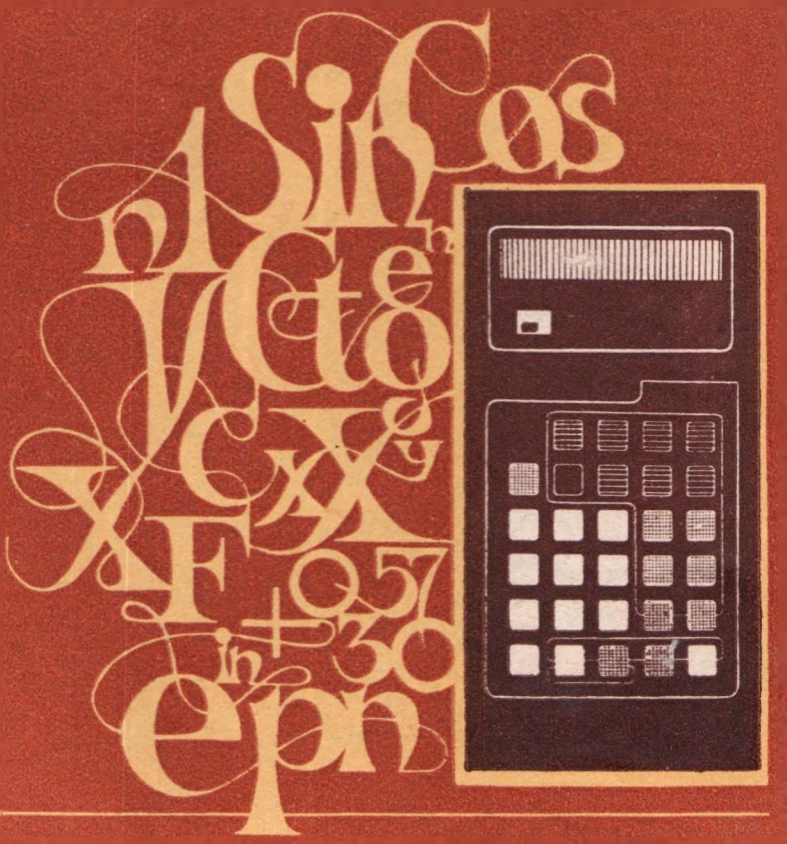
М. И. ЖАЛДАК
Н. В. МОРЗЕ

ОСНОВЫ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ



Допущено Государственным агропромышленным комитетом Украинской ССР как учебное пособие для учащихся средних специальных учебных заведений

~~Рекомендовано к изданию коллегией~~ Государственного комитета Украинской ССР по профессионально-техническому образованию в качестве методического пособия для средних профессионально-технических училищ

Основы информатики и вычислительной техники. / М. И. Жалдак, Н. В. Морзе. — К. : Вища шк. Головное изд-во, 1985. — 191 с.

В пособии рассматриваются арифметические и логические основы электронно-вычислительной техники с элементами схемотехники. Основы программирования излагаются на базе программируемых микрокалькуляторов типов «Электроника Б3-34», «Электроника МК-56», «Электроника МК-54». Раскрываются понятия о машинно-ориентированных и машинно-независимых языках программирования, трансляции, распределении памяти, отладке программ, автоматизации программирования. Описываются элементы алгоритмического языка высокого уровня БЕЙСИК. Даются примеры использования микропроцессорных систем.

Для преподавателей и мастеров производственного обучения средних профессионально-технических училищ; может быть использовано преподавателями других учебных заведений.

Табл. 70. Ил. 88.

Р е ц е н з е н т ы: кандидат технических наук *Ю. Г. Слесарев* (зав. лабораторией Всесоюзного НИИ аналитического приборостроения), *М. В. Зубец* (нач. отдела профподготовки Госпрофобра УССР), преподаватель *В. Д. Леонский* (Киевское СПТУ № 11)

Редакция ~~учебной и научной~~ литературы по информатике, ~~вычислительной технике, кибернетике и АСУ~~

Ж 2405000000—~~109~~ / М211(04)—86 БЗ ~~24-6-85~~

Зав. редакцией Г. Ф. Гороховская

§ 1. РОЛЬ И МЕСТО ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ В СОВРЕМЕННОМ ПРОИЗВОДСТВЕ

Усложнение современного производства, необходимость оперативной связи между различными предприятиями, потребность переработки в короткие сроки огромного количества информации выдвинули на первый план задачи автоматизации многих производственных циклов, управления предприятиями и целыми отраслями производства. «Решение проблем, которые перед нами стоят, использование возможностей, которыми мы располагаем, во многом зависят от уровня руководства народным хозяйством, уровня планирования и управления» — отмечалось на XXVI съезде КПСС.

Для эффективного управления технологическим циклом, предприятием или отраслью промышленности необходимо в сжатые сроки обрабатывать огромные потоки информации. Объективная возможность такого управления появилась лишь благодаря достижениям математики, кибернетики, теории информации и, особенно, электронно-вычислительной техники. На апрельском (1985 г.) Пленуме ЦК КПСС подчеркивалось, что первостепенное внимание должно быть уделено именно ускорению развития вычислительной техники, приборостроения, электротехники и электроники как катализаторов научно-технического прогресса.

Современная электронно-вычислительная машина (ЭВМ) — быстрый, надежный, безотказный исполнитель и советчик. Поэтому естественно, что математика и вычислительные машины проникли почти во все области человеческой деятельности.

В настоящее время ЭВМ широко применяются в народном хозяйстве и оказывают поистине революционное влияние на развитие научных исследований, экономики, инженерного проектирования, планирования, управления производством, на технику и технологию производства. Дело в том, что вычислительные машины перерабатывают информацию, а обрабатывать ее и затем принимать на этой основе решения человеку приходится в любой производственной

деятельности. Это относится и к труду станочника, который, внимательно следя за движением резца, перерабатывает в уме сведения о его местоположении и принимает решение о дальнейших действиях. Это же касается труда агронома, бухгалтера, ученого. Очевидно, что ускорение обработки информации, возможность при принятии решений использовать более точные и детальные сведения прямо влияют на конечный результат трудовой деятельности человека, а значит, и общества в целом. Поэтому важнейшим, определяющим элементом ускорения развития научно-технического прогресса являются ЭВМ.

ЦК КПСС, Совет Министров СССР уделяют большое внимание разработке и внедрению средств вычислительной техники в народное хозяйство. Основные направления экономического и социального развития СССР на 1981—1985 годы и на период до 1990 года, утвержденные XXVI съездом КПСС, предусматривают расширение комплексной автоматизации производства на базе вычислительной техники.

Цель автоматизации производства — повышение производительности труда, избавление человека от опасной, вредной и монотонной работы. Механизированное производство организуется по схеме «человек — машина». Управление машиной со стороны человека во многих случаях носит механический характер, требуя от него лишь минимальных творческих способностей (например, работа токаря, выполняющего определенную последовательность действий, многократно повторяемую на протяжении почти всего рабочего дня, сварщика, сборщика на конвейере, оператора технологической установки). Такую работу уже сейчас можно считать не самой квалифицированной и не самой высокопроизводительной. Производительность труда резко повысится, если функции по управлению машинами передать автоматам.

