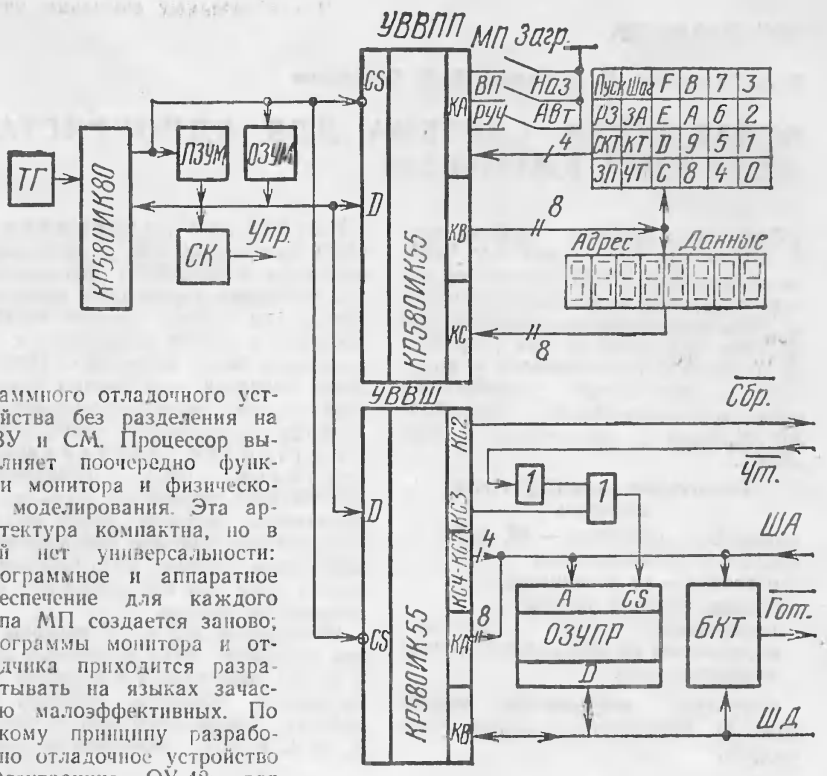


Рис. 3. БЗУ на основе КР580ИК80: СК — системный контроллер: УВВПП — устройство ввода-вывода передней панеля УВВШ — устройство ввода-вывода шины адреса, данных, управления

Если БЗУ (см. рис. 3) выполнить на базе персонального компьютера и дополнить ОУ кросс-ассемблером, то получится отладочная система, описанная в [3]. Преимущества такой отладочной системы очевидны. Кроме того, на ПК можно поставить контрольно-диагностические средства, а также обеспечить техническое документирование.

Особого внимания заслуживают отладочные средства для микроЭВМ К1814, К1820 и т. д., не имеющие входа типа Готов. В этом случае резко возрастает сложность отладочной модели. Для таких ЭВМ предпочтительнее осуществлять математическое моделирование. Отладочная система такого средства [4] не имеет аппаратного обеспечения, что позволяет вести разработку такой системы силами «чистых» программистов.

Возможен и промежуточный вариант архитектуры, когда отладочная система [4] соединяется с разрабатываемым МКУ через простой блок сопряжения (6 корпусов СИС), заканчивающимся сокетом. Это позволяет отлаживать разрабатываемое МКУ, хотя и без соблюдения реального масштаба времени. По данной архитектуре реализована отладочная система для микроЭВМ серии К1820.

В настоящее время разработаны и эксплуатируются отладочные устройства типа «СЭМ» и отладочные системы типа «МИКРОСОТ». Они охватывают широкий класс микропроцессоров и микроЭВМ. Архитектура этих средств подробно представлена в [5] и несколько отличается от вышеописанных типов структур отладочных средств.

Разработчики МКУ могут в соответствии со своей квалификацией выбрать ту или иную архитектуру отладочных средств. Необходимо отметить, что поиск оптимальных структур средств отладки продолжается, поэтому предлагаемые решения не следует считать обязательными. Наиболее перспективной является архитектура [3], базирующаяся на ПК и обеспечивающая выполнение следующих требований: реальный масштаб времени; универсальность программного ядра, реализующего запись-чтение ОЗУ программ пользователя, функции программатора, хранение содержимого ОЗУПР на магнитном носителе и т. п.; реализация автоматического и пошагового режимов, а также режима контрольных точек; выведение на дисплей содержимого внутренних регистров, трассировка программ с дизассемблированием; наличие кросс-ассемблера; возможность использования развитого программного обеспечения и периферийного оборудования ПК.

УДК 681.3.06 : 326

Р. А. Евлампиев, Е. В. Галузо, В. П. Голованов

ОТЛАДОЧНАЯ СИСТЕМА ДЛЯ ОДНОКРИСТАЛЬНОЙ МИКРОЭВМ КМ1816ВЕ48

Отладочная система на базе микроЭВМ «Электроника 60» или ДВК (в дальнейшем ЭВМ) состоит из пакета программ и отладочного модуля. Она предоставляет пользователю широкие возможности при разработке и отладке программного и аппаратного обеспечения устройств на базе однокристальной микроЭВМ КМ1816ВЕ48 (в дальнейшем микроЭВМ).

Технические характеристики системы:

объем ОЗУ программ — 4К байт;
число контрольных точек
останова — не ограничено;
основные режимы работы:
пошаговый
с остановом на контрольных точках
автоматический

Структура отладочного модуля (рис. 1). Рассмотрим основные узлы модуля.

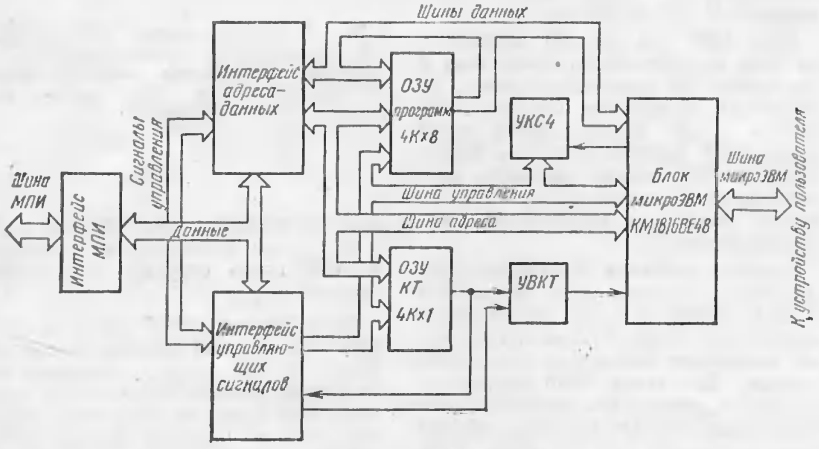


Рис. 1. Отладочный модуль

Интерфейс МПИ обеспечивает связь между каналом ЭВМ и аппаратурой модуля. Интерфейс собран на микросхемах средней степени интеграции.

Интерфейс адреса-данных (АД) (универсальный двунаправленный порт КР580ИК55) используется для установки адреса ОЗУ программ, записи или чтения данных из ОЗУ программ и чтения информации с шины адреса блока микроЭВМ. Направление передачи информации изменяется путем перепрограммирования регистра управляющего слова (РУС).

Интерфейс управляющих сигналов (УС) (микросхема КР580ИК55) формирует сигналы, управляющие работой всего модуля. Направление передачи информации не изменяется, поэтому РУС программируется один раз при начальной инициализации модуля.

Интерфейсы АД и УС связаны через интерфейс МПИ с каналом ЭВМ и имеют свои адреса в старшем банке памяти (область внешних устройств). Таким образом, к портам А, В, С и РУС интерфейсов можно обратиться программно, как к ячейкам памяти ЭВМ.

В ОЗУ программ (восемь микросхем КР537РУ2, что по объему составляет 4К байт) можно записывать или считывать программы, подготовленные для отладки на микроЭВМ. Интерфейс АД занимает шины ОЗУ программ, только когда на них отсутствуют сигналы от блока микроЭВМ. Очередность захвата шин обеспечивается программным путем с помощью интерфейса УС.

В ОЗУ контрольных точек (КТ) (микросхема КР537РУ2) шина адреса соединена с шиной адреса ОЗУ программ и блока микроЭВМ, а сигнал Запись КТ формируется интерфейсом УС отдельно от сигнала записи информации в ОЗУ программ. Поэтому можно устанавливать и сбрасывать КТ по всему объему адресного пространства, не меняя информации в ОЗУ программ. Сигнал с выхода ОЗУ КТ поступает на узел выделения контрольной точки УВКТ и на один из входов интерфейса УС. УВКТ формирует импульс для останова микроЭВМ, которая переходит в состояние ожидания. На шине адреса в этот момент находится адрес следующей команды. Если после этого из интерфейса УС придет сигнал продолжения, то микроЭВМ продолжит свою работу до следующей КТ.

Через узел коммутации сигналов чтения (УКСЧ) подается сигнал чтения либо от интерфейса УС для чтения информации из ОЗУ программ, либо от блока микроЭВМ для чтения команды из ОЗУ программ. Сигнал записи поступает в ОЗУ программ только от интерфейса УС, так как микроЭВМ не имеет возможности записывать информацию в ОЗУ программ.

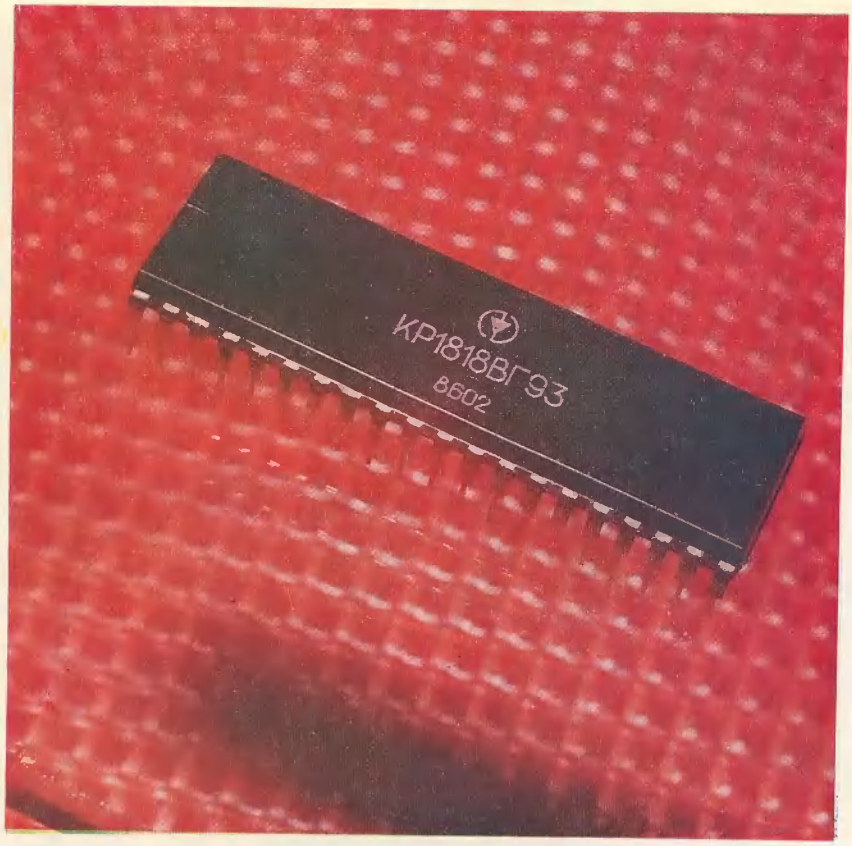
Блок микроЭВМ (рис. 2) построен на базе однокристальной микроЭВМ КМ1816ВЕ48 (D3). В состав блока также входит ряд микросхем серии К561 для формирования сигналов, необходимых при работе микроЭВМ.

Для отладки программного обеспечения и проверки правильности работы ее совместно с аппаратурой не-

ЛИТЕРАТУРА

1. Иванов В. И., Лобанов В. И., Митрофанов А. В. Отладочные средства для малоразрядных однокристальных микроЭВМ.— Микропроцессорные средства и системы, 1984, № 2, с. 42—45.
2. Ланцов А. Л. и др. Цифровые устройства на комплементарных МДП интегральных микросхемах.— М.: Радио и связь, 1983.— 261 с.
3. Евлампиев Р. А., Галузо Е. В., Голованов В. П. Отладочная система для однокристальной микроЭВМ. — Микропроцессорные средства и системы, 1986, № 3, с. 32—33.
4. Зобин Г. Я. и др. Система подготовки и отладки программ для однокристальной микроЭВМ КМ1816ВЕ48. — Микропроцессорные средства и системы, 1986, № 3, с. 27—30.
5. Иванов Е. А., Муренко Л. Л., Широков Ю. Ф. Универсальная отладочная система автоматизации проектирования микропроцессорных устройств.— Микропроцессорные средства и системы, 1984, № 3, с. 53—57.

Статья поступила 1 апреля 1985 г.



КР1818ВГ93

Микроконтроллер накопителя на гибком магнитном диске

управляет выводом информации из ЭВМ на гибкие магнитные диски и вводом информации с дисков в ЭВМ;

обеспечивает программирование номеров дорожки, сектора и стороны диска, длины сектора;

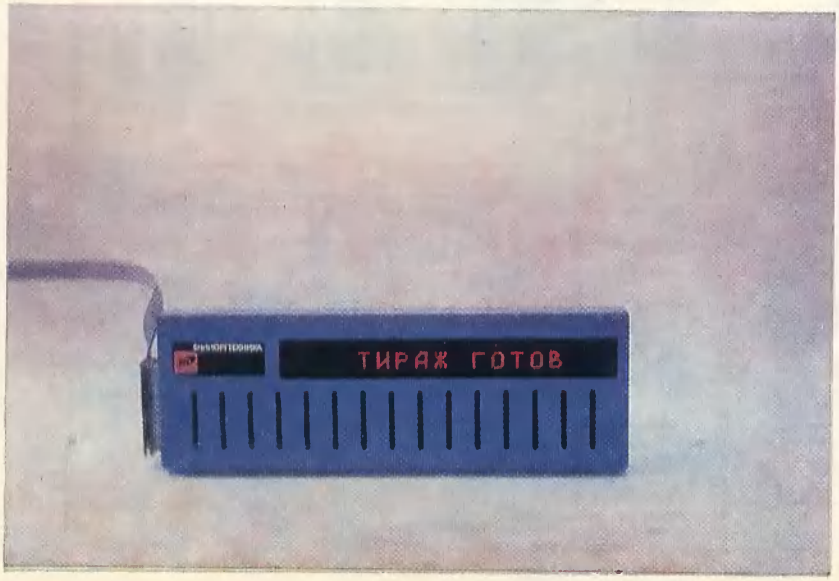
задает режимы поиска дорожки и установки магнитной головки в исходное положение, режимы чтения и записи информации, скорость перемещения магнитной головки.

Основные технические характеристики контроллера

Скорость обмена информацией, Кбит/с, не более	
одинарная плотность записи	250
двойная плотность записи	500
Тактовая частота, МГц	
ГМД размером 133 мм	1
ГМД размером 203 мм	2
Потребляемая мощность, мВт, не более	500
Число выполняемых команд	11
Напряжение источника питания, В	5, 12
Тип корпуса	2.123.40-2 (40-выводной, DIP)

Развитая система команд и сигналов управления позволяет применять контроллер в вычислительных устройствах и системах с НГМД различной сложности.

Контроллер используется в персональной ЭВМ «Электроника МС 0585».



ПУЛЬТОВОЙ ДИСПЛЕЙ

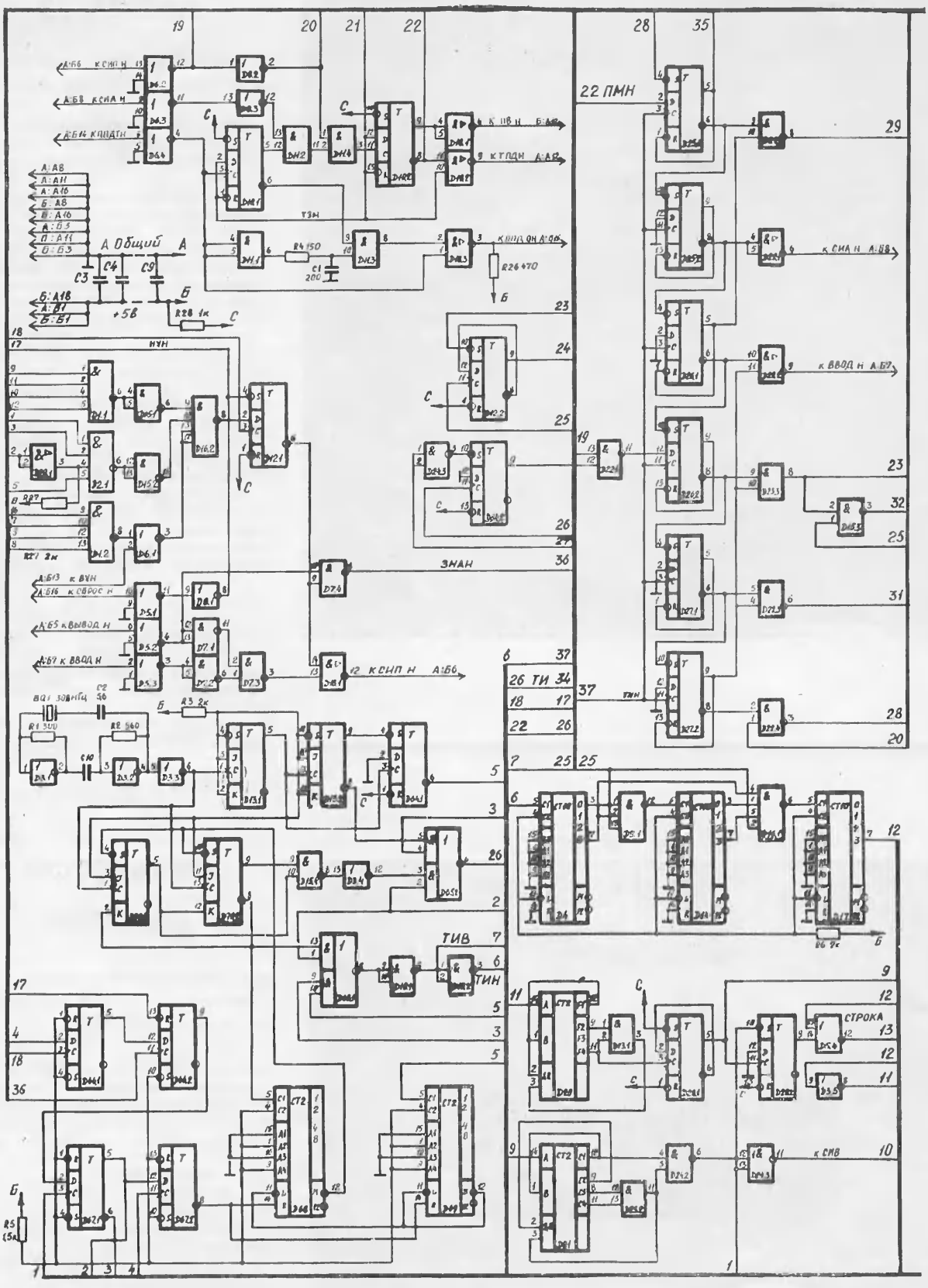
для микроконтроллеров с системой команд микропроцессора КР580ИК80А

Простое, доступное в изготовлении устройство отображения сравнительно небольших объемов алфавитно-цифровой информации.

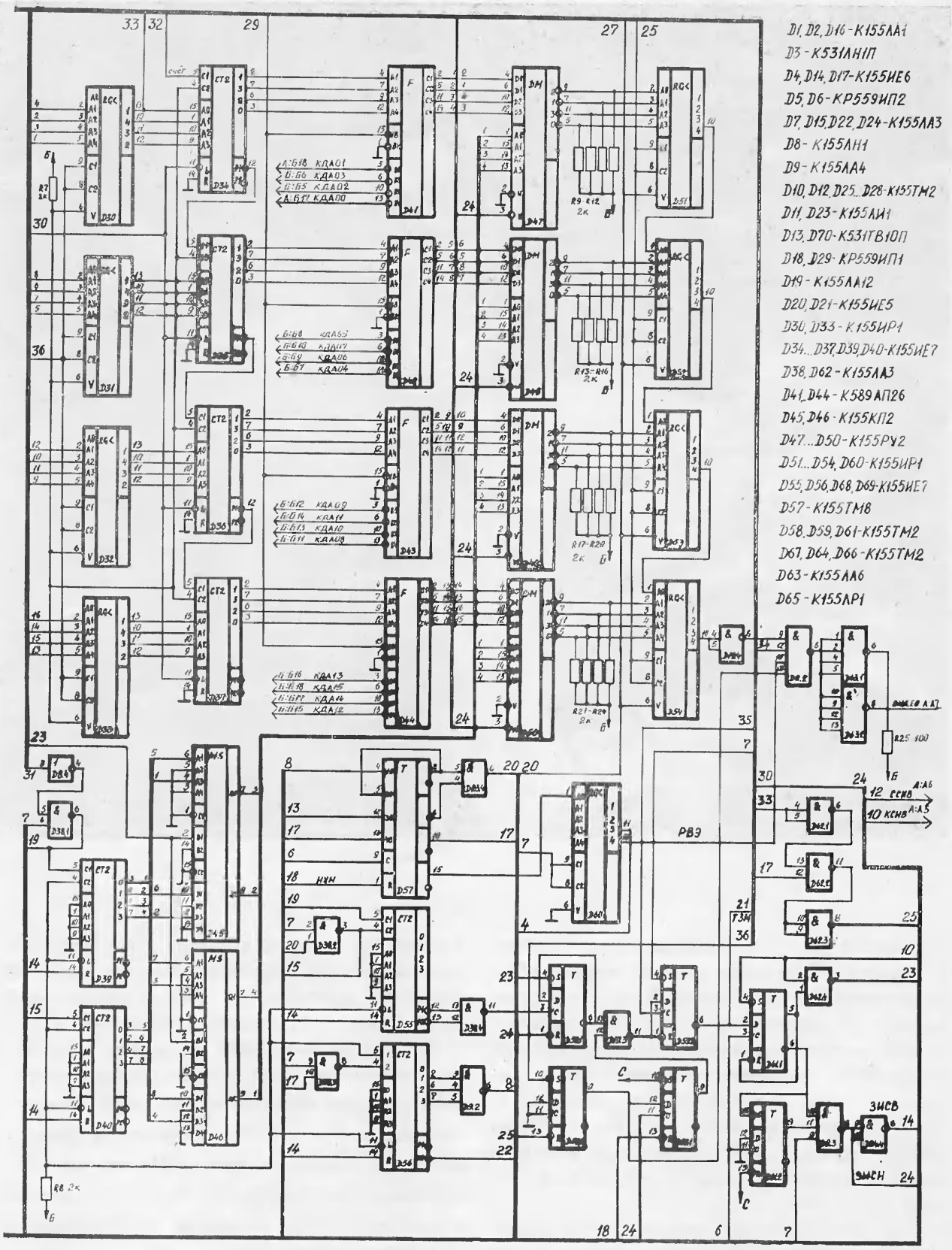
Обеспечивает четыре режима индикации:

— непрерывный;
— мигающий;
— «бегущая строка»;
— попеременный вывод пары сообщений.

Не требует дополнительных ресурсов микропроцессорной системы.



Принципиальная схема регенератора (к ст. М. М. Ляпунова и Б. И. Белякова на с. 43)



D1, D2, D16 - К155ЛА1
D3 - К531ЛН1П
D4, D14, D17 - К155ИЕ6
D5, D6 - КР559ИП2
D7, D15, D22, D24 - К155ЛА3
D8 - К155ЛН1
D9 - К155ЛА4
D10, D12, D25...D28 - К155ТМ2
D11, D23 - К155ЛИ1
D13, D70 - К531ТВ10П
D18, D29 - КР559ИП1
D19 - К155ЛА12
D20, D21 - К155ИЕ5
D30, D33 - К155ИР1
D34...D37, D39, D40 - К155ИЕ7
D38, D62 - К155ЛА3
D41, D44 - К589АП26
D45, D46 - К155КП2
D47...D50 - К155РУ2
D51...D54, D60 - К155ИР1
D55, D56, D68, D69 - К155ИЕ7
D57 - К155ТМ8
D58, D59, D61 - К155ТМ2
D67, D64, D66 - К155ТМ2
D63 - К155ЛА6
D65 - К155ЛР1

В школе № 314 Куйбышевского района г. Москвы оборудован кабинет информатики и вычислительной техники. В кабинете установлено 12 ученических микроЭВМ типа ДВК-1 и две учительских микроЭВМ типа ДВК-2М. МикроЭВМ соединены в простейшую локальную сеть, которая обеспечивает передачу программ на языке Фокал с диска ДВК-2М в память ДВК-1 и обратно, вывод на печать текстов программ и результатов расчета.

С января 1986 года в этом кабинете проводятся практические занятия для учащихся 9-х классов по курсу «Основы информатики и вычислительной техники». Здесь же проходят 8-часовой практикум по программированию и другие школы Куйбышевского района.

Хотя микроЭВМ ДВК-1 оснащены лишь интерпретатором языка Фокал, программирование для них ведется, по сути дела, на алгоритмическом языке. О том, как этого удалось добиться, читайте в статье «Метод программирования на Бейсике и Фокале на основе алгоритмического языка» З. М. Штильман и Б. М. Штильмана.

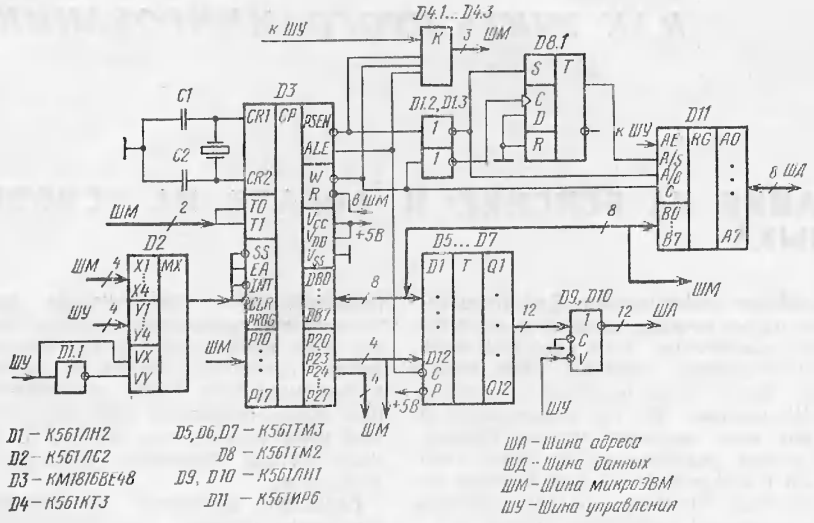


Рис. 2. Блок микроЭВМ КМ1816ВЕ48

обходимо микроЭВМ подключить непосредственно к шине (ШМ). Таким образом, процессор блока микроЭВМ является одновременно и процессором разрабатываемого микроконтроллера. Сигналы микроЭВМ не буферизованы. Это позволяет контролировать нагрузку на шины микроЭВМ со стороны пользователя. Чтобы в процессе ввода и вывода содержимого внутренних регистров не изменить состояние схемы устройства пользователя, три выходных сигнала (W ALE и PSEN) снимаются не с выводов микроЭВМ, а через КМОП-ключи (D4). Кроме того, четыре входных сигнала проходят в микроЭВМ через коммутатор (D2). При такой коммутации входные управляющие сигналы подаются либо программно через интерфейс УС, либо непосредственно от устройства пользователя.

Адрес, формируемый микроЭВМ для чтения команды, фиксируется на буферном регистре (D5...D7). Шина адреса освобождается через буфер D9, D10, так как выходы регистра не могут переключаться в высокоимпедансное состояние.

Прием и выдача информации из микроЭВМ осуществляется через многофункциональный регистр (D11), служащий для ввода команды из ОЗУ программ в микроЭВМ, вывода содержимого аккумулятора и для отключения сигналов DB0...DB7 микроЭВМ от шины данных (ШД). С помощью подпрограммы ввода-вывода запоминается содержимое счетчика команд микроЭВМ и участка ОЗУ программ (с адреса 000 до 0020), записывается программа чтения или модификации внутреннего регистра и запускается на выполнение. После выполнения подпрограммы восстанавливается содержимое счетчика команд и области ОЗУ программ с адреса 000 до 0020.

При чтении содержимое регистра через аккумулятор микроЭВМ передается в регистр D11, затем считывается через интерфейс АД в ЭВМ. При модификации новое содержимое заносится непосредственно во внутренний регистр. В ОЗУ программ заносится команда «Пересылка константы в регистр», и микроЭВМ запускается на выполнение этой команды.

Устройство пользователя подключается к шине микроЭВМ с помощью жгута, на конце которого распаивается 40-выводной разъем.

При отладке аппаратной части устройства пользователя микроЭВМ работает сначала в пошаговом режиме. При выполнении команд ввода-вывода проверяется правильность прохождения сигналов в устройстве. После этого участки подпрограмм ввода-вывода выполняются в автоматическом режиме. Работа аппаратуры проверяется в реальном масштабе времени.

Пакет программ отладочной системы. В состав пакета входят кросс-ассемблер для КМ1816ВЕ48, программа преобразования и программа-монитор.

Кросс-ассемблер транслирует исходный текст программы, формирует листинг и объектный файл. В системе может применяться любой кросс-ассемблер для данной микроЭВМ, если он формирует стандартный объектный модуль (стандарт фирмы INTEL).

Пример:

```
: 100000002345D48754DE90646461118712341 2DA10
: 040010001256EFF5D6
: 00000001FF
```

Этот модуль можно распечатать и скорректировать как обычный текстовый файл.

Программа преобразования. Эта программа преобразует символьные коды объектного модуля в двоичные коды команд микроЭВМ и формирует файл, загружаемый затем в ОЗУ программ отладочного модуля. Если в системе используется кросс-ассемблер, который сразу создает файл в формате отображения памяти, то процесс преобразования исключается.

Программа монитор выполняет следующие функции:

загрузка программы из файла с гибкого магнитного диска в ОЗУ программ модуля микроЭВМ;

запись отлаженной программы из ОЗУ программ на гибкий диск в формате отображения памяти;

чтение и модификация любой ячейки ОЗУ программ;

запись (стирание) контрольных точек останова по любому адресу ОЗУ программ (максимальное число — 4096 точек, т. е. в каждом адресе точка останова);

чтение и модификация содержимого всех регистров микроЭВМ (включая триггер банка памяти);

чтение и модификация содержимого внутреннего и внешнего ОЗУ (если используется внешнее ОЗУ);

запись на гибкий диск и считывание с него содержимого внутреннего и внешнего ОЗУ микроЭВМ;

выполнение программы по шагам с индикацией счетчика команд, кода команды и ее символьного представления на ассемблере после каждого шага или после N шагов;

выполнение программы в автоматическом режиме с остановом на контрольных точках. Можно также задать число пропуска точек останова, что очень удобно при отладке циклов;

выполнение программы в автоматическом режиме;

аппаратный сброс микроЭВМ;

формирование сигнала внешнего прерывания для микроЭВМ;

выбор памяти программ (микроЭВМ может работать с ОЗУ программ, установленным в модуле микроЭВМ, и с ПЗУ, установленным в устройстве пользователя).

Многие команды монитора отладочной системы аналогичны командам отладчика ODT в ОС РАФОС.

Таким образом, с помощью отладочной системы пользователь может проверять функционирование аппаратной части разрабатываемого устройства и отлаживать программное обеспечение микроЭВМ КМ1816ВЕ48.

Статья поступила 1 апреля 1985 г.

УДК 681.3.06-519.68

З. М. Штильман, Б. М. Штильман

МЕТОД ПРОГРАММИРОВАНИЯ НА БЕЙСИКЕ И ФОКАЛЕ НА ОСНОВЕ АЛГОРИТМИЧЕСКОГО ЯЗЫКА

С 1985 года в школах страны введен курс «Основы информатики и вычислительной техники». В учебнике для 9-го класса [1], а также в проекте перспективной программы курса на 90-е годы [2] введен специальный неформальный язык для записи алгоритмов — алгоритмический язык (или Е-язык по терминологии работы [3]). Этот язык построен на основе русской лексики, он позволяет отработать методы алгоритмизации, «не привязываясь» к конкретной ЭВМ. Е-язык с помощью минимальных средств поддерживает технологию структурного программирования [4] на этапе алгоритмизации. При переходе к построению программы рекомендуется выполнять программирование путем реализации алгоритмических конструкций в конкретном языке программирования [2]. Однако такой переход затруднен, поскольку конкретные производственные языки программирования имеют ряд недостатков для массовой школы. Они англоязычны, плохо приспособлены для структурного программирования (как, например, распространенные версии Бейсика или Фокала) либо слишком сложны (как, например, Паскаль).

О структурном программировании. В наше время, казалось бы, излишне доказывать необходимость структурного подхода к программированию во всех сферах использования ЭВМ, в том числе в курсе школьной информатики. Опыт свидетельствует об обратном. Поэтому для краткости приведем выдержку из работы крупных специалистов фирмы IBM [5]. «Мы все изучали курс элементарной математики и знаем, что недостаточно получить правильный ответ, надо еще уметь показать ход решения задачи. Объяснение здесь вполне понятно: без этого мы можем угадать ответы на простые задачи, но не сможем решать сложные. И только овладение способом решения простых задач позволит решать более трудные задачи. Структурное программирование обеспечивает нас систематическим методом создания правильных программ, а для его реализации требуется рациональное мышление, а не хитроумные догадки».

О разрыве теории с практикой. Конкретность мышления учащихся требует как можно более раннего использования ЭВМ в качестве исполнителя алгоритмов, приведенных в учебнике информатики. Для этого необходимо соответствующее программное обеспечение. Нам известны лишь две подобные системы. Это прежде всего всеобъемлющая система «Школьница» [6, 7], включающая в себя язык высокого уровня Рапира. Система реализована на ЭВМ типа СМ и микроЭВМ «Агат», которые не получили широкого распространения в школьных кабинетах вычислительной техники. Вторая — высокоинтерактивная система Е-практикум [3] — обеспечивает поддержку алгоритмического языка (Е-языка) в чистом виде, т. е. без ввода-вывода, не позволяя учащимся написать программу, которая ведет диалог с человеком. Система реализована пока лишь на ЭВМ типа СМ.

Таким образом, в отсутствие необходимого программного обеспечения для школьных компьютеров приходится кодировать команды алгоритмического языка средствами Бейсика или Фокала, например, по методике работы [8]. Здесь-то и кроется опасность утраты связей с исходным алгоритмом на алгоритмическом языке, который в данном случае является псевдокодом. Дело в том, что в программе на Бэйсике отсутствует текст алгоритма на псевдокоде, в готовой программе плохо просматриваются конструкции алгоритмического языка (цикл, ветвление, вспомогательный алгоритм), текст программы, как правило, существенно короче текста на псевдокоде. У учителей, а тем более у учащихся возникает искушение писать программы прямо на Бейсике или Фокале, минуя псевдокод, а значит, минуя алгоритм на алгоритмическом языке, утрачивая в итоге дисциплину структурного программирования. Опыт показывает, что именно это и происходит в большинстве школ, оснащенных ЭВМ. В результате при обучении возникает разрыв между теорией — строгим структурным подходом учебника информатики — и практикой программирования на Бейсике или Фокале без какой-либо структурности, но с обилием блок-схем. При этом часто теория приносится в жертву практике. Алгоритмический язык остается языком «безмашинного варианта» обучения, в то время как «машинный вариант» практически порывает с ним.

Основные принципы метода. Выходом из создавшегося положения на переходный период (до широкого распространения программных систем поддержки алгоритмического языка) нам представляется применение метода программирования на основе алгоритмического языка с построчным кодированием на производственный язык (Бейсик или Фокал). В основу метода положены следующие принципы:

единство основных компонентов программирования (проектирования, алгоритмизации, собственно программирования, отладки программ);

структурный подход к программированию на всех этапах разработки программ;

базовый язык — алгоритмический язык из учебника информатики, расширенный средствами ввода-вывода производственного языка;

производственный язык (как средство кодирования) — Бейсик или Фокал в зависимости от того, каким интерпретатором оснащена данная микроЭВМ.

Идея метода довольно проста. Большинство распространенных версий Бейсика и Фокала (и некоторых других языков) допускают запись комментариев в одной строке с выполняемым оператором (правее его), а также в отдельной строке. Это позволяет ввести в программу и сохранить в качестве комментариев ее текст на псевдокоде — алгоритмическом языке. Кроме того, Бейсик и Фокал допускают сокращение многих ключевых слов операторов: Бейсик — до трех первых букв, Фокал — до одной буквы. Это дает возможность сократить в тексте программы ту ее часть (коды), которая мешает естественному восприятию логики программы. Перейдем к описанию предлагаемого метода.

Базовый язык — это расширенная формализация языка учебника, он аналогичен Е-языку [3]. Рассмотрим специфику базового языка при условии, что производственным языком является Фокал. Программа на базовом языке состоит из секций «алгоритмов»: основного и вспомогательных. В отличие от Е-языка в базовом языке заголовок основного алгоритма отличается от заголовков вспомогательных: в нем нет списка формальных параметров, отсутствуют строки АРГ и РЕЗ. Базовый язык включает переменные двух типов: целые (ЦЕЛ) и вещественные (ВЕЩ). Кроме того, он включает целые (ЦЕЛ ТАБ) и вещественные (ВЕЩ ТАБ)

массивы, точнее, одно- и двумерные табличные величины. Если литерные величины поддерживаются в производственном языке, то они, естественно, включаются в базовый язык. В соответствии с синтаксисом Е-языка в базовом языке все переменные явно описываются (в заголовке алгоритма или в строке НАЧ). Имя переменной — произвольная последовательность букв латинского алфавита и арабских цифр, начинающаяся с буквы. При этом надо учитывать, что интерпретаторы производственного языка, как правило, распознают имена по первым двум символам. Поэтому величины, у которых эти два символа совпадают, будут рассматриваться как одна и та же величина.

Арифметические выражения могут включать имена переменных (произвольного типа), целые и вещественные константы (с десятичной точкой вместо запятой), встроенные функции (например, FSIN(X) — sin x, FITR(X) — целая часть и т. п. для Фокала), а также арифметические операции «+», «—», « * », «/» и возведение в целую положительную степень «↑». Конкретный вид функций и знаков операций определяется их допустимостью в конкретном производственном языке. Допустимы два вида скобок «(«,»)», «[«,»]». В качестве индексов табличных величин допустимы произвольные арифметические выражения. При этом в качестве индекса используется целая часть вычисленного значения арифметического выражения, а дробная часть отбрасывается, т. е. A[5/7] трактуется как A[0].

В условиях команд ветвления и повторения [1] содержатся логические выражения. В базовом языке они могут содержать арифметические выражения, связанные знаками отношений «=», «<», «>», «<=» (меньше или равно), «>=» (больше или равно), «<>» (не равно). Логические выражения могут содержать также логические операции И, ИЛИ, НЕ (¬) и скобки «(«,»)». При этом требуется, чтобы каждое элементарное логическое выражение (входящее в состав сложного выражения), содержащее единственный знак отношения, записывалось в отдельной строке. Например,

```
ЕСЛИ A>3 И
  B↑2—4*A*C>0
  ТО
  X1=(-B+FSQT(B↑2-(4*A*C))/2*A
ВСЕ
```

Этот пример демонстрирует также команду присваивания базового языка. В отличие от [1] здесь вместо символа «:=» используется «=». Каждая команда присваивания записывается в отдельной строке.

Команды ввода-вывода базового языка полностью заимствованы из Фокала [9]. С помощью команды TYPE допустим вывод значений отдельных переменных, арифметических выражений и литерных констант. Команда ASK вводит значения переменных, а также выводит литерные константы. Набор простых команд базового языка содержит также специальные команды для работы с табличными величинами TYPETAB (вывод), ASKTAB (ввод) и SETTAB (присваивание).

Синтаксис составных команд базового языка, а также команды вызова вспомогательного алгоритма совпадает с [1]. Однако, как уже отмечалось, программы, записанные на базовом языке, должны исполняться на ЭВМ, оснащенной интерпретатором производственного языка. Поэтому для команд, которые в нем отсутствуют, необходимо кодирование с помощью операторов производственного языка (в данном случае — Фокала).

Кодирование команд. На простых примерах продемонстрируем запись и кодирование простых и составных команд. Приведем программу вычисления функции:

$$y = \begin{cases} x \text{ при } x < 0, \\ 2x \text{ при } x \geqslant 0. \end{cases}$$

```
1.10 C
1.20 C
1.30 ;
1.40 IF(X)1.50, 1.80, 1,80; C
1.50 C
1.60 S
1.70 GO 1.90; C
1.80 S
1.90 C
2.10 ;
2.20 Q
```

Текст программы на базовом языке вводится в ЭВМ с отступом от левого края строки, например с 40-й колонки. Записанные в строках 1.10 и 1.20 заголовок алгоритма и описания переменных кодируются в Фокале буквой С (COMMENT), т. е. рассматриваются как комментарии. Команда ввода ASK воспринимается интерпретатором и кодируется «;» для сохранения отступа от левого края строки.

В Фокале имеется лишь арифметический оператор ветвления IF(A) n_1, n_2, n_3, где A — арифметическое выражение; n_1 — номер строки, куда следует перейти, если A<0; n_2 — номер строки для перехода, если A=0; n_3 — если A>0. С помощью этого оператора кодируются элементарные логические выражения. В данном примере с его помощью закодирована проверка условия X<0 (строка 1.40). Для кодирования команды ветвления (строки 1.40—1.90) использован также оператор GO n — безусловного перехода к строке с номером n. Здесь он кодирует слово ИНАЧЕ для обхода второй альтернативы (GO 1.90). Стоящие в строках 1.60 и 1.80 операторы присваивания кодируются оператором S(SET); после этого они воспринимаются интерпретатором. Ключевые слова ЕСЛИ, ТО, ИНАЧЕ, ВСЕ интерпретатором не воспринимаются и должны быть «закрыты» для него, т. е. кодированы буквой С как комментарии. Команда вывода в строке 2.10 кодируется аналогично команде ввода. Слово КОН кодируется оператором Q (QUIT) — останов программы. Во вспомогательных алгоритмах это слово кодируется оператором R (RETURN) — возврат (см. ниже).

```
АЛГ ФУНКЦИЯ
НАЧ ВЕЩ X, Y
  ASK "X=", X
  ЕСЛИ X<0
    ТО
      Y=X
      ИНАЧЕ
      Y=2*X
  ВСЕ
  TYPE «Y=», Y
КОН
```

Рассмотрим программу вычисления суммы квадратов N первых натуральных чисел.

```
1.10 C
1.20 C
1.30 ;
1.40 S
1.50 S
1.60 IF(I—N)1.70, 2.20, 2.20; C
1.70 C
1.80 S
1.90 S
2.10 GO 1.60; C
2.20 ;
2.30 Q
```

```
АЛГ СУМКВ
НАЧ НАТ N, ЦЕЛ S, I
  ASK "N=", N
  I=0
  S=0
  ПОКА I<N
  НЦ
    I=I+1
    S=S+I ¬2
  КЦ
  TYPE "S=", S
КОН
```

Команда повторения записана в строках 1.60—2.10. Она кодируется с помощью оператора IF — условного перехода на строку, содержащую НЦ, если условие повторения выполнено (здесь строка 1.70), и на строку, следующую после КЦ (здесь 2.20). Слово КЦ кодируется оператором GO — безусловного перехода на строку, содержащую слово ПОКА. Ключевые слова ПОКА, НЦ и КЦ «закрываются» для интерпретатора оператором С.

Кодирование вспомогательного алгоритма. Судя по работам [1] и [10], вызов вспомогательного алгоритма (в Е-языке) в терминах языков программирования соответствует обращению к процедуре с передачей параметров по значению. Его можно рассматривать «как обобщенный оператор присваивания результатам некоторых значений, вычисляемых вспомогательным алгоритмом по текущим значениям, передаваемым аргументам этого алгоритма» [10]. Этим определением мы и воспользовались для кодирования команды вызова вспомогательного алгоритма.

Дело в том, что ни Фокал, ни распространенные версии Бейсика не поддерживают вызов процедуры с параметрами. Присваивая аргументам вспомогательного алгоритма значения фактических параметров при входе в него, а при выходе передавая результаты, мы моделируем такой вызов. Однако для введения процедур надо обеспечить не только передачу параметров, но и аппарат локальных переменных, и возможность рекурсивного обращения. С каждым вызовом вспомогательного алгоритма возникает новый экземпляр памяти алгоритма, образуемый его аргументами, результатами и промежуточными величинами [10]. Моделировать это простыми средствами не удается. Поэтому при записи вспомогательного алгоритма на базовом языке необходимо использовать уникальные имена для параметров и промежуточных переменных, не совпадающие с именами основного алгоритма и других вспомогательных. При необходимости рекурсивного вызова, т. е. вызова самого себя, следует ввести второй экземпляр вспомогательного алгоритма, заменив его имя, имена параметров и промежуточных величин, и вызвать его.

Рассмотрим кодирование вспомогательного алгоритма на примере программы вычисления следующей функции:

$$u(x, y, z)=\frac{\max(x, y)+\max(x+y, xz)}{(\max(0{,}5,\ x+z))^2}.$$

(Эта задача содержится среди упражнений в [11].)

```
1.10 C                                  АЛГ ДРОБЬ
1.20 C                                  НАЧ ВЕЩ X,Y,Z,U,A,B,C
1.30 ;                                    ASK X,Y,Z
1.40 S M=X; S N=Y; DO 2; S A=K; C         MAX (X,Y,A)
1.50 S M=X+Y; S N=X * Z; DO 2; S B=K; C   MAX (Z+Y, X*Z, B)
1.60 S M=0.5; S N=X+Z; DO 2; S C=K; C     MAX (0.5, X+Z, C)
1.70 S                                    U=(A+B)/C ↑2
1.80 ;                                    TYPE "U=", U
1.90 Q                                  КОН
1.95 C - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
2.01 C                                  АЛГ MAX (ВЕЩ M,N,K)
2.05 C                                      АРГ M,N
2.10 C                                      РЕЗ K
2.15 C                                  НАЧ
2.20 IF(M—N)2.25, 2.25, 2.40; C           ЕСЛИ M<N
2.25 C                                      ТО
2.30 S                                      K=N
2.35 GO 2.45; C                             ИНАЧЕ
2.40 S                                      K=M
2.45 C                                    ВСЕ
2.50 R                                  КОН
```

В строках 1.40—1.60 содержатся три вызова вспомогательного алгоритма MAX с различными значениями фактических параметров. При кодировании первого вызова аргументам M и N алгоритма MAX (см. строку 2.05) присваиваем значения фактических параметров X и Y оператором S (SET). Оператором DO 2 осуществляем переход к исполнению группы строк 2. Это строки, у которых целая часть номеров равна 2. В них записан вспомогательный алгоритм MAX. Заголовок алгоритма со списком параметров, списки АРГ и РЕЗ кодируются как комментарии (буквой С). После вызова в строке 1.40 алгоритм MAX исполняется со значениями M и N, равными значениям X и Y. По окончании вычислений оператором R (RETURN) в строке 2.50 осуществляется переход к оператору, следующему за DO в строке 1.40. Здесь — это оператор присваивания S A=K. С его помощью результат вычислений (значение переменной K) присваивается фактическому параметру A. Собственно команда вызова вспомогательного алгоритма MAX (X, Y, A) «закрыта» как комментарий. Вызовы в строках 1.50 и 1.60 кодируются аналогично.

Из-за того, что вызов вспомогательного алгоритма кодируется оператором DO, каждый из вспомогательных алгоритмов должен занимать одну группу строк, т. е. это строки, номера которых имеют одинаковую целую часть. В примере — это строки 2.01—2.50. Тем самым мы получаем ограничение на размер вспомогательного алгоритма: не более 99 строк. В особых случаях, с которыми вряд ли столкнется учащийся, это ограничение нетрудно обойти, используя несколько групп строк и связывая их при кодировании операторами DO.

Кодирование алгоритмов работы с табличными величинами. Продемонстрируем это на примере программы сортировки пузырьком таблицы из 10 чисел.

```
1.10 C                                  АЛГ СОРТИРОВКА
1.20 C                                  НАЧ ВЕЩ ТАБ A[1 : 10], ЦЕЛ I,P, ВЕЩ B
1.30 FOR I=1,10;                          TYPETABI,"A[",I,"]=";  ASKTAB A[I]
1.40 S                                    {P =1
1.50 IF (P—1)3.30, 1.60, 3.30; C          {ПОКА P=1
1.60 C                                    {НЦ
1.70 S                                      P=0
1.80 S                                      I=0
1.90 IF (I—9)2.10, 3.20, 3.20; C            ПОКА I<9
2.10 C                                      НЦ
2.20 S                                        I=I+1
2.30 IF (A[I]—A[I+1])2.90, 2.90, 2.40; C    ЕСЛИ A[I]>A[I+1]
2.40 C                                          ТО
2.50 S                                          B=A[I+1]
2.60 S                                          A[I+1]=A[I]
2.70 S                                          A[I]=B
2.80 S                                          P=1
2.90 C                                      ВСЕ
3.10 GO 1.90; C                             КЦ
3.20 GO 1.50; C                           КЦ
3.30 FOR I=1,10;                          TYPETABI,"A[",I,"]=", A[I]
3.40 Q                                    КОН
```

Роль «пузырька» здесь играет целая величина P, принимающая значения 0 и 1. Ввод таблицы A осуществляется в строке 1.30 с помощью команд TYPETAB и ASKTAB. Первая служит для оформления ввода (выводит номер текущего элемента таблицы), вторая непосредственно осуществляет ввод. Кодируются эти команды с помощью оператора цикла FOR. В результате его исполнения все операторы Фокала, расположенные в одной строке с ним правее его, повторяются. Поскольку интерпретатор Фокала распознает операторы по первой букве, TYPETAB, ASKTAB и SETTAB будут выполняться как обычные TYPE, ASK и SET, но столько раз, сколько требует параметр цикла FOR. Задавая его равным размерности табличной величины (здесь 10), мы обработаем в цикле все элементы таблицы.