Автоматизация производственных процессов имеет большую, почти столетнюю, историю: копировальные автоматы, ткацкие станки с программным управлением, конвейерные линии. В настоящее время широкое распространение получили автоматические поточные линии, объединяющие в себе комплексы автоматически работающих станков. Но все это лишь первые шаги на пути к всесторонней, комплексной автоматизации — узкая ориентация станков и поточных линий на определенный вид изделий. Поэтому такие средства могут использоваться только там, где производство носит массовый, устойчивый характер.

Следующий этап в автоматизации — разработка всякого рода программируемых и за счет этого перенастраиваемых средств: станков с числовым программным управлением и промышленных роботов. Станок с программным управлением настраивается на обработку определенной детали с помощью программы. Чтобы начать обрабатывать другую деталь, следует ввести в устройство управления новую программу, что выполняется весьма просто. Аналогично функционируют и промышленные роботы, которые начали применяться для перемещения заготовок и деталей между станками, выполнения сборочных работ, электросварки. Они легко переналаживаются для выполнения самых различных операций.

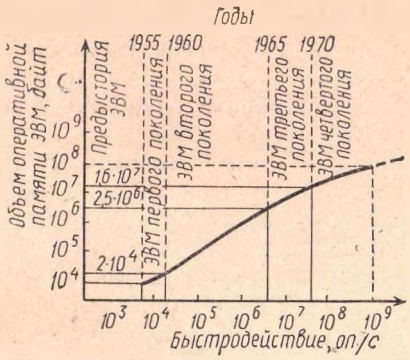


Рис. 1. Диаграмма смены поколений ЭВМ

ЭВМ используются для управления домнами и мартенами, прокатными станами, большими химическими производствами. Так как ЭВМ имеет программное управление, а замена программы осуществляется легко, ЭВМ легко и быстро можно переналаживать с одной работы на другую.

За более чем 30 лет своего существования ЭВМ прошли путь от быстродействия в одну тысячу операций в секунду до сотен миллионов операций в секунду (рис. 1). С 1955 г. каждые последующие пять лет в вычислительной технике принципы построения ЭВМ обновлялись. В настоящее время действуют ЭВМ четвертого поколения.

Проект первой в мире вычислительной машины, которая могла выполнять вычисления автоматически без вмешательства человека и при этом выбирать тот или иной путь продолжения вычислений в зависимости от найденных результатов, еще в 30-х годах прошлого столетия разработал известный английский ученый Чарльз Беббедж. В его машине предусматривались следующие устройства:

1. «Склад» для хранения чисел (в современных вычислительных машинах такое устройство называют *запоминающим*).

2. «Фабрика» для выполнения действий над числами (в современных вычислительных машинах такое устройство называют *арифметико-логическим*).

3. Устройство для управления операциями машины в нужной последовательности, в частности перенесением чисел из одного места памяти в другое.

4. Устройство для ввода данных в машину и вывода результатов.

Первую в мире программу для машины Беббеджа написала Ада Лавлейс, дочь знаменитого Джорджа Байрона. Однако Беббеджу не удалось увидеть свою машину в действии.

Первая автоматически действующая вычислительная машина была построена лишь в 1944 г. в США. Машина работала на механических и релейно-контактных элементах. Первой электронной ламповой вычислительной машиной стала машина «ЭНИАК» (1946 г., США), в континентальной Европе такая машина («МЭСМ») была разработана в 1950 г. в СССР.

Вычислительные машины, построенные на электронных лампах и имевшие широкое распространение в 50-е годы нашего столетия, принято считать машинами *первого* поколения. Это были очень сложные и громоздкие устройства.

Первыми серийными ЭВМ, которые начали выпускаться в начале 50-х годов в нашей стране, были машины «БЭСМ-1» и «Стрела». По сравнению с другой радиоэлектронной аппаратурой они были во много раз сложнее по устройсту и для тех лет обладали очень высоким быстродействием. Так, машина «БЭСМ-1», содержащая около 7000 электронных ламп и еще большее число резисторов, конденсаторов и других элементов, выполняла 8000 арифметических и логических операций в секунду. Организация слаженной работы такого огромного количества отдельных элементов уже являлась достижением инженерной мысли, требовала напряженного творческого труда ученых, инженеров и рабочих. Машина занимала большой зал, нуждалась в специальной вентиляции, стабильных источниках питания, профилактической смене ламп. Для поддержания ее работоспособности требовался штат опытных инженеров, способных быстро находить и устранять неисправности.

Считается, что эпоха машин первого поколения охватывала период от конца 40-х до середины 50-х годов. На смену этим машинам пришли ЭВМ *второго* поколения, выполненные на полупроводниковых элементах с быстродействием до нескольких миллионов операций в секунду. Представителями ЭВМ второго поколения в нашей стране были машины «Минск-22», «Минск-32» и др.

Третье поколение связывают, как правило, с ЭВМ, построенными на так называемых *интегральных микросхемах*.

Схемы называются интегральными по той причине, что в одном кристалле такой схемы объединены многие электронные устройства. Из интегральных микросхем можно более просто монтировать сложные узлы ЭВМ. В нашей стране хорошо известными представителями машин третьего поколения являются ЭВМ серии ЕС.

Четвертое поколение ЭВМ часто связывают с использованием в качестве элементной базы больших интегральных микросхем, электронных запоминающих устройств, микропроцессоров.

Микропроцессор — это логическое устройство, предназначенное для интерпретации программы и изготовленное по интегральной технологии. Микропроцессоры, выполненные, как правило, на одном кристалле большой интегральной микросхемы, являются строительным элементом ЭВМ различных типов и назначений.

Если к микропроцессору присоединить запоминающее устройство и устройство ввода — вывода информации, нацелив всю совокупность электронных устройств на выполнение определенной функции, то получится микрокомпьютер, или микропроцессорная система.

К машинам современного, четвертого поколения относятся многомашинные и многопроцессорные системы обработки информации, а также микрокомпьютеры. На переднем форзаце этой книги показано развитие ЭВМ и их элементной базы по поколениям.

В настоящее время широко обсуждаются проекты машин *пятого* поколения, которые будут резко отличаться от машин предыдущих поколений прежде всего тем, что их организация будет значительно в большей степени отвечать идеям создания искусственного интеллекта.

Кроме классификации ЭВМ по поколениям, существует и другая классификация, связанная с областями применения и назначением вычислительных машин (рис. 2).

Когда говорят о микро-ЭВМ, обычно подразумевают вычислительную машину, встроенную в прибор, инструмент, станок, научную и измерительную аппаратуру, в робот-манипулятор и т. д. На микро-ЭВМ возлагаются функции программного управления сложными робототехническими устройствами, целыми автоматическими производственными линиями. Сферы их применения непрерывно расширяются.

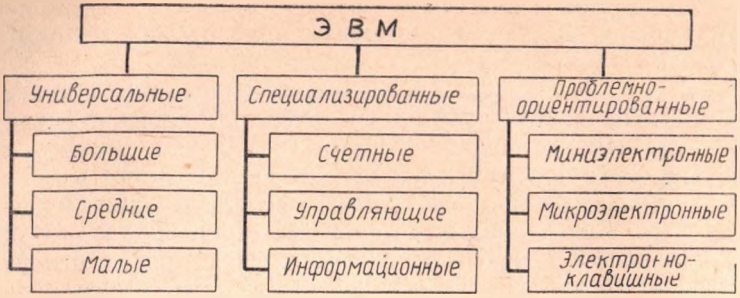


Рис. 2. Классификация ЭВМ

Микро-ЭВМ является микропроцессорной системой. Примером микро-ЭВМ могут быть микрокалькуляторы, которые образуют целое семейство вычислительных машин от самых простейших, выполняющих только четыре арифметических действия, до программируемых, на которых можно выполнять достаточно сложные инженерные расчеты и выводить результаты на печать.

К категории мини-ЭВМ относятся стационарные установки, используемые и как самостоятельные машины, и как машины для управления приборами и автоматическими линиями. В этом смысле нет особой разницы между микро- и мини-ЭВМ. На основе мини-ЭВМ строятся различного рода автоматизированные рабочие места разных целей и назначений. На базе микро- и мини-ЭВМ в настоящее время стали выпускаться так называемые *персональные компьютеры.*

К сверхбольшим или сверхпроизводительным вычислительным системам принадлежат ЭВМ с быстродействием, превышающим сотни миллионов операций в секунду.

Специалисты насчитывают более 200 тыс. применений ЭВМ в промышленности и быту и считают, что число таких применений ежегодно возрастает на 800. Особенно широкое распространение получили такие средства электронно-вычислительной техники, как микрокалькуляторы и микро-ЭВМ.

Появление микропроцессоров привело к резкому снижению себестоимости, размеров, потребляемой мощности и одновременно к повышению надежности микро-ЭВМ, что позволило широко применять микро-ЭВМ в системах автоматизации производства, автоматизированных системах сбора и обработки информации, управления и контроля. Микропроцессорная техника создает возможность сделать

процесс автоматизации всеохватывающим, становится средством, позволяющим экономить энергию, труд, материальные ресурсы.

Сегодня микропроцессорная техника и системы автоматизации становятся составной частью технологии любого производства. Однако автоматизировать технологию производства в отрыве от людей, владеющих данной технологией, невозможно. Если создается прокатный стан, то вся автоматизация, вся компьютеризация, все программное обеспечение должны разрабатываться коллективом, который создает этот стан. Если это автомобиль, то всю автоматизацию должны разрабатывать специалисты автомобильной промышленности, потому что никто лучше них не может сказать, где надо разместить микропроцессор и какую роль он должен играть.

Люди должны уметь переводить технологию всех автоматизируемых процессов на язык ЭВМ и микропроцессоров. Огромное значение при этом приобретает проблема общения человека с ЭВМ. Поскольку машина работает только по командам, задаваемым человеком, он должен уметь задать машине программу действий, т. е. последовательность команд, которые она должна выполнить. Процесс бурного проникновения электронно-вычислительной техники в разнообразные сферы жизни требует подготовки специалистов соответствующей квалификации.

§ 2. ПОНЯТИЕ ОБ ИНФОРМАТИКЕ

На современном этапе научно-технической революции, когда ЭВМ применяются практически во всех отраслях народного хозяйства, особое значение приобретают задачи и методы разработки, проектирования, создания, оценки функционирования систем переработки информации, использующих ЭВМ. Постановка этих проблем обусловлена появлением и распространением новой индустриальной технологии сбора, обработки, хранения и передачи информации.

Под и н ф о р м а ц и е й понимается любая совокупность сигналов, воздействий, сведений, которые некоторая система воспринимает от окружающей среды (*входная информация*), выдает в окружающую среду (*выходная информация*) либо хранит в себе (*внутренняя информация*). Все, что делает человек, так или иначе связано с использованием информации. Каждый человек в своей деятельности ежедневно сталкивается с необходимостью

решения различного рода задач, для чего он всегда обрабатывает некоторую информацию: известное значение—условие задачи —превращается в новое значение —решение задачи.

Методы переработки информации (в биологии, медицине, лингвистике, математике и других областях знаний) изучает наука, называемая *информатикой*. В современном представлении информатика — это наука о законах и методах получения, накопления, хранения, переработки и передачи информации с использованием ЭВМ.

Вплоть до последнего времени в использовании и обработке информации главным образом участвовал лишь человек. С появлением ЭВМ — универсальных автоматических устройств — стало возможным решать любые задачи по переработке информации. С ЭВМ, а значит, с рождением и распространением новой автоматической технологии сбора, обработки, хранения и передачи информации связано появление новой научной дисциплины — информатики.

Овладение «информационным взрывом» в различных сферах человеческой деятельности требует создания и использования специальных машинно-информационных систем для оперирования особенно интенсивными потоками информации. Структура, техническая основа, программное обеспечение таких систем должны быть приспособлены к выполнению специальных процедур по оптимальному сбору, хранению, переработке и выдаче больших массивов информации; эти системы должны быть рассчитаны на автоматизированные выборку и переработку данных, а также подготовку решений в соответствии с задачами функционирования и развития тех областей социальной деятельности, в которых такие системы применяются (управление, наука, производство, образование, медицина, военное дело и т. п.).

Предметом информатики как научной дисциплины и выступают во взаимодействии со средой машино-информационные системы, воплощающие в себе новую, неизвестную в прошлом, технологию сбора, переработки, передачи информации — технологию, которая переводит практику управления, регулирования материального производства, научных исследований, образования и других сфер человеческой деятельности на принципиально новый индустриальный уровень.

Одним из важнейших условий применения ЭВМ является построение алгоритма (программы), позволяющего из исходной информации получать по указанным в алго-

ритме правилам переработанную информацию. Программирование — дисциплина, которая исследует методы формулирования и решения задач с помощью ЭВМ; это важнейшая, определяющая составная часть информатики.

Современная информатика постепенно складывается как самостоятельная научная дисциплина, имеющая свою теорию. Вместе с тем в многочисленных прикладных областях, где применяются методы информатики, специфика приложений определяет ее новые прикладные направления. Информатика снабжает методами исследований многие предметные области. Жизнедеятельность общества, его важнейших подсистем — производства, науки, образования, медицины, обороны, государственного управления и других — прямо и все сильнее зависит от информатики, причем области эффективного применения последней непрерывно расширяются благодаря развитию электронно-вычислительной техники.

С наибольшей остротой проблемы формирования информатики встали в наши дни. Практика использования ЭВМ насчитывает уже несколько десятилетий. Их воздействие на жизнедеятельность общества становится широкомасштабным и глубоким в связи с появлением и широким распространением мини- и микро-ЭВМ, автоматизированных рабочих мест, станков с числовым программным управлением, программируемых роботов.

Информатика в своем развитом виде родилась в середине 70-х годов после создания микропроцессоров. Значимость информатики, ее социальных функций растет по мере перехода к новым поколениям ЭВМ. Так, переход к третьему поколению ЭВМ открыл новые возможности использования информационных систем в управлении. Среди новых сфер применения информатики назовем проведение «компьютерных» конференций, электронную корреспонденцию и прессу, электронные денежные переводы, курсы надомного обучения, рекламу, прогноз погоды, безденежную розничную торговлю, получение справок и выборку данных, выполнение некоторых функций библиотек и архивов, управление системами тревожной сигнализации, охрану окружающей среды и т. д.