Вывод отсортированной таблицы осуществляется командой TYPETAB в строке 3.30 («Знак «!» является символом управления кареткой и вызывает переход на новую строку с каждым выполнением оператора TYPE.)

Технология программирования предусматривает четыре этапа: проектирование алгоритма, составление алгоритма путем последовательного уточнения (без использования блок-схем), собственно программирование, отладка программы.

Рассмотренный метод записи программы на формализованном псевдокоде с построчным кодированием на Фокал не случайно назван методом программирования на основе алгоритмического языка. На всех этапах учащийся мыслит в терминах базового, т. е. алгоритмического языка. Программу записывают на базовом языке в правой части листа, затем кодируют слева. Кодирование выполняется по формальным правилам построчно с отвлечением от смысла кодируемых команд. При некотором опыте и наличии на микроЭВМ экранного редактора текстов можно ввести в ЭВМ программу на базовом языке без кодов, оставив левые части строк свободными, а затем закодировать программу прямо на экране дисплея. Исправление ошибок выполняется прежде всего в тексте программы на базовом языке, а затем формально исправляются коды.

Об универсальности метода и перспективах. Лица, хорошо знакомые с тонкостями Фокала, обратили внимание, что при кодировании команд практически не используются различные «хитрости» этого языка. Это позволило полностью формализовать процедуру кодирования. В таком формальном виде она легко переносится на Бейсик. При этом некоторые правила кодирования даже упрощаются, например, из-за наличия в Бейсике логического оператора IF, оператора GOSUB перехода к подпрограмме (без жестких ограничений на ее размер) и т. п.

При возможном расширении алгоритмического языка [10], например, за счет команды выбора или цикла по параметру эти расширения легко включить в базовый язык и сформулировать соответствующие правила кодирования. Очевидно, метод применим не только для алгоритмического языка. В качестве базового языка можно использовать произвольный структурный псевдокод. В настоящее время ведется отладка экранного редактора текстов для программ на Фокале (для ДВК-1). Сам редактор объемом около 1000 строк написан в соответствии с изложенным здесь методом, закодирован на Фокале, однако в качестве базового языка использована версия языка Э. Дейкстры [12]. В перспективе предполагается превращение редактора в учебный редактор-конвертор базового языка в Фокал. В этом случае отпадет надобность в ручном кодировании. Если же такое кодирование потребуется в учебных целях, то учащийся будет его выполнять под контролем интерактивной системы (редактора-конвертора) подобно работе в среде Е-практикума.

Нам представляется, что и в дальнейшем после разработки и внедрения специализированного программного обеспечения школьного курса информатики описанный здесь метод программирования сохранит свое значение как эффективный и простой метод установления связи между конструкциями алгоритмического и производственного языков.

Опыт реализации метода. В школе № 314 Куйбышевского района г. Москвы оборудован кабинет вычислительной техники на основе микроЭВМ ДВК-1/ДВК-2М. Подобная система описана в работе [13]. С января 1986 г. преподавание информатики в 9-х классах этой школы сочетается с практикой программирования на ЭВМ всех изучаемых в курсе алгоритмов. Работа ведется на основе изложенного здесь метода. На этой же основе проводится факультатив по информатике.

С 17 февраля 1986 г. в этой школе для ряда других школ Куйбышевского района проводится практикум по информатике. Интенсивность занятий довольно высока: практикум состоит из четырех занятий по 2 ч. Применяемый метод позволяет органично перейти от безмашинного варианта изучения информатики к практическому использованию и углублению ранее полученных знаний при программировании на микроЭВМ. Несмотря на то, что учащихся с самого начала предупреждают о том, что ЭВМ оснащена лишь интерпретатором Фокала, у большинства из них остается впечатление, что они программировали на алгоритмическом языке.

ЛИТЕРАТУРА

1. Основы информатики и вычислительной техники. Проб. учеб. пособие в 2-х ч. Ч. 1 / Под ред. А. П. Ершова и В. М. Монахова.— М.: Просвещение, 1985.
2. Программа курса основ информатики и вычислительной техники в IX—X классах (проект) / Под ред. акад. А. П. Ершова. Препринт НИИ СиМО АПН СССР, 1986.
3. Варсанофьев Д. В., Кушниренко А. Г., Лебедев Г. В. Е-практикум — программное обеспечение школьного курса информатики и вычислительной техники.— Микропроцессорные средства и системы, 1985, № 3.
4. Хьюз Дж., Мичтом Дж. Структурный подход к программированию: Пер. с англ.— М.: Мир, 1980.
5. Лингер Р., Миллс Х., Уитт Б. Теория и практика структурного программирования: Пер. с англ.— М.: Мир, 1982.
6. Звенигородский Г. А. Система математического обеспечения, ориентированная на школьный учебный процесс.— Управляющие системы и машины, 1980. № 5.
7. Звенигородский Г. А. Первые уроки программирования.— «Библиотечка «Квант».— М.: Наука, 1985. Вып. 41.
8. Каймин В. А., Питеркин В. М. Основы информатики и вычислительной техники.—М.: МИЭМ, 1985.
9. Фролов Г. И., Куправа Т. А., Чаиров Ю. Г., Капустина Г. Н. Программирование на языке Фокал. Учеб. пособие.— М.: МИЭТ, 1984.
10. Ершов А. П. Алгоритмический язык в школьном курсе основ информатики и вычислительной техники.— Микропроцессорные средства и системы, 1985, № 2.
11. Ускова О. Ф., Горбенко О. Д. Практикум по алгоритмическому языку. Методические разработки.— Воронеж: ВГУ, 1986.
12. Дейкстра Э. Дисциплина программирования: Пер. с англ.— М.: Мир, 1978.
13. Пресняхин Л. Н., Фролов Г. И., Куправа Т. А. и др. Учебный класс на основе диалоговых вычислительных комплексов.— Микропроцессорные средства и системы, 1985, № 3.

Статья поступила 27 марта 1986 г.

УДК 681.325.5

В. Н. Чернов, М. Г. Шестаков, Е. В. Устьян

ИСПОЛЬЗОВАНИЕ БИС СЕРИИ КР1802 ДЛЯ СОЗДАНИЯ АДАПТЕРА МАГИСТРАЛЕЙ СМ ЭВМ И «ЭЛЕКТРОНИКА 60»

Для различных периферийных устройств мини-ЭВМ СМ-4 и «Электроника-100/25» разрабатываются устройства сопряжения, электрические и функциональные параметры которых удовлетворяют требованиям интерфейса «Общая шина» [1].

Минимизировать аппаратурные затраты при создании подобных устройств сопряжения можно, используя ориентированные на интерфейс «Единый канал» микроЭВМ «Электроника 60» БИС обмена информацией КР1802ВВ1 и БИС интерфейса КР1802ВВ2 [2].

Устройство сопряжения (адаптер) из четырех БИС на функциональной схеме (рис. 1) использования БИС обмена информацией (СОИ) КР1802ВВ1 показано условно как один элемент. Такой блок содержит четыре программно доступных регистра форматом 16 бит, применимых для формирования трех двунаправленных шин через каналы БИС (В, С и Х) при организации соответствующих управляющих сигналов.

Один канал БИС (например, А) используется для загрузки данных с канала мини-ЭВМ. Для электрического согласования необходимо использовать блок из четырех микросхем шинного формирователя (ШФ), двунаправленные шины DB которого подключаются к шине данных канала (КД 00...КД 15), а шины для приема D1 и выдачи информации D0 соединяются и образуют внутреннюю двунаправленную шину данных УС. Для ШФ необходимо формировать сигналы выборки С и управления и выдачи информации СЕ.

Первый регистр (адрес 00) СОИ можно использовать в качестве счетчика адреса, наращиваемого по сигналу на вход CI, при режиме обмена массивом. Канал БИС Х можно использовать при организации прямого доступа, так как он обеспечивает электрическое согласование с «Единым каналом». Разряд адреса К Адр 17 необходимо дополнительно формировать как «Лог. 1» (высокий уровень).

Адрес с канала и сигналы К СА и К В/В1 принимаются через блок из пяти приемников (микросхемы КР559ИП2 или К589АП26), что достаточно для обмена словами. Дешифратор адреса (ДА) формирует сигналы управления для канала БИС А (ЕСА — выбор, WA — запись, RA — чтение) и адреса (АА0, АА1) для загрузки любого из четырех регистров.

Для формирования протоколов запроса канала достаточно одной БИС интерфейса (СИ) КР1802ВВ2. На принципиальной схеме использования СИ (рис. 2) уровни и условия формирования управляющих сигналов указаны только для тех входов, применение которых обязательно.

Схемы формирования сигналов пуска прерывания (ПР) и прямого доступа (ПД) строятся в зависимости от конкретного применения.

Сигнал Канал занят (КЗ) формируется, когда устройство сопряжения становится активным.

В протоколе запроса по одному из уровней прерывания используются входы AIN, INR, ASWB и INA. В протоколе запроса на прямой доступ — DAEI, DAR2, ACS и DAEO. На вход CLK подаются тактовые импульсы (ТИ) частотой до 5 МГц.

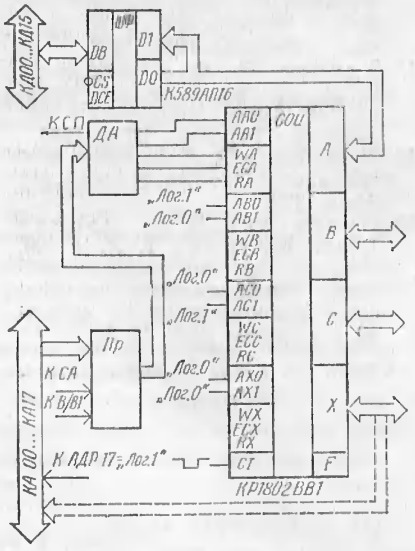


Рис. 1. Функциональная схема использования БИС обмена информацией (СОИ) КР1802ВВ1

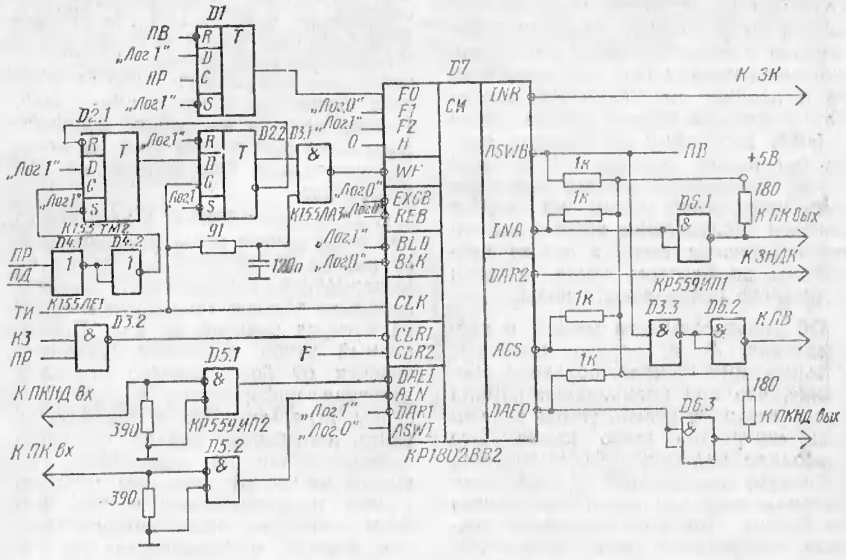


Рис. 2. Принципиальная схема использования БИС интерфейса (СИ) КР1802ВВ2

Сигнал пуска микроинструкции (WF) формируется при поступлении сигналов ПР и ПД и синхронизуется с тактовыми импульсами по фронту на микросхеме D2. Чтобы задержать сигнал WF относительно положительного перепада ТИ на 20... ...30 нс, используется RC-цепочка.

Временна́я диаграмма микроинструкции, выполняемая СИ после поступления WF, зависит от сигнала на входе F0. При запросе по уровню используется триггер D1 (обеспечивает на F0 «Лог. 1», а при запросе на прямой доступ на F0 — «Лог. 0»).

Особенность использования СИ — организация сброса сигнала К ПВ. При наличии сигнала пуска прерывания ПР происходит принудительный сброс СИ по установлению сигнала К3. При прямом доступе сигнал К ПВ должен быть сброшен перед началом последнего обмена. В качестве сигнала сброса пригоден сигнал признака равенства содержимого первого и четвертого регистров СОИ.

Дополнительно для согласования уровней по сигналам К ПК и К ПКНД необходимо использовать входные приемники D5 и выходные передатчики D6. Сигналы К ЗК и К ЗНДК с выхода СИ подаются в «Единый канал». Сигнал К ПВ объединяется по схеме ИЛИ из двух сигналов. Резисторы используются для электрического согласования.

Разработанный авторами адаптер состоит из восьми СОИ, одной СИ и 35 микросхем малой степени интеграции, размещенных на плате размером 240×280 мм. Адаптер имеет шесть программно доступных регистров и четыре двунаправленные системные шины. Формат регистров и шин равен 16 бит. Обмен данными через регистры по системным шинам возможен пословно и при прямом доступе. УС формирует один вектор прерывания по любому из уровней прерывания.

Применение БИС КР1802ВВ1 и БИС КР1802ВВ2 значительно сокращает аппаратурные затраты и упрощает разработку устройств сопряжения, удовлетворяющих требованиям интерфейса «Единый канал».

ЛИТЕРАТУРА

1. Малые ЭВМ и их применение / Под ред. Б. Н. Наумова.—М.: Статистика, 1980.— 231 с.
2. Березенко А. И., Корягин Л. Н., Назарьян А. Р. Микропроцессорные комплекты повышенного быстродействия.— М.: Радио и связь, 1981.— 168 с.

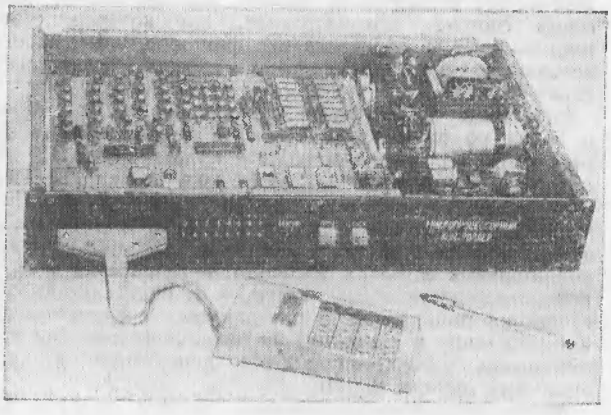
Статья поступила 30 января 1985 г.

УДК 681.323

Ю. А. Акуней, Б. В. Антонов, А. Г. Маликов, Т. В. Марусин, В. Н. Тер-Арутюнов

УНИВЕРСАЛЬНЫЙ МИКРОКОНТРОЛЛЕР «ЭЛЕКТРОНИКА МК-48» НА ОСНОВЕ ОДНОКРИСТАЛЬНОЙ МИКРОЭВМ К1816ВЕ48

Простейшие контроллеры технологического оборудования, имеющие развитый ввод-вывод (до 128 линий), простое управление, понятное оператору, не имеющему опыта работы с вычислительной техникой, а также средства разработки и отладки программ управления технологическим процессом, особенно необходимы там, где нет еще опыта внедрения сложных управляющих систем и технология производства не отвечает требованиям автоматизации. Микроконтроллер «Электроника МК-48» (см. рисунок) может встраиваться прямо в технологическое оборудование. Программа управления технологическим процессом хранится в РПЗУ микроЭВМ К1816ВЕ48. Микроконтроллер имеет 32 независимых дискретных выхода, 16 дискретных и 8 аналоговых входов. Объем управляющих программ составляет 4К байт. Потребляемая мощность не более 15 Вт; габаритные размеры 530×310×85 мм. Однокристальная микроЭВМ К1816ВЕ48 имеет встроенный тактовый генератор, ОЗУ объемом 64 байта, часть которого занимают два банка по восемь рабочих регистров и 8-уровневый стек, 8-разрядный таймер-счетчик, 27 линий ввода-вывода [1]. Применение микроЭВМ позволило ограничиться минимальным числом микросхем (всего 30 микросхем серий К1816, К561, К555, К155, КР580, К573) и разместить микроконтроллер на одной печатной плате размером 330×290 мм, что в значительной мере определило его надежность и невысокую трудоемкость изготовления. Конструктивно контроллер представляет собой блок прямоугольной формы, в котором установлены две печатные платы — блок управления и источник питания. Корпус рассчитан на щитовой монтаж.



Микроконтроллер «Электроника МК-48»

Программы управления в виде кодов команд ОЭВМ записываются пользователем в две микросхемы УФРПЗУ К573РФ2 (К573РФ5), установленные в панельки на плате управления. ПЗУ, расположенное на кристалле ОЭВМ, не используется. Переналадка контроллера для управления новым технологическим процессом заключается лишь в смене ППЗУ с программой управления.

Для повышения качества и надежности программ управления конкретными технологическими процессами, сокращения сроков их разработки, документирования созданных программ может быть использована система автоматизации проектирования программного обеспечения САПР-48 для ОЭВМ К1816ВЕ48 [2]. Она функционирует на ЭВМ типа СМ-4, «Электроника 100/25», «Электроника 60», ДВК-2М в операционной системе Рафос (ОС ДВК, Фодос) и позволяет создавать, модифицировать, ассемблировать исходные модули ПО, отлаживать программы управления технологическим оборудованием на программно-логической модели ОЭВМ, формировать загрузочные модули для записи в РПЗУ.

Организация ввода-вывода. Дискретный ввод-вывод в контроллере организован с помощью двух БИС программируемого интерфейса КР580ВВ55А. Оконечными элементами дискретного вывода являются 32 оптрона ADT110A, позволяющие коммутировать напряжения до 27 В на подключенной извне нагрузке при токе до 100 мА. Сигналы с дискретных входов поступают на 16 оптронов К262КП1А. Напряжение срабатывания

дискретного входа 2,4 В при токе 10 мА, что позволяет подключать его прямо к выходу ТТЛ логики.

Важной отличительной чертой микроконтроллера является гальваническая развязка по всем входам и выходам от управляемого оборудования, что исключает влияние помех управляемой схемы на работу контроллера. Гальваническая развязка аналоговых входов осуществлена с помощью так называемого «плавающего» конденсатора, который постоянно включен в цепь измеряемого напряжения и специальным коммутатором переключается на период измерения ко входу АЦП, тем самым выполняя функции аналогового запоминающего устройства и фильтра. Для преобразования аналогового сигнала в цифровую форму использован 10-разрядный интегральный АЦП К1113ПВ1А. Максимальная величина измеряемого напряжения положительной полярности — 10,23 В. Частота преобразования: номинальная — 1 Гц, максимальная — 70 Гц.

Органы управления и индикации. На передней панели микроконтроллера расположены лишь две кнопки: «Пуск» и «Продолжение». Обе кнопки опрашиваются программно, что позволяет задавать функциональное назначение их по своему усмотрению. В общем случае подразумевается, что кнопкой «Пуск» будет инициализироваться процесс управления, а для продолжения техпроцесса после запрограммированной паузы (например, для проведения вспомогательных ручных операций) следует пользоваться кнопкой «Продолжение».

Текущий шаг или цикл процесса управления индицируется восемью светодиодами на передней панели контроллера. Девятый светодиод «Авария» используется для индикации отклонения процесса от нормального режима. Все светодиоды включаются программно.

Сервисные возможности. Для удобства наладки, ремонта и проверки работоспособности микроконтроллера вместе с ним поставляется пульт контроля. Пульт позволяет вручную опросить все дискретные входы, проконтролировать величину напряжения на любом аналоговом входе и установить любые комбинации на дискретных выходах. Пульт контроля имеет размеры 220×110×30 мм, содержит 16 клавиш, 4 семисегментных индикатора и подключается к контроллеру на период наладки через разъем на передней панели. Функционирование пульта обеспечивается программой монитора, хранящейся в специальном ПЗУ.

В контроллере предусмотрены средства самотестирования. Тест самопроверки позволяет при подключении к входным и выходным разъемам контроллера несложной схемы быстро проверить работоспособность основных узлов, а в случае обнаружения неисправности локализовать место ее возникновения. В настоящее время выпущена опытная партия контроллеров.

За справками обращаться по адресу: 290034, Львов, бюро заказов.

ЛИТЕРАТУРА

1. Кобылинский А. В., Липовецкий Г. П.— Однокристальные микроЭВМ серии К1816.— Микропроцессорные средства и системы, 1986, № 1, с. 10—19.
2. Антонов Б. В., Глазер С. Ф., Маликов А. Г., Шабалин А. И. Система автоматизации проектирования программного обеспечения для однокристальной микроЭВМ К1816ВЕ48.— Микропроцессорные средства и системы, 1986, № 3, с. 25.

Статья поступила 20 ноября 1985 г.

УДК 681.32

Е. Г. Кормин

ПОВЫШЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ СИСТЕМ АВТОМАТИЗАЦИИ НА ОСНОВЕ МИКРОЭВМ «ЭЛЕКТРОНИКА 60М»

При создании массовых систем автоматизации экспериментальных исследований и технологических процессов на основе микроЭВМ возникает проблема обеспечения их длительной надежности. Основными причинами отказов систем являются сбои микроЭВМ.

Сбои микроЭВМ представляют собой однократные случайно появляющиеся отказы, не связанные с выходом из строя технических средств. В результате сбоя функционирование микроЭВМ может быть нарушено так же, как при аварийном отказе, если не принять специальных мер к восстановлению функционирования. Возникновение сбоя не катастрофично, поскольку его последствия могут быть ликвидированы программным путем с использованием алгоритмов с обратными связями [1, 2], метода контрольных точек [3], метода восстановления точки сбоя [4] и др.

Проблема заключается в определении самого факта сбоя и передаче управления программе восстановления: существуют разновидности сбоев, которые не могут быть выявлены ни программным путем, ни с помощью стандартного механизма аварийных прерываний ЭВМ, ни с помощью средств пассивного аппаратного контроля функционирования процессора [5].

Эти сбои заключаются в «зависании» процессора, например, на проверке какого-либо условия. Наиболее тяжелая форма такого сбоя — зависание управляющего микропроцессора на исполнении микрокоманды. В результате этого зависания выполнение всех программ, а также доступ к процессору всех внешних устройств прекращается, т. е. происходит отказ микроЭВМ и системы в целом.

Для предотвращения сбоев микроЭВМ и ликвидации их последствий в системах автоматизации традиционно используются методы пассивной и активной защиты, основанные на применении дополнительных технических средств, начиная от электромагнитных экранов, отдельных питающих фидеров, резервных источников и мотор-генераторов, вплоть до стопроцентного «горячего» резервирования [6] и использования сетевых структур, в которых живучесть систем нижнего уровня обеспечивается их поддержкой высоконадежными комплексами верхних уровней [7]. В массовых системах автоматизации, для которых стоимость — один из определяющих факторов, возможности использования дополнительных средств значительно ограничены.

Отсутствие эффективных средств защиты от сбоев и помех по питанию в комплекте наиболее массовых микроЭВМ «Электроника 60М» и «Электроника НЦ-80» потребовало разработки методов построения отказоустойчивых систем и средств. При этом была поставлена задача — достичь максимальной эффективности средств защиты при минимальных дополнительных затратах. Результат — разработка метода аппаратно-программной защиты, т. е. определенная организация аппаратных и программных средств систем автоматизации и протокола их взаимодействия при минимальном увеличении объема аппаратных и программных средств систем.

Метод аппаратно-программной защиты основан на четырех положениях:

1. Функционирование систем автоматизации — цикли-

ческое (измерение значений функций при изменяющихся значениях аргумента, изменение параметров воздействия во времени и т. д.). Искажение результатов единичного цикла не влияет катастрофически на работу автоматизированного комплекса. В противном случае могут быть приняты меры (например, за счет введения информационной избыточности) для защиты от такого влияния. Это позволяет при обнаружении сбоя не заботиться о продолжении текущего цикла либо восстановлении его результатов, а перейти к исполнению следующего, что дает возможность применения достаточно простого способа восстановления функционирования ЭВМ — метода контрольных точек, т. е. возобновления работы программы после сбоя, начиная с последней пройденной из совокупности заранее определенных контрольных точек программы. Для этого при прохождении контрольных точек ЭВМ делает «снимок вычислительного процесса», используя энергонезависимое устройство памяти. В качестве контрольных точек наиболее целесообразны характерные точки цикла. При определении факта сбоя специальная программа восстановления, используя эту информацию, может продолжить вычислительный процесс.

2. Для выявления отказов, не определяемых программным путем (прекращение исполнения программы), можно использовать метод взаимодействия аппаратных и программных средств системы [8]. Метод заключается в том, что при нормальной работе системы процессор периодически делает контрольные обращения к «Охранному таймеру», который должен генерировать сигнал начального пуска процессора при нарушении периодичности контрольных обращений.

3. В системах автоматизации объем оперативной информации, жизненно важной с точки зрения функционирования системы («снимок вычислительного процесса»), невелик. Это позволяет использовать для ее хранения энергонезависимые ЗУ, выполненные на микромощных КМОП-микросхемах ОЗУ большой емкости с использованием индивидуального резервного батарейного питания.

4. Наиболее критичный элемент систем автоматизации (с точки зрения их отказоустойчивости) — запоминающие устройства памяти программ, поскольку разрушение или искажение информации в этом ЗУ (вызванное, например, сбоем процессора или помехой по сети питания [9]) приводит к отказу ЭВМ.

Для ликвидации данных отказов перспективно использование «кремниевого программного обеспечения»— размещение программ в ПЗУ. Наибольший эффект получается при использовании перепрограммируемых ПЗУ, позволяющих в случае необходимости изменять программное обеспечение систем с минимальными затратами времени и средств. Современные интегральные ПЗУ высоко надежны и позволяют отказаться от внешних устройств хранения и ввода программ, что также увеличивает надежность систем.

Метод аппаратно-программной защиты использован при создании систем автоматизации массовых экспериментальных исследований. В основе реализации метода — включение в состав систем оригинального модуля.

Модуль «Регенератор функционирования программ» (РФП) обеспечивает аппаратную поддержку механизма восстановления функционирования систем после сбоев (рис. 1). Этот модуль включает в себя «Охранный» таймер, ПЗУ «Рестарт», энергонезависимое ОЗУ (рис. 2).

«Охранный» таймер определяет факт сбоя и вырабатывает сигнал перезапуска, поступающий на шину К ПИТН В магистрали микроЭВМ. Контроллер ситуации состоит из мультивибратора со сбросом и дешифратора команды обращения к модулю по адресу 173376_8. Выход дешифратора подключен к входу «Сброс» мультивибратора, выход мультивибратора — к шине К ПИТН В магистрали микроЭВМ.

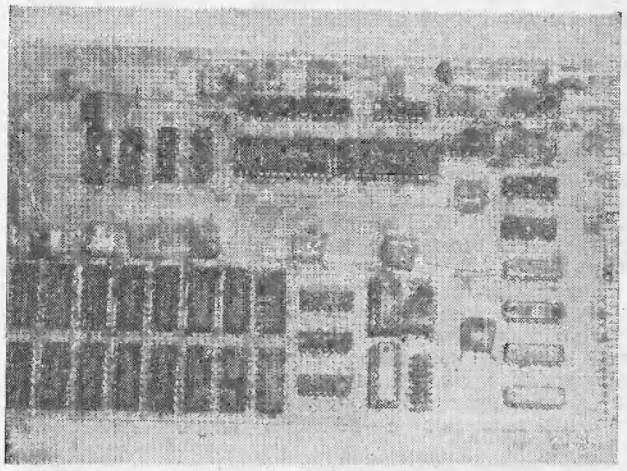


Рис. 1. Модуль «Регенератор функционирования программ»

Сброшенное состояние мультивибратора соответствует высокому уровню сигнала на шине К ПИТН В, разрешающему работу процессора. Появление импульса на выходе одновибратора, а следовательно, и на шине К ПИТН В приводит к выполнению процессором последовательности операций:

переходу на подпрограмму обслуживания внутреннего прерывания с адресом вектора 24;

выдаче через 3 мс (максимум) сигнала первоначальной установки системы К СБРОСН;

выполнению микропрограммы выбора режима пуска и перехода к выполнению программы с адреса 173000_8 (в основном режиме пуска).

Чтобы импульса перезапуска не было, мультивибратор должен поддерживаться в сброшенном состоянии сигналами с дешифратора. Для этого микроЭВМ должна периодически обращаться к РФП по адресу 173376_8, причем период обращений должен быть меньше длительности паузы между импульсами мультивибратора. При нарушении периодичности обращений исполняется процедура начального запуска микроЭВМ.

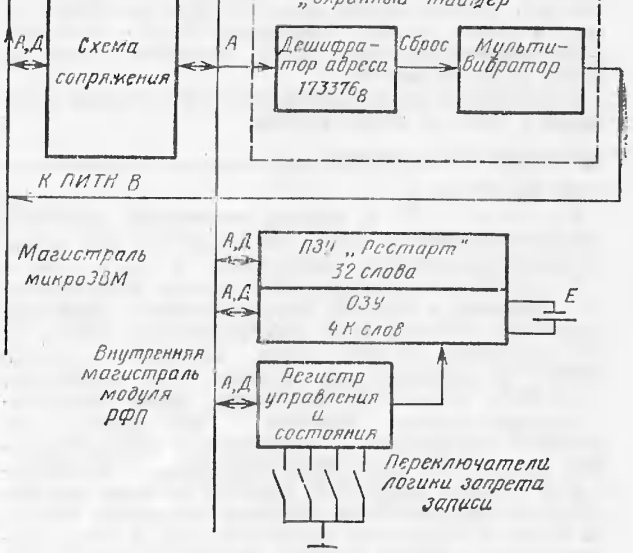


Рис. 2. Структурная схема модуля «Регенератор функционирования программ»

ПЗУ «Рестарт» (32 слова) с начальным адресом 173000_8 хранит программу перезапуска.

Энергонезависимое ОЗУ (4К слов) со страничной организацией (4 страницы по 1К слов) снабжено схемами постраничной защиты от несанкционированного доступа и общей схемой защиты от искажения информации при переходных процессах, связанных с включением-выключением питания. Схемы постраничной защиты препятствуют записи в ячейки запрещенной страницы. При этом модуль не отвечает сигналам подтверждения СИП на обращения микроЭВМ с командой записи, вследствие чего происходит аварийное прерывание.

Управление схемами защиты возможно программное, с помощью регистра команд, и ручное, с помощью переключателей, доступных пользователю, причем состояние переключателей может быть считано в микроЭВМ. Энергонезависимое ОЗУ может занимать в адресном пространстве микроЭВМ любой из 8 банков, которые выбирают с помощью переключателей на модуле.

При включении-выключении микроЭВМ появляются случайные комбинации сигналов в магистрали микроЭВМ и на выходах микросхем, входящих в логику управления энергонезависимым ОЗУ (ЭОЗУ). Следствием этого может быть запись случайных данных в ЭОЗУ. Для предотвращения этого на входах выбора кристалла микросхем ЭОЗУ — CS включена схема защиты, принудительно устанавливающая высокий уровень на входе CS при переходе сигнала К ПИТН В к низкому уровню.

ЭОЗУ выполнено на микросхемах КР537РУ3А. Резервный источник — внешний, подключаемый либо к резервной шине питания магистрали микроЭВМ, либо непосредственно к контактам на плате (тем самым достигается возможность переноса блока). Напряжение резервного источника 2...5 В. Ток, потребляемый в режиме хранения при напряжении питания 3 В,— не более 10 мкА. Конструктивно модуль РФП — это блок размером 252×146 мм, выполненный в стандарте микроЭВМ «Электроника 60М» (ОСТ 11.305.903) [10]. Элементная база — микросхемы серий К555, К556, К589.

Программное обеспечение для перезапуска при сбоях систем организуется так: «Ловушка прерывания» должна прекращать исполнение программы при любом аварийном прерывании. В программе выделяются характерные точки, при прохождении которых в ЭОЗУ переписывается вся оперативная информация, необходимая для восстановления функционирования. Нормальное состояние логики управления ЭОЗУ — состояние с запрещенной записью. Запрет снимается только на время переписи данных.

В программах предусматривается периодическое обращение к РФП по адресу 173376_8.

Начальная часть программы перезапуска, реализующая подготовку стека и переход к основной части программы, расположенной в ПЗУ программ, записывается в ПЗУ «Рестарт».

Метод аппаратно-программной защиты и модуль РФП использованы при создании систем автоматизации экспериментальных исследований и технологических процессов. При опытной эксплуатации этих систем, работающих с мощным энергопотребляющим оборудованием, функционирование восстанавливается после всех видов сбоев, вплоть до кратковременного отключения питания.

Разрабатываются типовые методы организации отказоустойчивых систем со средствами регенерации функционирования.

Телефон для справок о приобретении технической документации — 222-87 (г. Гатчина Ленинградской обл.).

ЛИТЕРАТУРА

1. Колоскова Г. П., Колосков В. А. Вопросы проектирования помехоустойчивых алгоритмов управления.— В кн.: Помехи в цифровой технике-82. Республиканский дом техники, 1982, с. 72—75.
2. Колосков В. А., Денисова Г. П. Об учете особенностей логических алгоритмов при выделении точек перезапуска в системах с восстановлением.—В кн.: Агрегатирование, контроль и диагностика сложных систем.— Киев, 1978, с. 64—71.
3. Карась В. М. Повышение устойчивости вычислительных процессов к сбоям ЭВМ программным способом.— УСиМ, 1982, № 3, с. 94—99.
4. Архангельский А. Н., Карибская Л. В., Николаев А. В., Сергунин А. В. Метод восстановления процесса вычислений после сбоя ЭВМ «Электроника 60».— ПСУ, 1984, № 9, с. 12, 13.
5. Олянишин О. А., Фридштанд В. Д. Повышение надежности систем автоматического управления на базе ЭВМ «Электроника 60».— Там же, 1985, № 9, с. 13, 14.
6. Foster S. Powerful small systems.— Computer Designs, 1983, 15, № 8, с. 101—111.
7. Григорьев Г. И., Доморацкий С. Н., Куклин Г. Н. и др. Особенности автоматизированных систем обработки информации на научно-исследовательских судах.— I междунар. школа по автоматизации научных исследований. Тез. докл.— Пущино, 1982.
8. Oppenheimer C. P. Reliable designs begin with the basics.— Computer Design, 1983, № 9, с. 93—98.
9. Носов В. В., Левин Д. З. Обеспечение помехоустойчивости устройств управления, построенных на базе микропроцессорной техники.— ПСУ, 1983, № 12, с. 21, 22.
10. ОСТ 11.305.903.

Статья поступила 8 августа 1985 г.

КРАТКИЕ СООБЩЕНИЯ

УДК 681.326-181.4

Кутлер Г. П. **К вопросу построения надежных телеинформационных систем на микроЭВМ для оперативно-информационных комплексов в энергетике.** — В энергетике в многоуровневой системе диспетчерского управления в качестве централизованного приемопередающего элемента сети, обеспечивающего работу по 64 дуплексным каналам связи, используется микроЭВМ. На диспетчерских пунктах, где отсутствует мини-ЭВМ, микроЭВМ выполняет также оперативно-информационные функции. При приеме оперативной информации многоканальным централизованным устройством весь объем информации концентрируется в одной микроЭВМ. Поэтому наиболее важным вопросом при разработке подобных комплексов является вопрос о надежности работы системы. В статье рассматриваются общие вопросы повышения надежности, использованные в данных телеинформационных системах (ТИС), такие как резервирование микроЭВМ и каналов связи, и различные способы повышения живучести системы за счет разработки более надежного математического обеспечения (МО).

В статье приводятся различные виды повреждений, способы их выявления и меры по повышению живучести и готовности системы, которые могут быть использованы при разработке системы на микроЭВМ. К этим мерам относятся автоматическое переключение основного и резервного оборудования, перезагрузка исходных данных, повторные передачи и переспросы, периодическое обновление полного объема информации, технологический контроль, использование метода недоверия и модульно-функционального метода при построении МО и т. д. Особо рассматривается автоматический рестарт как наиболее общий метод восстановления работоспособности системы в целом.

За справками обращаться по адресу: 125445, Москва, Ленинградское шоссе, д. 112/1, к. 4, кв. 1044.

Заключительные три статьи этого раздела посвящены особенностям практического использования серийно выпускаемых отечественной промышленностью средств отображения информации.

УДК 681.142 : 003.6

М. М. Ляпунов, Б. И. Беляев

ПЕРСОНАЛЬНАЯ ГРАФИЧЕСКАЯ СТАНЦИЯ ПЕГАС

Персональная графическая автономная станция (ПЕГАС) представляет собой универсальное устройство, способное отображать черно-белые изображения на растре 400×275 точек и обладающее собственной вычислительной мощностью. ПЕГАС была разработана для использования в качестве рабочего места конструктора и технолога-программиста в интегрированной гибкой производственной системе КАПРИ [1].

ПЕГАС выполнена на основе и в конструктиве алфавитно-цифрового дисплея 15 ИЭ-00-013. Видеомонитор, клавиатура и источник питания используются без каких-либо доработок. Вместо дисплейного процессора и плат логики дисплея установлены микроЭВМ МС 1201.1 и два функциональных блока, разработанных специально для ПЕГАСа (рис. 1). На крейте дисплея монтируются проводные соединения, соответствующие системному каналу микроЭВМ. Четыре свободных позиции крейта могут быть использованы для подключения дополнительной стандартной или специальной аппаратуры.

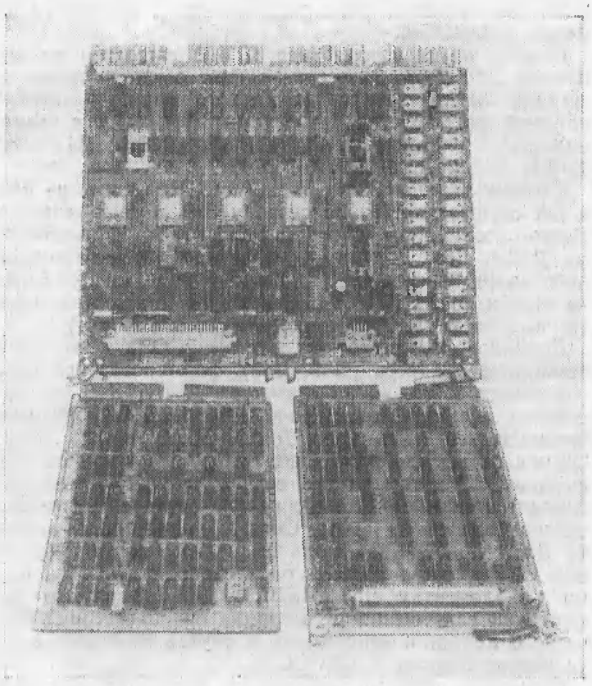


Рис. 1. Функциональные блоки ПЕГАСа

[1] См. статьи Адамова Е. О. и др. в журнале «Микропроцессорные средства и системы», 1985, № 4, с. 53—62.

Аппаратурная простота ПЕГАСа достигнута за счет переноса ряда функций с аппаратурного уровня на программный. Такие функции, как генерация символов, отрезков, дуг, изображение и движение маркера, реализуются программно на встроенной микроЭВМ, вычислительная мощность которой достаточна для выполнения прикладных программ наряду с перечисленными функциями.

В составе одного из функциональных блоков имеются два реверсивных координатных счетчика, предназначенных для подключения к ПЕГАСу графического манипулятора (например, типа «мышь»). Поскольку обработка показаний счетчиков ведется программно, манипулятор можно использовать для перемещения по экрану маркера или фрагментов изображения и для других функций.

Через устройство последовательного обмена микроЭВМ ПЕГАС подключается к вычислительной сети или отдельной ЭВМ.

В качестве памяти под растровое изображение использован участок ОЗУ встроенной микроЭВМ (этот участок далее называется растровой памятью). Изображение на экране представляет собой точечный растр прямоугольной формы с 400 точками по горизонтали и 275 по вертикали. Одной точке растра соответствует один бит ОЗУ; 16-разрядное слово растровой памяти отображается горизонтальной строчкой из 16 расположенных рядом точек растра. Для растровой памяти можно использовать любой участок ОЗУ длиной 6875 слов (адрес начала растровой памяти переключается программно). Так как из 32К адресного пространства микроЭВМ 4К выделены под встроенное ПЗУ и внешние устройства, а 7К задействованы под растровую память, то объем математического обеспечения графической станции может достигать 21К 16-разрядных слов.

Равномерный точечный растр позволяет отображать как графическую информацию, так и алфавитно-цифровую. При горизонтальном расположении строк символов на экран вмещается до 30 строк по 66 символов в каждой. Для отображения 80-символьных строк графическая станция имеет специальный режим работы, при котором вместо сплошного растра аппаратно создается линейчатый (80 вертикальных полос шириной 5 точек).

Чтение информации из ОЗУ для вывода ее на индикатор осуществляется в режиме прямого доступа к памяти.

Структура персональной графической станции ПЕГАС представлена на рис. 2.

Регенератор служит для циклического чтения растровой памяти и развертки ее содержимого на экране видеомонитора. Он содержит регистр начального адреса (РНА) — 16-разрядный регистр, программно доступный для записи, предназначенный для хранения адреса начала растровой памяти. Для включения регенерации равномерного растра необходимо занести в РНА значение адреса командой MOV ADDR, @#177766; обращение MOV ADDR, @#177767 включает регенера-

Рис. 2. Структура ПЕГАСа

цию линейчатого растра; выключить регенерацию можно командой RESET или занесением в РНА нечетного числа.

Комбинированный интерфейсный блок (КИБ) предназначен для связи микроЭВМ с клавиатурой и манипулятором. Он содержит четыре 16-разрядных регистра, программно доступных для чтения и записи: регистр состояния (РС), регистр данных (РД), регистр-счетчик X-координаты (РХ) и регистр-счетчик Y-координаты (РУ). Перечисленные регистры имеют адреса 177500...177506 (при желании адреса могут быть изменены перестановкой перемычек). РС служит для ввода битовой информации. Разряды 00...03 отражают состояние фиксируемых и функциональных клавиш клавиатуры. Разряд 06 устанавливается программно и разрешает прерывание, которое происходит при переходе в «Лог. 1» разряда 07, подключенного к кнопке графического манипулятора. Остальные разряды РС не задействованы, но выведены на разъем, установленный на задней панели дисплея, и могут быть использованы по усмотрению пользователя. РД предназначен для вывода битовой информации. Задействован только один разряд (00), переключающий русский и латинский регистры клавиатуры. Остальные разряды также выведены на разъем. Регистры РХ и РУ — это реверсивные счетчики, постоянно подключенные к датчикам манипулятора.

Регенератор и КИБ построены на базе ИС серий К155, К531, К559, К589, и каждый размещается на стандартной плате размерами 135×240 мм².

Структура регенератора (рис. 3). Регенератор содержит буферное ОЗУ, предназначенное для временного хранения видеоинформации. Необходимость включения в схему буферного ОЗУ вызвана ограниченным быстродействием системного канала микроЭВМ. Временные характеристики развертки требуют чтения одного слова каждые 2,08 мкс. Цикл чтения слова составляет 1,2...1,6 мкс; время предоставления канала может достигать 8...10 мкс. Узел управления захватывает канал для чтения нескольких слов растровой памяти и освобождает его, когда буферное ОЗУ заполнено. Чтение информации из буферного ОЗУ производится синхронно с работой генератора импульсов и блока синхронизации; прочитанные 16-разрядные слова разворачиваются сдвиговым регистром в видеосигнал. Емкость БОЗУ — 16 16-разрядных слов.

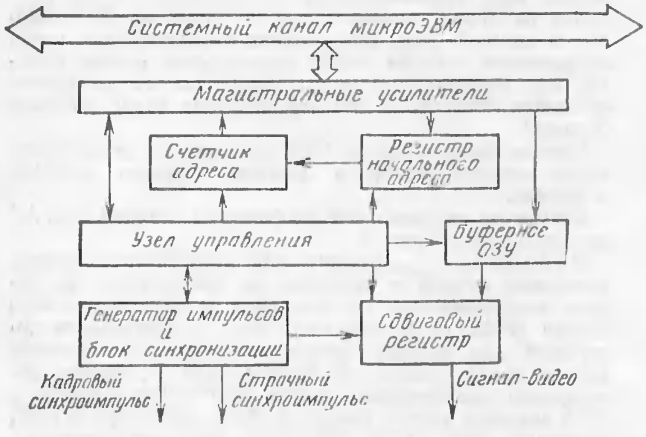


Рис. 3. Структура регенератора

Наибольшие сложности при разработке принципиальной схемы регенератора (рис. 4) были вызваны ограниченным быстродействием элементов и системного канала микроЭВМ.

Генератор импульсов и блок синхронизации. Времязадающий генератор (Д3.1...Д3.3) работает на частоте 30,8 МГц. Тактовый генератор (Д13. Д64.1, Д70, Д65, Д19.1, Д19.2, Д66...Д69) вырабатывает сигналы ТИВ и ТИН (тактовый импульс — высокий и низкий), являющиеся основными синхронизирующими импульсами в схеме регенератора. Тактовый генератор работает в одном из двух режимов в зависимости от состояния триггера Д67.1. Если на Д67.1 (5) низкий уровень, ТИВ представляет собой равномерную последовательность импульсов частотой 7,7 МГц и обусловливает равномерный растр. В противном случае ТИВ генерируется как последовательность пятиимпульсных пакетов. Частота импульсов в пакете — 10,27 МГц; периодичность пакетов такова, что средняя частота импульсов ТИВ одинакова в обоих режимах.

Делитель частоты 1/500 (Д4, Д9.1, Д14, Д16, Д17) вырабатывает сигнал ССИВ (строчный синхроимпульс); делитель 1/308 (Д3.5, Д20, Д28.1, Д21, Д23.2, Д24.2, Д24.3) — кадровый синхроимпульс КСИВ.

Магистральные усилители (Д41...Д44, Д5, Д6, Д18, Д29) обеспечивают связь регенератора с системным каналом микроЭВМ.

Регистр начального адреса (РНА) выполнен на элементах Д30...Д33. Занесение в него осуществляется по сигналу ЗНАН (запись начального адреса — низкий), который формируется элементами Д7.4, Д5.2 и схемой опознавания адреса 177766/177767 (Д29.1, Д1, Д2.1, Д15.1, Д15.2, Д6.1, Д16.2, Д12.1).

Счетчик адреса (СА) (Д34...Д37). Перезапись из РНА в СА осуществляется по сигналу ЗИСН (занесение исходного состояния), который вырабатывается элементами Д62.4, Д61, Д9.3 после завершения развертки каждого кадра. Сигнал СЧЕТ, формируемый на Д15.3 после чтения по КПДП очередного слова растровой памяти, вызывает увеличение содержимого СА на 2.

Буферное ОЗУ состоит из собственно ОЗУ (Д47...Д50), коммутатора адреса (Д45, Д46), формирователей адресов записи и чтения на счетчиках Д39 и Д40 соответственно, счетчика заполнения Д55 и схемы арбитража чтения/записи (Д56, Д9.2, Д57.1, Д57.2, Д24.3, Д64.2). Сигнал на выходе последнего задерживает запись в буферное ОЗУ в том случае, если в данный момент происходит чтение либо до его начала осталось менее трех тактов (импульсов ТИВ). Сигнал на выходе 6 элемента Д58.1 приобретает высокий уровень, когда в буферном ОЗУ имеется хотя бы одна свободная ячейка. Схема на элементах Д58.2, Д59.1, Д62.1...3 служит для фиксации разрешения регенерации при занесении в РНА четного адреса и запрещении в случае нечетного адреса или по сигналу К СБРОС Н.

Сдвиговый регистр выполнен на элементах Д51...Д54, Д19.4, Д2.2 и Д63.

Узел управления частично описан выше. Кроме схем опознавания адреса, фиксации разрешения регенерации и формирователя сигналов ЗИСН и ЗИСВ в него входят узел захвата магистрали (Д8.2, Д8.3, Д10, Д11) и узел обмена данными (Д22, Д25...Д27, Д23.3, Д12.2). Узел захвата магистрали запускается высоким уровнем

сигнала ТЗМ (требование захвата магистрали), который вырабатывается триггером Д59.2. По получении сигнала предоставления прямого доступа К ППД1 Н узел захвата магистрали сигналом ПМН (предоставление магистрали) запускает цепочку триггеров в узле обмена данными. Триггеры в цепочке последовательно устанавливаются в 0, формируя при этом сигнал разрешения выдачи адреса (Д22.2), канальные сигналы К СИА Н, К ВВОД Н, сигнал коммутации адреса записи буферного ОЗУ (Д23.3) и синхроимпульс записи буферного ОЗУ (Д12.2). В процессе срабатывания триггеров цепочки могут возникать задержки по ожиданию сигнала К СИП Н и сигнала разрешения со схемы арбитража (Д64.2). По окончании сигнала К СИП Н все триггеры цепочки устанавливаются в 1. Если к этому моменту в буфере ОЗУ есть свободное место, то сигнал ТЗМ не сбрасывается и очередной цикл чтения происходит без освобождения магистрали.

В настоящее время около 10 экземпляров ПЕГАСа находятся в опытной эксплуатации, которая показывает высокую надежность станции. Программное обеспечение ПЕГАСа постоянно развивается. Реализованы следующие пакеты:

быстродействующий алфавитно-цифровой знакогенератор с функциями плавного сдвига текста вверх и вниз по экрану;

версия интерпретирующей системы БИМС с функциями вывода на экран текста, точек, отрезков, дуг, заливки контуров и моделирования движения объектов;

пакет «ТРАССА», предназначенный для контроля и коррекции управляющих программ для токарных станков с УЧПУ «Электроника НЦ-31» путем моделирования процесса резания на экране (см. фото 1).

В стадии разработки находятся версия БИМС с функциями геометрических расчетов и пакет «ТРАССА-3» для автоматизированной разработки управляющих программ путем моделирования процесса резания.

С запросами о получении технической документации обращаться по адресу: 123182 Москва, Д-182, Пл. И. В. Курчатова, Институт атомной энергии имени И. В. Курчатова.

Статья поступила 8 августа 1985 г.