Информатика — это не просто одна из очередных новых технологий. Она активно преобразует другие технологии материального и нематериального производств и в конечном счете формирует новый стиль работы и новый уклад жизни. Чем шире вторгается информатика в жизнь обще-

ства, тем больше зависимость ее дальнейшего развития от экономических, политических, культурных факторов.

Информатика дифференцируется в зависимости от конкретных областей применения — экономики, политики, военного дела, науки. Она дифференцируется также в зависимости от уровня развития технических систем, отличаясь разными техническими средствами, носителями информации, способами взаимодействия машин с людьми. Например, информатика, базирующаяся на втором поколении ЭВМ, принципиально отличается от информатики, базирующейся на четвертом поколении ЭВМ. Можно вести речь об общей информатике и ее более конкретных (прикладных) ответвлениях: экономической информатике, политической, научной, военной информатике.

Средства информатики, определяющие ее приложения к проблемам средней школы, формируют прикладную дисциплину — *школьную информатику* — ту ветвь информатики, которая занимается исследованием и разработкой программного, технического, учебно-методического и организационного обеспечения применения ЭВМ в учебном процессе школ и профессионально-технических училищ.

Глава 1

ТЕХНИЧЕСКИЕ СРЕДСТВА ЭВМ

§ 3. СТРУКТУРНАЯ СХЕМА ЭВМ

Любая ЭВМ состоит из следующих основных устройств (рис. 3): процессора, запоминающего устройства (памяти) и устройств ввода — вывода.

Процессор является основой ЭВМ и предназначен для обработки информации и управления остальными устройствами ЭВМ. Инструкции о том, как необходимо преобразовывать информацию, называются *командами*. Последовательность команд представляет собой *программу* обработки информации.

Процессор обеспечивает выполнение команд программы, переключение с одной программы на другую, реакцию на сбои в работе машины. Он объединяет в себе арифметико-логическое устройство *АЛУ* и устройство управления *УУ*.

В современных ЭВМ различают также процессоры, обеспечивающие ввод — вывод информации. Такие процессоры называются *каналами*, а процессор, который обеспечивает переработку информации,— *центральным процессором*.

Арифметико-логическое устройство является основой процессора и предназначено для обработки информации, т. е. для выполнения арифметических и логических операций над данными, поступающими из запоминающего устройства (ЗУ).

Арифметико-логические устройства, входящие в состав современных ЭВМ, могут выполнять около 200 различных операций. Кроме арифметических (сложить, вычесть, умножить, разделить), в наборе операций АЛУ содержатся операции, позволяющие редактировать информацию, переносить ее из одного поля памяти в другое и т. д.

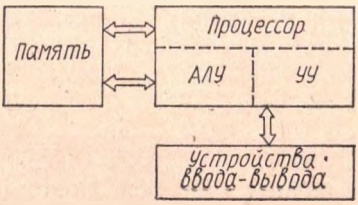


Рис. 3. Структурная схема ЭВМ

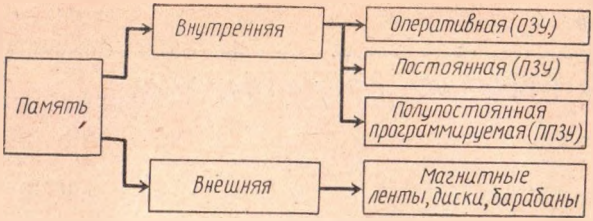


Рис. 4. Классификация запоминающих устройств ЭВМ

Устройство управления координирует работу всей ЭВМ. Это устройство выбирает команды выполняемой программы из памяти, расшифровывает содержащуюся в них информацию и управляет всем ходом вычислительного процесса в соответствии с заданной программой. УУ определяет, какую именно работу должна выполнить машина, и затем подключает соответствующее устройство для выполнения этой работы.

Для хранения программ и обрабатываемой информации предназначено ЗУ, в котором хранятся все сведения, необходимые для решения задачи (исходные данные, результаты обработки информации, программы).

Запоминающие устройства делятся на *внешние* и *внутренние* (рис. 4). Внешние ЗУ реализуются в основном на магнитных лентах, дисках или барабанах. Принцип записи информации на таких магнитных носителях такой же, как и на обычном магнитофоне. Внешние магнитные ЗУ могут хранить записанную информацию практически неограниченное время. Объем информации, которая может быть записана на такие устройства, также практически неограничен.

Внутренние ЗУ конструктивно объединены с другими электронными блоками, в частности с процессором. Эти устройства подразделяются на *оперативные запоминающие* (ОЗУ), *постоянные запоминающие* (ПЗУ) и *полупостоянные запоминающие* (ППЗУ). ОЗУ предназначены в основном для хранения исходных данных, промежуточных и конечных результатов в процессе выполнения арифметических и логических операций. ОЗУ состоит из набора регистров (ячеек), в каждом из которых может быть записано одно число. Регистры пронумерованы и номера их называются *адресами* чисел, хранящихся в регистрах.

Каждая ячейка содержит определенное количество n двоичных разрядов. Поэтому в ячейке памяти может быть

помещен набор из n нулей и единиц, т. е. n-разрядное *двоичное число*, которое называется *содержимым ячейки*.

Ячейки памяти обладают тем свойством, что содержимое ячейки может быть многократно использовано. Запись числа или команды в ячейке (подобно записи на магнитофонной ленте) сохраняется до тех пор, пока в ячейку не будут записаны новое число или команда.

Длина ячейки памяти в разных ЭВМ может быть различной. В микро-ЭВМ широко распространена память с 8-разрядной ячейкой.

Наименьшей единицей информации (один двоичный разряд) является бит. Слово «бит» произошло от английских слов «BInary digiT» — двоичная цифра.

Наряду с битом употребляются производные единицы — *байт* (8 бит), *килобайт* (2^{10} = 1024 байт, сокращенно — *К*) и мегабайт (2^{20} байт, сокращенно — *М*). Байты пронумерованы, начиная с нуля. Номер байта называется *адресом* содержимого этого байта.

При сравнении объемов памяти различных ЭВМ их обычно выражают в байтах. Так, основная память машины ЕС-1033 имеет емкость до 1 млн. байт (1 *М*), а объем основной памяти вычислительной машины ЕС-1060 может достигать 16 *М*.

Использование ОЗУ дает возможность осуществлять быстрый попеременный ввод и вывод (запись и считывание) информации, причем для записи и считывания доступна любая отдельная ячейка ОЗУ. Поэтому ОЗУ называется также *памятью с произвольной выборкой*.

Постоянные ЗУ предназначены для хранения стандартных программ, необходимых при использовании данного цифрового устройства для решения типовых задач. ПЗУ, в отличие от ОЗУ, используются только для считывания информации. Запись информации в ПЗУ осуществляется один раз навсегда при их изготовлении.

Информация, хранящаяся в *полупостоянных (программируемых)* ЗУ (ППЗУ), может быть заменена новой. Использование ППЗУ дает возможность при необходимости перенастроить вычислительную машину на решение новых задач.

Устройства ввода—вывода (УВВ) предназначены для ввода информации в ЗУ вычислительной машины, а также для вывода информации из ЗУ. При этом некоторые из них могут быть использованы только для ввода или только для вывода информации, другие же — как для ввода, так и для вывода информации.

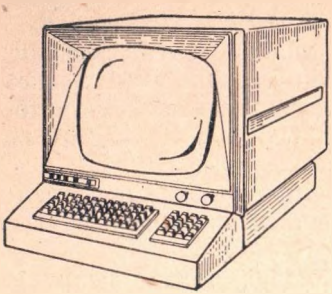


Рис. 5. Общий вид дисплея

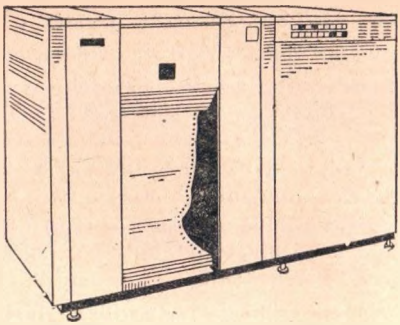


Рис. 6. Общий вид алфавитно-цифрового печатающего устройства

Одним из наиболее распространенных устройств ввода и вывода информации в ЭВМ являются *световые индикаторы* или экраны (*дисплеи*), как на рис. 5. Вводимая информация набирается на клавиатуре вручную и тут же отображается на экране дисплея с целью контроля ввода. Выводимая информация отображается на том же экране. Управление выводом осуществляется посредством той же клавиатуры.

Часто ввод и вывод информации выполняются с помощью электрифицированной пишущей машины. В отличие от дисплея, здесь вводимая и выводимая информация печатается на бумаге.

Рассмотренные устройства ввода и вывода информации дают возможность работать в режиме диалога с ЭВМ, т. е. давать задания машине на выполнение той или иной работы, тут же получать ответы, в зависимости от полученных ответов давать ЭВМ новые задания.

Информация может также выводиться через специальное *алфавитно-цифровое* печатающее устройство на бумагу (рис. 6).

Вводимая информация может подготавливаться на специальных носителях с помощью устройств подготовки данных, работающих независимо от ЭВМ. Затем с таких носителей информация с большой скоростью переписывается в ОЗУ.

В качестве носителей информации используются *бумажные* и *магнитные* ленты и карты. Бумажные ленты и карты называются перфолентами и перфокартами. Информация, наносимая на них, кодируется с помощью двух символов (0 и 1), и наносится в виде набо-

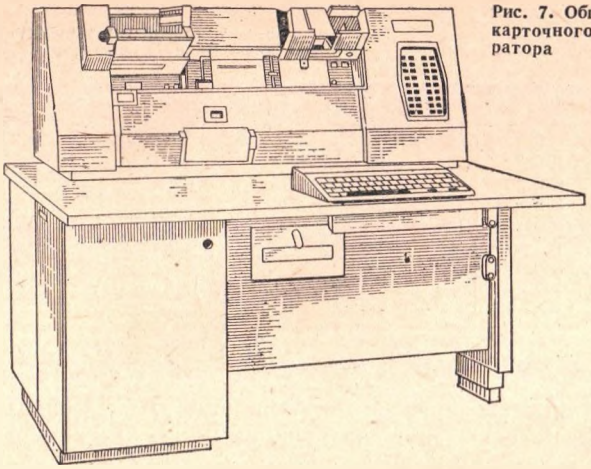


Рис. 7. Общий вид карточного перфоратора

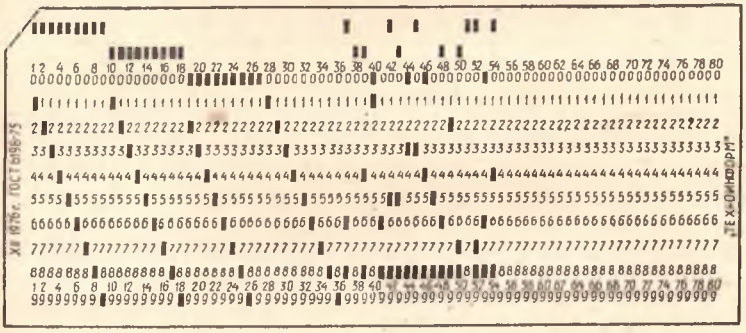


Рис. 8. Перфокарта

ров отверстий посредством устройств, называемых п е р ф о р а т о р а м и. При этом символу 1 соответствует наличие отверстия в определенной позиции, а символу 0 — отсутствие отверстия. На рис. 7 показан общий вид *карточного* перфоратора.

Перфокарта представляет собой картонный прямоугольник, на котором нанесена цифровая координатная сетка (рис. 8). Верхний угол срезан под углом 45° для установления положения перфокарты в пакете перфокарт. Вертикальные столбики координатной сетки называются *колонками*, горизонтальные строки — *позициями*. Колонки пронумерованы от 1 до 80. В каждой колонке имеется 12 позиций

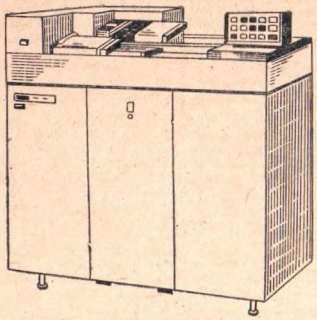


Рис. 9. Общий вид устройства ввода информации с перфокарт

Рис. 10. Перфолента

(10 из них обозначены цифрами от 0 до 9, в надцифровом поле размещены еще две позиции — 11- и 12-я). Каждый вводимый в память ЭВМ символ (цифра, буква, знак математической операции, разделительный знак) занимает на перфокарте отдельную колонку, причем каждому символу соответствует определенная комбинация пробивок в колонке перфокарты.

При вводе информации в память ЭВМ с перфокарт каждая перфокарта попадает в специальное устройство ввода. В нем установлены фотоэлементы соответственно каждой позиции на перфокарте, а также источник света. При прохождении перфокарты между источником света и фотоэлементами свет через отверстия в перфокарте попадает на фотоэлементы, в которых вследствие этого возникают импульсы тока. Эти импульсы усиливаются и поступают в соответствующие устройства ЭВМ. Скорость ввода — до 1200 перфокарт в минуту.

Процесс нанесения отверстий на перфокарты называется *перфорированием*. На рис. 9 показан общий вид устройства ввода информации с перфокарт.

Кроме перфокарт используются и перфоленты — узкие бумажные ленты, информация на которые наносится в виде круглых отверстий (рис. 10). Принцип ввода информации с перфолент такой же, как и с перфокарт. Информация на перфоленту наносится строками, расположенными поперек перфоленты. В каждую строку заносится комбинация отверстий, соответствующая отдельному символу. Процесс нанесения информации на перфоленту выполняется на устройстве, называемом *ленточным* перфоратором (рис. 11). С помощью устройства ввода (рис. 12) информация с перфоленты с большой скоростью переносится в ОЗУ вычислительной машины.