УДК 681.325.5 : 181.4

Н. Н. Насруллаев, О. К. Нусратов, С. Б. Ситков, Р. К. Симонян, Е. Д. Дворянкина

МНОГОТЕРМИНАЛЬНАЯ СИСТЕМА ОТОБРАЖЕНИЯ ИНФОРМАЦИИ

Современные системы отображения информации (СОИ) содержат управляющую ЭВМ и группу пультов операторов и должны обрабатывать и отображать информацию, непрерывно поступающую в высоком темпе из внешнего канала, а также анализировать ее. Включая в состав пультов оператора микропроцессорные средства, можно анализировать информацию в автономном режиме этих устройств. Это значительно расширяет функциональные возможности СОИ. Матричные газоразрядные индикаторы позволяют создавать табло коллективного пользования (увеличивается область применения СОИ для испытаний, экспериментов и т. д., где информация предоставляется группе специалистов-наблюдателей, коллективно следящих за ходом процесса). МикроЭВМ и микропроцессорные средства позволяют строить универсальные и гибкие СОИ [1].

Многотерминальная система отображения информации СОИ, созданная в СКБ «Кибернетика» АН Азерб. ССР, позволяет:

отображать статическую и динамическую информацию в графической и символьной форме на экранах табло коллективного пользования, пультов оператора;

анализировать выведенную информацию;

масштабировать (сдвигать, растягивать) графическое изображение по осям координат;

совмещать несколько графиков в одном кадре.

Программное обеспечение многотерминальной системы реализовано в ОС «РАФОС» и состоит из монитора и пакетов программ, работающих независимо друг от друга, связанных между собою только через монитор и общую область памяти. Объем ОЗУ микроЭВМ — не менее 32К слов.

Система отображения информации (СОИ) содержит (рис. 1) диалоговый вычислительный комплекс «Электроника НЦ 80-20/2» (ДВК-2), контроллер аппаратуры управляющего комплекса, контроллеры устройств ввода-вывода, один или несколько пультов оператора и табло коллективного пользования.

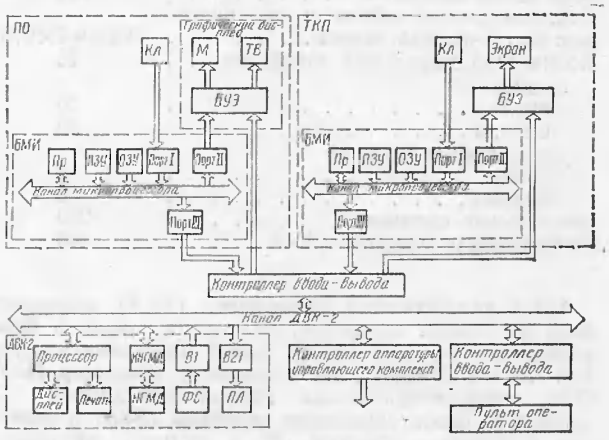


Рис. 1. Структурная схема системы отображения информации

Основные характеристики системы отображения информации

Вид выводимой информации	Графический, символьный
Тип обмена	Асинхронный, параллельный
Разрядность шин данных, бит	16
Внутрисистемный интерфейс типа	Q bus
Адресное пространство	32К
Режимы обмена	Программный ввод-вывод, прерывание по вводу и выводу, прямой доступ к памяти по вводу и выводу

Пропускная способность каналов связи между устройствами СОИ и ДВК-2 зависит от характеристик процессора ДВК-2 и режима обмена и равна (К слов/с):

АУК — ДВК-2	250
ПО — ДВК-2	200
ТКП — ДВК-2	200

В состав пульта оператора (ПО) входят: блок маркерных измерений (БМИ), клавиатура (Кл) и графический дисплей, выполненный на основе телевизионного приемника.

Технические характеристики пульта оператора

Тип экрана	ЭЛТ
Цвет свечения телевизионного приемника	Красный, синий, зеленый
Монитора	Черно-белый
Размер экрана по диагонали, см	
телевизионного приемника	61
монитора	16
Информативная емкость, точка	
телевизионного приемника	512×256
монитора	512×256
Объем ЗУ, бит	
телевизионного приемника	128К×3
монитора	128К
Время высвечивания одной точки экрана, нс	100
Разрядность шины данных клавиатуры, бит	8
Максимальное расстояние от контроллера ввода-вывода, м	200
Соединительный кабель с контроллером ввода-вывода, марка	ТСВ4×2×0,5
Количество сигнальных линий, шт.	36
данные	
ввода	16
вывода	16
управляющие сигналы	
ввода	2
вывода	2
Напряжение питания, В	220
Потребляемая мощность, В·А	600

Табло коллективного пользования (ТКП) содержит: блок маркерных измерений, клавиатуру, экран и блок управления экраном (БУЭ). Блоки маркерных измерений ПО и ТКП идентичны и содержат: процессор, ПЗУ, ОЗУ, порты ввода-вывода. Графический дисплей ПО состоит из блока управления экранами (БУЭ) и экранных устройств — монитора (М) и цветного телевизионного приемника (ТВ).

Технические характеристики табло коллективного пользования

Количество экранов, шт.	2
большой	1
малый	1
Размер экрана (h×l), мм	
большой	1544×3088
малый	1544×772
Размер индикаторного элемента — точки (h×l), мм	3×3
Информативная емкость экрана, точка	
большого	512×256
малого	128×256
Объем ЗУ большого экрана, бит	128К
Яркость свечения, кд/м²	90
Цвет свечения	Зеленый
Видимость, м	20
Максимальное расстояние от контроллера ввода-вывода, м	200
Соединительный кабель с контроллером ввода-вывода, марка	ТСВ41×2×0,5
Количество сигнальных линий, шт.	36
данные	
ввода	16
вывода	16
управляющие сигналы	
ввода	2
вывода	2
Напряжение питания, В	220
Потребляемая мощность, В·А	3500

СОИ работает в трех режимах: I — режим подготовки данных; II — режим в темпе эксперимента (реального времени); III — режим повторного воспроизведения.

В режиме I СОИ работает автономно, т. е. без связи с внешним управляющим комплексом. Оператор в диалоговом режиме с системного дисплея формирует и редактирует файл данных, строя на экране изображение расчетных (эталонных) кривых.

Используются пакеты программ формирования и редактирования файлов данных; построения графического изображения.

В режиме II информация поступает по каналу управляющего комплекса в темпе эксперимента через контроллер аппаратуры управляющего комплекса в ДВК-2, где она обрабатывается и одновременно накапливается на ГМД. Обработанная информация по каналу ДВК-2 через контроллеры ввода-вывода поступает в блоки управления экранами графического дисплея ПО и ТКП и отображается на экранах.

Используются пакеты программ для приема данных из управляющего комплекса в БУФЕР ДВК-2 и их регистрации на ГМД; директивного управления (изменение типа кривой, масштаба и т. д.); чтения БУФЕРА, обработки и отображения данных в графическом и цифровом виде.

В режиме III обрабатывается и выводится на экраны графических дисплеев ПО и ТКП информация, записанная на ГМД в режимах I и II. При этом каждый оператор может отображать необходимый файл, масштабировать, анализировать с помощью маркера параметры выведенных кривых (графиков), совмещать кривые, передавая в процессор ДВК-2 соответствующие директивы из клавиатур ПО и ТКП.

Используются пакеты программ для работы с библиотекой файлов данных; построения графического и символьного изображения; для проведения маркерных измерений.

В режимах I и III обычно готовится (до эксперимента) и обрабатывается (после эксперимента) большое количество информации. Поэтому работа графических устройств (ПО и ТКП) в СОИ организована в режиме разделения времени, что обеспечивает одновременную и независимую работу нескольких операторов. При проведении анализа с помощью маркера изображение строится и перемещается под управлением БМИ. Координаты изображения маркера в БУЭ ПО и ТКП передаются без процессора ДВК-2, в который передаются лишь координаты измеряемой точки. После обработки

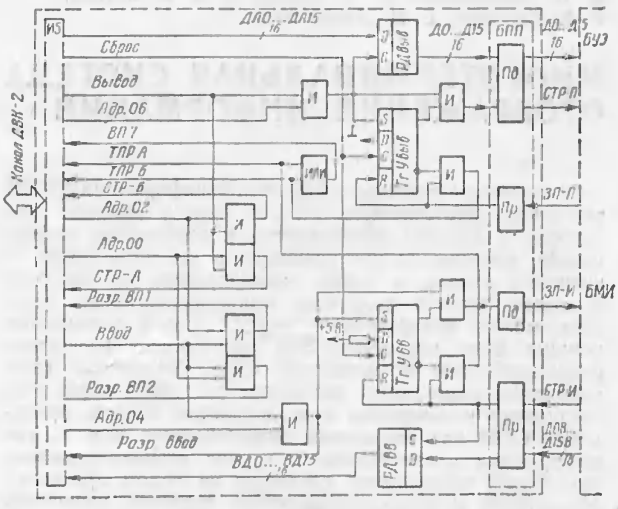


Рис. 2. Функциональная схема контроллера ввода-вывода

в ДВК-2 на экраны ПО, ТКП выводятся параметры замеряемой точки.

ДВК-2 выполняет функции дисплейного процессора, управляет обменом с аппаратурой управляющего комплекса, перфоленточным устройством в режиме подготовки данных и печатью при документировании [2].

Контроллер ввода-вывода предназначен для подключения к каналу ДВК-2 одного ПО и одного ТКП и состоит из двух идентичных узлов — контроллера ПО и контроллера ТКП.

Контроллер ПО (ТКП) (рис. 2) содержит интерфейс пользователя И5, 16-разрядные адресуемые регистры данных ввода (РДвв) и вывода (РДвыв), триггеры управления вводом (Тгвв) и выводом (Тгвыв), элемент ИЛИ, группу элементов И, выполненных на основе микросхем серии К155, а также блок приемопередатчиков на основе микросхем серии К109, К155 и согласующих резисторов.

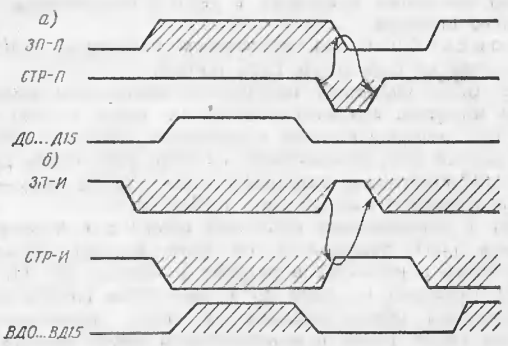


Рис. 3. Временная диаграмма работы контроллера ввода-вывода при обмене информацией с блоком управления экраном (*а*) и блоком маркерных измерений (*б*)

Блок БПП усиливает и согласует по уровню сигналы, передаваемые по каналу связи с ПО (ТКП). Адреса регистров и векторов прерываний устанавливаются перемычками в интерфейсе пользователя И5. Обменом между ПО (ТКП) и контроллером ввода-вывода управляют сигналы (рис. 3):

ЗП-И — запрос источнику на выдачу информации;

СТР-И — строб от источника информации — на шинах выставлены данные;

ЗП-П — запрос приемника на выдачу информации;

СТР-П — строб приемнику информации — на шинах данных выставлена информация.

Сигналы ЗП-П и СТР-И инициируют сигналы ТПР А, ТПР Б, ВП7. Последний интерпретируется как разряд 07 регистра команд состояний ввода-вывода (РКСвв, РКСвыв). Сигналы Разр. ВП1, Разр. ВП2 — сигналы обращения к РКСвыв, РКСвв соответственно в цикле ВВОД. Наличие одного из них разрешает прохождение сигнала ВП7 на шину Д07 канала ДВК-2. Сигналы СТР-А, СТР-Б служат для записи информации в разряд 06 РКСвв, РКСвыв соответственно (триггеры разряда 06 РКСвв, РКСвыв расположены в интерфейсе пользователя И5).

Контроллер аппаратуры управляющего комплекса подключает аппаратуру управляющего комплекса (АУК) к ДВК-2. Контроллер состоит из устройства обмена с АУК и блока таймера.

Устройство обмена с АУК (рис. 4) содержит устройство прямого доступа к памяти ИЗ, триггер флага (ТгФл), триггер режима (ТгР), триггер управления (ТгУ), селектор адреса, переключатель, формирователи, группу элементов И, выполненных на основе микросхем серии К155, а также блок БПП, идентичный блоку БПП контроллера ввода-вывода.

Технические характеристики контроллеров ввода-вывода, аппаратуры управляющего комплекса

Тип обмена с ПО, ТКП	Асинхронный, параллельный
с аппаратурой управляющего комплекса	Синхронный, параллельный
Разрядность шин данных со стороны ПО, ТКП, бит	
входных	16
выходных	16
Со стороны аппаратуры управляющего комплекса	
входных	16
выходных	16
Разрядность шин управления со стороны ПО, ТКП	4
Со стороны аппаратуры управляющего комплекса	
входных	4
Режимы работы контроллера ввода-вывода	Программный ввод-вывод, прерывание по вводу и выводу
Режим работы контроллера аппаратуры управляющего комплекса	Прямой доступ к памяти в режимах ввод и вывод
Максимальное расстояние от канала ДВК-2, м	1,5
Напряжение питания, В	5
Потребляемая мощность, В·А	
контроллера ввода-вывода	30
контроллера аппаратуры управляющего комплекса	18,5

СОИ — один из абонентов, подключенных к каналу управляющего комплекса, имеет адрес, задаваемый переключателем. При одном обращении АУК к СОИ пе-

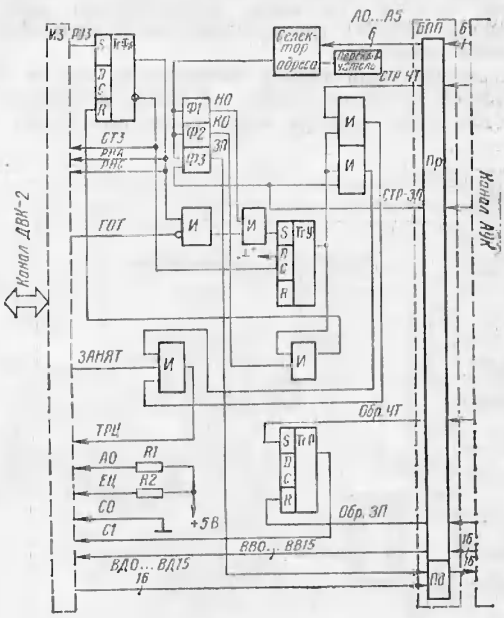


Рис. 4. Функциональная схема устройства обмена информацией с аппаратурой управляющего комплекса

редается или принимается массив данных, каждое слово которого сопровождается сигналом СТР-ЧТ (синхроимпульс в режиме передачи информации из АУК в СОИ) либо СТР-ЗП (синхроимпульс в режиме приема информации АУК из СОИ). Массивы информации записываются в БУФЕР ОЗУ ДВК-2 (в режиме «Чтение») либо считываются из него (в режиме «Запись»).

В начале каждого обмена формирователь Ф1 формирует сигнал Начала обмена (НО). При условии, что ТгФл установлен (установка выполняется под управлением программы сигналом Фл1, поступающим по цепи РУ3, и означает разрешение программы на обмен с ОЗУ ДВК-2) и что сигнал «ГОТ» — низкий, сигнал «НО» устанавливает ТгУ, тем самым разрешая после прихода сигнала СТР-ЧТ либо СТР-ЗП выработку сигнала ТРЦ (сигнал «Занят» высокий). Сигнал ТРЦ выставляется по каждому слову. На выходе ТгФл формируются сигналы РНА, РНС.

По окончании обмена АУК снимает с адресной шины адрес СОИ, при этом на выходе формирователя Ф2 формируется сигнал «Конец обмена» КО, который сбрасывает триггер ТгФл, тем самым формируя сигнал Фл2, поступающий по цепи Ст3 в устройство прямого доступа к памяти И3 (сигнал Фл2 информирует программу о конце обмена с ОЗУ ДВК-2). Длина передаваемых и принимаемых массивов задается АУК.

Сигналы Обр.ЧТ (обращение к абоненту в режиме передачи информации на АУК в СОИ) и Обр.ЗП (обращение к абоненту в режиме передачи информации из СОИ в АУК) определяют направление обмена. По цепям С1, С0 передается код, определяющий вид канального цикла ДВК-2 при обмене в режиме ПДП.

На выходе формирователя Ф3 формируется сигнал ЗП («Запись»), разрешающий передачу данных в канал АУК.

Блок программируемого таймера автономно отрабатывает программно задаваемые временные интервалы (ВИ) с формированием сигналов прерывания, счетом астрономического времени и используется при обмене СОИ с АУК в режиме реального времени.

Блок программируемого таймера (рис. 5) содержит интерфейс пользователя И5, программируемый таймер (Т), буферный регистр управления (БРУ), генератор опорной частоты (Г), схему формирования запросов прерывания (СФЗП), группу элементов И, элемент ИЛИ и элемент задержки.

Программируемый таймер выполнен на основе БИС КР580ВИ53, остальные узлы — на основе микросхем серии К155. Блок таймера ведет одиночный отсчет ВИ

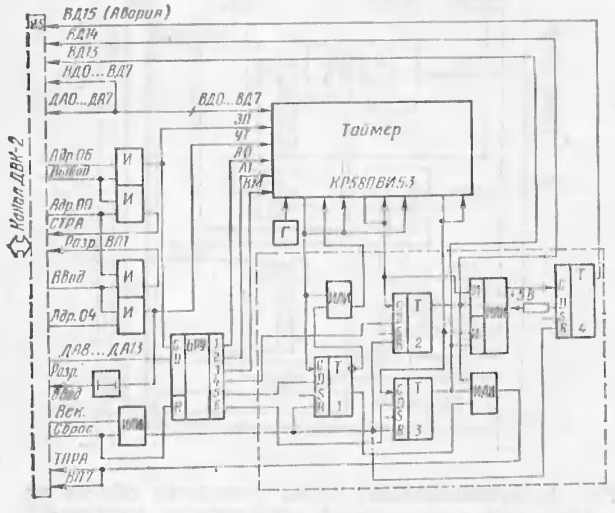


Рис. 5. Функциональная схема блока таймера

и циклический отсчет одновременно двух интервалов ВИ1 и ВИ2, обеспечивающих соответственно прием данных в БУФЕР, процесс чтения БУФЕРА, обработки и отображения информации. Диапазон ВИ — от 1 мкс до нескольких секунд. Длительность ВИ задается программно и передается по цепям ДА0...ДА7 в таймер Т (записывается в него управляющими сигналами А0, А1, ВМ, ЗП).

Канал «0» используется как делитель частоты для каналов «1» и «2» при циклических отсчетах и для одиночного отсчета ВИ. Каналы «1» и «2» используются для отсчета ВИ1 и ВИ2.

По окончании отсчета одного из временных интервалов в схеме СФЗП формируется сигнал ТПРА. Номер канала, выставившего ТПРА, а также сигнал «Авария» (формируется в случае неотработки предыдущего запроса ТПРА) передаются по цепям ВД13, ВД14, ВД15 соответственно и считываются в процессор под управлением программы сигналом Разр. ввод. По этому же сигналу по цепям ВД0..ВД7 в ДВК-2 считывается код текущего времени.

Процессор блока маркерных измерений (БМИ) реализован на базе МПК БИС КР580.

ПЗУ БМИ содержит программы построения изображения маркера, его перемещения по полю экрана ПО или ТКП, анализа клавиш клавиатуры, обмена с ДВК-2. ПЗУ (2К×8 бит) реализовано на базе БИС серии К573 с ультрафиолетовым стиранием, ОЗУ (2К×8 бит) — на базе статической памяти типа К134РУ6.

Порт I обеспечивает байтовый обмен с клавиатурой ПО или ТКП, реализован на базе одного элемента КР580ВВ55 и работает в режиме I «Ввод» [3]. Порты II, III (каждый на базе двух элементов КР580ВВ55) обеспечивают обмен словами с БУЭ графического дисплея (БУЭ ТКП) и контроллером ввода-вывода соответственно. Элемент, в который записывается старший байт слова, работает в режиме 0 — «Вывод», а элемент, в который записывается младший байт слова,— в режиме I — «Ввод». Порт II передает координаты изображения маркера и код действия (зажечь, погасить точку) над точкой в БУЭ графического дисплея ПО (БУЭ ТКП). Порт III передает координаты измеряемой точки графика и коды клавиш основного поля клавиатуры в ДВК-2.

БМИ входит в состав ПО и ТКП, логические блоки которых выполнены на печатных платах размером 110×135 мм и размещены в типовых каркасах К2КБ13.

Клавиатуры ПО и ТКП идентичны и содержат основное и вспомогательное наборные поля и выполнены по схеме матрицы, в узлах которой включены контактные группы клавиш.

Основное поле клавиатур предназначено для передачи оператором в ДВК-2 рабочих директив и содержит клавиши русского, латинского, греческого алфавитов и служебные клавиши.

Вспомогательное поле клавиатур служит для работы оператора с БМИ и содержит клавиши направления перемещения маркера (для перемещения маркера в восьми направлениях: вверх, вниз, вправо, влево, по диагоналям и в начальную позицию) и управляющую клавишу «Измерение» (для получения на системной строке графических дисплеев ПО и экранах ТКП значения точки графика, в которой находится маркер). Клавиатура и Порт I БМИ обмениваются информацией по асинхронному байтовому каналу под управлением сигналов СТР-И, ЗП-И.

Графический дисплей — это высокоскоростное устройство вывода растрового типа на базе телевизионного приемника [4]: цветного телевизора (ТВ) типа УЛПЦТИ-61 («Горизонт-736», «Фотон-716», «Янтарь Ц-355») и черно-белого монитора (М) типа «Электроника ПОЭ ИФА16». БУЭ графического дисплея (рис. 6) содержит блок приемопередатчиков (БПП), блок ин-

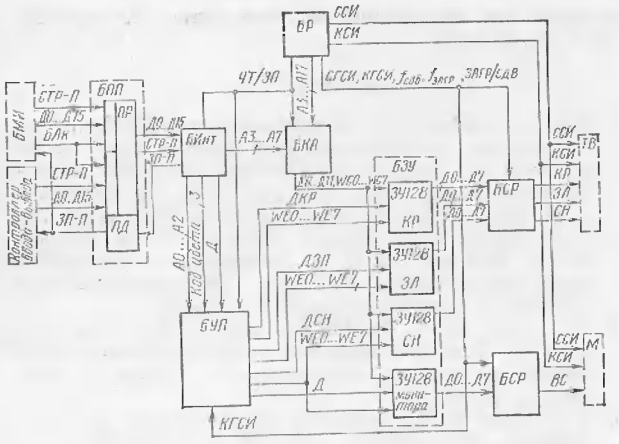


Рис. 6. Функциональная схема блока управления экраном графического дисплея

терфейса (БИнт), блок БЗУ, блок управления ОЗУ (БУП), блок регенерации изображения (БР), блок коммутации адресов ЗУ (БКА), блоки сдвиговых регистров (БСР). Блоки БИнт, БУП, БКА, БР, БСР выполнены на основе микросхем серии К155. БЗУ выполнен на базе статического ЗУ типа К541РУ1А (в БУЭ можно использовать БЗУ, выполненный на базе динамического ЗУ типа К565РУ3, К565РУ6). Блок БПП идентичен блоку БПП контроллера ввода-вывода.

Изображение на экранах графического дисплея строится по точкам заданием координат X, Y (интерпретируются как адреса БЗУ, по которым записываются или стираются точки) и кода цвета (для телевизионного приемника). Координаты X, Y, код действия (зажечь, погасить точку, полное стирание экрана) поступают из контроллера ввода-вывода (вывод основного изображения) по цепям Д0...Д15 либо по цепям Д'0...Д'15 из БМИ четырьмя словами в следующей последовательности: код цвета, код координаты X, код координаты Y, код действия. Содержимое БЗУ разворачивается («Чтение») со скоростью перемещения луча по экрану. Информация в БЗУ записывается в промежутках между тактами «Чтения».

БИнт синхронизирует поступление информации из контроллера ввода-вывода (либо БМИ) и запись ее в БЗУ, а также формирует сигнал запроса на ввод одного слова (ЗП-П) и хранит информацию одной точки изображения.

БУП задает режим работы БЗУ («Чтение», «Запись») (WE0...WE7). Блоки ЗУ128 хранят изображение объемом в 1 кадр по каждому цвету для ТВ и М.

БКА коммутирует адрес ячейки ЗУ128 из БР в режиме «Чтение» и из БИнт в режиме «Запись». БР управляет режимом развертки содержимого БЗУ на экраны ТВ и М и вырабатывает управляющие сигналы ССИ, КСИ, СГСИ, КГСИ, коды адресов ячеек ЗУ128 (А3...А7), а также $f_{СДВ}$ (частота сдвига БСР), $f_{ЗАГР}$ (частота загрузки БСР), ЗАГР/СДВ (режим «Загрузка»/«Сдвиг» БСР), ЧТ/ЗП («Чтение»/«Запись» — режимы работы БУЭ). БСР преобразует содержимое восьми ячеек ЗУ128, считываемых за один такт «Чтения», в полные видеосигналы «КР» (красный), «СН» (синий), «ЗЛ» (зеленый).

Экран ТКП составлен из модулей газоразрядных индикаторных панелей типа ИМГ 3-01 [5].

Блок управления экраном (БУЭ) ТКП содержит БПП, блок интерфейса (БИнт), БЗУ, блок согласования (БС), блок управления памятью (БУП). БИнт, БУП и БС выполнены на микросхемах серии К155, а БЗУ — на базе статической памяти типа К541 РУ1. БПП идентичен БПП контроллера ввода-вывода. Изображение на экране строится по точкам заданием адреса модуля, координат X, Y внутри модуля и кода действия (зажечь, погасить точку, полное стирание экрана).

БУЭ ТКП работает в режимах I (ввод информации), II (ввод с сохранением информации в БЗУ), III (вывод информации из БЗУ), задаваемых кодом режима. Адрес модуля, координаты X, Y, режим работы БЗУ и код действия (в режимах I, II) подаются по цепям Д0...Д15 из контроллера ввода-вывода (вывод основного изображения) либо по цепям Д'0...Д'15 из БМИ (при выводе изображения маркера) тремя словами в следующей последовательности: код координаты X, код координаты Y, код действия и код режима.

В БИнт хранятся адрес модуля, координаты одной точки, код режима, код действия и формируется сигнал запроса на ввод одного слова (ЗП-П).

БС преобразует уровни сигналов для модулей ИМГ 3-01.

БЗУ хранит информацию основного изображения. Адреса ячеек БЗУ — координаты точки, а код действия — информация, записываемая в ячейку БЗУ. При перемещении движущегося изображения (маркер или другой объект) по фону основного изображения последнее не искажается. При этом не требуется его повторный вывод из ДВК-2.

БУЭ работает в режимах III (восстановление точек основного изображения, через которые проходит движущееся изображение) и I (вывод движущегося изображения на экран без запоминания в БЗУ).

На рис. 7, 8 приведены примеры отображения на экране ТКП графической и символьной информации.

Пульт управления и табло коллективного пользования информационно и программно совместимы, имеют единую систему команд и единый интерфейс. Блоки буферных ОЗУ в блоках управления ПО и ТКП освобождают память ДВК-2 от хранения дисплейного файла, а канал ДВК-2 — от регенерации изображения. Блоки маркерных измерений в составах ПО и ТКП позволяют одновременно с обработкой информации (поступающей из аппаратуры управляющего комплекса в ДВК-2), выведя обработанную информацию на экраны ПО и ТКП, анализировать ее, сокращая время занятости процессора ДВК-2. Перечисленные аппаратные реализации отдельных узлов системы повышают быстродействие многотерминальной СОИ.

Опытные образцы СОИ эксплуатируются в качестве подсистем в АСУ вибродинамическими испытаниями, в АСУ моделированием навигационных задач и в качестве технических средств в учебных заведениях.

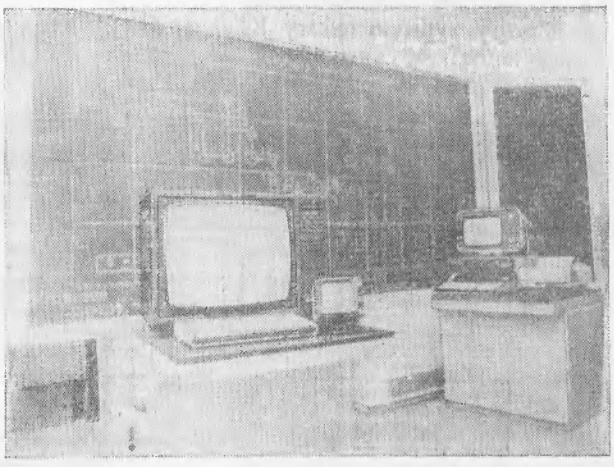


Рис. 7. Отображение на табло коллективного пользования графической информации

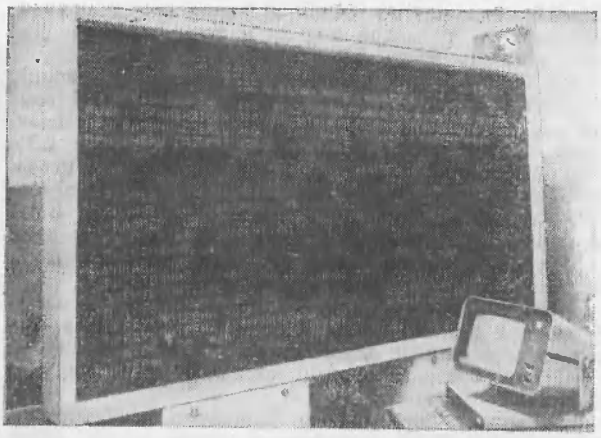


Рис. 8. Отображение на табло коллективного пользования символьной информации

Справки о приобретении технической документации и аппаратуры можно получить по адресу: 370141, г. Баку, ул. Ф. Агаева, 9. Телефоны для справок: 39-74-96, 39-74-94.

ЛИТЕРАТУРА

1. Артамонов А. Г., Болотников В. Н. и др. Применение микропроцессоров в дисплейных системах.— В кн.: Современные методы и устройства отображения информации / Под ред. М. И. Кривошеева и А. Я. Бретбарта.— М.: Радио и связь, 1981, с. 70, 71.
2. Джхунян В. Л., Борщенко Ю. И., Отрохов Ю. Л., Шишарин С. А. Одноплатные микроЭВМ ряда «Электроника МС1201», Микропроцессорные средства и системы, 1985, № 2, с. 8—13.
3. Алексенко А. Г., Галицын А. А., Иванников А. Д. Проектирование радиоэлектронной аппаратуры на микропроцессорах.— М.: Радио и связь, 1984, с. 11—30.
4. Якушев В. С. Цветной телевизионный КАМАК-дисплей.— Автометрия, 1984, № 4, с. 93—96.
5. Карпов В. Г., Милюков М. М., Мясин В. А. Новый подход в построении экранов коллективного пользования. Модуль ИМГ-3.— ПСУ, 1982, № 9, с. 21.

Статья поступила 20 декабря 1985 г.

УДК 681.142

В. С. Безобразов, В. А. Димов, А. В. Мякотин, В. Ю. Сохранов, А. А. Шишкевич

КОНТРОЛЛЕР ГРАФИЧЕСКОГО ДИСПЛЕЯ

Диалоговые вычислительные комплексы (ДВК — 1М, 2М, 3М2) могут выводить на экран графическую информацию. Это достигается введением в ДВК контроллера графического дисплея (КГД) (рис. 1).

Контроллер обеспечивает формирование черно-белого графического изображения размером 286 строк по 400 точек при скорости вывода графической информации до 500К байт/с и работает в трех программно устанавливаемых режимах: выдача на экран растрового монитора только символьной информации; выдача только графической информации; выдача на экран символьной и графической информации одновременно.

В структурную схему КГД (рис. 2) входят следующие функциональные узлы: ГОЗУ, РОИ, РСГИ, УССМ, РИ, УСМ, Г.

Графическое оперативное запоминающее устройство (ГОЗУ) информационной емкостью 16К байт выполнено на восьми БИС динамического ОЗУ типа К565РУ6. Выбранное число точек в строке (400) обеспечивает равный шаг между ними по вертикали и горизонтали. Такая организация графического экрана по сравнению с экраном размером 256×512 точек [1, 2] при равной информационной емкости ГОЗУ существенно упрощает алгоритмы построения окружностей, поворотов фигур и других графических преобразований. Кроме того, она позволяет получать твердую копию изображения без искажений пропорций на мозаичном АЦПУ УВВПЧ-30-004, входящем в комплект ДВК.

Системная магистраль КГД представляет собой систему сигнальных связей, назначение

Рис. 1. Структурная схема ДВК

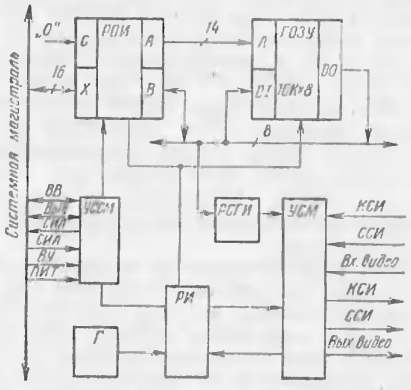


Рис. 2. Структурная схема КГД

и конструктивная реализация которых соответствует интерфейсу ОСТ 11.305.903-80. Связь КГД с другими устройствами ДВК, подключенными к системной магистрали, осуществляется устройством сопряжения с системной магистралью через регистры обмена информацией. КГД всегда является «пассивным» устройством на системной магистрали, цикл обмена не более 4 мкс.

Регистры обмена информацией (РОИ) выполнены на БИС обмена информации (ОИ) КР1802ВВ1 и представляют собой четыре доступных по записи и чтению из системной магистрали 16-разрядных регистра: регистры управления (РУ), адреса (РА), данных (РД) и регистр-счетчик (РС). Все регистры имеют выход на системную магистраль (канал X) и на внутренние магистрали адреса (канал А) и данных (канал Д) ГОЗУ. Адреса регистров РУ, РД, РА и РС на системной магистрали: 176640, 176642, 176644, 176646 соответственно.

Регистр управления используется для разрешения и запрещения выдачи графической и символьной информации на экран монитора и имеет следующий формат: разряды 0...13 не используются; разряды 15 — разрешение выдачи графической информации, при записи в этот разряд «1» на экране дисплея отображается содержимое ГОЗУ; разряд 14 — запрет выдачи символьной информации, при записи в этот разряд «1» символьная информация на дисплей не выдается.

Во время включения питания ДВК при переходе сигнала ПИТ из низкого уровня в высокий в разряды 14 и 15 РУ аппаратно записывается код 00, что соответствует разрешению выдачи символьной и запрещению выдачи графической информации.

Регистр адреса служит для задания адреса элемента изображения в ГОЗУ и имеет формат, в котором разряды 14, 15 не используются, разряды 0...13 — адрес байта графической информации. Принята следующая адресация элементов изображения в ГОЗУ КГД по строкам и байтам в строке:

Строка	Адресация элементов изображения				
	00 000	00 001	00 002	.....	00 049
1	00 050	00 051	00 052	.....	00 099
2	...	...	...	.....	...
.	...	...	...	.....	...
.	14 250	14 251	14 252	.....	14 299
286	1	2	3	.....	50
	Номер байта в строке				

Регистр данных предназначен для буферного хранения байта графической информации и имеет формат, в котором разряды 8...15 не используются, разряды 0...7 — байт графической информации, причем крайней левой точке элемента изображения, состоящего из восьми горизонтальных точек, соответствует нулевой разряд байта. «Лог. 1» в байте графической информации соответствует светящаяся точка на экране монитора. При обмене с системной магистралью производится считывание или запись байта графической информации в ГОЗУ через РД по адресу, предварительно установленному в РА.

Регистр-счетчик в РОИ используется для аппаратного формирования адреса следующего байта графической информации в ГОЗУ, который считывается во время вывода изображения на экран монитора. При этом канал С РОИ служит для записи адреса байта, с которого начинается формирование изображения.

Регистр сдвига графической информации (РСГИ), построенный на ИС К155ИР1, преобразует байт графической информации, считанный из ГОЗУ в последовательный код, который поступает через устройство сопряжения с монитором (УСМ) на вход видеосигнала монитора.

Распределитель импульсов (РИ) задает временную диаграмму работы КГД и синхронизирует с ней асинхронные обращения к КГД по системной магистрали. Процессор ДВК получает возможность доступа в ГОЗУ один раз за 1 мкс. Это позволяет обойтись без сигналов готовности или логики прерываний при обмене.

Устройство сопряжения с монитором (УСМ) по сигналам кадровой и строчной синхронизации (КСИ, ССИ) монитора «Электроника МС6105.01» синхронизирует работу генератора (Г), РС в РОИ и РИ. УСМ из видеосигнала, поступающего из РСГИ, формирует результирующий видеосигнал в соответствии с режимом, предварительно заданным программистом в РУ РОИ. Для обеспечения синхронизации графического изображения с разверткой, формируемой в логическом блоке символьного дисплея, частота кварцевого генератора в КГД в 4 раза превышает частоту выдачи графических точек и составляет 30,8 МГц. Это позволяет снизить до практически незаметного уровня эффект фазового биения графических и символьных точек, что возможно при расхождении частот генераторов дисплея и КГД.

Программирование КГД. Так как аппаратные средства КГД дают только возможность доступа к восьми смежным точкам в строке, необходимо специальное программное обеспе-

чение для установки битов ГОЗУ так, чтобы при выводе на экран формировалось изображение, с определенной точностью соответствующее требуемым геометрическим фигурам. Потенциально большой объем операций над ГОЗУ, а также учет того, что за созданием изображения наблюдает человек с его естественным желанием «получить картинку быстро», предъявляют жесткие требования к скорости работы программ.

В разработанном для КГД ПО для сокращения времени доступа к биту ГОЗУ используются три массива: адресов ГОЗУ, соответствующих началу строк, объемом 286 слов; смещений к адресу в строке объемом 400 байт; масок для установки бита объемом 400 байт. Получение адреса байта ГОЗУ и маски для заданных X и Y производится следующей последовательностью команд микроЭВМ:

```
MOV    X, R0                  занесение X
MOV    Y, R1                  занесение Y
ASL    R1                     переход к словной адресации
MOVB   XADDR(R0), R2
ADD    YADDR(R1), R2          получение адреса ГОЗУ
MOVB   MASK (R0), R3          получение маски для бита
```

Таким образом, имея на регистрах X и Y, для доступа к биту ГОЗУ необходимы пять команд: три для получения адреса и маски и две для операций с РА и РД. При использовании КГД совместно с микроЭВМ «Электроника МС 1201.02» операция установки точки занимает в среднем 25 мкс.

Для растровой развертки векторов используется программная реализация цифрового дифференциального анализатора (ЦДА), которая дает большую скорость интерполяций при выбранной архитектуре КГД и микроЭВМ, чем широко используемый метод Брезенхэма [3], за счет более эффективного использования регистров центрального процессора. При работе этой программы переход от бита к биту в строке сопровождается лишь сдвигом маски в регистре ЦП, и, возможно, увеличением РА на единицу. Переход в строках сопровождается прибавлением константы +50 или —50 к РА. В результате при выводе векторов для вычисления и вывода очередной точки необходимо выполнить максимум 10 команд, операнды которых расположены на регистрах. Среднее время вывода точки при развертке вектора в ГОЗУ составляет 30 мкс.

Разработанное для КГД базовое программное обеспечение для достижения быстродействия широко использует методы, аналогичные описанным, и позволяет производить: построение символов двух шрифтов с поворотами и масштабированием; построение прямых и дуг восемью типами линий; задание окон и полей вывода с автоматическим масштабированием

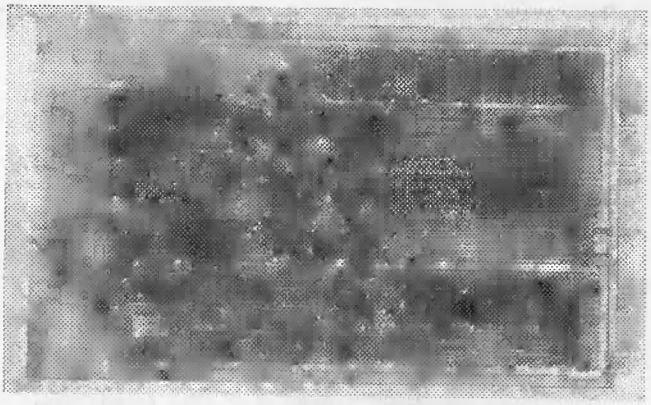


Рис. 3. Контроллер графического дисплея

по полю вывода; заполнение гранично-определенных областей в соответствии с восемью образцами штриховки. Все операции выполняются в режимах заполнения, стирания, инверсии.

На основе этого базового графического программного обеспечения произведена постановка графического пакета для языка Фортран, совместимого по вызовам с широко распространенным пакетом ПЛОТ-10, что позволяет использовать с КГД программы, написанные для дисплея Тектроникс 4010. Базовое программное обеспечение позволяет также использовать языки Паскаль и ассемблер для выполнения графических операций.

Конструктивно КГД выполнен на полуплате микроЭВМ «Электроника 60» (рис. 3) и имеет габаритные размеры не более 240×135×12,5 мм. Подключение КГД к системной магистрали ДВК осуществляется через розетку типа РППМ-288 ГеО.364.209ТУ, к выходам кадровой и строчной синхронизации, входу и выходу видеосигнала — через два разъема ОНП-КГ-56-10 НЦО.364.077ТУ.

Напряжение питания КГД +5 В от источника питания ДВК, потребляемая мощность не более 10 Вт. КГД в составе ДВК «Электроника НЦ8020/3М2» сохраняет работоспособность при условиях эксплуатации по группе 2 ГОСТ 21552—76.

За справками обращаться по тел. 534-54-71 (Москва).

ЛИТЕРАТУРА

1. Бахмацкий В. Д., Белый В. Г., Большинский С. М. и др. Графическая приставка к растровому знаковому дисплею.— Приборы и техника эксперимента, 1984, № 2, с. 53—57.
2. Семенов П. А. Микроконтроллеры на базе БИС КР580 для микроЭВМ «Электроника 60», «Электроника НЦ-80-01Д».— Микропроцессорные средства и системы, 1985, № 3, с. 42—45.
3. Фоли Дж., Дэм А. вэн. Основы интерактивной машинной графики. Т 2 — М.: Мир, 1985, с. 139—143.

Статья поступила 7 марта 1986 г.

УДК 681.322.1

В. Ю. Романов, В. Н. Барышников, М. А. Воронов, Ф. И. Паначев

ПЕРСОНАЛЬНАЯ ЭВМ «ИРИША»: ПЕРИФЕРИЙНЫЕ УСТРОЙСТВА, ИСТОЧНИК ПИТАНИЯ

(Продолжение цикла статей по компьютеру «Ириша». Начало см. в «МП» № 3, 1985 г., № 1, 2, 1986 г.)

Периферийные устройства

Телевизионный монитор. Монитором может служить специальное устройство или доработанный бытовой телевизор. Модуль телевизионного адаптера [1] вырабатывает несколько различных выходных сигналов, предназначенных для работы как с черно-белым, так и цветным монитором.

На выходе «Видео» ПЭВМ формируется полный телевизионный сигнал, содержащий все необходимые для работы стандартного телевизора сигналы синхронизации. Этот сигнал может быть непосредственно подан на входы телевизоров, предназначенные для работы с видеомагнитофонами. У большинства моделей телевизионных приемников: «Электроника ВЛ100», «Электроника 407», «Сапфир 401», «Юность 405», «Юность

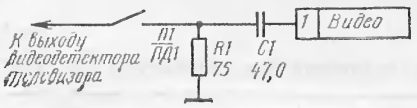


Рис. 1. Схема организации входа для подачи сигнала непосредственно на выход видеодетектора

406» и др. такой вход отсутствует, но его несложно организовать путем простой доработки (рис. 1). Доработка сводится к установке дополнительного разъема и переключателя П1 типа ПД1. Переключатель позволяет использовать телевизор и для просмотра передач. Суть переделки состоит в организации входа для подачи сигнала непосредственно на выход видеодетектора через соответствующий переключатель. Если переключатель разомкнут, то схема не влияет на работу приемника. При замыкании переключателя выходной сигнал видеодетектора шунтируется за счет подключения низкоомного резистора R1 и телевизор работает как монитор. Для устранения возможных помех рекомендуется переключатель каналов телевизора установить на свободный канал или поставить в промежуточное положение.

Полный видеосигнал не несет информации о цвете, поэтому для получения цветного изображения необходимо использовать сигналы управления цветом R, G, B, вырабатываемые модулем телевизионного адаптера. Серийные цветные телевизоры входов R, G, B не имеют и требуют более сложной доработки. Такую доработку простыми средствами можно выполнить в тех конструкциях телевизионных приемников, которые имеют отдельные видеоусилители по каждому из первичных цветов. Этому требованию удовлетворяют все унифицированные телевизоры, выпускаемые в настоящее время. Доработка ламповых телевизоров более ранних моделей, имеющих один широкополосный видеоусилитель, слишком сложна и нецелесообразна.

Ниже описывается доработка телевизоров УПИЦТ-32-10 («Юность Ц-404», «Шилялис Ц-410»), 2УСЦТ-51-3 («Горизонт Ц-355», «Рекорд ВЦ-311») и УПИМЦТ-61-С-2 («Рубин Ц-202» и так далее) с размерами экрана 32, 51 и 61 см по диагонали. Телевизоры с меньшим размером экрана, как показала опытная эксплуатация, не позволяют получать удовлетворительное качество изображения из-за низкой разрешающей способности кинескопов.

Для организации входов R, G, B в телевизорах указанных типов необходимо установить плату согласующих усилителей и дополнительный разъем. Принципиальная схема согласующей платы приведена на рис. 2,

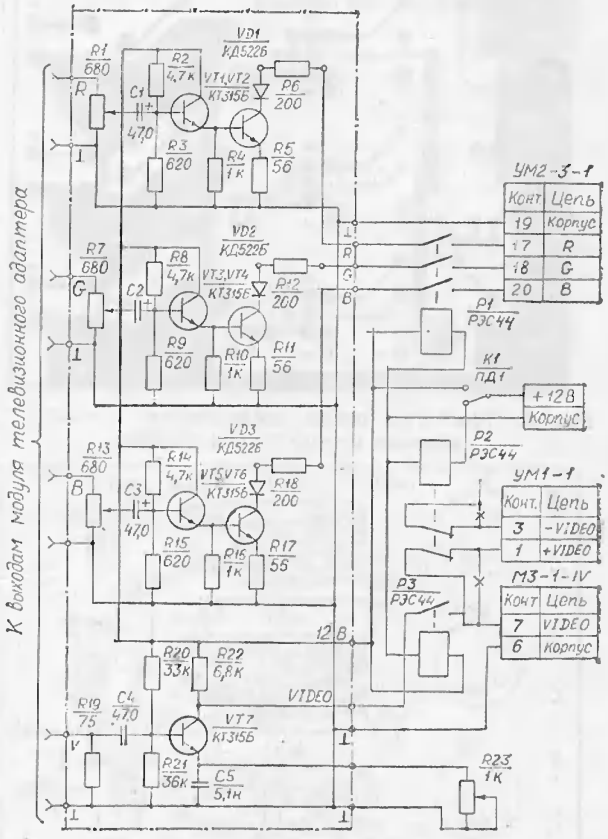
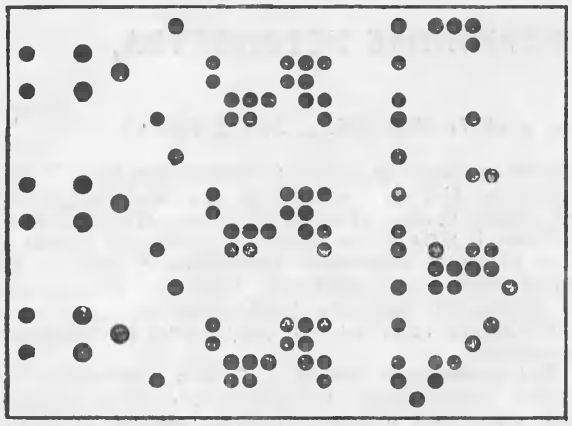


Рис. 2. Принципиальная схема согласующих усилителей: *х* — разрывы проводников для установки дополнительных реле

топология и монтажная схема на рис. 3 и 4 соответственно. На плате расположены три усилителя каналов R, G, B и усилитель сигнала синхронизации. Переменные резисторы R1, R7, R13 служат для регулировки размаха видеосигнала по каждому цвету, а R23 — для установки амплитуды сигнала синхронизации. При доработке телевизоров УПИМЦТ-61-С-2 емкость С4 должна быть заменена на диод (Д220 или ему подобный). Для телевизоров УПИЦТ-32-10 резисторы R6, R12, R18 должны быть закорочены перемычками. Номера кон-

а)



б)

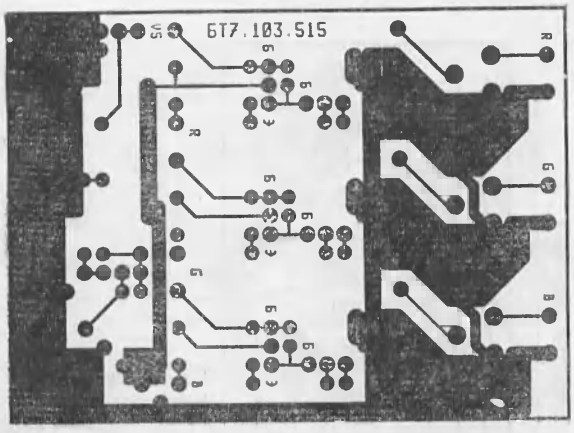


Рис. 3. Топология платы согласования: *а* — лицевая сторона; *б* — оборотная сторона

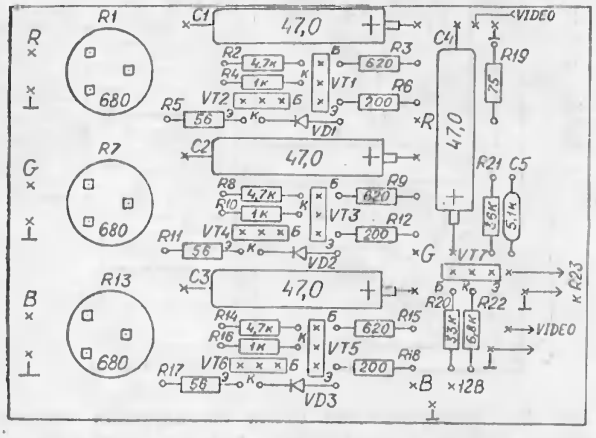


Рис. 4. Монтажная схема платы согласования

тактов на разъемах (см. рис. 2) соответствуют обозначениям на принципиальных схемах указанных телевизоров.