Рис. 11. Общий вид ленточного перфоратора

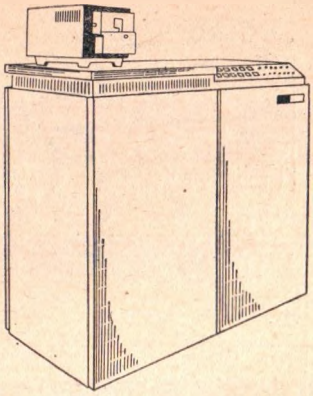


Рис. 12. Общий вид устройства ввода информации с перфоленты

Через специальные устройства вывода информация может быть также перенесена на перфоленты или перфокарты автоматически.

В последнее время широкое распространение получила подготовка исходной информации на магнитных носителях — картах и лентах. *Магнитная карта* — это пластиковый прямоугольник, на который нанесен слой магнитного вещества. По внешнему виду она напоминает перфокарту. При нанесении информации на магнитные карты, как и на магнитные ленты, намагничиваются соответствующие места носителя. Намагниченному месту соответствует символ 1, ненамагниченному — 0. Объем информации, который можно поместить на магнитную карту, в 100 раз превышает объем информации, вмещающейся на перфокарте.

Одним из главных преимуществ магнитных носителей перед перфоносителями есть то, что магнитные носители могут быть использованы для записи информации практически неограниченное число раз, в то время как перфоносители — только один раз.

Информация может как вводиться с магнитных лент и дисков, так и выводиться на них с помощью одних и тех же устройств ввода — вывода (рис. 13, 14).

Таким образом, обработка информации в ЭВМ происходит по следующей схеме:

1. С помощью устройства ввода исходные данные и наборы инструкций для обработки информации преобразуются в совокупность электрических сигналов, которые определенным образом воздействуют на память ЭВМ (т. е.

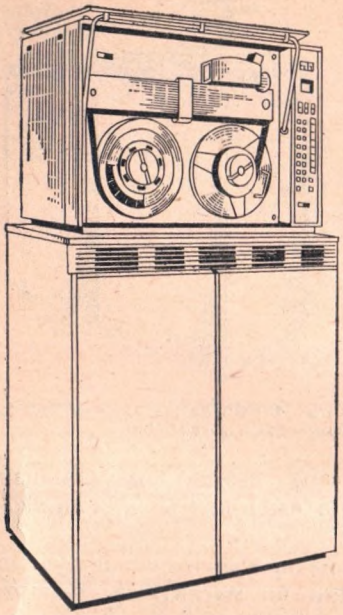
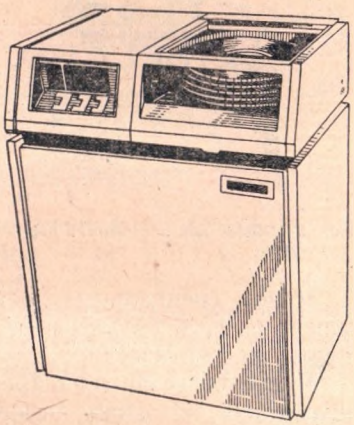


Рис. 13. Общий вид накопителя информации на магнитной ленте

Рис. 14. Общий вид накопителя информации со сменным пакетом магнитных дисков



происходит занесение исходной информации в память ЭВМ в некоторой специальной форме).

2. Из памяти извлекается порция управляющей информации (команда). Как правило, это содержимое одной ячейки. В этой информации закодирована исчерпывающая инструкция о том, какие действия должны выполнить устройства машины в ближайший отрезок времени.

3. Устройство управления расшифровывает управляющую информацию и в зависимости от ее значения выдает участвующим в операции устройствам ЭВМ приказы о конкретных действиях. Согласно этой информации УУ настраивает АЛУ на выполнение нужной операции, отыскивает в памяти числа, которые участвуют в данной операции, и пересылает их в АЛУ, отсылая результаты операций для дальнейшего хранения в память.

4. После того как устройства выполнят порученную им работу, они «сообщают» об этом УУ. Тогда либо из памяти ЭВМ извлекается следующая управляющая информация, либо движение информационных потоков в машине приостанавливается.

5. Далее полученные результаты с помощью устройств вывода представляются в требуемой форме.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Назовите основные устройства ЭВМ. 2. Нарисуйте структурную схему ЭВМ. 3. Для чего предназначен процессор? 4. Какое назначение АЛУ? 5. Какие функции выполняет УУ? 6. Для чего предназначено ЗУ? 7. Какое основное отличие внешних и внутренних ЗУ? 8. Каким образом реализуются внешние ЗУ? 9. Назовите основные типы внутренних ЗУ. 10. Для чего предназначено ОЗУ? 11. Что называется адресом числа, записанного в ОЗУ? 12. Для чего предназначено ПЗУ? 13. В чем заключается основное отличие использования ОЗУ от ПЗУ? 14. Чем отличается принцип использования ППЗУ от использования ПЗУ и ОЗУ? 15. Для чего предназначены УВВ информации? 16. Какие устройства позволяют работать в режиме диалога с ЭВМ? 17. Для чего предназначены перфокарты и перфоленты? 18. В каком виде информация наносится на перфоленты и перфокарты? 19. Какое преимущество магнитных носителей информации перед перфоносителями? 20. Опишите схему обработки информации в ЭВМ.

§ 4. АРИФМЕТИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Системы счисления. Совокупность средств для изображения и наименования чисел называется *системой счисления*. Существующие системы счисления подразделяются на непозиционные и позиционные.

В *непозиционных* системах смысл каждого символа (цифры) не зависит от позиции, занимаемой символом в изображении числа. Примером такой системы счисления есть римская система. Так, у числа XXX (тридцать), записанного в этой системе счисления, цифра X в любом месте обозначает десять единиц.

В *позиционных* системах счисления значение каждой цифры изменяется с изменением ее положения (позиции) в ряду цифр, изображающих число. Так, система записи десятичных чисел основана на том, что десять единиц каждого разряда объединяются в одну единицу соседнего, более старшего разряда. Например, в числе 5875 первая цифра (5) означает количество тысяч, вторая (8) — сотен, третья (7) — десятков, четвертая (5) — единиц.

В большинстве современных ЭВМ информация кодируется с помощью только двух символов — 0 и 1. Это связано с тем, что многие физические элементы, используемые при конструировании ЭВМ, имеют два четко выраженные и легко отличимые состояния (например, наличие или отсутствие тока в электрической цепи, повышение или понижение напряжения на выходе некоторого элемента относительно заданного уровня, попадание или непопадание света

на фотоэлемент). Одному из таких физических состояний ставится в соответствие символ 0, другому — символ 1.

С помощью только двух символов (0 и 1) изображаются также и числа при выполнении над ними арифметических операций. При этом числа представляются в так называемой *двоичной системе счисления*.