Возможны два варианта доработки. В первом превращается телевизор только в RGB-монитор. Во втором режим монитора дополняет прямое использование телевизора по назначению. В варианте 1 согласующая плата монтируется в телевизорах УПИЦТ-32-10 («Юность Ц-404», «Шилялис Ц-410») на кронштейне селектора каналов, а магнитофонный выход телевизора служит дополнительным разъемом для организации R-, G-, B-входов и сигнала «видео». В этом варианте модули видеоусилителей и радиоканала могут быть изъяты.

В варианте 2 кроме согласующей платы требуется установить дополнительные реле для переключения режимов работы. На рис. 2 показано положение этих реле при работе телевизора в режиме монитора. Реле рекомендуется устанавливать вблизи коммутируемых цепей, в разрыв проводников. Для крепления платы усилителей необходим кронштейн. Переключатель для отключения звука телевизора в варианте 2 используется в качестве переключателя режимов работы телевизор — монитор (К1).

Клавиатура. Общая компоновка ПЭВМ «Ириша» предполагает наличие выносной клавиатуры, соединенной гибким кабелем с системным блоком. Вариант компоновки с отдельной клавиатурой имеет ряд преимуществ: он позволяет достаточно просто подключить одну из серийно выпускаемых клавиатур, например типа 15ВВВ-97-006, для чего достаточно изготовить соединительный кабель (табл. 1). Однако известные и доступные клавиатуры серийного производства дороги, имеют большую потребляемую мощность и габаритные размеры.

Таблица 1

Таблица соединений для клавиатуры 15ВВВ-97-006(005)

Модуль процессора (разъем СН051-20-2-В)		Клавиатура 15ВВВ-97-006(005) (разъем СН051-40-2-В)	
Контакт	Сигнал	Контакт	Сигнал
1	$\overline{\text{STB}}$	4	$\overline{\text{СТР}}$
2	$\overline{\text{PESET}}$	5, 1	$\overline{\text{ОЧС}}$, $\overline{\text{СБРОС}}$
3	$\overline{\text{IBF}}$	10	ПРД
4, 6, 8 10, 12, 14	ОВ	6—9, 11, 13, 17 21—27, 30, 38	ОВ
5	КВ7	20	ЛАТ.Д
7	КВ6	32	ВК7
9	КВ5	34	ВК6
11	КВ4	28	ВК5
13	КВ3	18	ВК4
15	КВ2	12	ВК3
16, 18, 20	5 В	40	5 В
17	КВ1	14	ВК2
19	КВ0	16	ВК1

Примечание. Сигналы $\overline{\text{ОЧС}}$ и $\overline{\text{СБРОС}}$ перемкнуть на разъеме клавиатуры. Кабель выполнить витыми парами или ленточным кабелем с чередующимися сигнальными линиями и линиями, соединенными с ОВ.

Клавиатура ПЭВМ «Ириша» выполнена в виде печатной платы, на которой смонтированы клавиши, соединенные в координатную матрицу, и схема управления (рис. 5). Плата установлена в декоративный корпус размером 328×152×30 мм. Клавиатура потребляет от источника питания напряжением 5 В не более 30 мА в режиме покоя, а при нажатии на клавишу (в момент передачи кода процессору) — не более 100 мА.

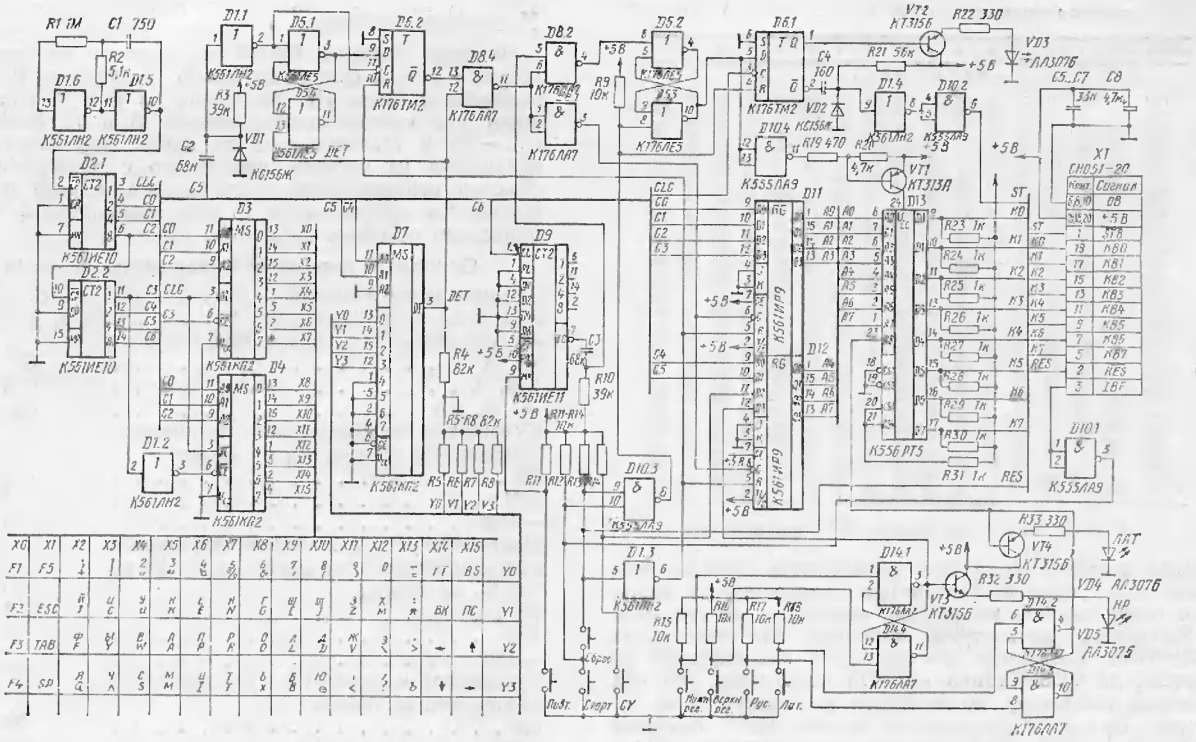


Рис. 5. Схема управления клавиатурой

Схема управления клавиатурой обеспечивает опрос состояния матрицы клавиш, фиксацию координат нажатой клавиши, устранение дребезга контакта, преобразование координат символа согласно ГОСТ 13052—74 и последующую передачу этого кода в параллельном виде в модуль процессора. Задающий генератор выполнен на микросхеме D1 и вырабатывает тактовую частоту 100 кГц для счетчика D2. С выходов счетчика сигналы поступают на дешифраторы столбцов и строк контактной матрицы D3, D4, D7. Матрица клавиш состоит из 16 столбцов и 4 строк; таким образом, на полный цикл ее опроса требуется 64 такта. В момент фиксации нажатия клавиши содержимое счетчика, являющееся координатами нажатой клавиши, переносится в промежуточный регистр D11, D12. Из информации о координатах нажатой клавиши и состоянии сигналов СУ, РУС/ЛАТ, НР/ВР в ПЗУ D13 вырабатывается код символа для передачи в процессор [2]. Схема устранения дребезга построена на микросхеме D6 и работает следующим образом. В исходном состоянии, когда ни одна из клавиш не нажата, оба триггера D6.1, D6.2 установлены в состояние «Лог. 0». При этом на инверсном выходе триггера D6.2 установлена «Лог. 1». При появлении сигнала DET, возникшего во время опроса матрицы, через логический элемент D8.1 стробируется запись содержимого счетчика в промежуточный регистр и активируется узел передачи кода. Одновременно триггер D6.1 устанавливается в состояние «Лог. 1». По завершении опроса матрицы триггер D6.2 устанавливается в состояние «Лог. 1» и его инверсный выход блокирует прохождение сигнала DET. Если в следующих циклах клавиша остается нажатой, то это не приводит к срабатыванию схемы передачи кода. Когда клавиша отпущена, по сигналу конца опроса триггеры D5.1, D5.4 сбрасываются и, если в течение следующего цикла ни одна из клавиш не нажата, сбрасывается и триггер D6.2. Узел снова готов к работе.

Таким образом, на каждое нажатие клавиши схема устранения дребезга вырабатывает только один запускающий импульс. Дребезг клавиш для этой схемы не имеет особого значения, так как процесс фиксации моментов нажатия и отпускания клавиши разнесены во времени. Запускающий импульс, выработанный схемой устранения дребезга, поступает в узел передачи кода и обрабатывается в два этапа. Сначала устанавливается в активное состояние триггер, собранный на элементах D5.2 и D5.3 и управляющий питанием ПЗУ, а через 10 мкс после включения питания в активное состояние устанавливается и триггер D6.1. В это время схема на элементе D1.4 вырабатывает импульс длительностью 3...5 мкс, стробирующий выходную информацию. После формирования строба клавиатура блокируется до приема сигнала подтверждения IBF, свидетельствующего о том, что выработанный код принят. Из обратного фронта этого сигнала вырабатывается сигнал сброса блокировки, и клавиатура готова к передаче следующего кода.

В клавиатуре предусмотрена возможность многократного повторения передачи последнего набранного кода. Узел автоповтора состоит из счетчика D9 и формирователя импульсов на элементах C3, R10, D8.2. В исходном состоянии на соответствующий вход счетчика подается сигнал Сброс, блокирующий его работу. При нажатии на клавишу ПОВТ узел автоповтора формирует импульс, запускающий схему передачи кода. При этом сигнал записи в промежуточный регистр не вырабатывается. Матрица клавиш собрана из кнопок ПК8. Поскольку сигналы опроса принимаются КМОП-микросхемами, имеющими высокоомный вход, то для замыкания узлов матрицы можно использовать любую контактную пару с переходным сопротивлением менее 1 кОм, например проводящую резину. Кодировка символов, вырабатываемых клавиатурой, определяется содержимым ПЗУ D13 (рис. 6).

В матрицу не входят клавиши переключения регистров, выбора рабочего алфавита, управления (СУ), автоматического повтора и сброса. Клавиатура имеет две клавиши сброса, различающиеся по функциям,— СБРОС и ЛОК. СБРОС. При нажатии на клавишу ЛОК. СБРОС элемент D1.3 вызывает разблокирование

FILENAME > OLD.KBD.ROM/1

ADRESS X=	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
								- DATA (HEX) -								
00X	81	85	2B	31	32	33	34	35	36	37	38	39	30	3D	09	08
01X	82	1B	0A	03	15	0B	05	0E	07	1B	1D	1A	08	2A	0D	0A
02X	83	1B	06	19	17	01	10	12	0F	0C	04	16	1C	3E	1A	1C
03X	84	20	11	1E	13	0D	09	14	1B	02	00	3C	3F	7F	1D	19
04X	81	85	2B	21	22	23	24	25	26	27	28	29	30	3D	09	08
05X	82	1B	6A	63	75	6D	65	6E	67	7B	7D	7A	6B	2A	0D	0A
06X	83	1B	66	79	77	61	70	72	6F	6C	64	76	7C	3E	1A	1C
07X	84	20	71	7E	73	6D	69	74	78	62	60	3C	3F	5F	1D	19
08X	81	85	3B	31	32	33	34	35	36	37	38	39	30	2D	09	08
09X	82	1B	0A	03	15	0B	05	0E	07	1B	1D	1A	08	3A	0D	0A
0AX	83	1B	06	19	17	01	10	12	0F	0C	04	16	1C	2E	1A	1C
0BX	84	20	11	1E	13	0D	09	14	1B	02	00	2C	2F	7F	1D	19
0CX	81	85	3B	31	32	33	34	35	36	37	38	39	30	2D	09	08
0DX	82	1B	4A	43	55	4B	45	4E	47	5B	5D	5A	4B	3A	0D	0A
0EX	83	1B	46	59	57	41	50	52	4F	4C	44	56	5C	2E	1A	1C
0FX	84	20	51	5E	53	4D	49	54	5B	42	40	2C	2F	7F	1D	19
10X	81	85	2B	21	22	23	24	25	26	27	28	29	30	3D	09	08
11X	82	1B	0A	03	15	0B	05	0E	07	1B	1D	1A	08	2A	0D	0A
12X	83	1B	06	19	17	01	10	12	0F	0C	04	16	1C	3E	1A	1C
13X	84	20	11	1E	13	0D	09	14	1B	02	00	3C	3F	7F	1D	19
14X	81	85	2B	21	22	23	24	25	26	27	28	29	30	3D	09	08
15X	82	1B	CA	C3	D5	CB	C5	CE	C7	DB	DD	DA	C8	3A	0D	0A
16X	83	1B	C6	D9	D7	C1	D0	D2	CF	CC	C4	D6	DC	3E	1A	1C
17X	84	20	D1	DE	D3	CD	C9	D4	DB	C2	C0	3C	3F	5F	1D	19
18X	81	85	3B	31	32	33	34	35	36	37	38	39	30	2D	09	08
19X	82	1B	0A	03	15	0B	05	0E	07	1B	1D	1A	08	3A	0D	0A
1AX	83	1B	06	19	17	01	10	12	0F	0C	04	16	1C	2E	1A	1C
1BX	84	20	11	1E	13	0D	09	14	1B	02	00	2C	2F	7F	1D	19
1CX	81	85	3B	31	32	33	34	35	36	37	38	39	30	2D	09	08
1DX	82	1B	BA	B3	F5	EB	E5	EE	E7	FB	FD	FA	E8	3A	0D	0A
1EX	83	1B	E6	F9	F7	E1	F0	F2	EF	EC	E4	F6	FC	2E	1A	1C
1FX	84	20	F1	FE	F3	ED	E9	F4	F8	E2	E0	2C	2F	DF	1D	19

Рис. 6. Карта прошивки ПЗУ клавиатуры

схемы передачи кода при неполучении сигнала подтверждения приема. Только при одновременном нажатии обеих клавиш клавиатура формирует сигнал RES.

Индикатор клавиатуры выполнен на светодиодах АЛ307БМ. Состояние триггера D6.1 индицируется на светодиоде VD3. Сигнал на VD3 показывает, что код передан процессору, но не считан (в клавиатуру не поступил сигнал подтверждения приема IBF). Клавиши ВР, НР и РУС, ЛАТ устанавливают триггеры D14.1, D14.4 и D14.2, D14.3, управляя через адресные входы D13 кодом на выходе. Состояние триггеров индицируется на светодиодах VD4 ЛАТ и VD5 НР и отражает режим работы клавиатуры.

Источник питания

Источник питания ПЭВМ смонтирован на передней несущей панели системного блока. Электрическая принципиальная схема ИП приведена на рис. 7. Источник имеет два канала стабилизаторов: 5 и 12 В. Источник — 15 В стабилизатора не имеет. Это напряжение получается из сигнала, снимаемого с дополнительной обмотки накопительной индуктивности канала 5 В, и появляется одновременно с этим напряжением. Ниже приведены основные параметры ИП.

Основные параметры источника питания

Входное напряжение, В	18...24
Выходное напряжение, В	+5, +12, —15
Максимальный ток нагрузки канала:	
5В, А	5
12В, А	0,2
—15В, мА	50
Суммарная нестабильность выходного напряжения канала, %, не более:	
5В	3
12В	3
—15В	10
Нестабильность выходного напряжения при изменении тока нагрузки от 10 до 100 %, не более:	
5В	1
12В	1
—15В	3
Суммарное напряжение пульсаций всех типов, мВ, не более:	
5В	75
12В	100
—15В	100

Рис. 7. Электрическая принципиальная схема источника питания

Канал стабилизатора 12 В собран по компенсационной схеме на базе интегрального стабилизатора КР142ЕН1Г (D4) с проходным транзистором КТ814Б (VT1-2). Работа этого канала, за исключением узла блокировки, особых пояснений не требует. Узел блокировки обеспечивает следующий порядок появления выходных напряжений: —15, 5, а затем 12 В, и запрещает формирование напряжения 12 В при отсутствии на выходе напряжений 5 и —15 В. При снятии входного напряжения порядок сброса выходных напряжений обратный. В узел входят два транзистора VT7 и VT8. При нормальной работе ИП транзистор VT7 открыт, VT8 закрыт. При уменьшении напряжения —15 В напряжение на базе транзистора VT8 повышается, он открывается, а транзистор VT7 закрывается, блокируя работу стабилизатора D4. При исчезновении напряжения 5 В VT7 также закрывается из-за отсутствия напряжения на базе, блокируя работу канала 12 В. Переменным резистором R46 устанавливается точное значение выходного напряжения этого канала.

Канал стабилизатора 5 В, через который проходит основная мощность, потребляемая ПЭВМ, построен по схеме ключевого стабилизатора с широтно-импульсной модуляцией (ШИМ). Его рабочая частота около 25 кГц. Узел генератора ШИМ сигнала выполнен на базе транзисторной сборки КР198НТ1А (D2). Он состоит из задающего мультивибратора и управляемого одновибратора. Использование одновибратора в схеме ШИМ позволило повысить помехозащищенность стабилизатора и тем самым предотвратить случайные сбои. На транзисторах второй сборки КР198НТ1А (D3) выполнены усилитель разбаланса и часть схемы ограничителя тока. Выходное напряжение ШИМ управляет работой ключевого каскада — VT6, VT5 и VT1-1. В качестве сило-

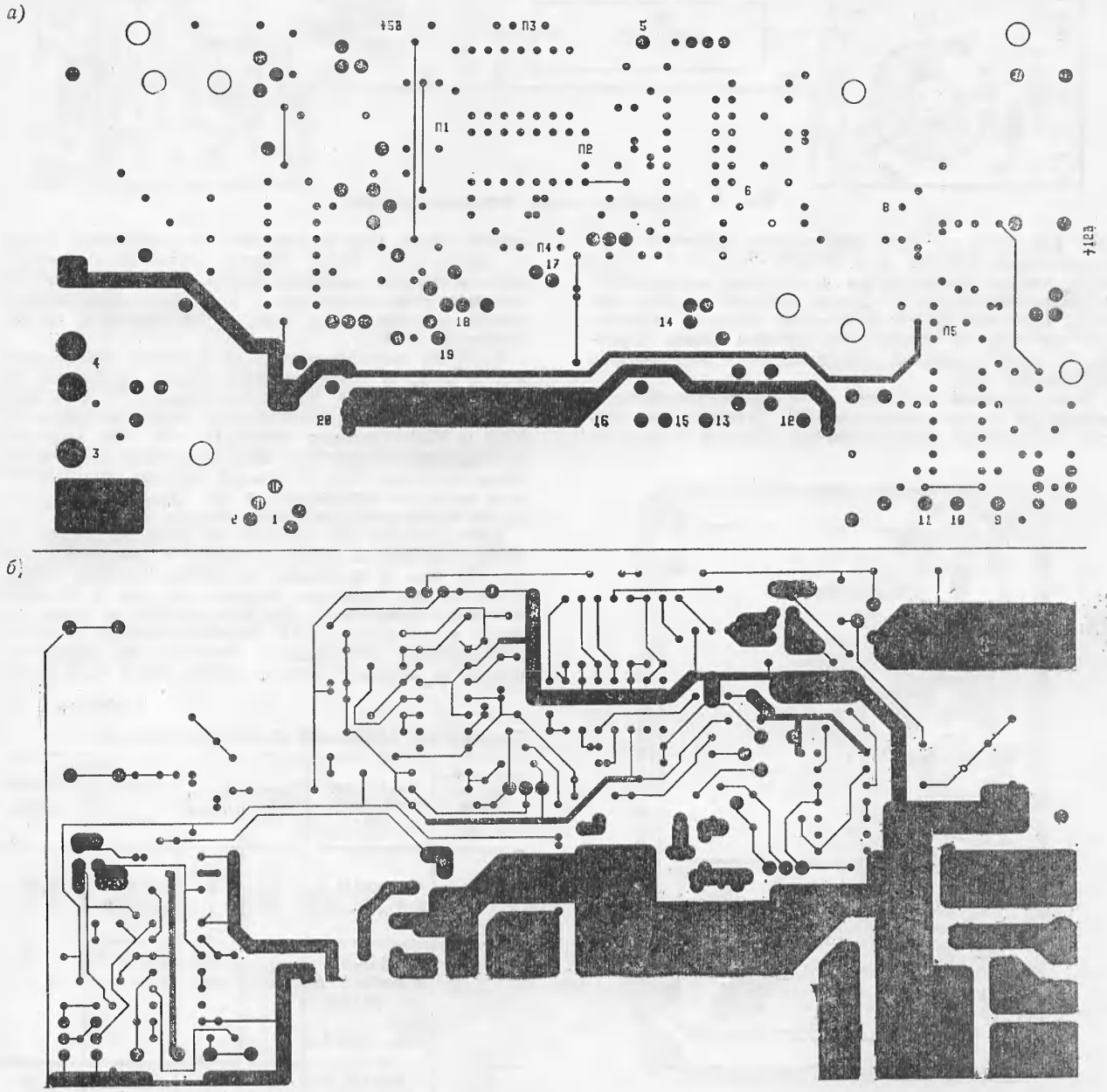


Рис. 8. Топология печатной платы источника питания: *а* — лицевая сторона; *б* — оборотная сторона

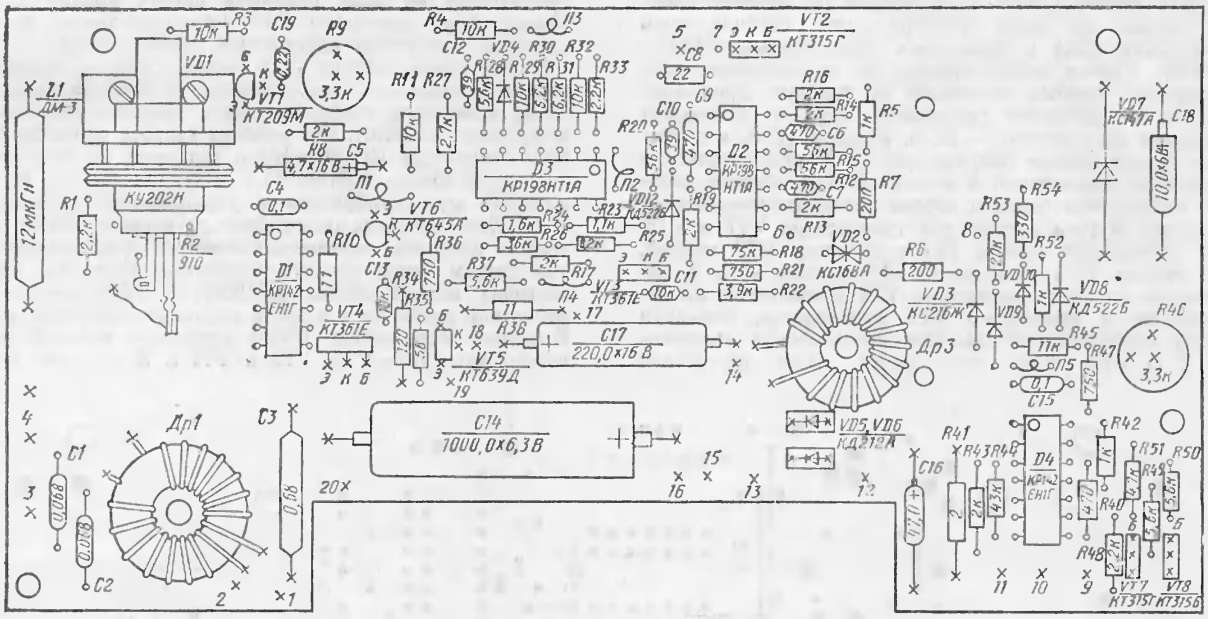


Рис. 9. Монтажная схема источника питания

вого ключевого элемента используется составная пара транзисторов КТ639Д и КТ805БМ. Выходной сигнал стабилизатора сравнивается с опорным напряжением в дифференциальном усилителе, который, в свою очередь, управляет ШИМ. В качестве опорного напряжения используется напряжение питания схемы управления, вырабатываемое интегральным стабилизатором КР142ЕН1Г (D1).

В ограничитель выходного тока, кроме транзисторов сборки D3, входит транзистор VT4. Датчиком тока служит специальный безындуктивный резистор в цепи силового ключа. При выходе тока за допустимые пределы срабатывает схема защиты, ограничивая длительность открытого состояния ключа. При этом из стабилизатора напряжения канал 5 В фактически превращается в стабилизатор тока, ограничивающий его на уровне 5,5...6 А.

Контроль перенапряжений на выходах стабилизаторов 5 и 12 В осуществляется схемой, состоящей из транзисторов VT1 и VT2 и тиристора VD1. При превышении допустимых напряжений через стабилитроны VD2 и VD3 начинает протекать ток, что приводит к отпиранию транзисторов защиты, вызывающему включение тиристора VD1. Открытый тиристор закорачивает цепь входного напряжения, а это приводит к пережиганию предохранителя в цепи питания 24 В.

Конструктивно ИП состоит из печатной платы, на которой находятся схема управления и выходные фильтры. На рис. 8 приведена топология печатной платы. Расположение элементов показано на рис. 9. Технологические перемычки П1...П5 используются во время наладки и регулировки ИП. Тепловыделяющие элементы смонтированы специальным способом на дюралевом основании передней панели, выполняющей роль тепло-

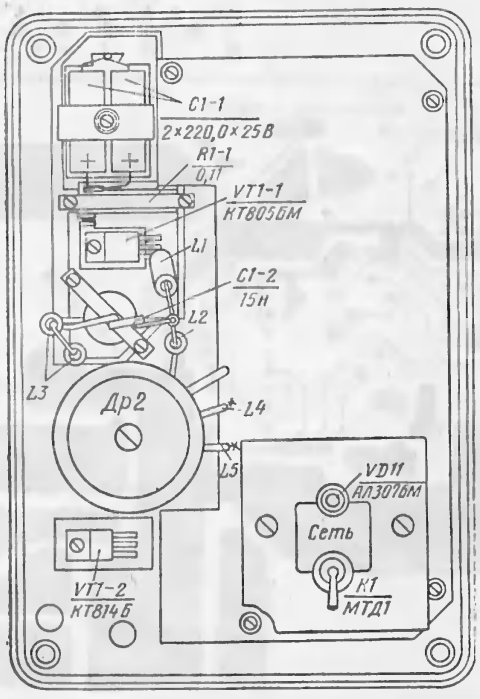


Рис. 10. Компоновка передней панели системного блока

Таблица 2

Параметры дросселей источника питания

Обозначение на схеме (рис. 7 . . .10)	Сердечник: типоразмер, мм	Номер обмотки	Число витков	Провод	
				марка	диаметр, мм
Др1	М2000НМ-А К17×10×4,5	2 2	8 8	МГШВ МГШВ	0,35 0,35
Др2	МП 140-4 К36×25×15 два размером К36×25×7,5	1 2	44 105	ПЭТВ-1 ПЭТВ-1	1,04 0,41
Др3	МП 140-4 К12×6×4	1	15	МГШВ	0,35

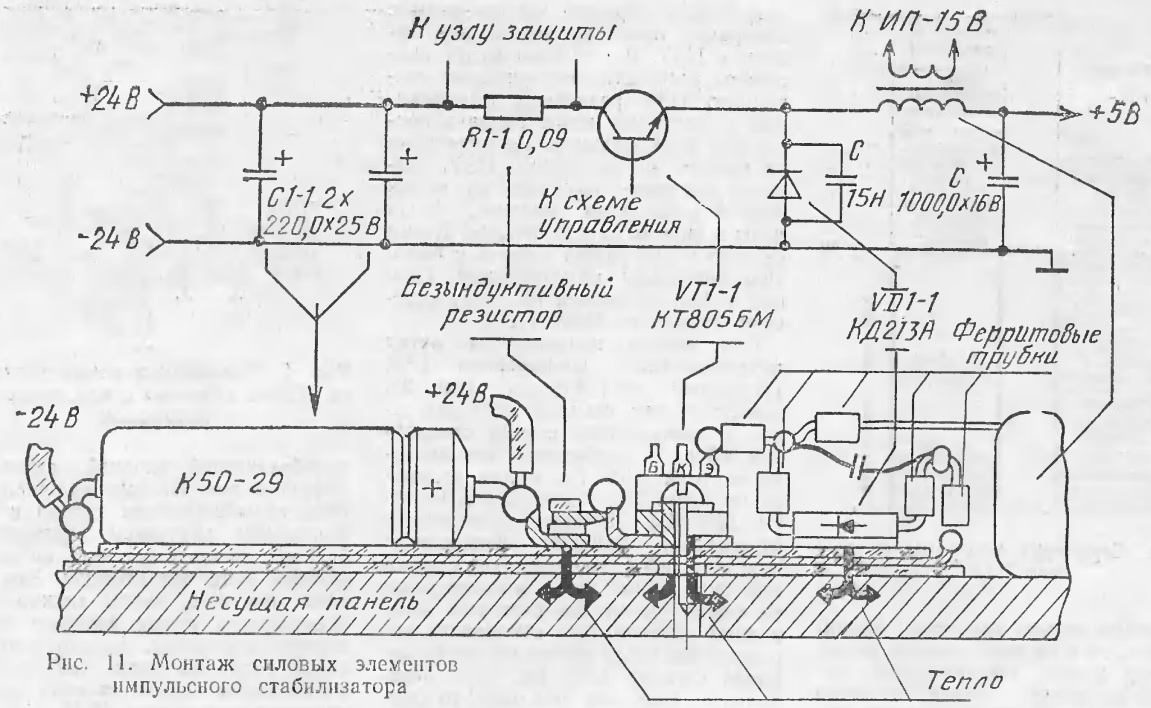


Рис. 11. Монтаж силовых элементов импульсного стабилизатора

отвода (рис. 10). Специальный монтаж силовых элементов импульсного стабилизатора (рис. 11) позволил резко снизить уровень коммутационных помех за счет уменьшения площадей излучающих контуров при одновременном улучшении технологичности. Силовые элементы монтируются к основанию через многослойную прокладку. Первый слой образует — слюда толщиной Н=0,05 мм, над ним расположен слой медной фольги (Н=0,1 мм), соединенный с общим выводом стабилизаторов. Над этим слоем расположен еще один слой слюды (Н=0,05 мм), на котором и смонтированы силовые элементы. Безындуктивный резистор представляет собой свернутую пополам полоску из нержавеющей стали марки Х18Н10Т толщиной 0,03 мм и сопротивлением 0.09 Ом. Он крепится к основанию ИП специальной планкой.

Для уменьшения помех на выводах диода VD1-1 и трансформатора Др2 надеты ферритовые трубки, обозначенные L1...L5 на рис. 10. Параметры моточных изделий ИП приводятся в табл. 2

ЛИТЕРАТУРА

1. Романов В. Ю., Барышников В. Н., Воронов М. А., Паначев Ф. И.— Графические возможности персональной ЭВМ «Ириша».— Микропроцессорные средства и системы, 1986, № 1, с. 61—72.
2. Модуль процессора персональной ЭВМ «Ириша»/ Барышников В. Н., Быстров В. П., Воронов М. А. и др.— Микропроцессорные средства и системы, 1986, № 2, с. 52—62.

Статья поступила 7 апреля 1986 г.

УДК 681.322.068

В. Н. Барышников, М. А. Воронов, Ю. В. Галутин, В. Ю. Романов, А. Л. Рушайло-Арно

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ПЭВМ «ИРИША»

Программное обеспечение ПЭВМ «Ириша» составляют резидентные программы, содержащиеся в системном ПЗУ, и программы, загружаемые в оперативную память с внешних носителей. Первые жестко связаны со структурой и возможностями аппаратных средств ПЭВМ и в известном смысле являются их продолжением, эмулируя работу стандартных периферийных устройств. В модуле процессора предусмотрены две розетки для установки микросхем системного ПЗУ: двух микросхем К573РФ2 или К573РФ5 емкостью 2К байт каждая (минимальный комплект) или микросхем емкостью 8К байт. Общий объем ПЗУ может изменяться от 4К байт минимального комплекта до 16К байт. На рис. 1 показана общая структура программ, записываемых в ПЗУ. Эти программы работают в первом распределении памяти, где располагается дисплейная часть ОЗУ. В процессе работы большинства программ производится переключение карт распределения памяти, что приводит к замене области ПЗУ и дисплейной части ОЗУ на равные по объему сегменты рабочей части памяти. Доступ к программам ПЗУ происходит в этом случае через специальную таблицу входов, хранимую в области связей оперативной памяти. Структура входов программ, записанных в ПЗУ, позволяет автоматически расширять возможности путем простой замены микросхем меньшей информационной емкости на микросхемы большей емкости. В настоящее время подготовлена к записи в ПЗУ версия интерпретатора Бейсик, близкого по возможностям к Бейсику стандарта MSX объемом около 16К байт.

Программы, загружаемые с внешних носителей, могут быть разбиты на две категории. Если ПЭВМ уком-

Рис. 1. Структура программ системного ПЗУ

плектована только кассетным магнитофоном, то в их число входит интерпретатор Бейсик (загружаемый вариант), редактор текстов, игровые программы, а также прикладные программы, подготовленные на дисковых вариантах ПЭВМ. Для дисковых вариантов подготовлена операционная система, аналогичная по возможностям системе ОС 1800. Она позволяет работать со всем набором языков программирования высокого уровня. Для деловых применений может оказаться полезной система подготовки дел ЭКОНОМИКА. Для вариантов ПЭВМ с расширенным объемом памяти ставится операционная система CP/M версии 3.0. Общая структура программного обеспечения ПЭВМ «Ириша» показана на рис. 2.

Описание программного обеспечения ПЭВМ «Ириша» мы начинаем с программ, предназначенных для записи в ПЗУ. В основном будут приведены распечатки содержимого системного ПЗУ различных модификаций с описанием точек входа в программы и их функций. Но отдельные программы из системного ПЗУ, значение которых выходит за рамки данной реализации машины, будут даны в виде исходных текстов. К ним относится программа работы с бытовым кассетным магнитофоном. Способ и формат записи на ленту отвечают стандарту MSX [1, 2].

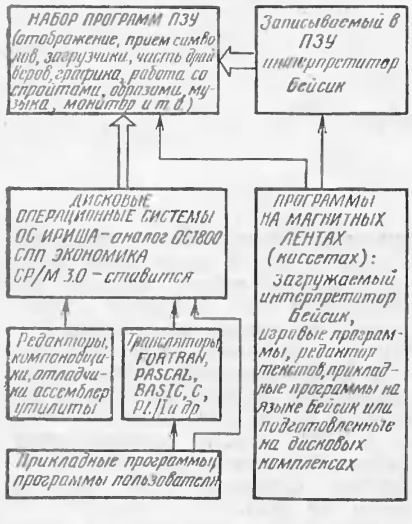


Рис. 2. Общая структура программного обеспечения ПЭВМ «Ириша»

Для записи используется метод двухчастотного кодирования FSK (Frequency shift Keying) (рис. 3), известный как стандарт «Kansas city». В соответствии с этим стандартом «Лог. 1» необходимы два периода частоты 2400 Гц, а для кодирования «Лог. 0» — один период частоты 1200 Гц. Данные записываются отдельными байтами асинхронно: каждому байту предшествует стартовый бит, равный нулю, а после записи формируются два стоповых бита, равных единице. При считывании определяется число переходов за $^3/_4$ периода сигнала 1200 Гц. Если переходов не было или был один, то фиксируется «Лог. 0». Если прошли два или три перехода, то «Лог. 1». Если же переходов было больше трех, то фиксируется ошибка. Между отдельными байтами могут быть промежутки чистой ленты произвольной длины. Изменение скорости движения ленты в пределах нескольких процентов не имеет значения, поскольку синхронизация производится по стартовому биту.

MSX-стандарт предусматривает разделение блоков данных специальными маркерами-заголовками: коротким и длинным. Короткий заголовок представляет собой 4000 периодов сигнала частоты 2400 Гц и используется для разделения блоков внутри файлов. Длинный состоит из 16 000 периодов той же частоты и служит для выделения отдельных файлов.

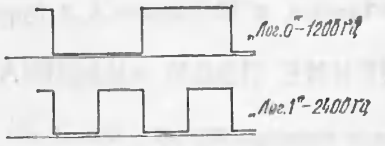


Рис. 3. Кодирование данных для записи на магнитофон

Применение стандарта MSX позволяет записывать и хранить на одной кассете МК-60 до 700К байт информации при скорости считывания до 100 байт/с.

Аппаратная часть интерфейса очень проста (рис. 4). Для записи требуется только буферный усилитель с фильтром верхних и нижних частот, а для воспроизведения — компаратор,

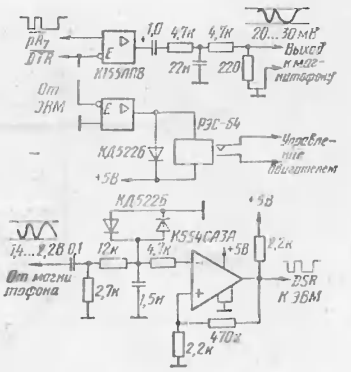


Рис. 4. Упрощенная схема узла связи ПЭВМ «Ириша» с кассетным магнитофоном

преобразующий входной сигнал в цифровой вид. На выходе канала записи вырабатывается сигнал с эффективным значением напряжения 5...7 мВ, который подается на микрофонный вход магнитофона. Это сделано для того, чтобы можно было использовать самые дешевые магнитофоны-диктофоны, имеющие только такой вход. На этом же разъеме магнитофона находится вход для управления двигателем ЛПМ.

Канал воспроизведения использует выходной сигнал, снимаемый с разъема для подключения головных телефонов. Он устойчиво работает при входном напряжении 1,5...2 В. Необходимый выходной уровень сигнала магнитофона устанавливается регулятором громкости экспериментально при воспроизведении пробных записей. При желании можно использовать линейный выход магнитофона, но в этом случае требуется предварительный усилитель.

При записи и воспроизведении блоков данных может понадобиться некоторый промежуток времени для обработки и подготовки следующего блока. Поскольку это время точно не определено и может иметь значительную величину, интерфейс предусматривает канал управления двигателем ЛПМ магнитофона. В кассетных магнитофонах типа «Электроника 302» существует возможность управления протяжкой ленты путем размыкания цепи питания двигателя с помощью переключателя, смонтированного на микрофоне.

Формирование сигналов для записи на магнитофон и обработка цифрового сигнала после компаратора производятся программно. В режиме записи программа управляет выходом pA7 порта А БИС КР580ВВ55. При этом сигнал DTR, снимаемый с БИС КР580ВВ51, должен быть установлен активным уровнем. Этот же сигнал управляет двигателем магнитофона. Прием сигнала осуществляется через вход DSR БИС КР580ВВ51. В отсутствие сигнала на этом входе устанавливается напряжение «Лог. 0». Все

```
;.....................................
;       ВЕКТОР ВХОДОВ
;
        JMP     TAPREAD ; ПРОЧИТАТЬ БЛОК.
        JMP     TAPWRITE; ЗАПИСАТЬ   "-.
;.....................................
;
;    ПРОЦЕДУРА ЧТЕНИЯ БЛОКА ДАННЫХ
;      С КАССЕТНОГО МАГНИТОФОНА.
;       ПАРАМЕТРЫ:
;              (HL)- АДРЕС БУФЕРА;
;              (BC)- ЧИСЛО БАЙТОВ.
;       РЕЗУЛЬТАТ: ФЛАГ CY = 0 НОРМАЛЬНО;
;                          = 1 ОШИБКА;
TAPREAD:
        PUSH    H       ; СОХРАНИТЬ РЕГИСТРЫ.
        PUSH    B       ;
        CALL    TAPION  ; НАЙТИ ЗАГОЛОВОК.
        POP     B       ; ВОССТАНОВИТЬ РЕГИСТРЫ.
        POP     H
LOOP:                   ; ЦИКЛ ЧТЕНИЯ.
        PUSH    H       ; ЗАПОМНИТЬ АДРЕС В БУФЕРЕ.
        PUSH    B       ;     -"-    СЧЕТЧИК.
        CALL    TAPIN   ; ПРОЧИТАТЬ БАЙТ.
        POP     B       ; ВОССТАНОВИТЬ СЧЕТЧИК.
        POP     H       ;     -"-        АДРЕС.
        JC      STOPCAS ; ЕСЛИ БЫЛА ОШИБКА - ВЫХОД.
        MOV     M,A     ; ЗАНЕСТИ БАЙТ.
        DCX     B       ; УМЕНЬШИТЬ СЧЕТЧИК.
        INX     H       ; УВЕЛИЧИТЬ АДРЕС.
        MOV     A,B     ; ПРОВЕРИТЬ НА КОНЕЦ.
        ORA     C
        JNZ     LOOP    ; КОНЕЦ ЦИКЛА

     КОНЕЦ ЧТЕНИЯ БЛОКА
;
STOPCAS:
        PUSH    H       ; СОХРАНИТЬ
        PUSH    PSW     ; РЕГИСТРЫ.
        CALL    TAPIOF  ; ВЫКЛЮЧИТЬ МОТОР.
        POP     PSW     ; ВОССТАНОВИТЬ
        POP     H       ; РЕГИСТРЫ.
        RET             ; ВЫХОД.

;.....................................
;
;    ПРОЦЕДУРА ЗАПИСИ БЛОКА ДАННЫХ
;      НА КАССЕТНЫЙ МАГНИТОФОН.
;
;       ПАРАМЕТРЫ:
;                (A) - ТИП ЗАГОЛОВКА;
;                (HL)- АДРЕС БУФЕРА ДАННЫХ;
;                (BC)- ДЛИНА БУФЕРА

TAPWRITE:
        PUSH    H       ; СОХРАНИТЬ
        PUSH    B       ; РЕГИСТРЫ.
        CALL    TAPOON  ; ЗАПИСАТЬ ЗАГОЛОВОК.
        POP     B       ; ВОССТАНОВИТЬ
        POP     H       ; РЕГИСТРЫ.

WLOOP:                  ; ЦИКЛ ЗАПИСИ.
        PUSH    B       ; ЗАПОМНИТЬ СЧЕТЧИК.
        PUSH    H       ;     -"-    АДРЕС БАЙТА.
        MOV     A,M     ; ВЗЯТЬ ОЧЕРЕДНОЙ БАЙТ.
        CALL    TAPOUT  ; И ЗАПИСАТЬ ЕГО.
        POP     H       ; ВОССТАНОВИТЬ АДРЕС.
        POP     B       ;     -"-      СЧЕТЧИК.
        INX     H       ; УВЕЛИЧИТЬ АДРЕС.
        DCX     B       ; УМЕНЬШИТЬ СЧЕТЧИК.
        MOV     A,B     ; ПРОВЕРКА НА КОНЕЦ БУФЕРА.
        ORA     C
        JNZ     WLOOP   ; КОНЕЦ ЦИКЛА.
        PUSH    H       ; ВЫКЛЮЧИТЬ
        CALL    TAPOOF  ; МОТОР.
        POP     H
        RET             ; ВЫХОД

;-------------------------------------
;
;   КОНСТАНТЫ ДЛЯ РАБОТЫ С КАССЕТНЫМ
;          МАГНИТОФОНОМ
;
;.....................................
;
;
;       ЗАПИСЬ  ( ПОРТ WRITE )

WHLEVEL EQU     10000000B ; ВЫСОКИЙ УРОВЕНЬ
WLLEVEL EQU     00000000B ; НИЗКИЙ УРОВЕНЬ
                          ; ЧТЕНИЕ
;       ЧТЕНИЕ  ( ПОРТ READ )

RHLEVEL EQU     10000000B ; ВЫСОКИЙ УРОВЕНЬ
RLLEVEL EQU     00000000B ; НИЗКИЙ УРОВЕНЬ

;       НОМЕРА ПОРТОВ

CONTROL EQU     13H     ; УПРАВЛЕНИЕ РЕЖИМОМ
READ    EQU     07H     ; ЧТЕНИЕ БИТА
WRITE   EQU     10H     ; ЗАПИСЬ БИТА
MOTPORT EQU     07H     ; УПРАВЛЕНИЕ ДВИГАТЕЛЕМ
KEYBOARD EQU    11H     ; КЛАВИАТУРА

;       КОМАНДЫ УПРАВЛЕНИЯ

CREAD   EQU     0EH     ; ЧТЕНИЕ
CWRITE  EQU     0FH     ; ЗАПИСЬ

;       БИТЫ УПРАВЛЕНИЯ ДВИГАТЕЛЕМ

MOTORON EQU     00000010B ; ВКЛЮЧИТЬ МОТОР
MOTOROFF EQU    00000000B ; ВЫКЛЮЧИТЬ  "-

THIGH   EQU     18*256+21 ; ПРОДОЛЖИТЕЛЬНОСТЬ "1"
TLOW    EQU     40*256+38 ;          -"-      "0"

HEADER  EQU     4000    ; ДЛИНА КОРОТКОГО ЗАГОЛОВКА
LOWLIM  EQU     0FFBEH
WINWID  EQU     0FFBCH

;-------------------------------------
;  ---------------------------------
;
;       ВЫКЛЮЧИТЬ МОТОР.
;
;    ПРИМЕЧАНИЕ: РАЗРЕШАЕТ ПРЕРЫВАНИЯ.
;
;  ---------------------------------
;
;       1>ПОСЛЕ ЗАПИСИ
TAPOOF:
        PUSH    B
        PUSH    PSW
        LXI     B,0
CTWOF1:
        DCX     B       ; ЗАДЕРЖКА ДЛЯ
        MOV     A,B     ; МЕЖБЛОЧНОГО ИНТЕРВАЛА.
        ORA     C
        JNZ     CTWOF1
        POP     PSW
        POP     B

;  ---------------------------------
;
;       2) ПОСЛЕ ЧТЕНИЯ
TAPIOF:
        PUSH    PSW
        MVI     A,MOTOROFF
        OUT     MOTPORT ; ОСТАНОВИТЬ МОТОР.
        MVI     A,CREAD ; ВКЛЮЧИТЬ ЧТЕНИЕ.
        OUT     CONTROL
        POP     PSW
        EI
        RET
;  ---------------------------------
;       ЗАПИСЬ ЗАГОЛОВКА.
;   ПАР: AKK.
;        (A)= 0 - ПИШЕТСЯ КОРОТКИЙ
;             ЗАГОЛОВОК ( 4000 ЕДИНИЦ)
;             В ПРОТИВНОМ СЛУЧАЕ ДЛИННЫЙ.
;
;  ПРИМЕЧ. ЗАПРЕЩАЮТСЯ ПРЕРЫВАНИЯ !!!
;
TAPOON:
        ORA     A       ; УСТАНОВИТЬ ФЛАГ ДЛИНЫ
        PUSH    PSW     ; СОХРАНИТЬ ФЛАГ.
        MVI     A,MOTORON
        OUT     MOTPORT ; ВКЛЮЧИТЬ МОТОР.
        LXI     H,0
MOTRWT:
        DCX     H       ; ЗАДЕРЖКА ДЛЯ "РАСКРУТКИ"
        MOV     A,H     ; ДВИГАТЕЛЯ.
        ORA     L
        JNZ     MOTRWT
        POP     PSW     ; ЗАНЕСТИ В (BC)
        LXI     B,HEADER; ДЛИНУ ЗАГОЛОВКА.
        JZ      SYNCW1
        LXI     B,HEADER*4
```

```
SYNCWI:
        MVI     A,CWRITE; ЗАПИСЬ.
        OUT     CONTROL
        DI
SYNLP1:
        CALL    BIT1OUT ; ЗАПИСЬ ОДНИХ ЕДИНИЦ.
        CALL    RETRET  ; КОМПЕНСАЦИЯ.
        DCX     B
        MOV     A,B
        ORA     C
        JNZ     SYNLP1
        RET
;-------------------------------------------
;
;       ЗАПИСЬ БАЙТА.
;
TAPOUT:
DATAW:
        LXI     H,?LOW  ; ЗАПИСАТЬ СТАРТОВЫЙ БИТ.
        PUSH    PSW
        MOV     A,L
        SUI     0AH     ; КОМПЕНСАЦИЯ.
        MOV     L,A
        CALL    BITOUT  ;   ЗАПИСЬ.
        POP     PSW
                        ; ЗАПИСАТЬ СОБСТВЕННО БАЙТ.
        MVI     B,8     ; 8 - БИТ.
DATAWL:
        RRC             ; ОЧЕРЕДНОЙ БИТ В ФЛАГ CY.
        CC      BIT1    ; ЗАПИСАТЬ "1"
        CNC     BIT0    ;  -"-    "0".
        DCR     B
        JNZ     DATAWL  ; ПОСЛЕДНИЙ БИТ?
                        ; ЗАПИСАТЬ СТОП-БИТЫ.
        CALL    BIT1
        CALL    BIT1
        ORA     A
        RET

;.....................................
;
;       ЗАПИСЬ БИТОВ
;
;       "0"
                        ; ЧИСЛО ТАКТОВ:
BIT0:                   ;.............
        LXI     H,?LOW  ;       10
        NOP             ;        4
        NOP             ;        4
        CALL    BITOUT  ;       17
RETRET:                 ;
        RET             ;       10
                        ;-----------------
                        ; 45+(40+38)*15+65=
                        ;    = 1280
                        ;-----------------
;       "1"
BIT1:
        CALL    BIT1OUT ;       17
        CALL    BIT1OUT ;       17
        RET             ;       10
                        ;=========================
                        ; 44+2*[(18+21)*15+65+18)]=
                        ;       = 1280
                        ;.........................

;       ЗАПИСЬ  ОДНОГО "ВСПЛЕСКА"
;       ЧИСЛО ЦИКЛОВ =
;         15 * [L] + 15 * [H] + 65  \+18\
;
BIT1OUT:
        LXI     H,?HIGH ;       10
        NOP             ;        4
        NOP             ;        4
                        ;      ======
                        ;       18
BITOUT:
        PUSH    PSW     ;       11 (+)
                        ; НИЗКИЙ УРОВЕНЬ.
KEEPL:
        DCR     L       ;        5
        JNZ     KEEPL   ;       10
                        ;      ======
                        ;       15 (*)
                        ; УСТАНОВИТЬ ВЫСОКИЙ УРОВЕНЬ
        MVI     A,WHLEVEL;       7
        OUT     WRITE   ;       10
                        ;      ======
                        ;       17 (+)
```

```
KEEPH:
        DCR     H       ;        5
        JNZ     KEEPH   ;       10
                        ;      ======
                        ;       15 (*)
                        ; УСТАНОВИТЬ НИЗКИЙ УРОВЕНЬ
        MVI     A,WLLEVEL;       7
        OUT     WRITE   ;       10
        POP     PSW     ;       10
        RET             ;       10
                        ;      ======
                        ;       37 (+)

;-------------------------------------
;       РАСПОЗНАТЬ ЗАГОЛОВОК БЛОКА.
;.....................................
;
;   ЗАГОЛОВОК РАСПОЗНАЕТСЯ ПО 1000
;   ПЕРВЫХ СИГНАЛОВ ВЫСОКОГО УРОВНЯ,
;   ПОСТОЯННОЙ ДЛИТЕЛЬНОСТИ (ЕДИНИЦ).
;   ПО ДЛИТЕЛЬНОСТИ ОДНОГО СИГНАЛА
;   ОПРЕЛЕЯЕТСЯ ДЛИНА ОДНОГО БИТА.
;
;       ВЫПОЛНЕНИЕ ПРЕКРАЩАЕТСЯ, ЕСЛИ
;   БЫЛО НАЖАТИЕ КЛАВИШИ НА КЛАВИАТУРЕ.
;   РЕЗ: CY= 0 - ВСЕ В ПОРЯДКЕ;
;            1 - НАЖАТИЕ.
;
;   ПРИМЕЧ. ЗАПРЕЩАЮТСЯ ПРЕРЫВАНИЯ !!!
;
TAPION:
        DI
        MVI     A,CREAD ; ЧТЕНИЕ.
        OUT     CONTROL
        MVI     A,MOTORON; ВКЛЮЧИТЬ МОТОР.
        OUT     MOTPORT
SYN05:
        LXI     H,1000  ; ЧИСЛО СИГНАЛОВ ДЛЯ
                        ; ОПРЕДЕЛЕНИЯ ЗАГОЛОВКА.
SYN10:
        MOV     D,C     ; ЗАПОМНИТЬ ПОСЛЕДНЕЕ
                        ; ЗНАЧЕНИЕ ДЛИТЕЛЬНОСТИ.
        CALL    CNTFUL  ; ОПРЕДЕЛИТЬ ДЛИНУ НОВОГО
                        ; СИГНАЛА.
        RC              ; ВЫЙТИ, ЕСЛИ НАЖАТА КЛАВИША.
        MOV     A,C
        CPI     0DEH    ; МАКС. ДЛИТЕЛЬНОСТЬ?
        JNC     SYN05   ; ЕСЛИ БОЛЬШЕ - С НАЧАЛА.
        CPI     5       ; МИНИМ. ДЛИТЕЛЬНОСТЬ.
        JC      SYN05   ; ЕСЛИ МЕНЬШЕ - С НАЧАЛА.
        SUB     D       ; СРАВНИТЬ С ПРЕДЫДУЩИМ
                        ; ЗНАЧЕНИЕМ. ВЫЧИСЛИТЬ
                        ; ОТКЛОНЕНИЕ (МОДУЛЬ).
        JNC     SYN11   ; РАЗНОСТЬ ОТРИЦАТЕЛЬНА?
        CMA             ; ДА - A:= -A.
        INR     A
SYN11:
        CPI     8       ; МОДУЛЬ ОТКЛОНЕНИЯ МЕНЬШЕ 8
        JNC     SYN05   ; НЕТ - ПОВТОРИТЬ С НАЧАЛА.
        DCX     H
        MOV     A,H
        ORA     L       ; ПРОВЕРИТЬ НА КОНЕЦ.
        JNZ     SYN10
                ; ОПРЕДЕЛИТЬ СРЕДНЮЮ ДЛИТЕЛЬНОСТЬ
SYN20:          ; ПО 256 СИГНАЛАМ.
        LXI     H,0     ; ОБНУЛИТЬ СУММУ.
        MOV     B,L     ; В (B) - НУЖЕН НОЛЬ.
        MOV     D,L     ; ЧИСЛО БИТОВ - 256.
SYN30:
        CALL    CNTFUL  ; ОПРЕДЕЛИТЬ ДЛИТЕЛЬНОСТЬ.
        RC              ;
        DAD     B
        DCR     D
        JNZ     SYN30
        LXI     B,06AEH ; КОРРЕКЦИЯ.
        DAD     B
                ; В [H] - "СРЕДНЕЕ" ЗНАЧЕНИЕ(СР).
;
; ОПРЕРЕДЕЛЕНИЕ "КОНСТАНТ"  ДЛЯ ПОСЛЕДУЮЩЕГО ЧТЕНИЯ.
;
;  LOWLIM - МИНИМАЛЬНАЯ ПРОДОЛЖИТЕЛЬНОСТЬ
;           СТАРТОВОГО БИТА. [H]*1.5.
;  WINWID - ШИРИНА ОКНА ОПРЕДЕЛЕНИЯ БИТА.
;
        MOV     A,H     ; [A] <- (СР)
        RAR
        ANI     7FH     ; [A] <- (СР)/2
        MOV     D,A     ; [D] <- (СР)/2
        DAD     H
```

```
        MOV     A,H     ; [A] <- (CP)/2
        SUB     D       ; [A] <- (CP)*1.5
        MOV     D,A     ; СОХРАНИТЬ ДЛЯ РАСЧЕТА
                        ;    "WINWID".
        SUI     6       ; КОМПЕНСАЦИЯ "RDBIT".
        STA     LOWLIM  ;
;
;       ОПРЕДЕЛЕНИЕ "WINWID"
;ПРОДОЛЖИТЕЛЬНОСТЬ ЦИКЛА В "CNTFUL"-39Т
;       -"-         -"-  В "RDBIT" -52Т
; ШИРИНА ОКНА РАВНА ТРЕМ ДЛИНАМ ОДНОГО
; СИГНАЛА \ (CP)/2 \
; WINWID = [(CP) * 3 / 2] * 39/52 =
;        = [D] * 3/4.
;
        MOV     A,D
        ADD     A
        ADD     D       ; * 3
        ANI     0FCH
        RRC
        RRC             ; / 4
        SUI     03H     ; КОМПЕНСАЦИЯ ДЛЯ RDBIT.
        STA     WINWID  ; ЗАПОМНИТЬ.
        ORA     A       ;
        RET             ; ВЫХОД.

;- - - - - - - - - - - - - - - - - - - -
;
;       ЧТЕНИЕ БАЙТА
; ВЫХОД: (A) - БАЙТ
;        CY = 1 - ОСТАНОВ ИЛИ ОШИБКА
;           = 0 - НОРМАЛЬНО.
TAPIN:
        LDA     LOWLIM  ; ЗАГРУЗИТЬ В (D)
        MOV     D,A     ; МИНИМАЛЬНУЮ ДЛИНУ
DATAR:
        CALL    BREAK   ; НАДО ЧИТАТЬ?
        RC
        IN      READ    ; ОЖИДАНИЕ ВЫСОКОГО
        RLC             ; УРОВНЯ.
        JNC     DATAR
DATAR0:                 ; ДОЖДАЛИСЬ!
        CALL    BREAK
        RC
        IN      READ    ; ОЖИДАНИЕ НИЗКОГО
        RLC             ; УРОВНЯ.
        JC      DATAR0
                        ; ДОЖДАЛИСЬ!
        MVI     E,0     ;
        CALL    CNTHLF  ; ОПРЕДЕЛИТЬ ДЛИНУ НИЗКОГО
                        ; УРОВНЯ.
DATAR1:
        MOV     B,C     ; СОХРАНИТЬ СТАРУЮ ДЛИНУ.
        CALL    CNTHLF  ; ОПРЕДЕЛИТЬ НОВУЮ.
        RC
        MOV     A,B     ; ВЫЧИСЛИТЬ  СУММАРНУЮ
        ADD     C       ; ДЛИТЕЛЬНОСТЬ.
        JC      DATAR1  ; ПОВТОРИТЬ ЕСЛИ СУММАРНАЯ
                        ; ДЛИТ. БОЛЬШЕ 255.
        CMP     D       ; СРАВНИТЬ С МИН. ДЛИНОЙ(D)
        JC      DATAR1  ; ЕСЛИ МЕНЬШЕ - ПОВТОРИТЬ.

;.....................................
;
;       СТАРТОВЫЙ БИТ НАЙДЕН !
;  И В (E) СТАРШИЙ БИТ РАВЕН 0 ПРИ НОРМАЛЬНОЙ
;       ПОЛЯРНОСТИ И РАВЕН 0 ПРИ ОБРАТНОЙ.
;
        MVI     L,8     ; ЧИСЛО БИТОВ В БАЙТЕ
DATARL:                 ;  /СЧЕТЧИК ЦИКЛА/.
        CALL    RDBIT   ; СКОЛЬКО СМЕН УРОВНЯ
                        ; /РЕЗ. В (A) И (C)/.
        CPI     3+1     ; ЕСЛИ БОЛЬШЕ 4,ТО
        CMC             ; ВЫХОД С ОШИБКОЙ.
        RC
        CPI     2       ; УСТАНОВИТЬ ФЛАГ CY,ЕСЛИ
        CMC             ; ЧИСЛО СМЕН УРОВНЯ 2 ИЛИ 3,
        MOV     A,D     ; РОТАЦИЯ РЕГИСТРА (D)
        RAR             ; ВМЕСТЕ С CY.
        MOV     D,A
        MOV     A,C     ; ЧЕТНОСТЬ ЧИСЛА СМЕН?
        RRC             ; ЕСЛИ ЧЕТНОЕ -
        CNC     CNTHL0  ;  СДЕЛАТЬ НЕЧЕТНЫМ.
        CALL    CNTHLF  ; ЖДАТЬ КОНЦА "БИТА".
        DCR     L       ; УМЕНЬШИТЬ СЧ. ЦИКЛА.
        JNZ     DATARL  ; КОНЕЦ?
        CALL    BREAK
        MOV     A,D     ; В(A) - ПРОЧИТАННЫЙ БАЙТ.
        RET             ; ВЫХОД.

;.....................................
;
;       ОПРЕДЕЛЕНИЕ ЧИСЛА СМЕН УРОВНЯ
;       ВО ВРЕМЯ, ОПРЕДЕЛЯЕМОЕ "WINWID".
;
RDBIT:
        LDA     WINWID  ; ЗАНЕСТИ В РЕГИСТР(B)
        MOV     B,A     ; ШИРИНУ ОКНА.
        MVI     C,0     ; ОБНУЛЕНИЕ СЧЕТЧИКА СМЕН.

                        ;ТАКТЫ
RDBITL:                 ;.....
        IN      READ    ; 10 ; БЫЛА СМЕНА УРОВНЯ?
        XRA     E       ;  4 ;
        JP      NOTRAN  ; 10 ;  НЕТ.
        MOV     A,E     ;  4 ;
        CMA             ;  4 ;
        MOV     E,A     ;  4 ;
        INR     C       ;  5 ;  УВЕЛИЧИТЬ СЧЕТЧИК,
        DCR     B       ;  5 ;  КОНЧИЛОСЬ ОКНО?
        JNZ     RDBITL  ; 10 ;
                        ;=====
                        ; 52
        MOV     A,C     ; ВЫХОД.
        RET
NOTRAN:
        NOP             ;  4 ; ЗАДЕРЖКА
        NOP             ;  4 ; НА 32 ТАКТА,
        NOP             ;  4
        INR     A       ;  5
        DCR     B       ;  5
        JNZ     RDBITL  ; 10
        MOV     A,C
        RET

;.....................................
;
;   ОПРЕДЕЛЕНИЕ ДЛИТЕЛЬНОСТИ ОДНОГО
;               СИГНАЛА.
CNTHLF:
        CALL    BREAK   ;       17+44
        RC              ;        5
CNTHL0:
        MVI     C,0     ;        7
CNTHL1:                 ;    .........
        INR     C       ;        5 !
        JZ      TIMOUT  ;       10 !
        IN      READ    ;       10 !
        XRA     E       ;        4 !
        JP      CNTHL1  ;       10 !
        MOV     A,E     ;        4
        CMA             ;        4
        MOV     E,A     ;        4
        RET             ;       10
TIMOUT:                 ; ВЫХОД ПО ПЕРЕПОЛНЕНИЮ
        DCR     C       ; СЧЕТЧИКА. ВОЗВРАЩАЕТ 255,
        RET
CNTFUL:                 ; ОПРЕДЕЛЕНИЕ ПОЛНОЙ
        CALL    BREAK   ; ДЛИНЫ СИГНАЛА.
        RC
        IN      READ    ; УРОВЕНЬ
        RLC
        JC      CNTFUL  ; НИЗКИЙ? - НЕТ
        MVI     E,0     ; - ДА
        CALL    CNTHL0
        JMP     CNTHL1

;.....................................
;
;       ПРОВЕРИТЬ СТАТУС КЛАВИАТУРЫ.
; РЕЗУЛЬТАТ:
;       CY = 1 - БЫЛО НАЖАТИЕ
BREAK:
        IN      KEYBOARD ;       10
        RLC              ;       04
        INR     A        ;       05
        INR     A        ;       05
        INR     A        ;       05
        INR     A        ;       05
        RET              ;       10
                         ;      ====
                         ;       44
```

Рис. 5. Исходный текст программы для работы с бытовым кассетным магнитофоном

временные соотношения при формировании сигнала для записи и обработки входного сигнала задаются программно по времени исполнения команд, поэтому время должно быть точно определено. Если программа находится в системном ПЗУ модуля процессора, то такты ожидания готовности, вносящие неопределенность, не используются и время исполнения команд может быть точно задано тактовой частотой процессора. Все временные соотношения в программе управления магнитофоном (рис. 5) рассчитаны для тактовой частоты 1,7777 МГц.

Для работы с магнитофоном используются две процедуры: TAPREAD — считать блок и TAPWRITE — записать блок. Обе процедуры выполняют перенос блока данных длиной, задаваемой регистровой парой B, C, с магнитной ленты в ОЗУ или наоборот. Адрес блока ОЗУ задается регистровой парой HL.

Процедура TAPREAD производит поиск на ленте заголовка и автоматически подстраивается под скорость записи, затем переносит в ОЗУ следующий за заголовком блок данных заданного размера. Таким образом эта программа использует заголовки не только в качестве разделителя данных, но и для автоматической настройки на частоту записи. Это позволяет читать записи, сделанные на других ЭВМ, со скоростью от 200 до 3000 бит/с. Определяемые программой параметры записываются в ячейки памяти и могут быть использованы другими программами. Если во время чтения возникла ошибка, то по возвращении из подпрограммы флаг СУ будет установлен в единицу. При безошибочном чтении флаг равен нулю. Процедура TAPREAD может быть прервана нажатием на любую клавишу клавиатуры, что устанавливает признак ошибки чтения.

Процедура записи TAPWRITE выполняет запись блока данных с предварительной записью заголовка. Тип заголовка указывается в аккумуляторе. Если он равен нулю, происходит запись короткого заголовка, если единице — длинного. Данные записываются с фиксированной скоростью 1200 бит/с. В процессе работы с магнитофоном можно обращаться и к другим подпрограммам, таким как отдельная запись заголовка или чтение-запись отдельного байта. Однако необходимо учитывать, что во время работы подпрограмм прерывания запрещены, что важно при разработке систем реального времени.

Публикуя исходный текст программы работы с кассетным магнитофоном, авторы надеются на обсуждение, в ходе которого будут сделаны полезные замечания. При соответствующей доработке программа может быть использована для других вновь разрабатываемых или уже существующих типов ЭВМ. Единый способ записи позволит упростить обмен программами и текстами между различными ЭВМ.

ЛИТЕРАТУРА

1. MSX Technical Data Book, Hardware / Software specifications., фирменный материал.
2. Austin Lesea, Rodnay Zaks Mikroprozessor — Interface — Techniken, vogel — verlag., с. 122—135.
3. Барышников В. Н., Быстров В. П., Воронов М. А., Паначев Ф. И., Романов В. О. Модуль процессора ПЭВМ «Ириша». — Микропроцессорные средства и системы, 1986, № 2, с. 52—62.

Статья поступила 7 апреля 1986 г.

УДК 681.327.2

С. Ф. Горбачев, А. П. Демин

ОПЕРАТИВНОЕ ЗАПОМИНАЮЩЕЕ УСТРОЙСТВО С ВНЕШНИМ СКОРОСТНЫМ КАНАЛОМ ВВОДА-ВЫВОДА ИНФОРМАЦИИ В МИКРОЭВМ «ЭЛЕКТРОНИКА 60»

В большинстве микроЭВМ возможны два вида обмена с внешним объектом: программный и прямой доступ к памяти (ПДП). Стандартные устройства связи с микроЭВМ, а также устройства, приведенные в [1—3], позволяют производить обмен в одном из этих режимов.

Недостатком программного режима является невысокая скорость обмена, которая для микроЭВМ «Электроника 60» составляет 20 мкс. Кроме того, регенерация динамической памяти затрудняет организацию цикличного обмена. Режим ПДП, хотя и позволяет обмениваться с большей скоростью, но при этом занимает канал микроЭВМ, в результате чего становится невозможным выполнение программы во время обмена.

Часто возникает необходимость вести обмен с большой скоростью, не занимая при этом канал микроЭВМ. Для решения таких задач предназначено разработанное устройство (рис. 1), в котором скорость обмена информацией с внешним объектом составляет 0,2 мкс. Устройство полностью согласовано по интерфейсу с микроЭВМ «Электроника 60» [4] и может работать в качестве стандартного ОЗУ типа П2 с любым номером банка памяти при соответствующей распайке перемычек.

С микроЭВМ устройство связано через канальные приемопередатчики D4 ... D7. Связь с внешним объектом через канальные приемопередатчики D1 ... D3 позволяет производить обмен 12-разрядными словами. Микросхемы памяти D32 ... D47 совместно с дешифратором D28 и микросхемами D29 ... D31 образуют блок памяти с организацией 4К×16 разрядов. Обмен информацией блока памяти с микроЭВМ организован в соответствии со стандартом [4] и отличается лишь тем [1], что сигналы К Ввод Н и К Вывод Н, поступающие из микроЭВМ, не проходят в устройство до тех пор, пока на выходе элемента D19,3 не установится уровень «Лог. 0». Это необходимо для завершения текущего обмена устройства с

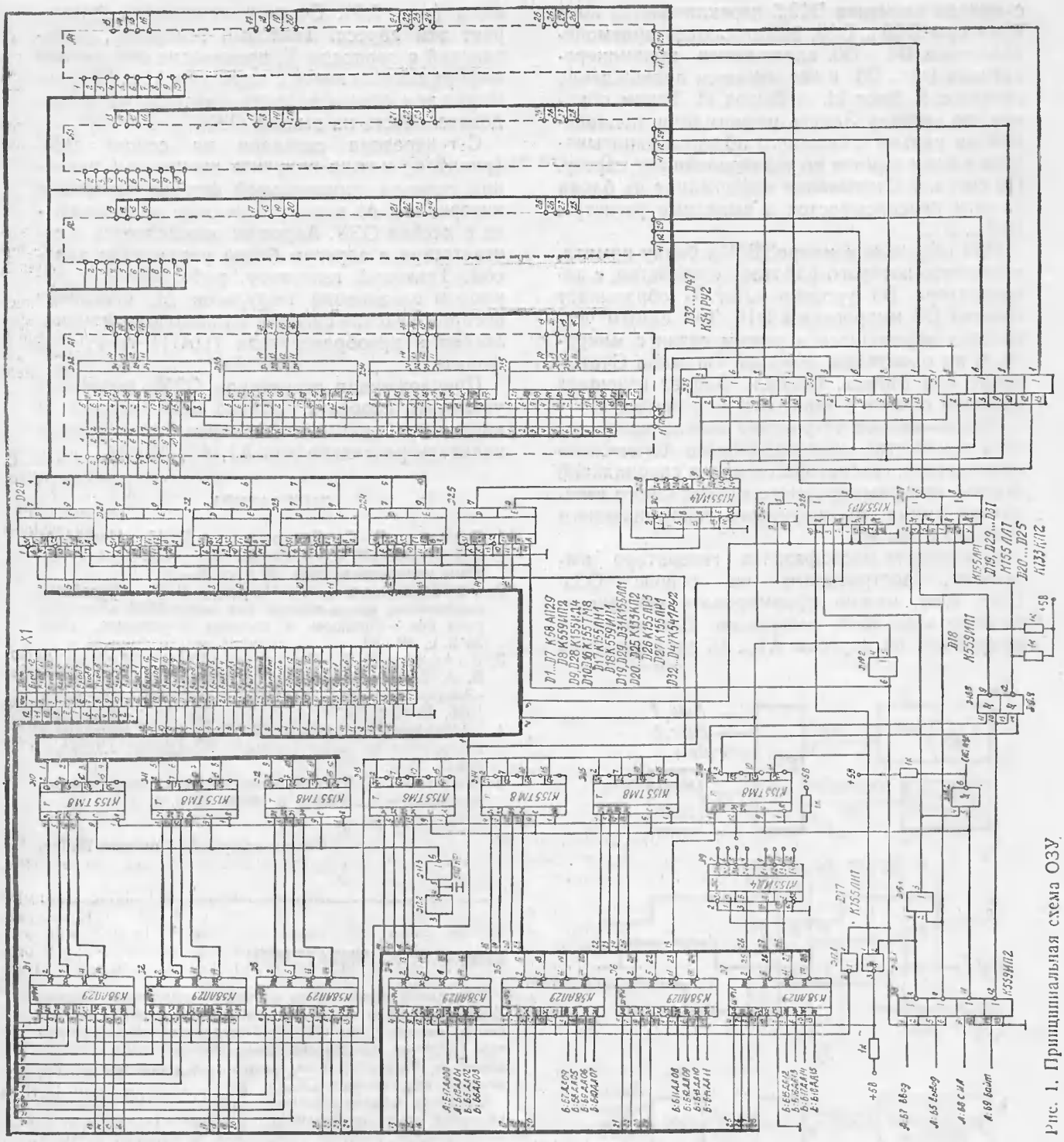


Рис. 1. Принципиальная схема ОЗУ

внешним объектом. С помощью коммутатора (D20 ... D25) можно задавать адрес ячейки блока памяти с регистра адреса (D13 ... D15) при обмене информацией с микроЭВМ либо с разъема при обмене информацией с внешним объектом.

Для обмена информацией с внешним объектом необходимо задать адрес и подать сигналы Считывание или Запись, длительность которых не должна превышать времени ожидания сигнала К СИП Н, равного 10 мкс [4], чтобы не происходило сбоя при одновременном обращении к устройству микроЭВМ и внешнего объекта. По сигналу Запись или Считывание

с выхода элемента D19,2 переключаются коммутаторы D20 ... D25, выключаются приемопередатчики D4 ... D6, включаются приемопередатчики D1 ... D3 и запрещается прохождение сигналов К Ввод Н, К Вывод Н. Таким образом, по сигналу Запись информация, подаваемая на разъем с внешнего объекта, записывается в блок памяти по установленному адресу. По сигналу Считывание информация из блока памяти переписывается в выходные регистры D10 ... D12.

При обращении микроЭВМ к банку памяти, в качестве которого работает устройство, с дешифратора D9 уровень «Лог. 0» сбрасывает триггер D3 микросхемы D16. Тем самым устройство переводится в режим связи с микроЭВМ по окончании действий сигналов Считывание или Запись. Сигнал Запрет извещает внешний объект о занятости устройства.

Разработанное устройство можно использовать в качестве многоканального логического анализатора, генератора сигналов специальной формы, программируемого многофазного генератора импульсов, цифрового запоминающего осциллографа [5].

С помощью многофазного генератора импульсов, построенного на основе ОЗУ (рис. 2, *а*), можно сформировать временную последовательность импульсов. Для этого из микроЭВМ по адресам А1 ... А6 записываются коды (рис. 2, *б*). Счетчик циклически формирует эти адреса. Тактовый генератор, работающий с периодом Т, производит считывание информации из ячеек с адресами А1 ... А6, формируя тем самым заданную временную последовательность на выходе ОЗУ.

С генератора сигналов на основе ОЗУ (рис. 2, *в*) можно получить дискретные значения сигнала произвольной формы, которые с интервалом Δt в цифровом виде записываются в ячейки ОЗУ. Адрес их определяется в соответствии с адресом блока управления адресом. Тактовый генератор, работающий с периодом следования импульсов Δt, обеспечит воспроизведение сигнала на выходе цифроаналогового преобразователя (ЦАП) требуемой формы.

Предложенная структура ОЗУ позволяет увеличить скорость обмена информацией с внешним объектом, если адрес блока памяти задавать раздельно для А1, А2, А3, А4.

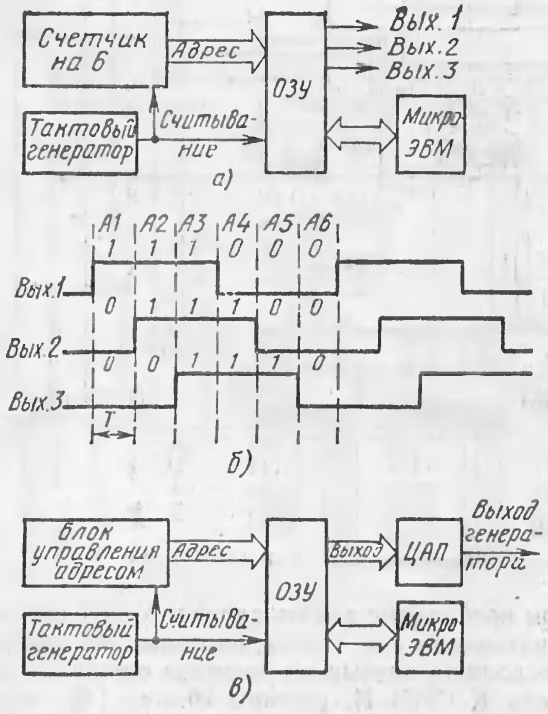


Рис. 2. Структурные схемы многофазного генератора импульсов (*а*) с временной диаграммой (*б*) и генератора сигналов специальной формы (*в*)

ЛИТЕРАТУРА

1. Ефимов В. С., Севрюкова Т. Н. Интерфейс для микроЭВМ «Электроника 60».— Приборы и системы управления, 1985, № 4, с. 36—37.
2. Овчинников В. Ф., Чернов В. Г. Устройство аналогового ввода-вывода для микроЭВМ «Электроника 60».— Приборы и системы управления, 1985, № 2, с. 32—33.
3. Бородин И. В., Радионов В. В., Кулешов В. А. Блок инициативных сигналов для микроЭВМ «Электроника 60».— Приборы и системы управления, 1985, № 8, с. 33.
4. Центральный процессор М2: Техническое описание и инструкция по эксплуатации.— М.: ЦНИИ «Электроника», 1970.
5. Мирский Г. Я. Микропроцессоры в измерительных приборах.— М.: Радио и связь, 1984, с. 118—156.

Статья поступила 8 августа 1985 г.

УДК 681.322.1+681.325.5

С. Н. Попов

СПЕЦПРОЦЕССОР НА БАЗЕ МИКРОПРОЦЕССОРА КР580ИК80А В КОМПЛЕКСЕ С МИНИ-ЭВМ

Мини- и микроЭВМ СМ-3, СМ-4, «Электроника 60», ДВК-2 могут успешно использоваться в качестве инструментальных для разработки программного обеспечения систем на базе МПК БИС серии КР580, если к ним подключить дополнительный процессор (спецпроцессор), имеющий собственную оперативную и постоянную память. Для операций ввода-вывода информации спецпроцессор использует периферийные устройства (диски, дисплеи, принтеры) базовой ЭВМ.

Производительность современных мини-ЭВМ позволяет обслуживать несколько спецпроцессоров одновременно без заметного ухудшения времени отклика. Например, в операционной системе РАФОС для ЭВМ класса СМ-4 могут быть одновременно запущены восемь оперативных и одно фоновое задания, что позволяет подключить до восьми спецпроцессоров и сохранить возможность обычного счета на мини-ЭВМ. Рассматриваемый спецпроцессор на базе микропроцессора КР580ИК80А (рис. 1) содержит небольшое число микросхем и отличается малым энергопотреблением. Спецпроцессор имеет оперативную память объемом 196К байт и постоянную память объемом 8К байт (рис. 2). Часть оперативной (128К байт) и постоянной (6К байт) памяти выполняет функции электронного диска (рис. 3). Сопряжение спецпроцессора с базовой ЭВМ осуществляется наиболее простым способом, а именно через параллельный интерфейс. Скорость передачи информации при этом достигает 30К байт/с.

В схеме спецпроцессора помимо стандартных решений имеется ряд особенностей. Прежде всего — это способ реализации режима регенерации информации в ди-

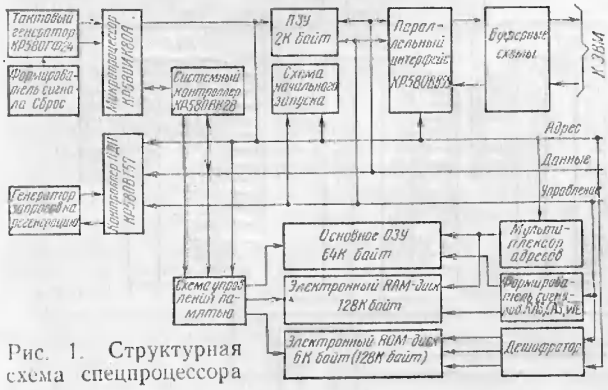


Рис. 1. Структурная схема спецпроцессора

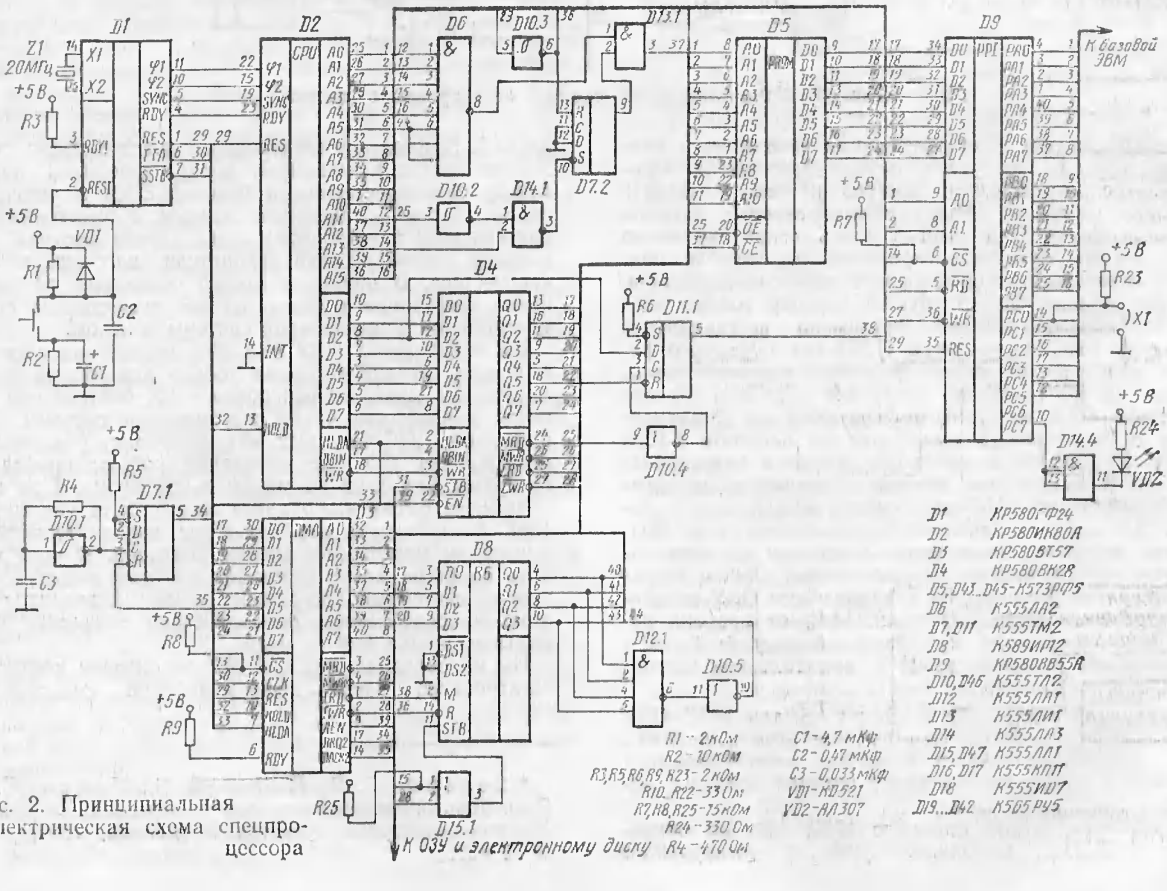


Рис. 2. Принципиальная электрическая схема спецпроцессора

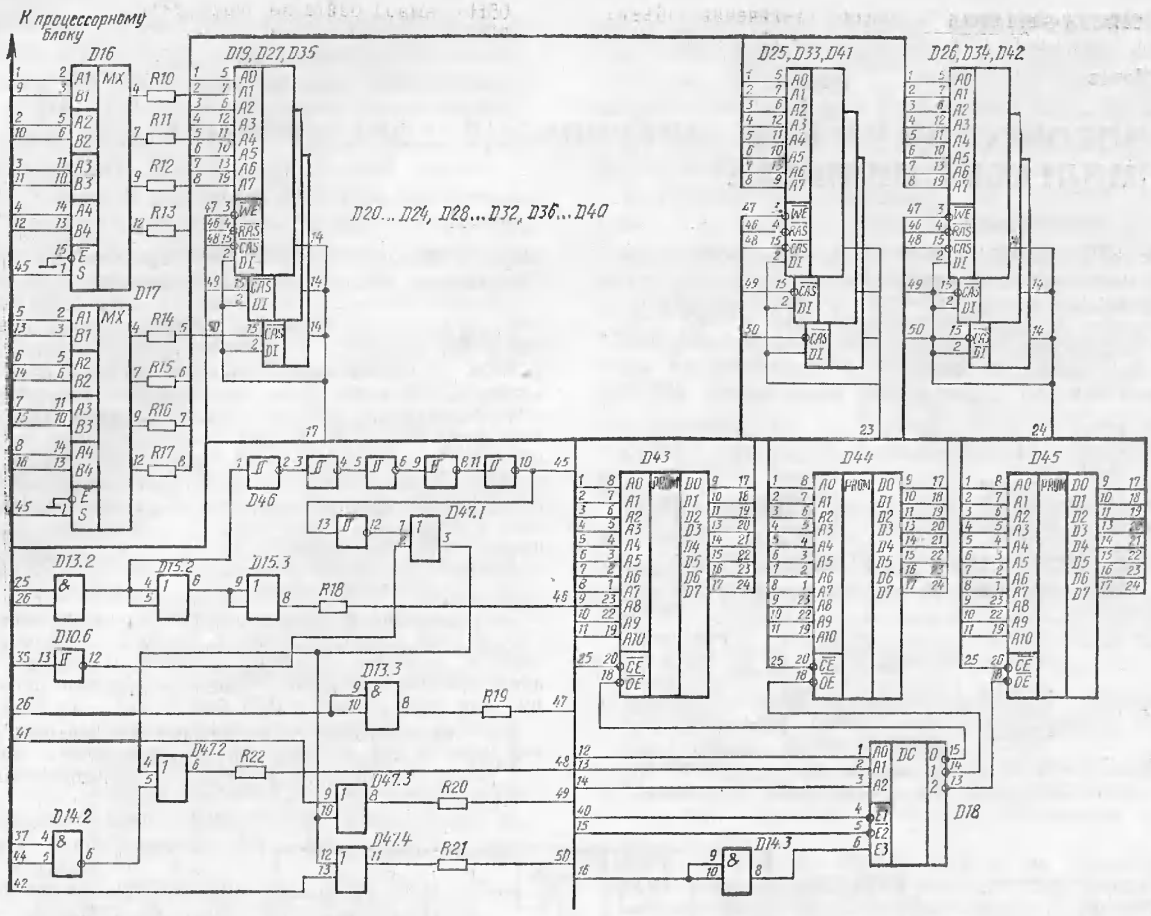


Рис. 3. Принципиальная схема «электронного диска»

намической памяти. Для регенерации используется контроллер прямого доступа к памяти КР580ВТ57. Применение режима прямого доступа к памяти (ПДП) позволило свести к минимуму аппаратурные затраты на регенерацию. При этом удалось сократить число микросхем на мультиплексирование адресов и полностью отказаться от специальных схем синхронизации. Для регенерации ОЗУ требуется перебор только семи младших адресов, поэтому применено нестандартное включение БИС контроллера ПДП без буферного регистра для старших восьми разрядов адресной шины. Запрос на регенерацию поступает каждые 15 мкс с триггера D7.1, который, в свою очередь, устанавливается от простейшего генератора на элементе D10.1. В БИС КР580ВТ57 задействован канал с автоинициализацией (канал 2), что позволило сократить программу обслуживания БИС контроллера ПДП. Длина пакета ПДП выбрана максимально возможной (16К байт), так что дополнительные затраты времени на автоинициализацию практически несущественны. Общие затраты времени на регенерацию динамического ОЗУ описанным способом составляют около 14 % от времени работы микропроцессора при тактовой частоте 2 МГц и могут быть снижены до 11 % при тактовой частоте 2,5 МГц.

Формирование сигналов $\overline{RAS}$ и $\overline{CAS}$ для БИС ОЗУ К565РУ5Д производится при помощи элементов D46... ...D47. Здесь используется присущая этим элементам задержка в срабатывании. Опыт показал хорошую повторяемость данной схемы.

Оперативная память емкостью 128К байт на микросхемах D27...D42 выполняет функции электронного диска *. Применение электронного диска вызвано стремлением свести к минимуму число пересылок данных между спецпроцессором и базовой ЭВМ и уменьшить время доступа к дисковым данным. Качественные характеристики электронного диска (время доступа) превышают характеристики накопителя на дисках типа «винчестер». В принципе можно отказаться от применения электронного диска, но это существенно снизит качество инструментальной системы в целом.

На микросхемах ПЗУ D43...D45 реализован электронный диск, предназначенный только для чтения информации. Его минимальный объем — 6К байт. В нем хранится резидентная часть операционной системы. Диск помечен в ДОС как диск «А», и поэтому обязательная перезагрузка ДОС по окончании работы прикладных программ проходит за минимальное время. При необходимости объем этого диска может быть увеличен до 128К байт, однако это потребует дополнительной буферизации магистралей микропроцессора. В этом варианте на диске могут храниться наиболее часто используемые системные программы: экранный редактор текстов, макроассемблер, символический отладчик, интерпретатор языка Бейсик и др.

На микросхемах D12.1, D10.5 реализован контроллер электронного диска информационной емкостью до

* Зеленко Г. В., Панов В. В., Попов С. Н. Электронный «квазидиск» для персональной ЭВМ.— Микропроцессорные средства и системы, 1984, № 4, с. 79—82.

256К байт. Используемый способ увеличения объема памяти, с которой работает 8-разрядный микропроцессор, позволяет переключать сегменты размером 64К байт и значительно сократить время пересылки данных между электронным диском и основной памятью.

Другой особенностью схемной реализации спецпроцессора является отказ от буферных схем как наиболее энергоемких элементов. Это позволило кроме снижения энергопотребления снизить требования к цепям развязки по питанию. По этой же причине возможна подача питающих напряжений непосредственно от ЭВМ, к которой подключен спецпроцессор. Ток по цепи 5 В не превышает 1,5 А. В спецпроцессоре используются микросхемы серии К555 с диодами Шотки, имеющие малые входные токи. Емкостная нагрузка на линиях адресной магистрали не превышает допустимую.

На микросхемах D6, D7.2, D13.1 выполнен блок начального запуска, позволяющий передать управление по адресу 0F800H после прихода сигнала Сброс. При произвольной длительности сигнала Сброс может произойти потеря информации в динамическом ОЗУ, поэтому в спецпроцессоре приняты меры по нормализации его длительности (элементы C1, C2, R1, R2, VD1).

Для формирования сигналов CS для БИС D3 и D9 используется упрощенная схема без специального дешифратора. Сигналы поступают непосредственно с адресных линий A12 и A13. Использование старшей половины адресной шины исключает необходимость в специальных мерах по блокировке сигналов $\overline{CS}$ сигналом AEN от контроллера ПДП. В режиме ПДП эти линии находятся в высокоимпедансном состоянии (в данной схеме контроллер ПДП не воздействует на них), что эквивалентно «Лог. 1». Базовые адреса внутренних регистров БИС D3 и D9 соответственно 00H и 10H.

В схеме не показаны буферные элементы на выходе БИС параллельного интерфейса D9, так как их тип зависит от способа подключения и типа ЭВМ, с которой работает спецпроцессор. Канал А БИС D9 настроен на ввод, а канал В на вывод информации в режиме 1. При необходимости связь с базовой ЭВМ может быть осуществлена и по последовательному каналу, хотя скорость обмена в этом случае существенно ниже. Для этого вместо БИС параллельного интерфейса необходимо установить БИС КР580ВВ51 с соответствующими схемами формирователей.

Микросхема ПЗУ D5 предназначена для хранения основной управляющей программы, состоящей из тестового монитора и программы поддержки обмена информацией через параллельный интерфейс. Тестовый монитор запускается в работу после прихода сигнала Сброс. Проводится проверка всех существенных компонентов спецпроцессора: ОЗУ, ПЗУ и параллельного интерфейса. В последнем случае на разъем, предназначенный для связи с процессором ввода-вывода, должна быть надета специальная заглушка, позволяющая соответствующим образом закоротить выходы интерфейса на его входы. Дополнительным условием запуска тестового монитора является наличие перемычки X1. При обнаружении неисправности загорается светодиод VD2.

Для связи спецпроцессора с базовой ЭВМ через параллельный интерфейс разработан простой байтовый протокол. В начальный момент спецпроцессор выдает в порт А БИС D9 байт синхронизации, имеющий значение 0AAH. Приняв этот байт, базовая ЭВМ подтверждает готовность к обмену выдачей аналогичного байта во входной порт спецпроцессора. После этого начинается обмен информацией.

Первый передаваемый байт от спецпроцессора к базовой ЭВМ — это всегда код операции. Приняты следующие коды:

01H — ввод байта с клавиатуры;
02H — вывод байта на экран дисплея;
03H — вывод байта на АЦПУ;
04H — ввод байта с перфоленты;
05H — вывод байта на перфоленту;
06H — проверка статуса клавиатуры;
07H — чтение блока с магнитной ленты;
08H — запись блока на магнитную ленту;
09H — чтение сектора с диска;
0AH — запись сектора на диск.

При необходимости работы с другими типами внешних устройств базовый набор кодов может быть расширен. При выводе информации из спецпроцессора следующий передаваемый байт — байт информации. При работе с диском после кода операции передается информация о номере диска (1 байт), номере дорожки (2 байта) и номере сектора (1 байт). После этого спецпроцессор передает или принимает 128 байт информации. Процесс заканчивается приемом спецпроцессором кода благополучного завершения записи или чтения сектора с диска. Аналогично строится работа с магнитофоном.

Программное обеспечение базовой ЭВМ должно поддерживать описанный выше протокол обмена. Способ реализации программного обеспечения существенно зависит от типа базовой ЭВМ и номенклатуры подключенного к ней периферийного оборудования. В зависимости от типа операционной системы, под управлением которой работает базовая ЭВМ, могут быть использованы различные по эффективности программные решения. Например, при использовании операционной системы РАФОС целесообразно оформить программу обслуживания спецпроцессора в виде драйвера внешнего устройства. Поскольку ДОС СР/М позволяет работать практически с любыми дисковыми накопителями, то на базовую ЭВМ могут быть возложены дополнительные функции: буферизация отдельных дорожек диска в ОЗУ процессора ввода-вывода и блокирование-деблокирование секторов. Последняя операция необходима в том случае, если размер физического сектора на диске больше 128 байт. Для обмена файлами, записанными в формате ОС базовой ЭВМ и в формате СР/М, могут быть использованы имеющиеся в составе ДОС СР/М программы обмена. Эти программы, как правило, допускают обмен только файлами, содержащими текстовую информацию.

Таким образом, предложенные технические и программные решения позволяют использовать любую ЭВМ в разработке программного обеспечения для микропроцессорных устройств на базе МПК БИС серии КР580. Предложенное устройство целесообразно применять даже с микроЭВМ СМ-1800, которая хоть и снабжена операционной системой, совместимой с СР/М (ОС 1800), но не имеет электронного диска.

Статья поступила 25 февраля 1986 г.

КНИЖНАЯ ПОЛКА

Заморин А. П., Шаров В. А. **Толковый словарь по вычислительной технике и программированию.** Основные термины — М.: Русский язык, 1987 (IV кв.) — 15 л.— 1 р. 00 к.

Словарь является справочным пособием по терминологии в области вычислительной техники и программирования. Он входит в серию толковых словарей по различным отраслям науки и техники, которую издательство начинает в 1987 г.

Содержит около 3 тыс. терминов по архитектуре, принципам работы, структуре ЭВМ, устройствам оперативной памяти, ввода-вывода, телеобработки, вычислительным комплексам, системам программирования, операционным и информационным системам.

Предназначается для широкого круга читателей: инженеров, преподавателей, студентов, учащихся средних специальных учебных заведений и всех, интересующихся вычислительной техникой.

УДК 621.385.001 : 621.387

С. Г. Жулай, В. В. Конько

МИКРОКОНТРОЛЛЕР ВЫВОДА ИНФОРМАЦИИ НА ГАЗОРАЗРЯДНУЮ ИНДИКАТОРНУЮ ПАНЕЛЬ

Рассматривается практическая реализация микропроцессорного контроллера вывода информации на газоразрядную индикаторную панель с самосканированием ИГПС2-222/5×7.

Указанная панель представляет собой систему ортогональных электродов (анодов и катодов), помещенных в газовую смесь между двумя стеклянными пластинами. Изображение любого знака синтезируется из вертикальных фрагментов, индицируемых подачей соответствующих информационных импульсов на индикаторные аноды панели синхронно с импульсами развертки, поступающими на катоды. Для уменьшения количества высоковольтных ключей последние объединены в пять групп (фаз); первый катод соединен с 6-, 11-, 16-м..., второй — с 7-, 12-, 17-м..., третий — с 8-, 13-, 18-м... и т. д. Селективное зажигание ячеек фрагментов при этом обеспечивается их последовательной подготовкой вспомогательным газовым разрядом, возникающим между катодами и дополнительно введенными анодами сканирования, через специальные отверстия в катодах. Тем самым выполняется так называемый режим самосканирования*. Общая инициализация газового разряда в панели осуществляется подачей достаточно длительного обнуляющего импульса на нулевой катод в начале развертки.

Технические характеристики данной панели.

Число строк	1
Число знакомест	32
Формат знакоместа, элемент	5×7
Промежуток между знакоместами, элемент	2
Минимальная частота развертки, Гц	100
Время действия обнуляющего катодного импульса, мкс	300±10 %
Время индикации фрагмента, мкс	50±5 %
Число индикаторных анодов	7

* Достижения в технике передачи и воспроизведения информации. Т. 3: Пер. с англ. / Под ред. Б. Кейзана— М.: Мир, 1980.— 312 с.

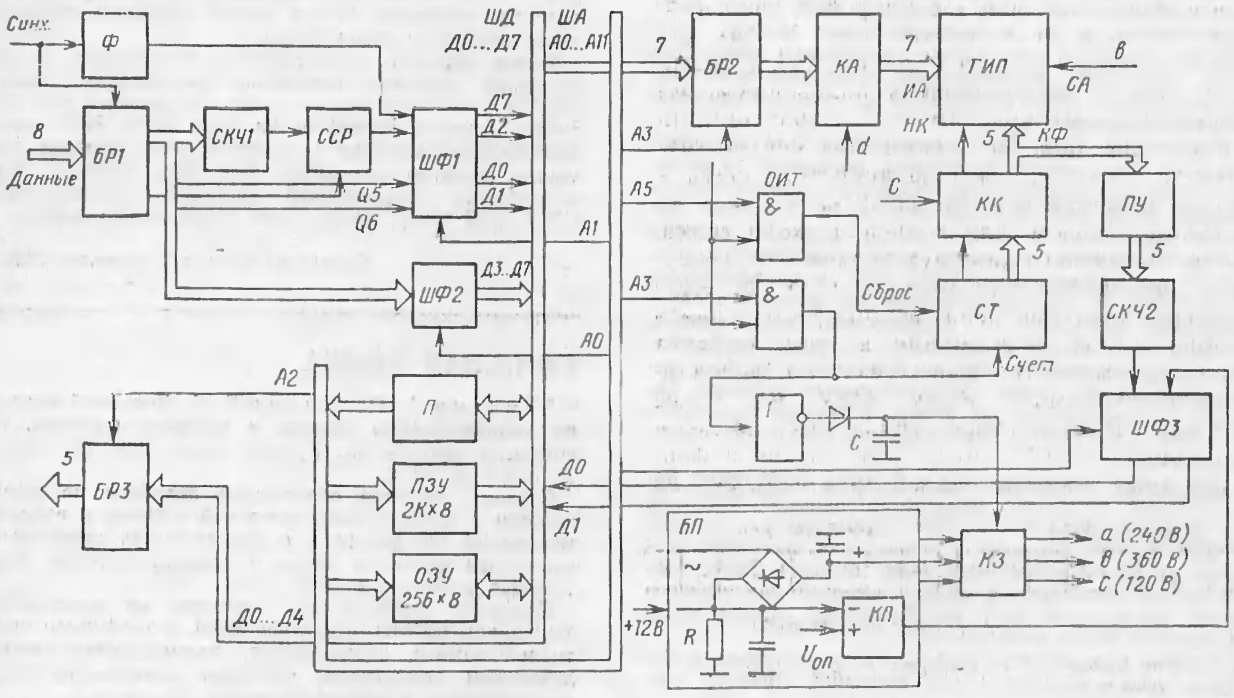


Рис. 1. Структурная схема контроллера вывода информации на газоразрядную индикаторную панель:

П — процессор на основе КР580ИК80А; Ф — флаг; БР — буферный регистр; СКЧ — схема контроля четности; ССР — схема сравнения; ШФ — шинный формирователь; КА — анодные ключи; КК — катодные ключи; КЗ — ключи защиты; СТ — счетчик; ПУ — преобразователь уровня; КП — аналоговый компаратор; БП — блок питания; НК — нулевой катод; КФ — катодные фазы; НА — индикаторные аноды; СА — аноды сканирования; ГИП — газоразрядная индикаторная панель

Контроллер программно и аппаратно реализует следующие основные функции: прием и хранение входной информации, представленной в виде прямых кодов КОИ-7, ее дешифрацию и осуществление предписываемых действий; развертывание изображения и синхронизацию работы отдельных узлов; автоматическую диагностику и идентификацию неисправностей.