Суть представления числа N в двоичной системе счисления заключается в том, что число N раскладывают по степеням числа 2 (два), т. е. записывают в виде

$$N = \alpha_n \cdot 2^n + \alpha_{n-1} \cdot 2^{n-1} + \cdots + \alpha_2 \cdot 2^2 + \alpha_1 \cdot 2^1 + \\ + \alpha_0 \cdot 2^0 + \alpha_{-1} 2^{-1} + \alpha_{-2} \cdot 2^{-2} + \cdots$$

аналогично тому, как в обычной десятичной системе счисления число N раскладывают по степеням числа 10 (десять). При изображении числа записывают только коэффициенты разложения, отделяя коэффициенты при отрицательных степенях от коэффициентов при неотрицательных степенях запятой (или точкой). Часть изображения, содержащую коэффициенты α_{-1}, α_{-2}, α_{-3}, ..., называют *дробной частью* числа, а часть, содержащую коэффициенты α_0, α_1, α_2, ...,α_n — *целой частью* числа.

Коэффициенты разложения α_n, α_{n-1}, ..., α_0, α_{-1}, α_{-2}, ... должны быть неотрицательными и меньшими числа, по степеням которого раскладывается число N. В данном случае коэффициенты могут принимать только два значения: 0 и 1. Достичь этого можно всегда.

Действительно, пусть, например, разложение числа 15 по степеням 2 записано в виде

$$15 = 3 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0.$$

Но так как $3 = 2 + 1$, то число 15 можно представить в виде следующего разложения по степеням 2:

$$15 = (2 + 1) \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 1 \cdot 2^3 + 1 \cdot 2^2 + \\ + 1 \cdot 2^1 + 1 \cdot 2^0.$$

Любое число N можно разложить по степеням любого целого числа, большего единицы. Если изображение числа N получено с помощью разложения по степеням некоторого целого числа q, большего единицы, и если коэффициенты разложения изображаются с помощью q неделимых символов (называемых также *цифрами*), то говорят, что число N представлено в системе счисления с основанием q. Например, в системе счисления с основанием 2 коэффициенты изображаются с помощью символов 0 и 1; в системе счисления с основанием 16 — с помощью символов 0, 1, 2, 3, 4,

5, 6, 7, 8, 9, A, B, C, D, E, F, а в системе счисления с основанием 8 — с помощью символов 0, 1, 2, 3, 4, 5, 6, 7.

Однако изображение, например, 101 в двоичной системе счисления будет обозначать число $1 \cdot 2^2 + 0 \cdot 2^1 + 1 \times \times 2^0 = 5$ (пять), а в десятичной — число $1 \cdot 10^2 + 0 \times 10^1 + 1 \cdot 10^0 = 101$ (сто один). Поэтому, чтобы указать, в какой системе счисления изображено число, рядом с его изображением указывают основание системы счисления, например $101_{(2)}$, $101_{(10)}$.

Само основание двоичной системы счисления в данной системе изображается с помощью двух символов 10, что соответствует разложению

$$2 = 1 \cdot 2^1 + 0 \cdot 2^0,$$

т. е. $2_{(10)} = 10_{(2)}$.

Аналогично и основание q любой системы счисления в той же системе изображается с помощью двух символов 10, что соответствует разложению

$$q = 1 \cdot q^1 + 0 \cdot q^0.$$

Перевод изображений из одной системы счисления в другую. Переход от изображения числа N в десятичной системе счисления к его изображению в двоичной системе осуществляют в два этапа: последовательно находят изображения целой и дробной частей данного числа.

Пусть дано, например, целое число N_0 и пусть его разложение по степеням 2 имеет вид:

$$N_0 = \alpha_n \cdot 2^n + \alpha_{n-1} \cdot 2^{n-1} + \cdots + \alpha_2 \cdot 2^2 + \alpha_1 \cdot 2^1 + \alpha_0 \cdot 2^0.$$

Для отыскания изображения этого числа в двоичной системе счисления необходимо найти коэффициенты его разложения по степеням 2:

$$\alpha_n,\ \alpha_{n-1},\ \ldots,\ \alpha_2,\ \alpha_1,\ \alpha_0.$$

Поделив нацело число N_0 на 2, получим частное

$$N_1 = \alpha_n \cdot 2^{n-1} + \alpha_{n-1} \cdot 2^{n-2} + \cdots + \alpha_2 \cdot 2^1 + \alpha_1 \cdot 2^0$$

и остаток от деления

$$r_1 = \alpha_0,$$

так как $0 \leqslant \alpha_0 < 2$. Поделив далее нацело N_1 на 2, получим частное

$$N_2 = \alpha_n \cdot 2^{n-2} + \alpha_{n-1} \cdot 2^{n-3} + \cdots + \alpha_2 \cdot 2^0$$

и остаток от деления

$$r_2 = \alpha_1.$$

Продолжив далее этот процесс, после каждого очередного деления будем получать остаток, который и будет коэффициентом при следующей степени 2, и через $(n + 1)$ шагов получим все коэффициенты $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_n$.

Теперь для изображения числа N_0 в двоичной системе счисления достаточно выписать последовательно коэффициенты разложения справа налево в порядке их нахождения, т. е. $\alpha_n\ \alpha_{n-1}\ \alpha_{n-2}\ \ldots\ \alpha_1\ \alpha_0$. Выписанный набор коэффициентов и будет изображением целого числа N_0 в двоичной системе счисления.

Пример. Найти разложение 9 по степеням 2.

Поделив нацело 9 на 2, получим частное 4 и остаток 1. Поделив далее нацело 4 на 2, найдем частное 2 и остаток 0. После деления очередного частного 2 на 2, получим частное 1 и остаток 0, а после деления 1 на 2 — частное 0 и остаток 1.

Так как последнее частное от деления равно нулю, то продолжать далее процесс деления не имеет смысла: все последующие частные и остатки от деления будут нулями.

Описанные вычисления удобно записать так:

$$\begin{array}{rl|l} 9 & & 2 \\ \hline 8 & 4 & 2 \\ \hline r_1 = 1 & 4 & 2 \\ \hline & r_2 = 0 & 2 \\ \hline & & r_3 = 0 \end{array} \qquad \begin{array}{rl|l} & & 2 \\ \hline & 2 & 2 \\ \hline & 1 & 2 \\ \hline & 0 & 0 \\ \hline & r_4 = 1 & \end{array}$$

Записывая теперь остатки от деления справа налево в порядке их получения, будем иметь двоичное изображение числа 9:

$$9_{(10)} = 1001_{(2)}.$$

Рассмотрим, как изображаются первые числа натурального ряда в двоичной системе счисления:

$0 = 0 \cdot 2^0$	$0_{(10)} = 0_{(2)}$
$1 = 1 \cdot 2^0$	$1_{(10)} = 1_{(2)}$
$2 = 1 \cdot 2^1 + 0 \cdot 2^0$	$2_{(10)} = 10_{(2)}$
$3 = 1 \cdot 2^1 + 1 \cdot 2^0$	$3_{(10)} = 11_{(2)}$
$4 = 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$	$4_{(10)} = 100_{(2)}$
$5 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$	$5_{(10)} = 101_{(2)}$
$6 = 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$	$6_{(10)} = 110_{(2)}$
$7 = 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$	$7_{(10)} = 111_{(2)}$
$8 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$	$8_{(10)} = 1000_{(2)}$
$9 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$	$9_{(10)} = 1001_{(2)}$

$10 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \quad 10_{(10)} = 1010_{(2)}$

$11 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \quad 11_{(10)} = 1011_{(2)}$