Аппаратная часть контроллера (рис. 1) построена по магистрально-модульному принципу. Отдельные функциональные модули объединены посредством инверсной 8-разрядной шины данных, 12-разрядной адресной шины и шины управления (на схеме не показана). Основой процессорного модуля является микропроцессор КР580ИК80А, сопрягаемый с внутренней магистралью системного контроллера КР580ВК38. Максимально возможная частота тактового генератора 2,5 МГц.

Входная байтовая информация через последовательный или параллельный интерфейс принимается в БР1, о чем сигнализирует нулевое состояние флагового триггера Ф. Старший (восьмой) разряд принимаемого слова — контрольный. В контроллере выявляется сбой в линии связи путем сравнения принятого контрольного в ССР разряда с сигналом выхода СКЧ1, осуществляющей проверку четности входного кода КОИ-7. При этом наличие сбоя идентифицируется логической единицей на выходе ССР. Входное слово считывается процессором через шину данных в определенном формате — в виде двух байтов (рис. 2), так как три нулевых разряда в младшем байте позволяют, как это будет видно в дальнейшем, без дополнительных программных затрат организовать адресацию всех семи фрагментов знакоместа (точнее, пяти фрагментов знака и двух — межзнакового промежутка).

Таким образом, для хранения кода знака, соответствующего каждому знакоместу, требуется две ячейки 8-разрядного ОЗУ. В контроллере предусмотрена возможность записи отображаемой информации в одну из двух страниц памяти. Все вместе взятое обусловливает необходимый объем ОЗУ не менее 128 байт. В индикаторе используется последовательный способ записи: при подаче информационных посылок запись производится на текущую страницу ОЗУ с увеличением адреса знакоместа на единицу. Управление записью, а следовательно, и отображением может осуществляться посредством набора команд, передаваемых подобно отображаемой информации (таблица). Их коды отличаются от кодов отображаемых знаков нулевыми значениями битов Q5, Q6.

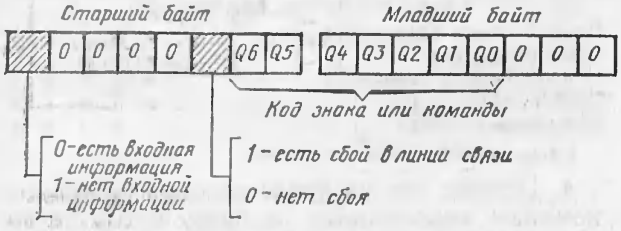


Рис. 2. Формат входного слова

Команды управления индикатором

Наименование команды	Код Q0 . . . Q6	Содержание команды
Обнуление страницы (ОС)	01Н	Занесение в ячейки ОЗУ, соответствующие текущей странице, кода пробела
Перевод страницы (ПС)	0АН	Запись в ОЗУ и отображение другой страницы
Перевод адреса знакоместа на один шаг вправо (ШВ)	09Н	Запрет модификации ячеек ОЗУ, соответствующих текущему знакоместу
Возврат на шаг (ШН)	08Н	Повторная запись в ячейки ОЗУ, соответствующие предыдущему знакоместу
Обнуление адресного счетчика (ВК)	0DH	Запись в ОЗУ текущей страницы продолжается с начального адреса страницы

Каждому отображаемому знаку соответствует хранящийся в ПЗУ набор кодов фрагментов, определяющих начертание знака на индикаторной панели в формате 5×7 элементов.

Индикация любого фрагмента на произвольном знакоместе инициируется записью его кода в БР2 с одновременным перебросом катодного счетчика СТ в соответствующее состояние управляющим сигналом OUT (Вывод), вырабатываемым процессором. Сброс СТ, осуществляемый тем же сигналом, но при обращении по другому адресу (см. ниже), обеспечивает формирование обнуляющего катодного импульса в начале развертки.

С помощью ПУ и СКЧ2 автоматически контролируется отсутствие пробоя в высоковольтных ключах катодных фаз, поскольку этот вид неисправности сильно влияет на долговечность индикаторной панели. Наличие пробоя какого-либо ключа вызывает появление «Лог. 1» на выходе СКЧ2. В контроллере предусмотрена также возможность обнаружения срыва развертки изображения. Периодически проверяется ток анодов сканированием индикаторной панели. Отсутствие тока свидетельствует о постоянном или временном срыве развертки и контролируется по отсутствию падения напряжения на резисторе R в источнике питания анодов сканирования. При этом

на выходе компаратора КП устанавливается «Лог. 1».

Коды названных, а также других типов неисправностей, о которых будет сказано ниже, в виде слова состояния индикатора размещаются в БРЗ, откуда они могут быть извлечены пользователем.

Для защиты индикаторной панели при случайных остановах или зацикливании программы функционирования контроллера предусмотрено автоматическое отключение высокого напряжения, осуществляемое с помощью ключей защиты. Последние в нормальном состоянии открыты смещением, создаваемым емкостью С, которая заряжается регулярно поступающими на счетный вход катодного счетчика импульсами (с частотой примерно 20 кГц).

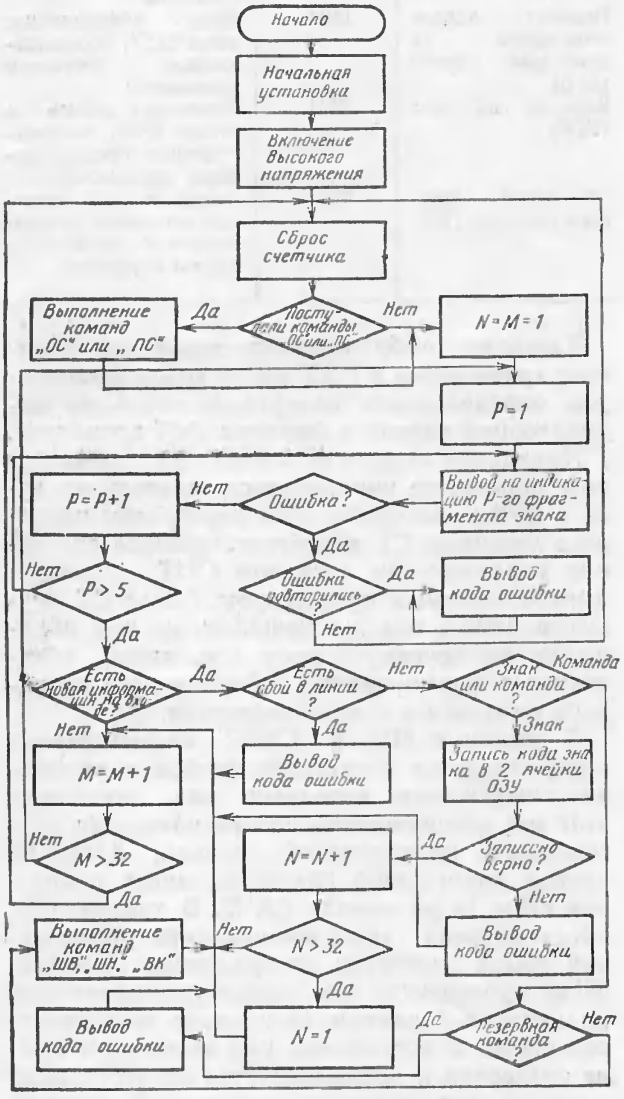


Рис. 3. Обобщенный алгоритм работы контроллера:
N — номер знакоместа при записи в ОЗУ; M — номер индицируемого знакоместа; P — номер фрагмента индицируемого знака

В аварийной ситуации указанные импульсы отсутствуют, конденсатор разряжается, а КЗ закрываются, блокируя тем самым поступление высокого напряжения на индикаторную панель.

Для экономии аппаратных средств адресация внешних устройств осуществляется отдельными разрядами адресной шины (см. рис. 1).

На рис. 3 приведен алгоритм функционирования рассматриваемого контроллера. Для данного алгоритма и его программной реализации целесообразно пояснить следующее:

1. Во время индикации каждого из пяти фрагментов знака на любом знакоместе (50 мкс ±5 %) производится программный опрос ШФ3 с целью обнаружения пробоя катодного ключа или срыва развертки. Для исключения вероятности ложного обнаружения в результате какого-либо случайного сбоя заключение о наличии неисправности принимается только в случае, если данный код ошибки повторится при запуске развертки. При этом выполнение программы работы контроллера прекращается и высокое напряжение, следовательно, отключается от индикаторной панели. Текущая информация, принимаемая процессором из ШФ3, сохраняется в ячейке ОЗУ с адресом 0840H.

2. В алгоритме предусмотрено получение и анализ новой информации из ШФ1, ШФ2 только в период развертывания межзнаковых промежутков (в течение примерно 100 мкс). Это обеспечивает максимальную скорость приема 3300 байт/с.

3. Программно реализуется идентификация еще двух видов ошибок, связанных в первом случае с неисправностью ОЗУ, а во-втором — с получением неучтенного кода команды (резервная команда). При этом используется так называемый локальный способ контроля ОЗУ: каждый раз проверяются только те ячейки ОЗУ, в которые в данный момент производится запись. Выбор подобного способа контроля обусловлен простотой его программной реализации. При обнаружении резервной команды последняя игнорируется, а в БРЗ выдается соответствующий код ошибки.

Полный список идентифицируемых в контроллере неисправностей и ошибок, а также соответствующие им значения отдельных разрядов слова состояния индикатора

Пробой ключа катодной фазы	*	*	*	*	1
Срыв развертки	*	*	*	1	*
Неисправность ячеек ОЗУ	*	*	1	*	*
Сбой в линии связи	*	1	*	*	*
Резервная команда	1	*	*	*	*

(* — произвольное значение)

4. Первые две из представленных в таблице команды выполняются не сразу после их получения, а несколько позже — во время действия обнуляющего катодного импульса в на-

чале следующего периода развертки. Это объясняется длительным временем их программной реализации.

5. Необходимое время индикации каждого фрагмента, включая пустые фрагменты межзнакового промежутка, а также время действия обнуляющего катодного импульса обеспечиваются в разных ситуациях (при различных ветвлениях программы) созданием соответствующих программных задержек.

При программной реализации рассмотренного алгоритма некоторые регистры общего назначения процессора имеют специальное назначение: регистр C используется как счетчик числа знакомест при индикации; регистровая пара HL в любой момент времени содержит абсолютный адрес ячейки ОЗУ, соответствующий текущему знакоместу при записи; указатель стека адресует пару ячеек ОЗУ, содержащую код индицируемого на данном знакоместе знака. При этом формат слова, записанного в указанную пару ячеек, аналогичен представленному на рис. 2 с той разницей, что в его старшем байте могут быть отличны от нуля лишь два младших разряда. В программе данное слово используется как абсолютный адрес первого фрагмента индицируемого знака. Адреса остальных фрагментов, размещенных в последующих ячейках ПЗУ, образуются путем последовательного прибавления к начальному адресу единицы. Точно так же происходит обращение к подпрограммам выполнения поступающих команд: смещенный на три разряда влево код команды служит начальным адресом подпрограммы. Подобный прием в сочетании со стековой адресацией позволяет не только сократить общий объем программы, но, что самое главное, до минимума уменьшить время выполнения отдельных наиболее ответственных ее участков (индикация фрагментов, прием и запись входной информации, выполнение команд). Последнее, в свою очередь, позволяет реализовать основные функции контроллера программным способом, что значительно сокращает аппаратные затраты (рис. 4).

```
;                ACCEMБЛЕР МВО/ОС-1800
; ***************************************************
; *          CHAR:ПОДПРОГРАММА ИНДИКАЦИИ ЗНАКА      *
; ***************************************************
;       ИСПОЛЬЗУЕМЫЕ РЕГИСТРЫ: A,D,E,H,L
;       ПОДПРОГРАММЫ:           НЕТ

        DATA    EQU     0BH     ;БР2
        STATUS  EQU     10H     ;ШФ3
        ERROR   EQU     04H     ;БР3

        CHAR:   POP     D       ;АДРЕС ПЕРВОГО ФРАГМЕНТА ЗНАКА
        LAB1:   LDAX    D       ;ПОЛУЧЕНИЕ КОДА ТЕКУЩЕГО
                                ;ФРАГМЕНТА
                INX     D       ;ФОРМИРОВАНИЕ АДРЕСА СЛЕ-
                                ;ДУЮЩЕГО ФРАГМЕНТА
                OUT     DATA    ;ИНДИКАЦИЯ ФРАГМЕНТА
                CPI     00H
                JNZ     LAB5    ;К ИНДИКАЦИИ МЕЖЗНАКОВОГО
                                ;ПРОМЕЖУТКА
                IN      STATUS  ;СЧИТЫВАНИЕ ВЫХОДОВ ШФ3
                CPI     00H     ;ЕСТЬ ОШИБКА?
                JNZ     LAB2
                XCHL
                LXI     H,0840H
                CMP     M       ;ОШИБКА ПОВТОРИЛАСЬ?
                JNZ     LAB3
                MOV     M,A
                XCHL
                JMP     LAB6    ;К СБРОСУ СЧ
        LAB2:   MVI     03H     ;НАЧАЛО БЛОКА ПРОГРАММНОЙ
                                ;ЗАДЕРЖКИ НА 25 МКС
        LAB4:   DCR     A
                JNZ     LAB4
                JNP     LAB1
        LAB3:   OUT     ERROR   ;ВЫВОД КОДА ОШИБКИ
                JMP     LAB3    ;ЗАЦИКЛИВАНИЕ ПРОГРАММЫ
        LAB5:   RET
```

Рис. 4.

Построение аналогичных контроллеров на основе однокристальных МП с более высоким быстродействием, чем у КР580ИК80А, позволит еще большую часть выполняемых функций поручить программе. Это положительно скажется на массогабаритных и стоимостных характеристиках устройств отображения информации, построенных на базе индикаторных газоразрядных панелей постоянного тока.

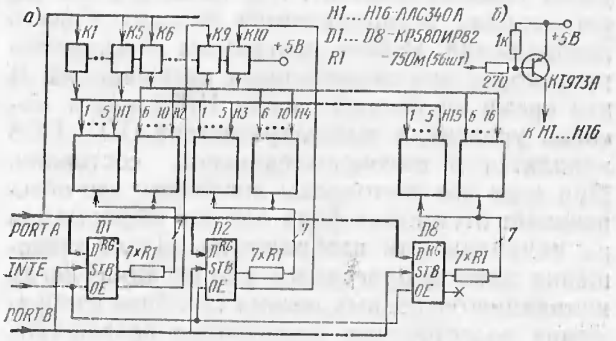
Статья поступила 8 августа 1985 г.

УДК 681.326

Сем. С. Селицкий, Ст. С. Селицкий, В. Н. Сидоров

ПРОСТОЙ ПУЛЬТОВОЙ ДИСПЛЕЙ

Пультовой дисплей содержит 16 алфавитно-цифровых знакомест, отображаемых светодиодными матричными индикаторами АЛС340А (см. рисунок). Для управления индикаторами используются восемь байтовых регистров КР580ИР82 и десять токовых ключей КТ973А. К микропроцессорной системе дисплей подключается через два порта вывода PORT A и PORT B. Вынесение этих портов (тоже регистры КР580ИР82) вместе с их адресным дешифратором на плату дисплея дает возможность подключать его непосредственно к системной шине.



Пультовой дисплей:
функциональная схема (*а*), схема ключей K1...K10 (*б*)

Управляющая программа объемом 700 байт в системе команд КР580ИК80А содержит три основных блока: диспетчер разделения времени, компилятор и блок регенерации изображения. Благодаря диспетчеру разделения времени управление дисплеем осуществляется на фоне любой пользовательской программы, к которой при этом не предъявляется никаких дополнительных требований, но ее выполнение в режиме разделения времени замедляется приблизительно в 1,5 раза.

Компилятор преобразует текст сообщения, хранимого в текстовом буфере в кодах КОИ-7 (длина буфера 16 байт), в форму, необходимую для вывода его на индикацию. Используемый при этом знакогенератор содержит 96 символов — буквы русского и латинского алфавитов, цифры, знаки препинания и некоторые другие. В такой форме сообщение переписывается в буфер регенерации, имеющий длину 80 байт — по числу столбцов в изображении — и играющий роль «страничной» памяти дисплея.

Программа регенерации изображения последовательно выводит информацию индицируемых в данном цикле столбцов в порт данных PORT A и фиксирует ее в соответствующем регистре DD1 ... DD8, выводя импульс в один из разрядов порта управления PORT B. Затем в порт данных выводится унитарный код, обеспечивающий включение необходимого токового ключа. Если в данном цикле регенерации необходимо включение токовых ключей К9 или К10, то информацию об этом программа регенерации заносит в старший бит регистров DD1 или DD2 одновременно с информацией, индицируемой в данном цикле в девятом столбце первой пары знакомест Н1, Н2 или в десятом столбце второй пары знакомест Н3, Н4 соответственно. Таким образом в каждом цикле регенерации осуществляется индикация восьми столбцов изображения (по числу байтовых регистров). Скважность регенерации каждого столбца Q=10 (по числу токовых ключей), т. е. за десять циклов регенерации осуществляется развертка всего изображения. Работа программы регенерации происходит при запрещенных прерываниях. В это время системный сигнал INTE имеет высокий уровень, а выходы регистров DD1...DD8 находятся в высокоимпедансном состоянии. При этом все светодиоды погашены, что обеспечивает отсутствие фона ложной информации на индицируемом изображении. После завершения работы программы регенерации, когда индикация очередных восьми столбцов изображения подготовлена, прерывания разрешаются. Выходы регистров DD1 ... DD8 переходят в активное состояние, включая соответствующие светодиоды матричных индикаторов.

На время индикации диспетчер передает управление пользовательской программе, а по истечении времени индикации управление по прерыванию вновь передается программе регенерации для вывода на дисплей очередных восьми столбцов изображения.

Пультовой дисплей работает в четырех режимах: непрерывный, мигающий, попеременный вывод пары сообщений, «бегущая строка». Необходимый режим индикации указывается в оглавлении, содержащем до 126 адресов исходных сообщений, каждое из которых после вывода на дисплей может программно модифицироваться. Однако модификация исходных сообщений является уже функцией программы пользователя.

Все сообщения, содержащие более 16 символов, по умолчанию выводятся в режиме «бегущая строка». Сообщения короче 16 символов, выводимые в любом режиме, кроме «бегущей строки», автоматически центрируются.

На дисплей могут выводиться как фиксированные сообщения, хранимые в банке сообщений, так и организуемые пользователем в ОЗУ. Банк сообщений формируется в кодах КОИ-7, записывается в ПЗУ и размещается в задаваемой программой области адресного пространства в пределах первых 16К байт. Для вывода сообщения на дисплей достаточно переслать его порядковый номер в соответствующую ячейку ОЗУ. Так же просто осуществляется программная модификация изменяемой части исходного сообщения и исходного режима индикации.

Для работы программы необходимы 240 байт ОЗУ, три канала цифрового программируемого таймера КР580ВИ53 и один уровень прерывания. Эта же программа (при изменении трех параметров) может управлять дисплеем, содержащим 24 знакоместа. В аппаратуру дисплея, кроме восьми индикаторов, требуется добавить лишь пять токовых ключей, изменив соответствующим образом ее функциональную схему. Теперь каждый байтовый регистр управляет не двумя, а тремя индикаторами. Скважность регенерации каждого столбца изображения вместо десяти становится равной 15, т. е. развертка всего изображения осуществляется за 15 циклов регенерации. Информация о необходимости включения токовых ключей К11 ... К15 выводится в старшие биты байтовых регистров DD3 ... DD7 по аналогии с информацией о включении ключей К9 и К10. Дополнительных ресурсов микропроцессорной системы при этом не требуется, так как указанных выше 240 байт ОЗУ достаточно и в этом случае.

Статья поступила 25 декабря 1985 г.

УВАЖАЕМАЯ РЕДАКЦИЯ!

Очень удачное, на мой взгляд, решение — публикация статей по принципу подборок на конкретную тему. Великолепны подборки «Программирование ППЗУ», «Контроллер ЭЛТ». Один из сложных моментов при проектировании микроЭВМ — это контроллер регенерации динамического ОЗУ. Приведенная в журнале «Радио» схема реализует алгоритм скрытой, прозрачной регенерации. К сожалению, контроллер регенерации «Радио» не позволяет использовать в МП-системах тактовый генератор К580ГФ24 и системный контроллер К580ВК28. Если есть возможность, опубликуйте или вышлите материал по такому контроллеру регенерации.

С уважением
Валл Владимир Иванович, г. Алма-Ата

УДК 681.327.8.06

Ю. А. Еремин, А. Г. Морозов

КОНТРОЛЛЕР ДИНАМИЧЕСКОГО ОЗУ ДЛЯ МИКРОПРОЦЕССОРНЫХ УСТРОЙСТВ

Для построения ОЗУ микропроцессорных устройств используется статическая и динамическая память произвольного доступа. При разработке ОЗУ объемом больше 10К байт для удовлетворения требований по стоимости, габаритам и энергопотреблению предпочтение отдается динамическим запоминающим устройствам. Работа этих устройств поддерживается специальной схемой-контроллером, которая управляет обращением к ОЗУ и регенерацией. Задача разработчика, как правило, сводится к уменьшению доли контроллера в общем объеме схемотехники ОЗУ, чтобы затраты на контроллер не зачеркнули преимуществ динамического ОЗУ.

В настоящее время в составе микропроцессорных устройств широко применяется микропроцессор КР580ИК80. Контроллеры динамических ОЗУ таких устройств для обеспечения режима скрытой регенерации часто используют информацию о состоянии процессора. Это увеличивает аппаратные затраты и не позволяет включать схему конкретного контроллера в состав других микроЭВМ, где не предусмотрена фиксация такой информации. Предлагаемый контроллер динамического ОЗУ при минимальных аппаратных затратах работает в режиме скрытой регенерации и не требует информации о состоянии процессора (рис. 1).

Контроллер включает узел формирования запросов обращения к памяти (элементы D1.1-3, D5.3, D6.1), узел формирования управляющих сигналов CAS, WE (элементы D1.4,

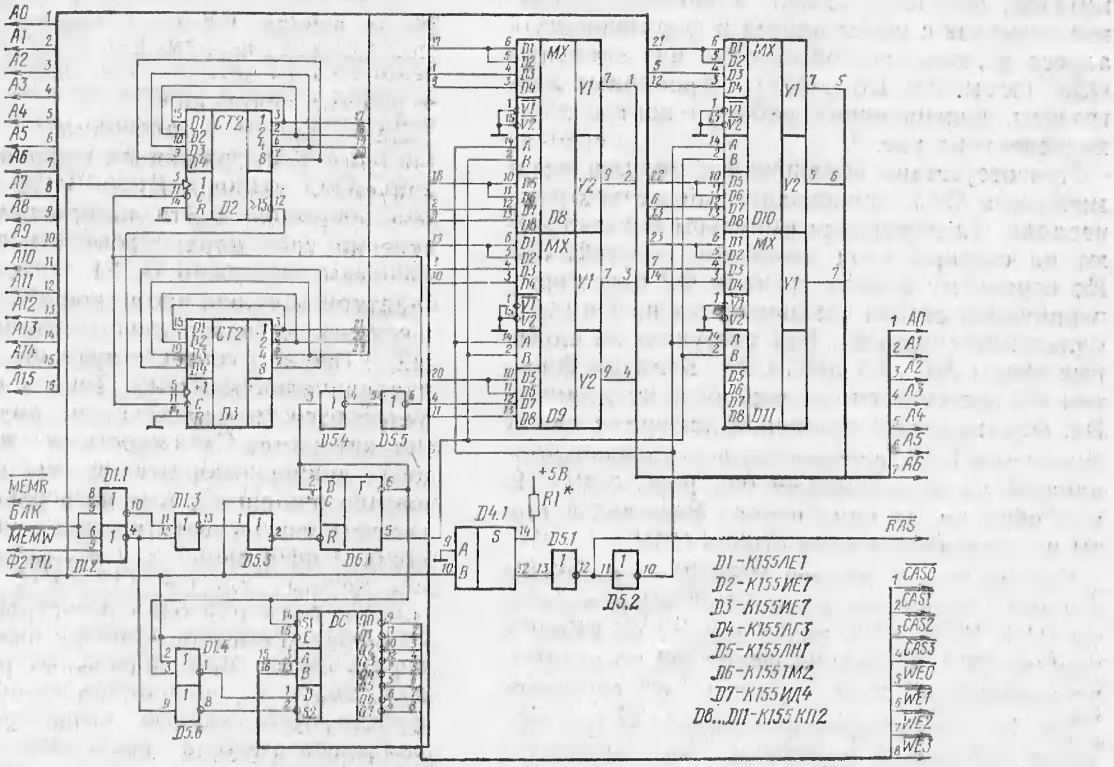


Рис. 1. Электрическая принципиальная схема контроллера

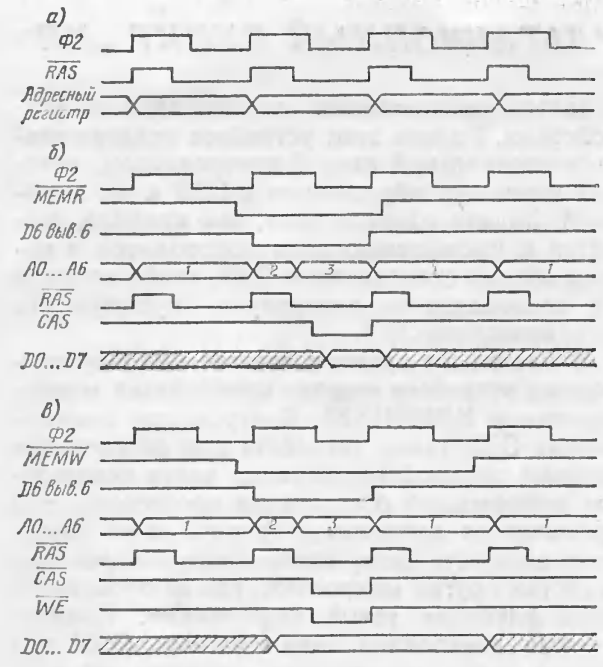


Рис. 2. Временные диаграммы:
режим регенерации (*а*), режим чтения (*б*), режим записи (*в*); *1* — адрес регенерируемой строки; *2* — младшие адреса адресной магистрали; *3* — старшие адреса адресной магистрали.

D5.6, D7), формирователь сигнала $\overline{RAS}$ (элементы D4, D5.1), формирователь адреса регенерируемой строки (элементы D2, D3), коммутатор, обеспечивающий мультиплексирование сигналов с шины адреса и формирователя адреса регенерируемой строки на элементы ОЗУ (элементы D8 ... D11). Временные диаграммы, поясняющие работу контроллера, приведены на рис. 2.

При отсутствии обращений к модулю динамического ОЗУ происходит процесс его регенерации. Тактирующие импульсы Ф2 поступают на счетный вход двоичного счетчика D2. По переднему фронту сигнала Ф2 адрес регенерируемой строки увеличивается на 1 и через мультиплексоры D8 ... D11 поступает на адресные входы A0 ... A6 БИС ОЗУ. Этим же фронтом Ф2 запускается одновибратор на элементе D4, формирующий временной интервал между сигналами $\overline{RAS}$, соответствующий паспортным данным на применяемый тип БИС ОЗУ. Таким образом, за один период тактовой частоты регенерируется одна строка ОЗУ.

Сигнал чтения памяти $\overline{MEMR}$, поступивший с шины управления микроЭВМ через элементы D1.1, D1.3, D5.3, разблокирует по входу R триггер D6.1. Передний фронт Ф2 по входу C устанавливает этот триггер в состояние «Лог. 1». Низкий уровень с вывода $\overline{Q}$ триггера после небольшой задержки на элементах D5.4, D5.5 поступает на вход C счетчика D1, D2, приостанавливает регенерацию на период обращения к памяти и сохраняет текущее состояние счетчика. Задержка обеспечивает срабатывание счетчика по переднему фронту Ф2. Сигнал с выхода $\overline{Q}$ триггера переключает мультиплексоры D8 ... D11, в результате чего на микросхемы ОЗУ поступают младшие разряды адреса ячейки, к которой происходит обращение. Сигналом $\overline{RAS}$ этот адрес запоминается во внутреннем адресном регистре БИС. После задержки на элементе D5.2 этот сигнал переключает мультиплексоры на выдачу старших разрядов адреса. Низкий уровень сигнала Ф2 при наличии низкого уровня на выходе $\overline{Q}$ триггера D6.1 разрешает работу дешифратора D7. В зависимости от состояния старших разрядов адреса микроЭВМ A15 и A16 формируется один из сигналов $\overline{CAS0}$ ... $\overline{CAS3}$, которым выбирается одна из четырех линеек БИС. Информация с выходов выбранных БИС поступает на шину данных микроЭВМ. Очередным передним фронтом сигнала Ф2 триггер D6.1 сбрасывается в состояние «Лог. 0», что приводит к переключению мультиплексоров и выдаче на адресные входы БИС ОЗУ адреса регенерируемой строки. Одновременно снимается сигнал $\overline{CAS}$ и разблокируется счетчик регенерации. По окончании действия сигнала $\overline{MEMR}$ схема возвращается в режим регенерации.

Цикл записи практически не отличается от цикла чтения. Сигналы записи $\overline{WE0}$ ... $\overline{WE3}$ формируются аналогично сигналам $\overline{CAS}$ при наличии низкого уровня сигнала $\overline{MEMW}$. Существенно, что в режиме записи сигналы $\overline{CAS}$ и $\overline{WE}$ поступают на управляющие входы БИС ОЗУ одновременно. При этом выходы БИС остаются в высокоимпедансном состоянии, что позволяет объединять вход и выход данных микросхемы ОЗУ.

Для использования в микроЭВМ других запоминающих устройств (статического ОЗУ, ПЗУ, ППЗУ и т. п.) предусмотрен вход блокировки запросов обращений к памяти БЛК. Устройство, адресное пространство которого перекрывается с адресами динамического ОЗУ, должно при дешифрации своего адреса выработать сигнал высокого уровня БЛК, который препятствует прохождению сигналов $\overline{MEMR}$, $\overline{MEMW}$ на вход R триггера D6.1.

Для работы контроллера необходимо, чтобы сигналы обращения к памяти $\overline{MEMR}$ и $\overline{MEMW}$ перекрывали один период тактирующих импульсов, т. е. начинались раньше переднего фронта тактирующего импульса, что нужно для своевременной разблокировки триггера D6.1, и заканчивались позже следующего положительного фронта, обеспечивая тем самым

регенерацию всех строк ОЗУ без пропуска. Необходимые временные соотношения управляющих сигналов достигаются, например, в МП устройствах, построенных с использованием системного контроллера К580ВК38 и тактового генератора К580ГФ24. При использовании системного контроллера К580ВК28 необходимо обеспечить задержку заднего фронта сигнала $\overline{\text{MEMW}}$, чтобы он поступал на вход триггера D6.1 после переднего фронта Ф2. Элементы, вносящие задержку, включаются между выходом D1.2 и входом D1.3.

Контроллер можно применять в микропроцессорных системах, построенных на основе других типов микропроцессоров, если в них соблюдаются необходимые соотношения между сигналами обращения к памяти и тактовым сигналом.

Многие записки на семинарах «МП» в Политехническом музее и адресованные редакции читательские письма содержат одну и ту же просьбу: «Объясните, пожалуйста, есть ли способы защиты авторского права на микропроцессорную технику и ее программное обеспечение?» На вопросы читателей отвечает заведующий отделом Всесоюзного научно-исследовательского института государственной патентной экспертизы.

УДК 681.325.5

Т. А. Арешев

ПАТЕНТНАЯ ОХРАНА МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

Аппарат правовой охраны нормами изобретательского права на средства вычислительной техники с использованием микропроцессоров (МП) в настоящее время недостаточно разработан. Экспертиза и заявители сталкиваются с серьезными трудностями.

Как известно, основной документ в этой области «Разъяснение № 4» Госкомизобретений не предусматривает защиту предложений, решающих поставленную задачу чисто алгоритмическим или программным путем.

Однако наличие в рассматриваемых объектах и аппаратных средств позволяет рассмотреть три возможных метода их правовой охраны изобретательским правом.

1. Охрана с указанием в формуле изобретения (ФИ) термина «Микропроцессор» с приведением в описании типа соответствующего МП, его конкретно используемых контактов и блок-схемы алгоритма работы МП.

2. Охрана с указанием в ФИ функциональных названий блоков (функции которых реализованы отдельным МП) с приведением в описании типа соответствующего МП, его конкретных используемых контактов и блок-схемы алгоритма работы соответствующего МП.

3. Охрана с указанием в ФИ функциональных названий блоков (функции которых реализованы отдельным МП) с приведением в описании блок-схемы алгоритма решения задачи соответствующего МП и эквивалентной данному алгоритму решения задачи функциональной схемы устройства.

При анализе возможности правовой охраны вычислительных средств с использованием МП в соответствии с предложенными методами необходимо учитывать тот факт, что правовая охрана должна предусматривать контроль (проверку) возможности решения поставленной задачи и возможности сравнения различных технических решений одной и той же или набора аналогичных задач.

Кроме того, для определения возможности контроля реализации поставленной задачи необходимо учитывать, что степень абстрагирования от реального процесса (физического, технологического, информационного) должна быть соизмерима с параметрами самого процесса. А именно, отдельные информационные символы, слова, над которыми производятся преобразования, должны быть информационно идентичны соответствующим параметрам процесса.

Сами преобразования должны быть прослеживаемы средним специалистом без привлечения каких-либо вспомогательных средств.

Если эти три условия не соблюдены, то решение задачи не удовлетворяет требованию контролируемости. Это означает, что патентная охрана таких объектов неопределенна.

Исходя из данных трех предпосылок можно оценить возможность использования каждого из методов охраны его преимущества и недостатки.

Для **первого метода правовой охраны** можно определить функциональную направленность устройства (из названия и области использования), но не функциональную направленность отдельных узлов, задачи которых решаются с использованием МП.

Представление решения в виде блок-схем алгоритмов не позволяет контролировать решение поставленной задачи и сравнивать различные технические решения одной или набора аналогичных задач.

Вследствие алгоритмического решения задачи в таком предложении невозможно выделить конкретные блоки и их функциональные взаимосвязи, составляющие совокупность признаков технического решения.

Вывод. Первый метод правовой охраны не является определенным и не может быть использован для охраны рассматриваемых технических решений.

Для **второго метода правовой охраны** можно установить функциональную направленность всего устройства в целом и отдельных узлов, задачи которых решаются с использованием МП.

Указание типа МП и его используемых контактов в некоторой степени ограничивает уровень охраны технических решений, так как каждый тип МП характеризуется своим набором сигналов и средств, организующих «процессорное ядро», и при смене типа микропроцессора изменяются условия решения задачи и ставится под сомнение возможность ее решения.

Следует отметить, что указание конкретного контакта некорректно, так как для нормального функционирования МП необходимо подключение всех его контактов. С другой стороны, указание конкретного вывода (контакта) влечет за собой (в случае использования другого контакта) неправомерную охрану объекта на основе использования совокупности признаков (на самом деле отсутствующей).

Кроме того, представление решения в виде блок-схем алгоритмов не позволяет контролировать возможность решения поставленной задачи и сравнивать различные технические решения одной или набора аналогичных задач.

Вследствие алгоритмического решения задачи в таком предложении невозможно выделить конкретные блоки и их функциональные взаимосвязи, составляющие совокупность признаков технического решения.

Вывод. Второй метод правовой охраны, как и первый, не является определенным и неприменим для охраны рассматриваемых технических решений.

Для **третьего метода правовой охраны** можно установить функциональную направленность устройства в целом и входящих в него узлов, задачи которых решаются с использованием МП.

Функциональные структурные схемы устройств, соответствующих приведенным блок-схемам алгоритмов используемых МП, позволяют контролировать решение поставленной задачи и сравнивать различные технические решения одной или набора аналогичных задач.

Третий метод можно использовать для охраны технических решений в области ВТ с использованием микропроцессоров.

Схема представления технического решения.

В формуле изобретения приводится совокупность функциональных узлов и блоков с их взаимосвязями, в том числе блоков, задачи которых решаются микропроцессорами.

В описании кроме остальных обязательных атрибутов приводятся разделы:

— «постановка задачи», в котором решаемая задача описывается словесно или с математическими соотношениями;

— «модель решения задачи», в котором описаны организационные и математические методы, используемые при решении поставленной задачи с выделением частей данного решения, выполняемых блоками на базе микропроцессоров;

— «алгоритм решения задач», в котором более подробно описываются общий алгоритм решения задачи и алгоритмы используемых микропроцессоров;

— «функциональные структурные схемы устройств», соответствующих реализуемым алгоритмам, в котором приводятся структурные схемы этих устройств, представляющих собой совокупность конкретных технических средств (и их взаимосвязей) и их функционирование.

Такое представление технических решений, не вынуждая эксперта необоснованно доверять или не доверять утверждениям заявителя о возможности решения поставленной задачи и ее частей, решаемых микропроцессорами, позволит экспертизе однозначно определить возможность решения задачи и, кроме того, новизну при сравнении известного и нового технических решений.

Такой метод патентования позволяет охранять средства вычислительной техники с использованием микропроцессоров при увеличивающемся потоке указанных технических решений, не ограничивая объем изобретения, сохраняя конкретность каждого технического решения и обоснованно оценивая осуществимость и новизну предложения.

Телефон для справок по затронутым вопросам: 240-55-32 по вторникам с 9 до 11 ч.

Статья поступила 17 января 1986 г.

УВАЖАЕМАЯ РЕДАКЦИЯ!

Являюсь подписчиком «МП» с 1984 года. Очень обрадовало известие, что журнал теперь будет издаваться 6 раз в год. Меня заинтересовала статья в № 1, 1986 г. о персональной ЭВМ «Ириша». Решил ее повторить. Прошу вас ответить на несколько вопросов:

1. Какие неточности есть в принципиальной и монтажной схемах ТВ-адаптера?

2. Какие микросхемы можно заменить микросхемами серии К155, в частности, возможна ли замена К555ИЕ7 на К155ИЕ7, К531ЛАЗП на К155ЛАЗ и др.?

3. Возможна ли замена микросхем К565РУ6 на К565РУ3?

Шугин А. Г., г. Владивосток

Ответ авторов цикла статей по компьютеру «Ириша».

После выхода журнала авторы провели контрольную сборку графического адаптера ПЭВМ «Ириша», используя материалы, приведенные в статье. В процессе сборки выявились следующие неточности в принципиальной и монтажной схемах:

1. На монтажной схеме положение резисторных сборок HP3 и HP7 показано неверно: их необходимо развернуть на 180°, чтобы общий вывод был соединен с шиной питания +5 В.
2. Элементы D38 ... D42 пронумерованы в обратной последовательности.
3. На принципиальной схеме сигналы MAS1 (с вывода 15 элемента D52) и MEMC (к выводу 2 элемента D55.1) должны быть без инверсии.
4. На вывод 2 элемента D53.1 приходит сигнал q_2.
5. На выводы 12, 13 элемента 59.4 поступает сигнал q_1, который с выхода 11 через жгут подается на выходы 2 и 5 элемента D50.
6. На вывод 15 элемента D3 приходит сигнал 10/M.

На вопросы относительно замены микросхем, указанных на принципиальной схеме, на микросхемы других серий и типов, но сходных по выполняемым функциям, можно ответить следующее:

— замена микросхем К565РУ5 на микросхемы К565РУ6 допустима, при условии, что в ПЭВМ будет установлен модуль дополнительного ОЗУ объемом не менее 48К байт. Замена на микросхемы К565РУ3 невозможна, так как они не обладают достаточным для данной схемной реализации быстродействием. Кроме того, в этом случае пришлось бы переделать трассировку платы, поскольку эти микросхемы требуют для работы трех напряжений питания;

— замена микросхем серии К555 на микросхемы К155 нежелательна, поскольку приводит к резкому повышению потребляемой мощности, что в свою очередь может привести к нарушению теплового режима ПЭВМ. Особенно это заметно при замене микросхем К555ИЕ7 на микросхемы К155ИЕ7;

— замена микросхем серий К531 и К155 на микросхемы серии К555 в ряде случаев возможна и даже желательна. Так, замена микросхем К531КП2 в адресном мультиплексоре, микросхем К155ТМ8, К155ТМ2, К531ИР18П, К531ИР19П на аналогичные по функциям микросхемы серии К555 позволяет уменьшить потребляемый модулем ток почти на 35 %.

Главному редактору журнала академику Ершову А. П.

Андрей Петрович!

Хочу обратить Ваше внимание на опасность массового распространения «компьютерной полуграмотности». Я имею в виду не недостаточное качество обучения на сегодня, я знаю Вашу точку зрения, что важно хотя бы начать. Вопрос в том, что многие готовы были бы считать вопрос о грамотности решенным (и «закрытым») в результате обучения чему-нибудь вроде Бейсика. И вопрос не в том, что Бейсик плох (или, может быть, хорош), а в том, что это совсем не то умение, которое поможет массовому специалисту, нуждающемуся в использовании вычислительных средств. Не потому, что не нужно в школе знать какой-то язык типа «мини-языка» Информатики-9 (не знаю, как его называть, но свято место пусто не бывает, и злые языки уже прозвали его «Ершолом»). Только этот язык (как и формируемое на его основе представление об алгоритме), скорее, относится к математическому курсу, а не к информатике (аналогично знанию прогрессий или символа Σ).

Всерьез специалисту нужно не кропать для себя мелкие самостоятельные программки (их возможности, особенно у непрофессионала, весьма ограничены), а уметь обращаться к готовым средствам, дополняя их понемногу собственным программированием. Но против чего я больше всего здесь возражаю, это против ориентации на «инстинктивное» программирование, по Г. Р. Громову. Думаю, что в сложном случае человек при таком подходе просто не справится с задачей и махнет рукой на программирование вообще.

Появилась и другая серьезная опасность. Старшие школьники и студенты, «дорвавшиеся» до машины и предоставленные самим себе, замыкаются в примитивном круге задач, превращаются в «фанатиков» (термин моих студентов, наблюдавших это явление в свою бытность в школе-интернате). Их больше всего начинает интересовать хитроумное использование каких-либо малоизвестных возможностей машины или операционной системы, возможность обойти разные запреты, недобро подшутить над соседом. Я уж не говорю о пролезании на машину с черного хода или ночных бдениях. Замкнувшись во всем этом, иные утрачивают способность перейти к более серьезным задачам, вылетают с факультета, а то и попадают в больницу. Или такой «пострел» уже на первом курсе начинает наводить справки, у кого бы купить за наличные бобину магнитной ленты (краденую, естественно). О заграничной породе «хакеров» (hackers) Вы, вероятно, лучше меня знаете. Это явление, может быть, не специфично для программирования: В. А. Залгаллер, учивший нас в университете геометрии (и заодно уму-разуму), рассказывал об опасности для способного школьника или студента застрять в круге задач элементарной математики.

И не стоит отвечать, что все эти ребята со временем «перебесятся», зато приобретут необходимую в других делах виртуозность. Я никак не против игры и не против виртуозности, но если мы не поведем этих людей вперед, ставя перед ними такие задачи, по сравнению с которыми всякие машинные трючки покажутся им неинтересными, то само это не пройдет. И не надо ссылаться на наше поколение, успешно миновавшее эту фанатизацию. Вот аналогия: моего сына кое-кто упрекает в злоупотреблении сладкой пищей. Я попытался вспомнить, почему мне такое не повредило. И вспомнил: денег не хватало.

Здесь о молодежи я сказал то же, что о взрослых, ликвидирующих «вторую безграмотность». Нельзя останавливаться на элементарных навыках программирования, а может быть, и вообще не стоит делать на них упор.

Доктор физ.-мат. наук **Г. С. Цейтин,**
математико-механический факультет
Ленинградского государственного университета

Григорий Самуилович!

Разделяя в целом Ваши предостережения об опасности «компьютерной полуграмотности», хотел бы сделать ряд замечаний в защиту различных сторон компьютерного всеобуча.

Мы воспринимаем термин «массовая культура» всегда с некоторыми оговорками, не упуская ни одного повода поговорить об ее издержках. Однако уже в силу объективных законов социальной психологии мы не можем от любого крупномасштабного явления оставить только его одну, «лучшую» сторону в надежде избавиться от другой, «худшей».

Полуграмотность, бескультурье, злоупотребление, сектантский фанатизм — это, к сожалению, универсальные стороны человеческого бытия, для которых предметная область или жанр являются всего лишь той или другой модной одеждой, прикрывающей более постоянную неприглядную суть.

Злой шутник найдет способ пошутить и без компьютера, а для молодого коммерсанта видеолента ничем не хуже дискеты с компьютерными играми. Вы правы — были бы деньги, а повод злоупотребить ими всегда найдется.

Возможность застрять на элементарных умениях не может отменять необходимость изучения основ. Точно так же эти основы не могут быть компенсированы способностью работы с конечным программным продуктом. Вы сами говорите: «и немного программировать самому». Так вот, электронные таблицы удивительно удобны, пока Вы находитесь в пространстве заданных возможностей. Однако даже самая невинная попытка выйти за эти пределы превращает удобнейший инструмент в очень неуклюжую систему программирования. И если разматывать эту цепочку нарастающих проблем, Вы не сможете остановиться, пока не упретесь в необходимость элементарной алгоритмической нотации, поддерживающей принципы структурного программирования.

Что до «Ершола», то мне лестна подобная персонификация. Однако справедливости ради я не могу считать алгоритмический язык Информатика-9 своим изобретением. Это общее достояние, сложившееся за последние 20 лет и уже увековечившее себя в сотнях статей, десятках книг и ряде систем и технологий программирования. Единственное, за что я берусь отвечать,— это тезис, что эта сложившаяся система обозначений достойна занять свое место рядом с грамматикой родного языка, рядом с языком математических символов.

В заключение об «**интуитивном** программировании» по Г. Р. Громову. Здесь, мне кажется, надо различать три вещи: 1) интуитивный, эмпирический, «снизу вверх» способ постижения грамматики языка; 2) уровень искусства во владении языком программирования и 3) выбор наиболее подходящего языка для реализации «автоформализации профессиональных знаний». Думается, что уже само осознание этих различий и возможность спорить об одном, не затрагивая другого, позволит найти рациональный баланс энтузиазма и критики.

А. Ершов

Статьей Г. Р. Громова журнал завершает публикацию материалов «круглого стола», который проходил в Большом зале Политехнического музея 4 октября 1985 г. (см. «МП», 1986, № 2, с. 92—93).

УДК 681.322.1-181.4+681.338.45

Г. Р. Громов

АВТОФОРМАЛИЗАЦИЯ ПРОФЕССИОНАЛЬНЫХ ЗНАНИЙ

Введение

Темп развития технологической цивилизации определяется в значительной степени темпом накопления профессиональных знаний. В свою очередь общая сумма потенциально доступных членам человеческого общества знаний зависит от достигнутого на данном историческом этапе уровня эффективности процесса «отчуждения» индивидуально генерируемых «частиц» знания от автора — их первичного источника и начального носителя; человека, который первым овладел тем или иным новым технологическим приемом, методом, средством и т. д.

Истоки информационной технологии

На ранних этапах развития цивилизации профессиональные навыки передавались в основном личным примером исполнения производственных действий (новых приемов охоты, обработки шкуры, костей животных и т. д.). Рациональные способы организации коллективных действий, синхронизация производственных усилий закреплялись для передачи из поколения в поколение ритуальными танцами, обрядовыми песнями, устными преданиями и т. д.

«Помехоустойчивость» социально-исторического канала передачи профессиональных знаний заметно возросла с открытием человеком элементов технологии длительного хранения на материальном носителе отдельных, наиболее характерных зрительных образов, связанных с накопленными знаниями. Произошло это по историческим меркам совсем недавно — каких-нибудь двадцать — тридцать тысяч лет назад. Именно тогда появились первые наскальные рисунки. Возраст человеческой цивилизации составлял к этому времени сотни тысячелетий.

Шесть тысяч лет назад технология регистрации на материальном носителе символьно кодированной информации о накопленных знаниях достигла того порогового уровня, с которого ведут отсчет эры письменности. Таким образом, за каких-нибудь двадцать тысяч лет человеком был пройден путь от наскальных рисунков до первых глиняных табличек с текстами. Это был путь поиска все более совершенных способов кодирования и расшифровки фиксируемых для длительного хранения на материальном носителе элементов знаний.

Начатый тогда процесс совершенствования носителей информации и инструментов для ее регистрации продолжается до сих пор: камень, кость, дерево, глина, папирус, шелк, бумага, люминофор, магнитные и оптические носители, кремний, цилиндрические магнитные домены и т. д.

Однако, накапливаемые в виде отдельных записей или книг профессиональные знания не могли непосредственно влиять на производственный процесс. Чтобы получить шанс «прорасти» новым знанием или повлиять на характер выполняемого другими людьми трудового процесса, книга должна была попасть «на благодатную почву» — на нее должен был «наткнуться» читатель, который в силу редчайшего стечения обстоятельств оказался уже подготовлен собственной биографией к свершению «таинства зачатия» новой идеи именно в данной профессиональной области знаний. Иными словами, только в том до невероятного редком случае, когда у автора книги и одного из немногочисленных читателей дорогого рукописного фолианта оказывался «резонанс» конструктивных идей, книга могла способствовать акту рождения нового знания. Понятно, какое воздействие на темпы развития технологической цивилизации должно было в этих условиях оказать изобретение печатного станка — машины для тиражирования зафиксированных на материальном носителе знаний.

Книгопечатание — первая информационная революция

Книгопечатание выполняло для роста накапливаемых человечеством профессиональных знаний ту же роль, какую играет, например, для растений рассеяние семян. Массовое тиражирование для последующего «рассеяния» на больших пространствах зафиксированной на материальном носителе информации о новых знаниях значительно повышало вероятность события, что хотя бы одно «семя знания попадет на благодатную почву», прозреет и в свою очередь даст «массовым тиражом» обогащенное новым знанием свое собственное «послание в будущее».

Стимулируемое книгопечатанием развитие наук ускоряло темпы накопления систематизированных по отраслям знаний. Эти знания теперь можно было быстро тиражировать и они становились доступными для многих нередко далеко удаленных друг от друга территориально и во времени участников внутриотраслевого трудового процесса. Например, при создании паровой машины основные технические решения были получены врачом Д. Папеном (1690 г.), шихтмейстером Колывано-Вознесенских заводов И. Ползуновым (1763 г.), лаборантом университета в Глазго Дж. Уаттом (1769 г.). «Паровая машина была первым действительно интернациональным изобретением...» — отмечал Энгельс.

За три столетия после изобретения в 1445 г. печатного станка оказалось возможным накопить ту «критическую массу» социально доступных знаний, при которой начался лавинообразный процесс развития промышленной революции.

«Наука — это та часть наших знаний, которую мы сумели понять настолько хорошо, что можем обучить этому ЭВМ. Там, где мы еще не достигли такого уровня понимания, речь пока идет лишь о профессиональном искусстве. Формальная запись алгоритма или программы ЭВМ, по-существу, позволяет нам выполнить весьма полезный тест глубины наших знаний, так как переход от искусства к науке просто означает, что мы поняли, наконец, как автоматизировать данную предметную область».

Д. Кнут

Знания, овеществленные через трудовой процесс в станки, машины, новые технологические процессы и другие организационно-технические новшества, становились источником новых идей и плодотворных научных направлений. *Регенеративный цикл: знания — общественное производство — знания, оказался замкнут, и спираль технологической цивилизации начала раскручиваться с нарастающей скоростью. Печатный станок сыграл при запуске этого процесса роль информационного ключа, резко повысив пропускную способность социального канала обмена знаниями.*

Математика — базовая методология индустрии знаний

С развитием промышленной революции становилась все более острой потребность в создании регулярно функционирующего научного аппарата по основным областям профессиональных знаний, чтобы процесс их отчуждения от автора все в меньшей степени оставался внутрицеховым «таинством», а приобретал со временем все более характерные черты одного из рутинных этапов типового производственного процесса. В это время начали закладываться те фундаментальные понятия и основные элементы технологии формализации профессиональных знаний, которые сегодня иногда объединяют более общим понятием «индустрия знаний».

Истоки этого процесса восходят, видимо, к тем незапамятным временам, когда жрецы — первые профессиональные хранители общинных сокровищниц знаний, начали постепенно отказываться от претензии на контроль над всей необъятной «магией» и переходить к индивидуальной специализации по областям «преимущественных интересов». Так возникли первые «специалисты»: астрономы — «звездочеты», психотерапевты — «заклинатели болезней» и т. д. Вместе с ростом социально-экономической потребности в более конкретном предметном знании из этих тысячелетиями изолированно развивавшихся «секторов» когда-то единой «магии» начали формироваться исторически «донаучные», а затем и некоторые современные научные и «околонаучные» дисциплины.

Например, жрецы-«звездочеты» достаточно высокой квалификации были у многих народов земли еще тысячелетия назад. Однако, с наступлением эпохи мореплавания «дефицит» специалистов этого профиля стал в буквальном смысле «катастрофически» очевиден. Острая социально-экономическая потребность вынудила сосредоточить на решении именно этой задачи «лучшие научные силы» средневековья и скоро стали заметны первые успехи в отчуждении «тайн неба» от их «богоизбранных» хранителей.

Успешно развивавшийся процесс формализации астрономических знаний позволил оснастить океанские парусники навигационными инструментами, а книги с астрономическими таблицами, схемами и точными формулами позволяли практически из любого грамотного отрока при необходимом прилежании за несколько лет воспитать корабельного «звездочета».

Успех (или провал) попытки отчуждения профессиональных знаний от их «богоизбранных» носителей до самого последнего времени определялся возможностью (или невозможностью) их формализации математическими методами. Области профессиональных знаний, которые оказались доступны для такого подхода, получили название «точных наук».

Процесс формализации знаний, как правило, сводился к тому, чтобы попытаться из всего многообразия сведений в избранной области человеческой деятельности выделить небольшую, но логически определяющую достаточно многое зону доступного математическим методам «формализуемого ядра». Это позволяло в случае успеха создать формально строгую «локальную систему знаний». В результате значительная часть содержательных сообщений об очередном «приращении» знаний в рамках данной предметной области могла быть исчерпывающе изложена на ограниченном формальными правилами языке, лишенном «недостатков разночтения». При этом оказывалось, что текст мог быть либо понятным и логически однозначно истолкованным для всего профессионального сообщества, либо квалифицирован как неправильный. Это означало, кроме прочего, что процесс передачи знаний от автора печатной работы заинтересованному читателю в рамках созданной «локальной системы знаний» оказывался уже не долгожданным редким событием, лишь на вероятность которого можно было, и то лишь косвенно, влиять тиражом издания, а практически достоверной рабочей процедурой.

Позитивный результат, достигнутый в формализации любой социально значимой области человеческих знаний, создавал эффект, с которым можно было сравнить лишь библейское чудо прозрения. Достаточно упомянуть известный поэтический комментарий к научному подвигу И. Ньютона:

«Был этот мир глубокой тьмой
окутан.
Да будет свет!
И вот явился Ньютон»...

Понятно поэтому, что основным критерием для отделения науки от искусства, ремесла и других форм «донаучной» деятельности был принят уровень их математизации. *«Учение о природе будет содержать науку в собственном смысле лишь в той мере, в какой может быть применена в ней математика» — отмечал Иммануил Кант.* До самого последнего времени в «учении о природе», казалось, не возникало сколько-нибудь заметных фактов, способных поколебать доверие академической общественности к этому давно уже канонизированному «критерию истинности». Более того, за последние десятилетия, и особенно в первые годы «эры ЭВМ», относительный вес «точных наук» в общей системе научного знания начал, как принято было считать, возрастать еще более быстрыми темпами.

В начале 50-х годов возник «кибернетический бум», суть которого заключалась в стремительно распространявшейся, главным образом, среди представителей «точного знания», эпидемии уверенности в том, что относительно медленное, с их точки зрения, развитие отдельных областей естествознания, например, биологии, вызвано в основном тем обстоятельством, что там работают совершенно дремучие, невежественные в математике специалисты, а образованным представителям «точного знания» до сих пор некогда было этим заняться: «Руки не доходят написать математическую модель клетки (а то и целостного организма) и закрыть, наконец, эту их древнюю канитель с пробирками и капельницами...»

Аналогичным «кавалерийским атакам» подверглись в тот период экономика, психология и многие другие ранее малодоступные для традиционной математики области научной деятельности. Видимо, наиболее точно описал атмосферу «математического клондайка» первого десятилетия компьютерной эры Клод Шеннон. В опубликованной тогда научно-публицистической статье «Бандвагон» он прямо предупреждал своих коллег: «Здание нашего несколько искусственно созданного благополучия слишком легко может рухнуть...»

За три столетия после изобретения в 1445 г. печатного станка оказалось возможным накопить ту «критическую массу» социально доступных знаний, при которой начался лавинообразный процесс развития промышленной революции.

...Печатный станок сыграл при запуске этого процеса роль информационного ключа, резко повысив пропускную способность социального канала обмена знаниями.

Экономика компьютеризации

В чем же заключалась реальная притягательная сила, так сказать, рациональное зерно «кибернетической эйфории» пятидесятых годов, для многих из тех широко известных своими конструктивными результатами выдающихся ученых, которых она, по крайней мере, в самое первое время, вовлекла в свою орбиту.

Видимо, главным образом это было вызвано тем обстоятельством, что *с появлением в середине XX века станков для обработки информации — ЭВМ, впервые в человеческой истории оказался возможным такой способ записи и долговременного хранения ранее формализованных математическими методами профессиональных знаний, при котором эти знания могли непосредственно, без промежуточного воздействия на человека, влиять на режим работы производственного оборудования. Процесс записи ранее формализованных профессиональных знаний в готовой для непосредственного воздействия на машины и механизмы форме получил название — «программирование ЭВМ».*

Можно было ожидать, что появление станков для непосредственного

включения в производственный процесс огромной массы накопленных человечеством профессиональных знаний, вызовет резкий, «взрывной» рост производительности труда. Однако, как известно, за первые 30 лет компьютерной эры этого так и не произошло. В США, например, *огромные и быстро растущие расходы на вычислительную технику сопровождались в период с начала 50-х до конца 70-х годов неуклонным снижением темпов роста производительности труда.* Основная причина наблюдаемого за последние десятилетия снижения темпов роста производительности труда в промышленно-развитых странах — непрерывный отток людей из сферы материального производства в «информационную сферу» народного хозяйства. Рост численности «информационных рабочих» «knowledge workers» вызван постоянным увеличением сложности индустриального общества и, как следствие, объема циркулирующих в нем информационных потоков. Однако, если машины и системы автоматизации в сфере материального производства постоянно совершенствовались и, соответственно, производительность труда там возрастала, то в сферу обработки информации, где трудятся ученые и специалисты, служащие, руководители всех уровней — средства автоматизации проникали до сих пор с большим трудом. Численность людей, занятых в информационной сфере, составляла поэтому к началу 80-х годов в большинстве промышленно развитых стран уже около 50 % от общего числа занятых во всех отраслях народного хозяйства и продолжала быстро расти. Естественно было бы в этих условиях задать очевидный вопрос: чем же объяснить, что за 30 лет эры ЭВМ уровень автоматизации труда людей, работающих в сфере обработки информации, в среднем почти не изменился?

Барьер формализованных знаний

Дело в том, что первые поколения ЭВМ были созданы для решения в основном лишь хорошо поставленных математических задач и, в первую очередь, задач чисто расчетного характера. «Ибо это недостойно рода человеческого, подобно рабам тратить часы на вычисления» — столетиями сокрушались математики. Созданная для решения такого типа задач вычислительная машина стала со временем полезным инструментом и в целом ряде других областей приложений. *В основном ЭВМ применялись там, где необходимо было решать корректно поставленные на формальном уровне задачи.* Как правило, это были наиболее приоритетные народно-хозяйственные и оборонные задачи из достаточно «продвинутых» по уровню накопленного задела математических методов областей «точных наук». Успешными оказались и попытки решения на ЭВМ так называемых информационных задач, существо которых обычно заключается в необходимости автоматизировать процесс хранения и быстрой обработки больших объемов хорошо структурированных данных по стандартным алгоритмам: поиск патентной информации или библиографических ссылок, некоторые банковские операции, продажа авиабилетов, и т. д.

Однако *все попытки в поисках эффективных областей приложений ЭВМ выйти за пределы ранее накопленного задела формализованных задач и ранее сложившегося в хозяйственном механизме фонда структурированных данных наталкивались на значительные, быстро растущие (по мере углубления понимания предметной области) трудности.*

Успех (или провал) попытки отчуждения профессиональных знаний от их «богоизбранных» носителей до самого последнего времени определялся возможностью (или невозможностью) их формализации математическими методами. Области профессиональных знаний, которые оказались доступны для такого подхода, получили название «точных наук».

Как заметил об этом К. Шеннон, *«ЭВМ выглядят как ученые схоласты. При вычислении длинной цепи арифметических операций ЭВМ очень значительно обгоняют человека. Когда же пытаются приспособить ЭВМ для выполнения неарифметических операций, они оказываются неуклюжими и неприспособленными для такой работы».*

Все это вынудило на рубеже 80-х годов несколько более критически оглянуться на пройденный путь. При этом выяснилось, что, как и предсказывал Шеннон, «в здании нашего несколько искусственно созданного благополучия» видны зияющие провалы.

Кроме упомянутого выше примера низведения тысячелетиями охраняемых таинств «жрецов-звездочетов» до уровня широко доступных практическому использованию законов небесной механики, можно было бы привести множество и других столь же убедительных примеров из самых различных областей «точных наук», показывающих, как процесс формализации знаний завершался, в случае успеха, разработкой математически строгой техники отчуждения «внутрицеховых» профессиональных тайн. Однако не менее важно и то, что далеко не все «форты» профессиональных знаний удавалось до сих пор брать «лобовой атакой», используя лишь тот арсенал средств формализации, которым располагала традиционная математика. Например, передача профессиональных знаний в таких важнейших областях современной науки, как медицина, происходит и сегодня во многом теми же средствами, что и тысячу лет назад. Причем происходит это вовсе не от недостатка внимания сторонников «точных методов» к этой древнейшей из наук. Как заметил академик И. М. Гельфанд, подводя итоги 15 лет работы по медицинской диагностике и прогнозированию возглавляемого им коллектива математиков и врачей, *«применение математических методов в медицине, несмотря на относительно длинную историю, все еще находится в начальной стадии. При первых же столкновениях с реальным медицинским материалом стало ясно, что те испытанные общие принципы, с которыми математики подходили к физическим и техническим задачам, в этой новой области плохо применимы. Аналогичное положение дел имеет место, по-видимому, и в других нетрадиционных для применения математики областях».*

Развивая мысль, которую И. М. Гельфанд вкладывает в заключительное обобщение, можно было бы отметить, что медицина далеко не единственный «крепкий орешек» для традиционных методов формализации профессиональных знаний. Дело здесь, видимо, в существенно разном уровне сложности задач, которые возникают при необходимости вычислить баллистическую траекторию, оценить ожидаемую температуру газа, или же механизм функционирования живой клетки, целостного человеческого организма, производственного предприятия, отрасли, и т. д.

«Совершенно естественно,— отмечал в этом контексте Ж. Адамар,— говорить об уме более интуитивном, когда зона комбинирования идей находится глубоко, и об уме логическом, если эта зона расположена достаточно поверхностно».

Структура фонда профессиональных знаний

Чтобы попытаться оценить, как соотносятся между собой по относительному объему различные «слои» накопленных человечеством профессиональных знаний, давайте проделаем в этом зале простой эксперимент. Я буду последовательно задавать вопросы, а вы попытайтесь отвечать на них каждый самому себе. Потом мы сопоставим ответы...

Давайте примем за 100 % объем ваших знаний в той области, где вы считаете себя наиболее сильным в профессиональном отношении. А теперь попытайтесь оценить, какую часть из полного объема ваших про-

фессиональных знаний вы могли бы передать ближайшему коллеге по работе, т. е. человеку, который понимает вас в рабочем контакте лучше других? Моя оценка: то, что вы смогли бы объяснить ближайшему коллеге, пользуясь любыми доступными вам средствами общения (речь, мимика, жесты, чертежи и т. д.), не превышает 20...10 % общего объема ваших знаний. Теперь предположим, что вы можете общаться с коллегой лишь письменно. Я полагаю, что в этом случае вы сможете передать, даже пользуясь всем богатством выразительных возможностей естественно-языковых текстов, уже не более 1 % объема ваших знаний. Наконец, предположим, что из всех возможных способов передачи знаний для вас оставлен только язык формальных описаний: математические формулы, известные языки программирования и т. д. В этом случае, я полагаю, основная часть здесь присутствующих смогли бы передать еще меньшую часть своих знаний, видимо, еще на несколько порядков меньше...

Вся область профессиональной человеческой деятельности, которая принципиально поддается пока формализации, а, значит, и автоматизации на базе ЭВМ,— это, образно говоря, тонкая поверхностная пленка формализованных знаний, лишь слегка прикрывающая поверхность океана накопленного человечеством неформального знания. Именно эта «пленка» и оставалась до самого последнего времени доступной областью для приложения машинных методов решения интеллектуальных задач. Отношение толщины этой «пленки», характеризующей доступную известным методам формализации часть человеческих знаний, к общей глубине профессиональных знаний, которыми оперируют в повседневной деятельности работники, труд которых предполагается автоматизировать, и является сегодня показателем потенциально достижимой эффективности внедрения ЭВМ. Попытки не замечать этих ограничений, или просто игнорировать их волевым образом нередко приводили к значительным организационно-экономическим просчетам, крупномасштабным потерям, как это, например, происходило в середине 70-х годов с печально памятными АСУ.

Иными словами, общая структура фонда накопленных человечеством профессиональных знаний может быть представлена в виде быстро сужающейся по высоте пирамиды. В основании этой «пирамиды знаний» лежит самый значительный по общему объему слой, который до самого последнего времени был практически недосягаем для какого бы то ни было «внешнего доступа». Элементы этого слоя — индивидуально накапливаемые «мастерами» знания и навыки, принципиально неотчуждаемые от

...с появлением в середине XX в. станков для обработки информации — ЭВМ, впервые в человеческой истории оказался возможным такой способ записи и долговременного хранения ранее формализованных математическими методами профессиональных знаний, при котором эти знания могли непосредственно, без промежуточного воздействия на человека, влиять на режим работы производственного оборудования.

их авторов традиционными методами формализации: «Могу сделать, но не знаю, как это объяснить».

Расположенный выше и, соответственно, значительно меньший по относительному объему «слой» образуют знания, которые, хотя и могут быть переданы, но лишь в процессе длительной совместной работы: «Делай, как я!»

Далее по высоте «пирамиды», а значит, и по порядку уменьшения относительного объема лежит слой знаний, которые доступны для передачи в рамках традиционной педагогической процедуры: «могу попытаться объяснить, например, в рамках семестрового курса лекций и месячного лабораторного практикума, но не уверен, что все это можно формально описать».

И, наконец, едва различимая по относительному объему на фоне ниже лежащих слоев знаний, «верхушка пирамиды» — формализованные знания.

До сих пор объектом автоматизации на базе ЭВМ мог быть лишь самый верхний, исчезающе малый слой задействованных в реальном производственном процессе профессиональных знаний. В этом, видимо, и заключается основная причина относительно слабого влияния, которое успели пока оказать ЭВМ на макроэкономические показатели динамики хозяйственного механизма индустриально-развитых стран. Есть основания предполагать, что позитивные сдвиги в экономической эффективности внедрения ЭВМ в народное хозяйство, которые на макроэкономическом уровне станут заметны, как ожидают, к началу 90-х годов, будут связаны с феноменом «персональных вычислений» (personal computing).

Процесс записи ранее формализованных профессиональных знаний в готовой для непосредственного воздействия на машины и механизмы форме получил название — «программирование ЭВМ».

Персональные вычисления

«Персональные вычисления» — это предоставленная миллионам людей возможность работать без посредников «один на один» с инструментом автоматизированной обработки информации. Отдельные «еретические эпизоды» в работе «непрограммирующих профессионалов» за пультом персонального компьютера, завершаемые созданием реально полезной прикладной программы, в тех областях приложений, где задел формализованных знаний ранее полностью отсутствовал, становились, по мере массового распространения ПЭВМ, все более частым явлением, а в последнее время начали обретать контуры массового производственного процесса. Первый самый мощный слой профессиональных знаний оказывается при этом «уязвим» для проникновения в него «человека с компьютером». Разумеется, «чуда» не происходит, и специалист, как и раньше, может «проникнуть» лишь в свою собственную «персональную зону» этого слоя, но ... выходит он из этой «зоны» нередко с работающей программой.

По общему характеру процесса разработки такая программа мало чем отличается от других «изделий», создаваемых в тех областях производственной деятельности, где уровень мастерства исполнителя слабо зависит от уровня владения им формальным аппаратом. По мнению итальянского ученого Дж. Атарди, процесс работы «непрограммирующего профессионала» за пультом ПЭВМ, характеризуется в первую очередь «приоритетом действия над планом». Как отмечал А. П. Ершов, это оказывается сегодня одним из наиболее характерных отличий «программирования для себя», которым заняты миллионы пользователей ПЭВМ от стиля работы значительной части профессиональных программистов, выполняющих, как правило, работу «на заказ». Регулярная производственная деятельность профессионального программиста по основной сути своей предполагает многоуровневое планирование и, соответственно, строго формализуемые условия «выполнения плана».

С другой стороны, объяснить (даже самому себе), как развивался за пультом ПЭВМ рабочий процесс, который через несколько десятков прошедших отладку и отвергнутых версий привел к единственному удовлетворяющему противоречивым производственным критериям варианту программы, «непрограммирующий профессионал», сознавая безнадежность ситуации, в большинстве случаев и не пытается. Принципиально новое качество такого «изделия» — программы ЭВМ перед любым дру-

гим сработанным аналогичным способом изделием заключается в том, что «динамика» функционирования системы или комплекса управляемых компьютером производственных агрегатов в этом случае может быть однозначно прочитана в «статике» исходного текста созданной программы. *Каким бы способом программа ни была разработана, если в конечном итоге опыт ее эксплуатации показывает, что она обеспечивает эффективный режим функционирования производственного оборудования, то это означает, что кроме непосредственного улучшения производственных результатов на данном рабочем месте получен и весьма важный «побочный» результат — формализованное описание найденного технического решения.* Понятно, что такого сорта «побочный продукт» может в ряде случаев оказаться сам по себе значительно более ценным, чем непосредственно наблюдаемый на данном рабочем месте производственный эффект внедрения ЭВМ.

Процесс автоформализации знаний и критерии «истинной науки»

В различных академических аудиториях нередко приходится сталкиваться с вопросом: «Зачем вам потребовалось для выделения некоторой совокупности элементов компьютерного творчества вводить новый термин — автоформализация? Это, ведь, кроме всего, еще и логический парадокс! *Как может человек, неважно каким инструментом он пользуется, формально описать то, чего он не в состоянии объяснить даже самому себе?* Наконец, согласитесь, что в ваших аргументах и доводах слишком много элементов, явно «потусторонних» по отношению к истинной науке».

Формулировка этого вопроса иногда меняется, но суть остается удивительно постоянной. Соответственно, и глава «Технология автоформализации профессиональных знаний» оказалась сразу же после выхода книги* предметом критики в значительной степени именно с этих «пуристских» позиций. Коротко ответить здесь на все связанные с такого типа дискуссиями вопросы, разумеется, трудно. Поэтому попытаемся, не останавливаясь на эмоциональных оттенках критерия «научности», рассмотреть лишь отдельные, с нашей точки зрения, наиболее существенные аспекты обозначившегося здесь узла методологических противоречий.

...«применение математических методов в медицине, несмотря на относительно длинную историю, все еще находится в начальной стадии.

При первых же столкновениях с реальным медицинским материалом стало ясно, что те испытанные общие принципы, с которыми математики подходили к физическим и техническим задачам, в этой новой области плохо применимы. Аналогичное положение дел имеет место, по-видимому, и в других нетрадиционных для применения математики областях».

И. М. Гельфанд

Итак, кроме нового «компьютеризованного изделия», обеспечивающего более эффективный режим работы устройств и оборудования, *важнейшим результатом персональных вычислений оказывается зафиксированный на машинном носителе, готовый к тиражированию «формализованный фрагмент» из ранее принципиально недоступного формализации «нижнего слоя» индивидуальных знаний.* Вновь созданная «непрограммирующим профессионалом» программа может затем либо использоваться на одном или нескольких рядом расположенных рабочих местах, или, в зависимости от ее конкретной потребительской ценности, быть использована в качестве документа, специфицирующего условия правильности другой, функционально ей эквивалентной программы, заказанной для исполнения бригаде профессиональных программистов. Например, когда требуется повысить эффективность предназначенной для тиражирования программы по критериям машинных ресурсов, и т. д.

Процесс формализации профессиональных знаний, осуществляемый в режиме персональных вычислений,— исторически новая форма интеллектуальной деятельности. Поэтому нам представлялось целесообразным очертить круг связанных с этим творческим процессом объектов исследования специальным термином — *автоформализация*.

Разумеется, ни ПЭВМ, ни какие-либо другие инструменты, не могут сами по себе резко изменить исторически сложившейся границы «зон определенности» профессиональных знаний. *Если мы говорим о том, что сегодня появляется возможность автоформализации профессиональных знаний средствами ПЭВМ, то должно быть ясно, что при этом существенно трансформируется и само понятие «формализации».* Судя по всему, это сегодня надо постоянно подчеркивать, чтобы избежать нередко возникающих в этом контексте терминологических недоразумений. Вместе с тем, любые попытки затронуть контуры «золотого храма» точных наук — монополии методов традиционной математики на процесс формализации профессиональных знаний, до сих пор воспринимались академически титулованными «жрецами-хранителями» его передовых форпостов как ересь, граничащая со святотатством.

Независимо от конкретного предмета и формы такого рода дискуссий, их концептуальная ось оставалась, как правило, дословно постоянной: «Математика ли это? Наука ли это?»

Мы, разумеется, не имеем здесь возможности сколько-нибудь подробно останавливаться на самых разнообразных иллюстрациях необычайно интересной для истории науки в целом устойчивости мифа о незыблемости «критериев строгой формализации» и их идентичности самому понятию научного метода. Изложение связанных с этим острых научных коллизий, блистательных взлетов и трагических тупиков, практических приложений можно найти в монографии советских математиков И. И. Блехмана, А. Д. Мышкиса, Я. Г. Поновко «Механика и прикладная математика: Логика и особенности приложений математики». Анализ сложившейся ситуации с позиции истории развития математики и естествознания, обширный фактический материал для самостоятельных выводов содержит недавно переведенная на русский язык книга американского математика М. Клайна «Математика. Утрата определенности». Как отмечает автор во вступлении: *«Наши предшественники видели в математике непревзойденный образец строгих рассуждений, свод незыблемых «истин в себе» и истин о законах природы. Главная тема этой книги — рассказ о том, как человек пришел к осознанию ложности подобных представлений и современному пониманию природы и роли математики»*...

Кратко поясним общий смысл, вкладываемый в понятие автоформализация, простым примером. Пусть пульт бортовой ЭВМ установлен на гусеничном вездеходе и создан аппаратно-программный комплекс для автоматического управления таким «самодвижущимся» средством, включающий набор датчиков, воспринимающих окружающую обстановку.

Условия задачи: В центре труднопроходимого болота находится зимовье. Один из местных жителей иногда бывает там во время охоты и возвращается обычно без особых сложностей, так как умеет выбирать трассу сравнительно безопасного движения. По профессии он водитель и при необходимости мог бы провести к зимовью и вездеход.

Вопрос первый. Может ли бригада квалифицированных математиков и программистов, изучив предварительно алгоритм управления заданным транспортным средством, понаблюдать затем в непосредственном общении за водителем во время его поездок на вездеходе к зимовью и соста-

* Громов Г. Р. Национальные информационные ресурсы: проблемы промышленной эксплуатации.— М.: Наука, 1984.— 240 с.

вить программу бортовой ЭВМ для автоматической проводки вездехода по этому же маршруту?

Судя по реакции зала, ответ, видимо, для большинства из присутствующих не оставляет заметных поводов для сомнения. Это невозможно! И дело, конечно же, не в особенностях того или иного конкретного участка труднопроходимой местности. Несмотря на резкий качественный скачок в оснащении, например, судов навигационным оборудованием, а в последнее время и бортовыми ЭВМ, профессия лоцмана, увы, пока еще далеко не является анахронизмом. Более того, отмеченные трудности отнюдь не ограничены лишь «проблемами транспорта».

Один из ярких примеров такого сорта тупиков среди задач профессионального программирования «на заказ» приводит руководитель отдела программирования фирмы ИБМ Дж. Фокс в книге «Программное обеспечение и его разработка»:

«При попытке автоматизировать нефтеочистительные заводы фирмы «Эксон», расположенные в Эдмонте (Канада) и в Антверпене (Бельгия) фирма ИБМ потеряла более 10 млн. долл. Выполняли работу две сотни моих хьюстонских сотрудников. Как-то один из разработчиков спросил инженера компании «Эксон», каким образом он узнает, когда надо включать тот или иной клапан управления потоком в трубопроводе. «Очень просто,— услышал он в ответ.— Я опускаю палец в струю и пробую»... «Запрограммируйте это»!— саркастически заключает Фокс.*

Даже самые взаимно доброжелательные и творчески напряженные беседы с «лоцманом», «инженером-технологом» или любым другим носителем, так называемых, слабоструктурированных профессиональных знаний, мало что могут прояснить программисту в содержательной постановке задачи. Дело, в том, что «квант времени» постижения существа сколько-нибудь нетривиальной из такого типа задач — жизнь... Поэтому на практике обычно возникает простая альтернатива: или повесить на трудноформализуемой прикладной задаче очередную стандартную «бирку» — «недозрела для автоматизации!» или поступить так, как рекомендовал поэт: «вот вам, товарищи, мое стило и можете писать сами». Попытаемся проследить, как могли бы развиваться события в этом последнем варианте.

Предположим, что бригада занятых на «задаче о вездеходе» программистов вместо того, чтобы продолжать попытки алгоритмизовать неуловимые для непосвященных способы оценки «таежной ситуации», разработала необходимые базовые средства: драйверы для управления исполнительными устройствами навигационной системы вездехода, связи с датчиками окружающей обстановки и т. д., например, в рамках одного из популярных языков программирования высокого уровня и пригласила «охотника» за пульт бортовой ЭВМ, чтобы он сам попытался написать программу управления вездеходом, реализующую его собственный (неосознанный пока) «алгоритм проводки» транспортного средства к зимовью.

Предположим также, что «охотник» владеет основами «второй грамотности» и может начать работу за пультом ЭВМ, скажем, в рамках системы Бейсик. Несколько дней или недель для освоения конкретной версии языка с встроенными проблемно-ориентированными функциональными расширениями, и ... начинается процесс создания варианта программы для движения на начальном, простейшем участке трассы, а затем — долгий, изнурительный процесс отладки. Можно, видимо, в самых общих чертах представить себе, как это могло бы происходить.

«Пуск!»— машина продвинулась на несколько метров и провалилась в трясину. Подъем, буксировка вездехода на исходную точку трассы, анализ ситуации. Автор внимательно осматривает местность вокруг гусеничного следа, вплоть до того участка, где эти следы исчезают под водой, долго водит пальцами по листингу программы. «Так ... кажется, ясно. В программу была заложена неполная информация: выбор направления движения с ориентировкой на более сухой мох осуществляется лишь в пределах зоны «талой воды». В случае, когда недавно прошел дождь, следует дополнительно учитывать также и цвет разводья и ориентироваться в движении на более «рыжие» участки, где обычно грунт оказывается плотнее». В текст программы вносятся необходимые изменения, предварительная отладка на машинном макете — «текстокарте» местности, и снова прогрет двигатель, команда «Вперед!» На этот раз машина прошла чуть дальше, и т. д.

Необходимый инструментарий, оценки эффективности, границы применимости...

Как и всякий чисто умозрительный, иллюстративный пример * рассматриваемый случай одинаково уязвим со всех сторон.

«Представить себе, чтобы «человек из тайги» мог на время отложить ружье и позабавиться с пакетом игровых программ — это еще куда ни шло... но, чтобы он написал программу реального времени..?» — резонно усомнится один.

«А не потребуется ли и для этого, как Вы его называете, процесса автоформализации, все тот же «квант времени», длиной в жизнь?»— сочувственно улыбнется другой.

И, наконец, третий решительно «закроет проблему» роковым вопросом: «А Вы уверены, что управляемая такой программой машина после скажем, двух-трех удачных рейсов пройдет трассу еще хотя бы раз? В другое время суток? При другой погоде?...»

Первый вопрос, видимо, адресован в значительной степени профессиональным программистам, которые создают базовые программные средства для инструментального обеспечения процесса автоформализации знаний. В этом контексте уместно еще раз отметить важность дальнейшего развития концепции «объектно-ориентированного программирования», согласно которой еще до первого контакта пользователя с ПЭВМ должен быть создан проблемно-ориентированный базовый инструментарий, который обеспечивает конечному пользователю регулярную возможность самостоятельно программировать достаточно сложные производственные процессы, в том числе и процессы реального времени, совершая лишь самые необходимые и ес-

«Совершенно естественно говорить об уме более интуитивном, когда зона комбинирования идей находится глубоко, и об уме логическом, если эта зона расположена достаточно поверхностно».

Ж. Адамар

* Желая подчеркнуть, что суть дела не в том, достаточно ли квалификации у программиста, чтобы решать ту или иную трудноформализуемую задачу, Фокс особо оговаривает, о ком именно в данном примере идет речь. Упоминаемые им «две сотни хьюстонских сотрудников» — сотрудники отделения фирмы ИБМ, работающие по контракту с управлением космических исследований (НАСА) в научном центре Хьюстона. Это, так называемые, «суперпрограммеры», многие из которых до того, как заняться «задачей о нефтеперегонном заводе», принимали участие в создании наиболее сложных из всех известных за рубежом программистских проектов: программное обеспечение посадки человека на Луну, управление орбитальными космическими лабораториями и др.

* Конкретный пример реализации одного из вариантов технологии автоформализации профессиональных знаний в рамках практически решаемой задачи автоматизации биотехнологических исследований рассматривается в упомянутой выше книге: «Национальные информационные ресурсы: проблемы промышленной эксплуатации», с. 139—143.

тественные, с точки зрения ранее накопленного профессионального опыта, действия «в мире информационных объектов», отображаемых на экране ЭВМ. Процесс такого «интуитивного» программирования приближается к процессу управления развитием сюжета в компьютерных играх *.

Суть *второго вопроса* можно было бы, видимо, дополнительно пояснить известным утверждением, что, в принципе, каждый человек мог бы стать академиком, только одному для этого потребуется тридцать лет, а другому — триста... Разумеется не всякую прикладную задачу и далеко не всякий «непрограммирующий профессионал» сумеет формализовать средствами персональных вычислений. Как заметил один из японских экспертов, «продавать ПЭВМ, говоря при этом, что можно получить с помощью ПЭВМ, это то же самое, что продавать авторучки, утверждая, что с их помощью можно написать нечто на уровне премии Акутагавы или Нобелевской премии».

Специалист в данной предметной области, вооруженный средствами ПЭВМ для формальной регистрации совершаемых им действий, может при определенных условиях зафиксировать в кодах машины какой-то след выполняемого им производственного процесса. Нередко это оказывается возможным и в тех случаях, когда проникнуть каким-либо иным способом извне в его «творческую кухню» оказывается практически невозможным. Однако, нет и не может быть какой-либо «предопределенности свыше» успеха или провала каждой отдельно взятой попытки автоформализации профессиональных знаний. Все, чем мы здесь располагаем сегодня,— это опыт, накопленный миллионами пользователей ЭВМ за последние десятилетия компьютерной эры, который со всей определенностью показывает, что вероятность успеха в решении трудноформализуемых производственных задач оказывается, как правило, значительно выше при движении профессионала в данной предметной области от объекта автоматизации к ЭВМ, чем наоборот — профессионального программиста от ЭВМ к автоматизируемому процессу. Еще в начале 70-х годов, с самых первых попыток организовать промышленное внедрение средств автоматизации на базе микроЭВМ, американские эксперты в области микропроцессорной техники вынуждены были признать: «для нас труднее понять, как работает автомобильная фирма, чем для них — как работают наши микропроцессоры».

И, наконец, *третий вопрос*. Разумеется, нет никаких гарантий в том, что созданная «охотником» навигационная программа обеспечит в какой-либо очередной раз благополучную проводку вездехода, сколько бы не состоялось предварительно успешных рейсов! Более того, именно поэтому мы и выбрали для иллюстрации столь экзотический пример, чтобы более ясно выделить *границы рационального использования методов автоформализации знаний.*

Дело в данном случае даже не в том, что любые сколько-нибудь сложные программы в новых областях приложений, как правило, начинают «спотыкаться» ... Суть вопроса в данном случае в другом. Ведь и в метро со всей его максимально достижимой стабильностью трассы и постоянством метеоусловий никто, как известно, не решился еще снять с поезда машиниста... Автоматика ему только помогает. Понятно, поэтому, что *областью приложений методов автоформализации профессиональных знаний должна быть лишь так называемая безкризисная зона производственной деятельности человека.* Что это такое? Поясним на простом примере.

Пусть создана база данных для машинного хранения и оперативного поиска документов, которые ранее хранились в рабочих столах исполнителей. Предположим, что обращение пользователя ПЭВМ за необходимым документом происходит с помощью популярной сегодня «метафоры рабочего стола», т. е. пользователь видит на экране ЭВМ условное изображение разного типа «ящиков», из которых он может, управляя курсором, «извлекать» изображения «канцелярских папок» и т. д. вплоть до нужного ему документа. Методами «объектно-ориентированного программирования» пользователь может запрограммировать различные алгоритмы поиска нужной ему информации, в том числе и весьма близко приближающиеся к тем «эвристикам», которые он использовал в «домашинную эру».

Многим знакома просьба сотрудника, который должен на какое-то время оставить свое рабочее место: «Пожалуйста, не трогайте моих бумаг, а то я потом ничего не найду!» Как же находит обычно человек в хаосе доверху забитых самыми разными бумагами ящиков своего письменного стола нужный ему документ? Мало кто, видимо, пытался всерьез задавать себе такие вопросы. Но, если тот же сотрудник, вооруженный ПЭВМ, смог по итогам длительного процесса автоформализации заложить алгоритм своего поиска в машину, причем так, что созданная им программа в 60 % случаев позволяет немедленно получить нужный документ, значит на данном рабочем месте оказалось возможным заметно повысить производительность труда. Понятно, что в остальных 40 % запросов, когда программа «отказывает», никакой «аварии» не произойдет, а будет выполнен «псевдоручной» поиск. В то же время даже малая вероятность «отказа» программы проводки упомянутого выше вездехода неприем-

Вся область профессиональной человеческой деятельности, которая принципиально поддается пока формализации, а, значит, и автоматизации на базе ЭВМ — это, образно говоря, тонкая поверхностная пленка формализованных знаний, лишь слегка прикрывающая поверхность океана накопленного человечеством неформального знания.

лема из-за «кризисной ситуации», которой чреват такой отказ.

Если доля ручных операций на каком-либо рабочем месте в результате автоформализации профессиональных знаний исполнителя снизилась, скажем на 30...40 %, то это в большинстве случаев уже вполне заметный прирост производительности труда, который, как правило, полностью окупает затраты на установку ПЭВМ. Тот факт, что в остальных 60...70 % случаев программа «пасует» и оператор вынужден, как и раньше, брать управление на себя, означает лишь, что пока не удалось на данном рабочем месте обеспечить учет большего числа факторов сложного производственного процесса. Традиционная альтернатива «все или ничего» в оценке работоспособности прикладной программы в данном случае не уместна, так как отношение числа успешных срабатываний к «отказам» оказывается лишь текущим показателем достигнутого выигрыша в росте производительности труда. Принципиально иная ситуация складывается, когда «отказ» связан с возможностью аварийного исхода. «Кризисная область» приложений ЭВМ и должна быть поэтому основным объектом так называемого «доказательного» программирования.

Итак, для четкого размежевания областей приложений математически безупречного «доказательного» программирования, выполняемого исключительно профессиональными программистами, и процесса автоформализации знаний, в который постоянно вовлекается все большее число «парапрограммистов» — самого массового контингента пользователей ПЭВМ, видимо, достаточно вспомнить три известных закона робототехники, сформулированные А. Азимовым: 1) робот ни при каких обстоятельствах не должен нанести вред человеку; 2) робот не должен наносить вред себе или другим роботам в тех случаях, когда это не противоречит первому закону робототехники;

* Подробнее с этим подходом можно познакомиться, например, в статьях А. П. Ершова и Г. В. Лебедева в последних номерах «МП».

3) робот выполняет любое указание человека в тех случаях, когда это не противоречит первым двум законам.

Понятно, что эксплуатация программ, полученных методами автоформализации профессиональных знаний, допустима лишь в рамках «третьего закона робототехники». Обеспечить безусловное выполнение «первого закона» и, по возможности, снизить риск нарушения «второго» — это профессионально выполняемая с максимально достижимым уровнем строгости задача «программирования на заказ».

Вернемся к «транспортной задаче». Пусть в цехе с многоуровневой системой автоматизации производственных процессов, необходимо запрограммировать траекторию движения транспортного робота так, чтобы в условиях частой смены маршрутов движения и одновременной работы большого числа таких роботов время простоя станков в ожидании доставляемых заготовок было бы минимальным. Предположим также, что до введения системы «сплошной автоматизации» в цехе работали электрокары с опытными водителями, которые исключали простой оборудования.

Иными словами, мы снова возвращаемся к задаче об «охотнике»... Только начать теперь надо будет с проверки выполнения первых двух «законов Азимова». Транспортный робот должен быть предварительно обеспечен аппаратно-программными средствами «нижнего уровня», которые полностью исключают возможность совершить наезд на человека, на какое-либо цеховое оборудование, столкновение роботов и т. д. В рамках тех возможностей управления движением, которые остаются после выполнения «условий безопасности», объектно-ориентированная программная система предоставляет водителю электрокара необходимые средства, чтобы экспериментировать с прокладкой маршрута, например, в режиме компьютерной игры.

Пусть в первом варианте такой программы робот успешно проходит маршрут лишь в 40 % случаев. Это значит, что в первом приближении 70 % ранее существовавшей численности водителей смогут теперь обеспечить тот же поток деталей, которые они раньше доставляли всей бригадой (предполагая, что 10 % будут теперь заняты лишь тем, чтобы находить и возвращать в точку старта «заблудившихся роботов»). Когда программа будет дополнительно усовершенствована и вероятность того, что робот не сойдет с оптимального маршрута, достигнет, скажем, 60 %, это будет означать, что может быть высвобождена почти половина водителей и т. д.

Последний пример нам потребовался, чтобы еще раз подчеркнуть: разработанная в режиме персональных вычислений программа должна оцениваться по совершенно иным критериям, чем профессионально создаваемый на заказ «программный продукт», потребительская ценность которого вообще не может обсуждаться до получения «сертификата» о безошибочном прохождении комплекса узаконенных тестов. С другой стороны — продукт автоформализации профессиональных знаний часто оказывается *локально полезным* еще задолго до окончательной доводки соответствующей программы. Более того, для многих из такого типа программ сама по себе постановка задачи на достижение близкого к 100 % уровня надежности может оказаться экономически просто бессмысленной, в то время как программа, которая надежно «срабатывает» даже менее, чем в половине практически интересных случаев, может дать весьма ощутимый прирост производительности труда на автоматизируемом рабочем месте.

Разные задачи, разные технологии, разные области приложений и, как следствие, принципиально различные критерии качества процесса формализации профессиональных знаний.

Понятие формализации наполняется, таким образом, новым «инструментальным» смыслом и далеко выходит за те жесткие дедуктивно-логические рамки, в которых оно сформировалось за первые 25 веков развития доминирующей сегодня «античной ветви» науки математики.

Успех или неудача акта формализации знаний все более нередко определяется не уровнем его логической доказательности, а принципиально иными «прагматическими» критериями (например, как показано выше, это могут быть... экономические критерии). Иными словами, с расширением областей приложений ЭВМ типичными становятся ситуации, когда традиционно неразрывная со времен «программы математизации знаний Пифагора из Самоса» концептуальная связь двух фундаментальных понятий: **формализация** и **логическая доказательность** — не сохраняется. Именно это обстоятельство и имелось в виду, когда мы отмечали выше, что с массовым переходом к технологии автоформализации знаний **трансформируется** само понятие — формализация.

Инверсная «триада»

Профессионал в данной предметной области, как правило, знает, как найти верное решение практически в любой конкретной ситуации, которая может возникнуть в ходе реального производственного процесса, однако не знает и, видимо, не может знать, какие именно рабочие ситуации и в какой последовательности сложатся на очередной «трассе решений».

Главное внешнее отличие режима «персональных вычислений» от традиционных методов программирования заключается поэтому в том, что к моменту пуска первого варианта программы ее автор не может быть уверен в том, что знает верный путь решения. Все, чем он располагает,— основанная на профессиональном опыте уверенность, что выход из любой конкретной ситуации будет найден непосредственно в контексте решения («по месту»). В значительной степени с этим обстоятельством и связана принципиальная сложность формализации интуитивно, видимо, существующих решений «слабоструктурированных» задач. Непреодолимая, как правило, трудность на пути формального описания такого типа задач — необходимость учета слишком большого числа факторов. Относительный вес этих факторов меняется от реализации к реализации заранее непредсказуемым образом... Именно это последнее обстоятельство ограничивало до сих пор возможность использования в решении на ЭВМ такого типа прикладных задач стандартной последовательности этапов: математическая модель, алгоритм, программа (типовая «триада» решения).

Моделирование предлагает, как известно, возможность *предварительного* выделения из большого числа факторов, реально влияющих на решение любой практической интересной задачи, относительно небольшого числа наиболее важных. Это и позволяет абстрагироваться при изучении математической модели от сложности всего остального «окружения» реального мира. Понятно, что в задачах, где априори невозможно выполнить разделение большого числа переменных на «существенные» и «несущественные», не может быть использован этот традиционный механизм абстракции. Приходится либо искать какие-то другие подходы к решению прикладной задачи, либо ... объявить такие задачи «донаучными» (выбор удобных здесь эпитетов необычайно богат), т. е. закрыть проблему старым, как мир, и поэтому безупречно надежным приемом — «зелен виноград»...

Альтернативный подход — автоформализация профессиональных знаний — позволяет, при необходимости, вернуться к математическому исследованию автоматизированного средствами персональных вычислений производственного процесса. В ряде случаев оказывается возможным использовать машинную выдачу текста созданной «непрограммирующим профессионалом» программы для реконструкции по ней хотя бы самых общих контуров неосознанно используемого автором алгоритма управления. При этом может быть сделана попытка использовать «восстановленный» алгоритм как базу для создания методами технологии программирования «на заказ» более эффективной по машинным ресурсам програм-

Основные этапы развития технологии знаний (оценка автора)

Знания				Производство			
						Темп производственного роста	
Основные вехи информационной технологии	Время от наших дней (годы)	Общая характеристика этапа технологии	Уровень технологии знаний	Доминирующий тип производственной культуры	Типовой производственный процесс	По валу: рост общего числа единиц выпускаемой продукции	По номенклатуре: рост разнообразия типов выпускаемой продукции
Наскальные изображения Письменность	30—20 тыс. 6 тыс.	**Домашинный этап:** первые инструменты счета из кости и камня, календарь, домеханические (солнечные, водяные и т. д.) часы, компас, бумага, книги	**Ручная** технология регистрации знаний	**Созерцательная:** статически-художественная	**Ремесленное** производство уникальных и мелкосерийных изделий	Медленный	Медленный
Книгопечатание	500	Первая информационная революция					
Телеграф Фото Радио ЭВМ	150 » 90 40	**Машинный этап:** печатный станок, механические часы, телеграф, фото, телефон, фонограф, радио, кино, магнитозапись, телевидение, ЭВМ	**Машинная** технология тиражирования и распространения знаний	**Механистическая:** формально-логическая	**Механизированное** массовое производство стандартизируемых изделий	Быстрый	Медленный
Персональные ЭВМ	10	Вторая информационная революция					
?	?	**Симбиотический** этап: «игровая компонента» ПЭВМ, индивидуальные рабочие станции, локальные сети, системы индивидуального доступа к отраслевым, региональным и мировым информационным ресурсам	**Человеко-машинная** технология автоформализации знаний	**Игровая:** динамически-художественная	**Гибкое** автоматизированное производство уникальных и мелкосерийных изделий	Быстрый	Быстрый *
?	?	Третья информационная революция					

* Примечание: Впервые оказывается, что машина в состоянии помочь человеку в расширении пределов его фантазии, поиске новых богатств разнообразия, а не только в увеличении физических возможностей к многократному повторению чего-то однажды им найденного. «Детонатором» этого процесса стала так называемая игровая компонента ПЭВМ. Подробнее об этом см. «Игровая компонента — первое функциональное отличие персональных ЭВМ»,— журнал «Микропроцессорные средства и системы», 1984, № 1, с. 46—47 или этот же раздел в упомянутой выше книге «Национальные информационные ресурсы:...», с. 124—127.

ние всех этих лет,— вспоминает Эйнштейн,— имелось чувство направления, движения вперед по направлению к чему-то конкретному. Очень трудно, конечно, выразить это чувство в словах, но оно безусловно имелось, и оно должно быть отделено от последующих размышлений о рациональной форме решения. Конечно, позади этого направления всегда имелось нечто логическое, но я имел его в виде обзора в зрительной форме».

Есть, видимо, основания предполагать, что для всех обозримых сегодня «эшелонов» научных исследований, профессионального искусства или интеллектуального ремесла (к последним относят, например, и повседневную практику программирования): от исторически признанных лидеров того или иного этапа развития науки до рядовых специалистов выполняется общая закономерность развития творческого процесса. *В задачах относительно простых для данной предметной области формальная логика «прокладывает» трассу решения, для наиболее сложных — обосновывает пройденный путь.*

Ведущая тенденция, которая сложилась уже в первые десятилетия компьютерной эры в «разделении труда» между двумя «партнерами» по диалогу «человек—ЭВМ», просматривалась достаточно четко: *человек решает наиболее «интуитивно нагруженную» часть производственной задачи, а машина — формально логическую.* Однако трудность в практической организации режима массового использования такого «диалога» заключалась в том, что вся тяжесть задачи предварительного «вычленения» формальной компоненты из всей совокупности профессиональных знаний в данной предметной области ложилась обычно на программиста. Принципиально непреодолимый характер возникающих при этом трудностей в тех наиболее интересных случаях, когда прикладная задача не имела ранее созданного «математического каркаса» решения, выше мы уже обсуждали.

«Феномен ПЭВМ» позволил сделать шаг к преодолению этого противоречия. Оказалось, *что может быть создан инструментарий, который позволит не только исполнять на ЭВМ формально поставленные задачи, но и помогать человеку в свершении «интимного» акта вычленения формализуемой компоненты из его индивидуальных профессиональных знаний. Необходимые для повышения эффективности такого процесса научные методы и технические средства образуют в совокупности «технологию автоформализации профессиональных знаний».*

Еще недавно доступные только «богоизбранным» единицам (да и то далеко не в любой профессиональной области) возможности не только решить прикладную задачу, но и формализовать затем найденную схему решения, становятся сегодня в режиме «персональных вычислений» рутинной для миллионов пользователей ПЭВМ рабочей процедурой. Вряд ли требуют особых комментариев нередко высказываемые в этой связи на страницах массовой печати рассуждения отдельных футурологов из процветающей ныне когорты технократических «экстремистов», которые всерьез обсуждают влияние «века информации» на процесс вызревания новых открытий или темпы рождения современных «платонов, ньютонов, ломоносовых и эйнштейнов». Однако было бы в некотором смысле «симметричным» заблуждением не замечать наблюдаемого сегодня ускорения темпов «дренажа» тонких методов творчества, традиционно связываемых с понятием «интеллектуальной элиты», в область массового «рабочего творчества». «Феномен ПЭВМ», с которым в первую очередь связывают сегодня «демократизацию» научного творчества,— только одна из наиболее заметных вех этого процесса.

ЗАКЛЮЧЕНИЕ

Точка зрения автора на общий характер исторического процесса развития технологии знаний представлена таблицей (см. с. 88).

Главные отличительные черты «второй информационной революции» в наиболее краткой их формулировке, видимо, могут быть сведены к трем основным пунктам:

1. **Взрывной процесс «демократизации» научного творчества** — все более широкое вовлечение в процесс формализации профессиональных знаний миллионов трудящихся самых различных специальностей, образовательного уровня и индивидуальных интересов.

2. **Резкое ускорение технологического цикла развития* ведущих отраслей общественного производства** — активное включение формализованных профессиональных знаний непосредственно в производственный процесс (например, когда эти знания обретают форму прикладных программ, управляющих станками с ЧПУ, технологическими участками, системами автоматизации научных экспериментов, производственных испытаний, обработки текстов и т. д.), минуя необходимую при «книжном тиражировании» стадию опосредованного воздействия на человека-исполнителя.

3. **Массовое тиражирование накапливаемых профессиональных знаний** в масштабах, сопоставимых с теми, которые ранее обеспечивал только печатный станок; однако, в отличие от эры книгопечатания, хранимых в **готовой для автоматизированного поиска «машинной форме»**, что создает необходимые технические предпосылки для постоянного расширения сферы практического использования индустриальных методов человеко-машинного производства новых элементов профессионального знания.

Отметим в заключение, что ПЭВМ, как и любой инструмент, лишь создает определенные технические предпосылки для повышения эффективности творческого процесса, но не в состоянии его вызвать.

Это уже наша с вами задача — воспитать поколение людей, для которых работа за пультом ПЭВМ в режиме автоформализации профессиональных знаний будет одной из наиболее естественных форм творческого самовыражения и, создать необходимые социально-экономические условия для постоянного стимулирования потребности в таком самовыражении (понятно, что именно последнее условие является и наиболее важным и наиболее трудновыполнимым).

В 1984 г. Д. Кнут, выступая с «Тьюринговской лекцией», предложил свой вариант определения границы между наукой и профессиональным искусством:

«Наука — это та часть наших знаний, которую мы сумели понять настолько хорошо, что можем обучить этому ЭВМ. Там, где мы еще не достигли такого уровня понимания, речь пока идет лишь о профессиональном искусстве. Формальная запись алгоритма или программы ЭВМ, по-существу, позволяет нам выполнить весьма полезный тест глубины наших знаний, так как переход от искусства к науке просто означает, что мы поняли, наконец, как автоматизировать данную предметную область».

Методы автоформализации оказываются основным средством включения в активный «производственный» фонд научного знания новых и все более мощных пластов профессионального искусства. Научная разработка и широкое практическое использование этих методов — важнейший фактор ускорения процесса массовой компьютеризации народного хозяйства.

* * *

КРАТКО ОБ ОТВЕТАХ НА ВОПРОСЫ

Все поступившие за время доклада записки с вопросами можно было бы, видимо, условно разделить на три заметно неравные группы.

Первая и самая большая группа — это записки с конкретными вопросами к тому или иному разделу доклада. Ответы на них, как правило, непосредственно встраивались в «ткань изложения» по мере их поступления, что, возможно, приводило к некото-

* Имеется в виду упомянутый выше регенеративный цикл: знание — производство — знание (см. с. 80).

мы, функционально эквивалентной исходному «макету». Такая потребность может возникнуть, скажем, когда оказывается экономически целесообразным начать массовое тиражирование найденного в режиме «персональных вычислений» технологического решения.

Наконец, в случаях, когда ставится задача объединить между собой отдельные «островки автоматизации», например, отдельные технологические участки, в автоматизированный цех, может потребоваться для разработки алгоритма управления следующего уровня поставить задачу еще более углубленного исследования ранее созданной программы — т. е. завершить «движение» по трем уровням абстракции исходной задачи: модель — алгоритм — программа — в направлении «снизу-вверх».

Итак, с нашей точки зрения, типовая «триада» не утрачивает концептуальной целостности независимо от того, используем ли мы, в зависимости от характера решаемой задачи, технологию «нисходящего» или «восходящего» проектирования. Там, где это возможно, разработчик программного комплекса идет в рамках традиционной технологии «сверху-вниз» от модели к программе. В случае когда этот путь оказывается неприемлем, используется «инверсная триада» и проектирование комплекса ведут путем его последовательного усложнения от частных решений к интегрированной системе «снизу-вверх». Понятно, что конкретный путь решения целиком и полностью диктует предметная область, а не тот или иной «догмат академической веры».

Проектирование систем автоматизации производства исключительно однонаправленно — «сверху-вниз» — было наиболее характерно для печально памятного процесса всеобщей «АСУ-низации». К началу 80-х годов эта волна прошла *, оставив практически во всех промышленно развитых странах многомиллионные убытки и стойкое отвращение разработчиков и заказчиков ко всякого рода «догматам» в компьютерной науке и технологии. Весьма дорогой ценой, но все-таки удалось при этом выяснить, что *избежать «реакции отторжения» сложившегося организма давно работающей организации (учреждения, цеха и т. д.) в ответ на внедрение в него извне «чужеродного тела» — функционирующей по строго формальным законам системы автоматизации, можно лишь при поэтапном «врастании» ее отдельных подсистем в «живое тело» организации.*

Общий смысл нового подхода: «дорожки сначала протаптывают, а потом уже асфальтируют...» В начале 80-х годов вместе с ПЭВМ появилась и концепция «архипелага автоматизации». Смысл ее американские эксперты сформулировали просто: use now, integrated later (*используйте* ЭВМ сразу, *объединяйтесь* в сети, многоуровневые структуры — потом...). *Сначала ЭВМ должна стать неотъемлемой, органичной частью действующей структуры организации, должны возникнуть и доказать свою локальную полезность «островки автоматизации» на базе компьютерной технологии и только затем уже можно будет поэтапно ставить вопрос о наиболее рациональных способах их объединения в иерархию формально взаимодействующих систем.*

Секрет особой «живучести» эволюционно усложняющихся систем, по сравнению с «циркулярно насаждаемыми» АСУ, заключается, кроме прочего, и в том, что в их структуру и алгоритм функционирования оказываются «запаяны» методами автоформализации многолетний опыт и профессиональные знания сотрудников о наиболее рациональных (а в некоторых случаях и уникальных) для данной организации способах формирования информационных потоков и управляющих воздействий.

В большинстве практически интересных случаев предварительное изучение предметной области позволяет оценить достижимый уровень определенности для совокупности внешних факторов, влияющих на условия конкретной задачи. Это создает необходимые предпосылки, чтобы на одном из первых этапов проектирования выбрать соответствующий уровень требований к внутренней логической строгости решения. Тем не менее, до самого последнего времени принято было считать, что независимо от степени исходной определенности условий, процесс решения задачи должен отвечать традиционным со времен «античной математики» требованиям логической точности выводов.

Давно уже считается общепринятым, что уровень точности числовых выкладок должен соответствовать реально достижимой в конкретной задаче точности регистрации исходных данных. В то же время попытки распространить тот же, по-существу, принцип «равнопрочности» основных этапов процесса преобразования исходной информации на уровень логической строгости ее решения воспринимаются, в лучшем случае, как нарушение неписаных правил хорошего математического тона, а чаще, как проявление логической небрежности, поощрение математической «полуграмотности» и т. д. Было бы уместным, видимо, еще раз напомнить здесь известное предупреждение Ф. Энгельса о том, что «если захочешь добиться математической достоверности в вещах, не допускающих этого, нельзя не впасть в нелепость или в варварство».

Если мы ясно сознаем, что не в состоянии учесть на требуемом уровне логической строгости **внешние** факторы, влияющие на условие задачи, то в какой степени должна нас заботить логическая безупречность **внутренней** схемы ее решения? Рациональным было бы, видимо, полагать, что вся трасса решения задачи от условия до результата должна быть «логически равнопрочной». Иными словами, чем больше внешней неопределенности в условиях, тем менее обоснованными оказываются строгие требования к внутренней логической доказательности формализуемого решения. И наоборот, для класса задач с гарантируемой степенью логической определенности условий требования «доказательного», логически безупречного вывода алгоритма ее машинного решения оказываются рационально обоснованными, естественно вытекающими из существа «постановки задачи».

Интуиция и формальная логика: дуализм творческого процесса

Следует особо отметить, что задача изучения характера взаимодействия интуитивной и логической компонент в творческом процессе (главным образом, в рамках «таинства» научного творчества) привлекала внимание исследователей еще задолго до того, как с появлением «феномена ПЭВМ» эта проблема оказалась в ряду наиболее актуальных, определяющих методологические основы массовой компьютеризации.

В 1969 г. советский психолог М. Г. Ярошевский высказал предположение о функциональном дуализме этих двух традиционно противопоставляемых компонент. По его мнению, *«имело бы смысл применить нечто сходное с «принципом дополнительности» в физике к трактовке отношений между формализуемыми (объективно отчуждаемыми от субъекта) и неформализуемыми, интимно-личностными, неотчуждаемыми от субъекта, компонентами творчества. Не утратило ли бы тогда свою антитеничность противопоставление интуитивного акта, относимого к сфере психологии, формализованной операции, относимой к логике, пронизывающее на протяжении веков учение о творчестве?»*

В качестве примера, иллюстрирующего характер взаимодействия этих компонент творческого процесса при решении «вершинных» задач человеческого интеллекта, М. Г. Ярошевский ссылается на фрагмент беседы Эйнштейна с одним из психологов, занятых «картографированием» трассы великих научных открытий: *«В тече-*

---

* Наследием этого «стихийного бедствия» остаются многочисленные вывески разного типа «АСУ-контор», которыми все еще оклеены фронтоны и внутренние дворы зданий центральной части многих наших городов.

рым повторам. Разумеется, это не исключает, а скорее предполагает, что в кулуарах семинара мы продолжим работу со всеми, у кого остались такого типа неясности, как и принято на всех наших встречах.

Вторая группа — это общие, риторические по стилю дискуссионные вопросы, посвященные некоторым философским, методологическим и педагогическим аспектам формирования критериев оценки профессионального уровня в той или иной предметной области, а также различного рода «метрикам» для соответствующего ранжирования профессиональных знаний, в том числе и по уже упомянутой в докладе шкале «относительной научности». Трудно предположить, что за одно заседание мы могли бы успеть здесь, хотя бы поставить все эти вопросы на обсуждение. Некоторые из них, имеющие ясно выраженную прагматическую основу, могут быть обсуждены на семинаре: «Технология автоформализации профессиональных знаний», который работает в Научном центре Пущино; другие, возможно, потребуют внимания более специальной аудитории.

В наиболее общей форме ответ на многие из вопросов этой группы, как нам представляется, содержится в замечании А. С. Макаренко:

«Вот, что я вам скажу. Не приходило ли вам в голову, что высшая профессиональность состоит именно в том, чтобы уметь без вреда для дела стать иногда выше этого самого профессионального к нему отношения? И, будучи во всеоружии знания и понимания в своей специальности, суметь взглянуть на вещи прямо и просто, вооружившись одним только здравым смыслом? Вы не согласны?»

. . . . . .

И, наконец, третья группа — это, на наш взгляд, наиболее «прицельные» из дискуссионных вопросов. Остановимся на них более подробно.

Значит ли все выше сказанное, что традиционные методы формализации знаний утрачивают свои позиции?

С нашей точки зрения, как раз наоборот, рост важности систематически используемых строгих методов «полного решения» формально поставленных задач непосредственно следует из ожидаемого быстрого расширения области их приложений. Значительную роль играют при этом методы автоформализации профессиональных знаний, которые, как мы пытались показать, нередко «прокладывают дорогу» для последующего использования более строгих методов в решении той или иной впервые поставленной на ЭВМ прикладной задачи.

Наконец, хотелось бы в этом контексте еще раз подчеркнуть, что по центральной своей сути концепция автоформализации означает не отрицание полезности какого-либо из ранее сложившихся методов формализации, а последовательное расширение области производственных приложений ЭВМ за те рамки, в которых традиционные методы давно и честно работают. По-существу, мы лишь отходим от сложившегося до сих пор «бинарного» принципа отбора прикладных задач: «все — или ничего», — который существенным образом ограничивал масштабы компьютеризации народного хозяйства.

Итак, есть области приложений, где традиционные методы позволяют получить доказательное «полное решение», но есть и такие, где методы традиционной математики пока не работают. Означает ли это последнее обстоятельство, что следует «декретно» приостановить процесс автоматизации в «трудных зонах», объявив их «донаучными», «недозревшими» и т. д.? Как заметил однажды Хевисайд, «мне не мешает испытывать удовольствие от вкусного ужина тот факт, что я не представляю себе точной картины процесса пищеварения.».

Иными словами, мы не обсуждаем полезность строительства автострад, а лишь обращаем внимание на то обстоятельство, что, несмотря на огромный и постоянно растущий грузооборот дорог с «твердым покрытием», их суммарная площадь составляет все еще лишь ничтожную часть земной поверхности. Поэтому, видимо, всегда, на любом уровне развития «дорожного строительства» будет оставаться актуальным вопрос, как быть «там, где кончается асфальт». Можно, конечно, ответить, оставаясь в рамках достаточно известной «пуристской» позиции: «Надо подождать! Если объект, о котором Вы беспокоитесь, действительно настолько важен, то в плановом порядке туда в свое время проложат асфальт...» Однако, известен и другой подход, согласно которому тропинки сначала протаптывают, а потом асфальтируют.

Тем, кто сегодня протаптывает тропинки на месте будущих автострад большой науки, и был адресован этот доклад.

МИКРОПРОЦЕССОРЫ В ПОЛИТЕХНИЧЕСКОМ

В 1986/87 учебном году журнал проведет в Политехническом музее два учебных цикла:

МикроЭВМ: основы применения

Цикл ориентирован на руководителей предприятий, профессорско-преподавательский состав вузов, школ, техникумов, а также на студентов и широкий круг специалистов различных отраслей народного хозяйства.

1. Персональные компьютеры: серийные изделия отечественной промышленности — 9.IX 1986 г.
2. Школьная информатика: второй звонок — 14.X 1986 г.
3. Компьютер в вузе: качественно новый учебный процесс—11.XI 1986 г.
4. Автоматизация учреждений (совместный семинар журналов «ЭКО» и «Микропроцессорные средства и системы») — 9.XII 1986 г.
5. МикроЭВМ в системах автоматизации проектирования и производства — 13.I 1987 г.
6. Компьютерные игры—10.II 1987 г.
7. Диалог человек — ЭВМ: вопросы психологии — 10.III 1987 г.
8. Микропроцессоры в массовых изделиях бытовой электроники — 14.IV 1987 г.
9. Компьютер и медицина — 12.V 1987 г.
10. Компьютер и музыкальное творчество — 9.VI 1987 г.

Микропроцессоры — рабочий инструмент инженера

Цикл ориентирован на разработчиков микропроцессорной техники, а также инженеров других специальностей, которым необходимо работать с новой информационной техникой.

1. Обзор основных классов отечественных микропроцессоров и микроЭВМ — 23.IX 1986 г.
2. Аппаратная структура и программное обеспечение типового микропроцессорного комплекса — 28.X 1986 г.
3. Особенности схемотехнической реализации и методы отладки микропроцессорных систем управления — 25.XI 1986 г.
4. Устройства связи микроЭВМ с объектами автоматизации — 23.XII 1986 г.
5. Периферийное оборудование микроЭВМ: дисплеи, гибкие диски, электронный «квазидиск», ЦМД-память, клавиатура, печатающие устройства — 27.I 1987 г.
6. Аппаратно-программная реализация многомашинных комплексов и локальных вычислительных сетей — 24.II 1987 г.
7. Операционные системы микроЭВМ — 24.III 1987 г.
8. Программируемые логические матрицы: основные классы, технические характеристики, типовые примеры применения — 28.IV 1987 г.
9. Техника работы с ПЗУ: физические основы и методы программирования, схемотехника универсальных и модульных программаторов — 26.V 1987 г.
10. Тенденции развития индустрии ЭВМ — 23.VI 1987 г.

Справки о семинарах можно получить по телефону 923-00-19 у методиста Политехнического музея Ермолаевой Татьяны Юрьевны. Абонементы можно приобрести в кассе Политехнического музея (подъезд № 9).

МИКРОСХЕМЫ СТАТИЧЕСКОГО ОЗУ КР132РУ6

Интегральные микросхемы КР132РУ6А и КР132РУ6Б представляют собой оперативное запоминающее устройство с произвольной выборкой статического типа, выполненное по n-канальной МДП-технологии с кремниевыми затворами и высокоомными нагрузочными резисторами. Информационная емкость ОЗУ 16 Кбит. ОЗУ совместимо по входным и выходным сигналам со схемами ТТЛ. Для питания используется один источник с напряжением 5 В ±10 %. Микросхема выпускается в пластмассовом 20-выводном корпусе (рис. 1) типа 2140.Ю.20-3. Размеры корпуса 25,0×7,5 мм. Назначение выводов показано в табл. 1.

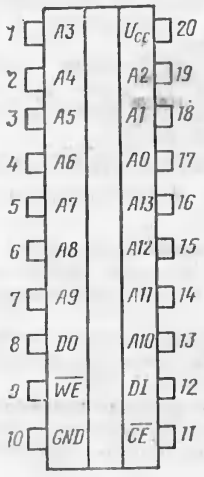


Рис. 1. Условное графическое изображение микросхемы КР132РУ6

Таблица 1

Назначение выводов

Вывод	Назначение
1 ... 4	Адресные входы строки A3 ... A6
5 ... 7	Адресные входы столбца A7 ... A9
8	Информационный выход D0
9	Вход сигнала записи $\overline{WE}$
10	Общий вывод GND
11	Вход сигнала «выбор микросхемы» $\overline{CE}$
12	Информационный вход D1
13 ... 16	Адресные входы столбца A10 ... A13
17 ... 19	Адресные входы A0 ... A2
20	Напряжение источника питания U_{cc}

Кристалл ОЗУ содержит накопитель в виде единой матрицы 128×128 бит, дешифраторы адреса строки и столбца, усилитель записи-считывания, адресные формирователи строк и столбцов, устройства ввода-вывода, генераторы смещения подложки и блок управления перечисленными узлами (рис. 2). Для выбора одного из 16 384 хранимых слов требуется только 14 адресных шин (A0...A13). При наборе двоичного кода на адресных входах микросхемы можно выбрать любую запоминающую ячейку в накопителе. Все входные сигнальные шины имеют буферные входные усилители и средства защиты от статического электричества (кроме входа $\overline{CE}$).

Выход микросхемы имеет три рабочих состояния (низкого уровня, высокого уровня и высокого выходного импеданса), позволяющих объединить выходы нескольких ОЗУ по схеме проводного ИЛИ по состоянию ключей в зависимости от присутствия команд $\overline{CE}$ и $\overline{WE}$.

При отключении источника питания микросхема переходит в микромощный режим хранения. При этом накопитель питается от напряжения сигнала $\overline{CE}$ (U_{CB}=5 В +10 %). Выходная информация из накопителя ЗУ для хранимой «Лог. 1» соответст-

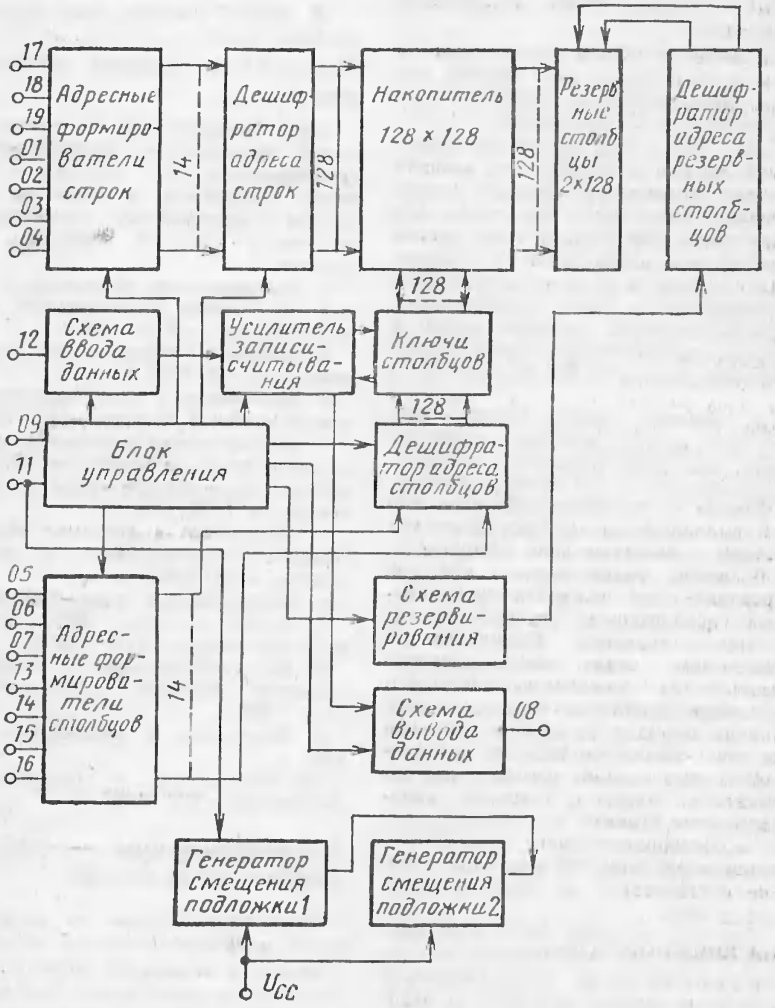


Рис. 2. Электрическая структурная схема

вует выходному напряжению «Лог. 1», а для хранимого «Лог. 0» — выходному напряжению «Лог. 0» на выходе D0 микросхемы.

Микросхемы КР132РУ6А и КР132РУ6Б работают в режимах записи (рис. 3), считывания (без разрушения информации) (рис. 4), считывания-модификации-записи (рис. 5) и хранения — обычном и микромощном. В микромощном режиме потребляемая мощность по сравнению с режимом обращения уменьшается в 10 раз. Временные параметры сигналов приведены в табл. 2.

(В скобках приведены данные для режима считывания-модификации-записи. Уровни отсчета временных параметров: U_{DIL}, U_{CEL}, $U_{WEL/REL}$=0,7 В, U_{DOL}=0,4 В, U_{DIH}, U_{CEH}, $U_{WEH/REH}$=2,2 В, U_{DOH}=2,4 В.)

Фронт сигнала $\overline{CE}$ (переход с высокого уровня на низкий) обеспечивает разрешение на выбор микросхемы в пределах системы, включающей ряд микросхем ОЗУ. Если данная микросхема не выбрана ($U_{\overline{CE}} \geqslant 2{,}4$ В), то она будет оставаться в положении хранения информации (выход закрыт). При этом шины A0...A13, D1, $\overline{WE}$ и D0 отключены от устройств ввода-вывода информации.

Вход $\overline{WE}$ разрешает записывать информацию при прохождении фронта сигнала $\overline{CE}$. При $U_{\overline{WE}} \leqslant 0{,}4$ В и прохождении фронта $\overline{CE}$ с высокого уровня на низкий происходит запись информации, при $U_{\overline{WE}} \geqslant 2{,}4$ В — считывание. Во время считывания или записи информации по входам A0...A13, D1, $\overline{WE}$ по фронту сигнала $\overline{CE}$ защелкивается во входных усилителях, после чего эти входы отключаются. Во время записи шина D0 отключается от устройства вывода информации (высокий выходной импеданс — выход закрыт) (табл. 3).

Для обеспечения максимального быстродействия микросхем длительность фронтов сигналов A0...A13, $\overline{WE}$, $\overline{CE}$ должна быть не более 2 нс. Основные электрические параметры микросхем представлены в табл. 4, предельные значения эксплуатационных параметров показаны в таб. 5.

При измерениях и эксплуатации микросхем должны быть приняты меры, исключающие возможность накопления электростатических зарядов на выводах. Допустимая величина статического потенциала не более 100 В. Микросхемы следует устанавливать на платы с металлизированными отверстиями.

Гарантийная наработка 150 000 ч в пределах гарантийного срока хранения (15 лет).

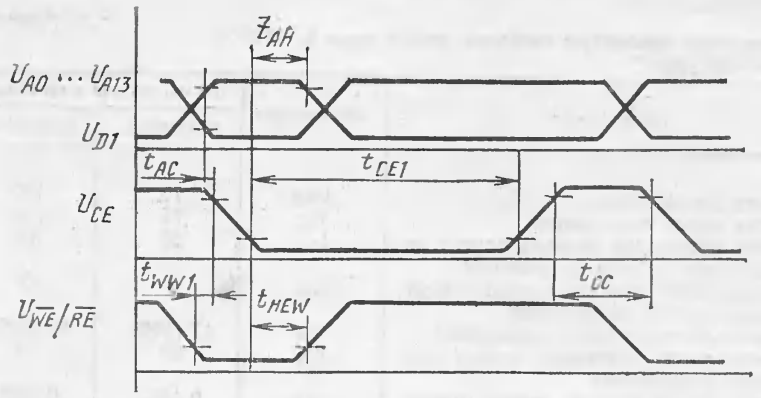


Рис. 3. Режим записи

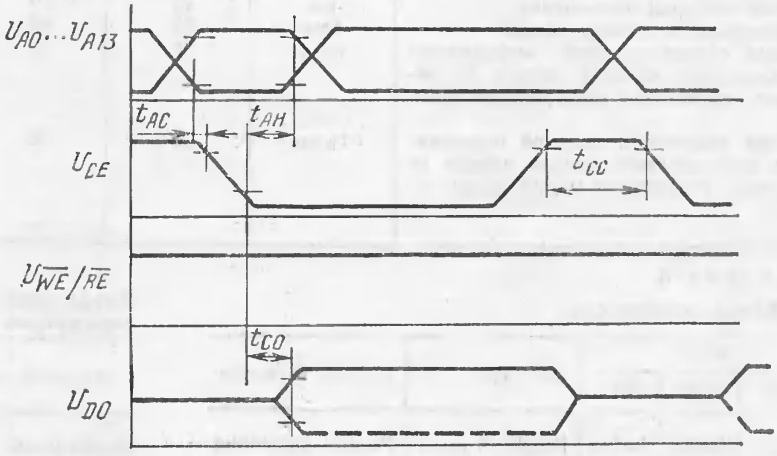


Рис. 4. Режим считывания

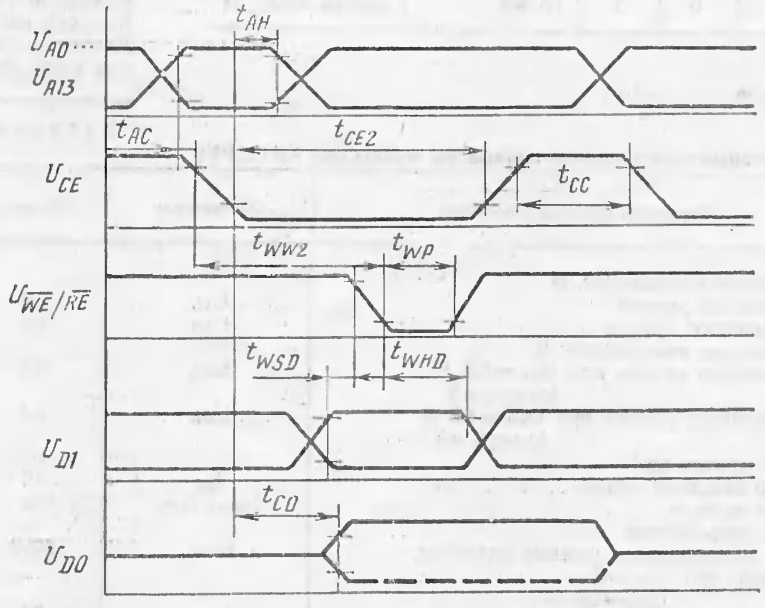


Рис. 5. Режим считывания-модификации-записи

НА КНИЖНОЙ ПОЛКЕ

Баласанян В. Э., Богдюкевич С. В., Шахвердов В. А. Программирование на микроЭВМ «Иск-

Таблица 2

Временные параметры сигналов микросхемы КР132РУ6
$T = -10 \ldots 70$ °С

Наименование	Обозначение	Норма, нс, для микросхемы	
		КР132РР6А	КР132РУ6Б
Время цикла записи	t_{WC}	75	120
Время цикла считывания	t_{RC}	75	120
Время удержания сигнала адреса относительно сигнала разрешения	t_{AH}	25	40
Время сдвига сигнала адреса относительно сигнала разрешения	t_{AC}	0	0
Длительность сигнала разрешения	t_{CE}	41 (106)	66 (146)
Длительность интервала между сигналами разрешения	t_{CC}	30	50
Время сдвига сигнала записи относительно разрешающего сигнала	t_{WW}	0 (40)	0 (55)
Время удержания сигнала записи относительно сигнала разрешения	t_{HEW}	25	40
Время выборки разрешения	t_{CO}	45	70
Длительность сигнала записи	t_{WP}	25	40
Время сдвига входной информации относительно сигнала записи (в режиме считывания-модификации-записи)	t_{WSD}	0	0
Время удержания входной информации относительно сигнала записи (в режиме считывания-модификации-записи)	t_{WHD}	35	50

Таблица 3

Таблица истинности

Вход			DO	Рабочее состояние
$\overline{CE}$	$\overline{WE}$	$\overline{DI}$		
1	Любое	Любое	Высокий импеданс	Режим хранения
0	1	Любое	0 или 1	Считывание
0	0	0	Высокий импеданс	Запись »Лог. 0»
0	0	1	То же	Запись «Лог. 1»

Таблица 4

Основные электрические параметры микросхем КР132РУ6

Параметр, единица измерения	Обозначение	Норма
Входное напряжение, В		
низкого уровня	U_{IL}	0,4
высокого уровня	U_{IH}	2,4
Выходное напряжение, В		
низкого уровня при $U_{CC}=5{,}5$ В $I_{OL} \leqslant 5$ мА	U_{OL}	0,4
высокого уровня при $U_{CC}=4{,}5$ В $I_{OH} \leqslant 2$ мА	U_{OH}	2,4
Ток утечки, мкА		
по каждому входу	I_{LI}	10
на выходе	I_{LOH}, I_{LOL}	50
Ток потребления		
в микромощном режиме хранения, мкА, при $U_{CC}=0$ $U_{CE}=5{,}5$ В	I_{CCS2}	2000
в режиме хранения, мА	I_{CCS1}	25

Таблица 5

Предельные значения эксплуатационных параметров

Параметр, единица измерения	Обозначение	Норма	
		мин.	макс.
Напряжение питания, В	U_{CC}	—	6
Напряжение на любом выводе, В	U	—0,3	U_{CC}
Напряжение на входе $\overline{CE}$, В	$U_{\overline{CE}}$	4,5	
Выходной ток, мА	I_O	—	10
Емкость нагрузки, пФ	C_H	—	120
Длительность фронтов входных сигналов, нс	$t_{0,1}$, $t_{1,0}$	—	500

НА КНИЖНОЙ ПОЛКЕ

ра-226»: Практ. руководство.— М.: Финансы и статистика, 1987 (1 кв.).

Описываются средства и способы программирования на новой отечественной микроЭВМ, предназначенной для эксплуатации непосредственно на рабочих местах в органах управления различного уровня. Рассматривается широко распространенный диалоговый язык программирования Бейсик. Изложение сопровождается учебными примерами.

З а п о л ь с к и й А. П., Ш к л я р В. Б., Ч и с т я к о в А. Н. **Персональные компьютеры Единой системы ЭВМ:** Науч.-попул.— М.: Финансы и статистика, 1987 (IV кв.).

Книга ставит целью способствовать своевременной подготовке массового читателя к применению персональных ЭВМ — нового направления в развитии вычислительной техники. Рассматриваются предпосылки появления персональных ЭВМ, состояние и перспективы их развития.

Книга знакомит с архитектурой персональных профессиональных ЭВМ Единой системы ЭВМ (ЕС ЭВМ), их техническими и программными средствами, возможностями использования в различных областях профессиональной деятельности. Книга написана доступным языком.

А п п а к М. А. **Автоматизированные рабочие места на базе микроЭВМ «Искра-226»:** Практ. руководство.— М.: Финансы и статистика, 1987 (1 кв.).— 6 л.: ил.— 30 к., 25 000 экз.

Рассматриваются методические, языковые и программные средства реализации автоматизированных рабочих мест на основе профессиональной микроЭВМ «Искра-226» для специалистов различного профиля.

Для экономистов, проектировщиков, конструкторов, программистов и работников других профессий, использующих в своей практике автоматизированные рабочие места, а также специалистов, занимающихся разработкой и внедрением баз данных и программных средств на микро- и мини-ЭВМ в отраслевых и региональных АСУ.

УДК 681.327.8.06

Коваленко В. А., Олейник А. В., Пархоменко Л. П., Солдатенко Л. М. **БИС контроллера КР1818ВГ93 для накопителя на гибком диске.**— Микропроцессорные средства и системы, 1986, № 3, с. 3.

Представлены основные характеристики БИС контроллера НГМД КР1818ВГ93, предназначенного для управления выводом информации из ЭВМ на гибкие магнитные диски и вводом информации в ЭВМ. Подробно описана система команд, приведены служебные коды и форматы массивов информации, временные диаграммы работы микросхемы. Дан пример включения контроллера с формирователями импульсов синхронизации и обеспечения предкомпенсации сигналов записи данных на ГМД.

УДК 681.325.5-181.48

Горовой В. В., Евдокимов В. А., Медведев В. И., Сахаров А. М. **БИС специализированного АЛУ К1815ИА1.** — Микропроцессорные средства и системы, 1986, № 3, с. 8.

Описана БИС специализированного АЛУ К1815ИА1, предназначенная для построения процессоров с параллельной обработкой информации. Приведены структурная схема, временная диаграмма работы, описание работы микросхемы. БИС содержит четыре одноразрядных АЛУ с общим управлением. По электрическим параметрам микросхема полностью совместима с микросхемами серий К583, К584, К155, К1533.

УДК 621.382.33

Кулешов В. И., Прибыльский А. В., Сякерский В. С., Яковлев Ю. В. **БИС ортогональной матрицы регистров сдвига К1815ИР1.** — Микропроцессорные средства и системы, 1986, № 3, с. 10.

БИС быстродействующей ортогональной матрицы регистров сдвига с перестраиваемой структурой информационной емкостью 8×4 бит предназначена для использования в высокопроизводительных конвейерных системах цифровой обработки данных. Приведены структурная схема, временная диаграмма работы, функциональные и электрические характеристики микросхемы.

УДК 621.3.049.77

Попов Ю. П., Милованов А. И., Медведев В. И., Васильев Л. В. **Преобразователь последовательно-параллельных кодов К1815ПР1.** — Микропроцессорные средства и системы, 1986, № 3, с. 12.

Микросхема К1815ПР1 предназначена для преобразования последовательно-параллельных дополнительных кодов чисел в прямые и прямых кодов в дополнительные в системах цифровой обработки сигналов. Может осуществлять изменение знака числа. По электрическим параметрам совместима с ТТЛ-схемами.

УДК 681.3.06

Лавров С. С. **Представление и использование знаний в автоматизированных системах.**— Микропроцессорные средства и системы, 1986, № 3, с. 14.

Приводится классификация знаний, используемых для построения моделей проблемных областей и спецификаций задач, а также классификация систем представления знаний и решения задач. Классификация идет по следующим критериям: степень использования различных видов знаний, форма представления знаний, вид получаемого решения задачи, степень универсальности системы. В заключение характеризуются основные черты системы решения задач на основе базы знаний, разрабатываемой с участием автора, и ее место в предложенной классификации.

UDC 681.327.806

Kovalenko V. A., Oleinik A. V., Parchomenko L. P., Soldatenko L. M. **Floppy-Disks Controller KR1818VG93.**— Microprocessor Devices and Systems, 1986, N 3, p. 3.

The main features of LSI floppy-disk controller KR1818VG93 are discussed. Instruction set, information formats and timing diagrams are presented. An example is given for building floppy-disk controller with synchronization pulse generator and signal precompensation.

UDC 681.325.5-1811.48

Gorovoi V. V., Evdokimov V. A., Medvedev V. I., Sacharov A. M. **Specialized CPU K18151A1.**— Microprocessor Devices and Systems, 1986, N 3, p. 8.

The paper describes specialized CPU K18151A1 intended for building parallel processing systems. Structural scheme, timing diagrams and functional features are presented. The LSI contains four single-bit CPUs with common control. It is compatible electrically with the series K583, K584, K155, K1533.

UDC 621.382.33

Kuleshov V. I., Pribylsky A. V., Sjakersky V. S., Yakovlev Yu. V. **Orthogonal Array of Shift Registers on LSI K1815IR1.**— Microprocessor Devices and Systems, 1986, N 3, p. 10.

Orthogonal 8×4 array of shift registers is intended for high-speed pipe-lined processing of digital information. The structural scheme, timing diagrams, functional and electrical characteristics are presented.

UDC 621.3.049.77

Popov Yu. P., Milovanov A. I. Medvedev V. I., Vasiliev L. V., Sai A. I. **K1815PR1 Serial/Parallel Transformer.**— Microprocessor Devices and Systems, 1986, N 3, p. 12.

LSI K1815PR1 is intended for transformation of serial / parallel codes from direct to complementary form and vice versa. The device is applied for digital signal processing. It is compatible with TTL — devices.

UDC 681.3.06

Lavrov S. S. **Knowledge representation in computerized systems.**— Microprocessor devices and systems, 1986, N 3, p. 14.

A classification of knowledge used in building of application domain models and in problem specification together with a classification of knowledge representation systems and problem solving systems are given. The classification is based upon the following criteria: the number of different kinds of knowledge represented, the form of knowledge representation, the form of solution given by the system the degree of universality of the system. The paper describes the main features of the knowledge base based problem solving being developed by the author and show its place in the proposed classification.

УДК 681.324.5+681.3.068

Мозговой Г. П., Семенова С. С., Семин Е. И., Трещалин О. В. **Мета-ассемблер МЕАСС для микропроцессорных систем с наращиваемой разрядностью.** — Микропроцессорные средства и системы, 1986, № 3, с. 20.

Предлагаемый мета-микроассемблер позволяет облегчить этап разработки микропрограммного обеспечения для любых микропроцессорных систем с микропрограммным управлением и наращиваемой разрядностью.

УДК 681.3.06

Кобылинский А. В., Сабадаш Н. Г., Тесленко А. К. **Система автоматизации программирования однокристальной микроЭВМ.** — Микропроцессорные средства и системы, 1986, № 3, с. 23.

Рассмотрена система программирования однокристальной микроЭВМ КМ1816ВЕ48, реализованная на комплексе технических средств «Электроника МС0401» и мини-ЭВМ СМ-4. Кратко описаны основные свойства языка ассемблера однокристальной микроЭВМ АСМ48, функции программы кросс-ассемблера и программы интерпретатора.

УДК 681.3.06

Белов А. М., Иванов Е. А., Муренко Л. Л. **Комплексы кросс-программ «Электроника Микросс».** — Микропроцессорные средства и системы, 1986, № 3, с. 27.

В краткой форме перечислены функции комплексов кросс-программ «Электроника Микросс-580» для БИС серии К580 и «Электроника Микросс-048» для однокристальных микроЭВМ серии КМ1816. Комплексы сданы в отраслевой фонд алгоритмов и программ.

УДК 681.142 : 003.6

Ляпунов М. М., Беляев Б. И. **Персональная графическая станция ПЕГАС.** — Микропроцессорные средства и системы, 1986, № 3, с. 43.

Станция ПЕГАС, разработанная на основе алфавитно-цифрового дисплея 15ИЭ-00-013 и микроЭВМ МС 1201.1, обладает собственной вычислительной мощностью для автономного решения прикладных задач и предназначена для обеспечения широкого круга пользователей.

УДК 621.385.001 : 621.387

Жулай С. Г., Конько В. В. **Микроконтроллер вывода информации на газоразрядную индикаторную панель.** — Микропроцессорные средства и системы, 1986, № 2, с. 70.

Рассмотрен конкретный пример реализации программно управляемого контроллера вывода алфавитно-цифровой информации на газоразрядную индикаторную панель. При построении контроллера использовалась микропроцессорная серия КР580. Возможна организация параллельного (скорость обмена информацией 3300 байт/с) и последовательного интерфейсов.

УДК 681.322.1-181.4+681.338.5

Громов Г. Р. **Автоформализация профессиональных знаний.** — Микропроцессорные средства и системы, 1986, № 3, с. 80.

Рассматривается процесс становления и развития «индустрии знаний». Кратко анализируются основные исторические этапы развития информационной технологии. Обсуждаются особенности процессов формализации знаний в режиме персональных вычислений, который определяется автором как автоформализация профессиональных знаний. Показано, что методы автоформализации становятся основным средством включения в активный фонд научного знания новых и все более мощных пластов индивидуального профессионального искусства.

UDC 681.325.5:681.3.068

Mozgovoi G. P., Semenova S. S., Semin E. I., Treschalin O. V. **MEASS Meta — Assembler for Microprocessor Systems with Exapandable Word Length.** — Microprocessor Devices and Systems, 1986, N 3, p. 20.

The meta — assembler is described providing microprogram development for microprocessor systems with microprogram control and variable word length.

UDC 681.3.06

Kobylinskiy A. W., Cobodazh N. G., Teslenko A. K. **Program development system for a single-chip microprocessor.** — Microprocessor Devices and Systems, 1986, N 3, p. 23.

A program development system the KN1816VE48 microprocessor is implemented on the dual processor system consisting from "Electronica MC0401" and SM-4 minicomputers. A brief description of the "ASM-48" assembly language and of the cross-compiler and the interptretet a given.

UDC 681.3.06

Belov A. M., Ivanov E. A., Murenko L. L. **"Electronica Micross" Cross-Programming Sets.** — Microprocessor Devices and Systems, 1986, N 3, p. 27.

The paper presents the functions of "Electronica Micress-580" and "Micross-048" cross-programming systems for LSI's K580 and KM1816.

UDC 681.142:003.6

Liapunov M. M., Belaev V. I. **PEGAS — Personal Graphical Station.** — Microprocessor Devices and Systems, 1986, N 3, p. 43.

The PEGAS station is built on the basis of 15IE 00 013 display with embedded microcomputer MS 1201.1. It is capable of supporting autonomous application programs.

UDC 621.385.001:621.387

Zhulai S. G., Konko V. V. **Microcontroller for Plasma Display.** — Microprocessor Devices and Systems, 1986, N 3, p. 70.

The paper presents an example of building controller for plasma display. It is based on KR580 microprocessor set. Serial and parallel interfaces can be built with output rate of 3300 butes/sec. Internal system testing is provided which includes diagnostics on incorrect instructions, RAM and ROM faulureset.c.

UDC 681.322.1-181.4+681.338.5

Gromov G. R. **«Autoformalization»: Knowledge acquisition of professional skills.** — Microprocessor Devices and Systems, 1986, N 3, p. 80.

The paper discusses the process of the development of the "knowledge industry". A brief analysis of the main historical stages of the evolution of the information technology is given. The specifics of the knowledge acquisition and representation processes in personal computing is discussed. It is shown that knowledge representation becomes the main form of the integration of new units of individual professional skills into the active body of the scientific knowledge.

МИКРОПРОЦЕССОРНАЯ ТЕМАТИКА В КНИГАХ ЭНЕРГОАТОМИЗДАТА

Для подготовки, повышения квалификации и повседневной работы специалистов издательство выпускает немало книг, посвященных разработке, отладке и применению микропроцессорных средств и устройств.

В работе Б. М. Кагана и В. В. Сташина «Основы проектирования микропроцессорных устройств автоматики» (1986) рассмотрены организация 8-, 16- и 32-разрядных микропроцессоров (МП), секционированных микропрограммируемых БИС, структура и функционирование основных БИС МП-наборов; описаны методы комплексирования, синхронизации программной и микропрограммной настройки БИС, способы программной реализации МП, типовые процедуры обработки данных и управления. В монографии В. Л. Григорьева «Программирование однокристальных микропроцессоров» (1987) будет изложена справочная информация о БИС К1810ВМ86 с фиксированными длиной слова и системой команд, а также программные ресурсы БИС и вопросы программирования на языке ассемблер; даны многочисленные примеры выполнения команд и программ решения несложных задач.

Книги Энергоатомиздата ориентированы на внедрение микропроцессорных систем во все отрасли энергетики: теплотехнику и тепло-, электро-, гидроэнергетику и, конечно, в ядерную энергетику.

Для технологов, занимающихся разработкой и внедрением систем автоматизации производственных процессов, в 1986 г. издана книга О. Е. Вершинина «Применение микропроцессоров для автоматизации технологических процессов». Рассказ об использовании микропроцессоров в АСУ ТП (разработке алгоритмов управления, проектировании аппаратных средств и кодировании программ) сопровождается примерами разработки программ для простых систем сбора производственной информации и управления технологическими процессами. Приведены сведения о составе микропроцессорных комплектов БИС, архитектуре центрального процессора, системе машинных команд, операционных системах и применении интерфейсных ЕИС.

Для разработчиков измерительной электронной аппаратуры издательство выпустит в 1987 г. три книги по использованию микропроцессорной техники. В. С. Гутников в монографии «Интегральная электроника в измерительных устройствах» рассмотрел применение аналоговых, цифровых и аналого-цифровых микросхем и дал примеры функциональных узлов, построенных на интегральных ТТЛ- и КМОП-схемах средней и большой степени интеграции. Книга В. С. Попова и И. Н. Желбакова «Измерение среднеквадратического значения напряжения» посвящена основным направлениям повышения точности и улучшения динамических характеристик средств измерения, в частности на основе применения микроЭВМ. И, наконец, монография М. В. Гальперина «Практическая схемотехника в промышленной автоматике», предназначенная разработчикам промышленной электроники, автоматики и измерительной техники, освещает вопросы схемотехники усилителей, компараторов, логических элементов и узлов, в том числе микропроцессорных устройств. Большое внимание уделено электромагнитной совместимости и помехоустойчивости систем.

В работе В. Г. и Э. Г. Файнштейнов «Микропроцессорные системы управления тиристорными электроприводами» (1986) изложены основы теории, методы расчета, алгоритмы управления МП-системами управления и способы их реализации; основные принципы построения и особенности специализированной микроЭВМ, информационно-логических, наладочных и исполнительных устройств. Даны сведения об исполнении комплектной системы управления, методах расчета настроечных параметров, приведены способы и приемы экспериментальных исследований и наладки.

Любителям, интересующимся электроникой, внедрением простых и экономичных автоматизированных систем, понравится книга Н. Како и Я. Яманэ «Датчики и микроЭВМ». В ней рассмотрены различные типы датчиков, классификация их по принципу действия и сферам применения, приводятся схемы сопряжения датчиков с микроЭВМ и конкретные примеры использования системы датчик—микроЭВМ (кондиционер, СВЧ-печь, автоматическая домашняя система, автомобиль, измеритель влажности зерна, медицинская диагностическая установка, интеллектуальный робот и т. д.).

Инженерно-технических работников, не имеющих специальной подготовки по вычислительной технике, заинтересует монография Р. Токхайма «Микропроцессоры. Курс и упражнения» (1987). Автор в элементарной форме излагает вопросы архитектуры, функционирования и программирования МП. Большое число решенных задач и иллюстраций помогает лучше усвоить материал.

Книга Д. Фокса и А. Фокса «Бейсик для всех» (1986), как и следует из названия, предназначена студентам и инженерам, не имеющим специальной подготовки по программированию, и даже школьникам. Книга представляет собой курс программирования на языке Бейсик для пользователей персональных компьютеров.

Энергоатомиздат выпустит в 1987 г. переведенную с английского книгу и для специалистов в области проектирования, производства и эксплуатации микросистем. Монография Г. Б. Уильямса «Отладка микропроцессорных систем» посвящена практическим вопросам отладки и применяемому для этого контрольно-измерительному оборудованию. В ней приведено описание логических и сигнатурных анализаторов, показаны схемотехнические решения и программное обеспечение, необходимые при отладке микросистем.

С. С. Матвеев

1р. 10к.

Индекс 70588

ВНИМАНИЕ

НАПОМИНАЕМ, что подписка на наш журнал принимается ПОКВАРТАЛЬНО. За месяц до начала любого квартала прекращается подписка на ту часть годового комплекта журналов, которая издается в следующем трехмесячном периоде.

ВНИМАНИЕ

С 1986 ГОДА ЖУРНАЛ ВЫХОДИТ 6 (ШЕСТЬ!) РАЗ В ГОД!

Если Вы подписались только на 4 номера, то имеется возможность оформить дополнительную подписку на 5 и 6 номера годового комплекта. Сделать это Вы можете в любом отделении «Союзпечать».

Цена одного номера — 1 руб. 10 коп. Индекс журнала — 70588.