$12 = 1 \cdot 2^2 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 \quad 12_{(10)} = 1100_{(2)}$

$13 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \quad 13_{(10)} = 1101_{(2)}$

$14 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \quad 14_{(10)} = 1110_{(2)}$

$15 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \quad 15_{(10)} = 1111_{(2)}$

Пусть теперь дано дробное число M_0 и пусть его разложение по степеням 2 имеет вид:

$$M_0 = \alpha_{-1} \cdot 2^{-1} + \alpha_{-2} \cdot 2^{-2} + \alpha_{-3} \cdot 2^{-3} + \dots .$$

Для того чтобы записать данное число M_0 в двоичной системе счисления, необходимо найти коэффициенты разложения α_{-1}, α_{-2}, ... Умножив M_0 на 2, получим

$$M_0 \cdot 2 = \alpha_{-1} \cdot 2^0 + \alpha_{-2} \cdot 2^{-1} + \alpha_{-3} \cdot 2^{-2} + \dots ,$$

причем $\alpha_{-1} \cdot 2^0 = \alpha_{-1}$ будет целой частью произведения, а в дробной части останется

$$M_1 = \alpha_{-2} \cdot 2^{-1} + \alpha_{-3} \cdot 2^{-2} + \dots .$$

Умножив теперь на 2 дробную часть M_1 полученного произведения, в целой части произведения $M_1 \cdot 2 = \alpha_{-2} \times \times 2^0 + \alpha_{-3} \cdot 2^{-1} + \dots$ найдем коэффициент α_{-2}, а в дробной части этого произведения будет

$$M_2 = \alpha_{-3} \cdot 2^{-1} + \dots .$$

Продолжая этот процесс далее, можно получить необходимое количество коэффициентов разложения дробного числа M_0 по отрицательным степеням 2.

Теперь для изображения дробного числа M_0 в двоичной системе счисления достаточно выписать последовательно вправо от запятой коэффициенты разложения в порядке их нахождения, записав в целой части 0, т. е. записать $0, \alpha_{-1}\alpha_{-2}\alpha_{-3}\dots$.

Пример. Найти разложение 0,125 по степеням 2.

Умножив 0,125 на 2, получим целую часть произведения 0 и дробную часть 0,250. Умножив далее дробную часть 0,250 на 2, будем иметь в целой части 0 и новую дробную часть 0,500. После очередного умножения в целой части произведения получим 1, а в дробной — 0. Так как последняя дробная часть равна 0, то нет необходимости продолжать этот процесс. Описанные вычисления удобно записать так:

$0,$	$\times \frac{125}{2}$
$(\alpha_{-1}=)\ 0,$	$\times \frac{250}{2}$
$(\alpha_{-2}=)\ 0,$	$\times \frac{500}{2}$
$(\alpha_{-3}=)\ 1,$	000

Таким образом,

$$0{,}125_{(10)} = 0{,}001_{(2)} = 0 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}.$$

Процесс перевода дробных чисел, в отличие от процесса перевода целых чисел, иногда может оказаться бесконечным.

Пример. Найти изображение числа в двоичной системе счисления по его изображению 0,1 в десятичной системе.

Имеем:

$0,$	$\times \frac{1}{2}$
$(\alpha_{-1}=)\ 0,$	$\times \frac{2}{2}$
$(\alpha_{-2}=)\ 0,$	$\times \frac{4}{2}$
$(\alpha_{-3}=)\ 0,$	$\times \frac{8}{2}$
$(\alpha_{-4}=)\ 1,$	$\times \frac{6}{2}$
$(\alpha_{-5}=)\ 1,$	$\times \frac{2}{2}$
$(\alpha_{-6}=)\ 0,$	$\times \frac{4}{2}$
$(\alpha_{-7}=)\ 0,$	$\times \frac{8}{2}$
$(\alpha_{-8}=)\ 1,$	6

т. е. 0,1 имеет бесконечное изображение:

$$0{,}1_{(10)} = 0{,}00011001\ldots_{(2)}.$$

Упражнения. Найти изображения чисел в двоичной системе счисления по их изображениям 0,2; 0,3; 0,4; 0,5; 0,625; 0,725; 0,85; 0,9; 0,75 в десятичной системе.

Правила перевода изображения произвольного числа N из десятичной системы счисления в двоичную следующие:

1. Находят коэффициенты разложения целой части числа N.
2. Определяют коэффициенты разложения дробной части числа N.
3. Полученные коэффициенты записывают влево и вправо от запятой в том порядке, в каком они найдены.

Пример. Найти изображение числа в двоичной системе счисления по его изображению 6,25 в десятичной системе.

По правилу 1 имеем:

$$\begin{array}{r|l} 6 & 2 \\ \hline 6 & 3 \;\big|\; 2 \\ \hline r_1 = 0 & 2 \quad 1 \;\big|\; 2 \\ \hline & r_2 = 1 \quad 0 \quad 0 \\ \hline & \quad r_3 = 1 \end{array}$$

Следовательно,

$$6_{(10)} = 110_{(2)}.$$

По правилу 2 находим:

$$\begin{array}{r|l} 0,\ & \times\begin{matrix}25\\2\end{matrix} \\ \hline (\alpha_{-1} =)\ 0,\ & \times\begin{matrix}50\\2\end{matrix} \\ \hline (\alpha_{-2} =)\ 1,\ & 00 \end{array}$$

Следовательно,

$$0{,}25_{(10)} = 0{,}01_{(2)}$$

и окончательно

$$6{,}25_{(10)} = 110{,}01_{(2)}.$$

Пример. Найти изображение числа в двоичной системе счисления по его изображению 16,225 в десятичной системе.

По правилу 1 имеем:

$$\begin{array}{r|l} 16 & 2 \\ \hline 16 & 8 \;\big|\; 2 \\ \hline r_1 = 0 & 8 \quad 4 \;\big|\; 2 \\ \hline & r_2 = 0 \quad 4 \quad 2 \;\big|\; 2 \\ \hline & \quad r_3 = 0 \quad 2 \quad 1 \;\big|\; 2 \\ \hline & \qquad r_4 = 0 \quad 0 \quad 0 \\ \hline & \qquad\quad r_5 = 1 \end{array}$$

Следовательно,

$$16_{(10)} = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0,$$

т. е.

$$16_{(10)} = 10000_{(2)}.$$

По правилу 2 находим:

$0,$	$\times \overset{225}{2}$
$(\alpha_{-1} =)\ 0,$	$\times \overset{450}{2}$
$(\alpha_{-2} =)\ 0,$	$\times \overset{900}{2}$
$(\alpha_{-3} =)\ 1,$	$\times \overset{800}{2}$
$(\alpha_{-4} =)\ 1,$	$\times \overset{600}{2}$
$(\alpha_{-5} =)\ 1,$	$\times \overset{200}{2}$
$(\alpha_{-6} =)\ 0,$	$\times \overset{400}{2}$
$(\alpha_{-7} =)\ 0,$	800

т. е. получаем бесконечное изображение:

$$0{,}225_{(10)} = 0{,}0011100\ldots_{(2)}.$$

Таким образом

$$16{,}225_{(10)} = 10000{,}0011100\ldots_{(2)}.$$

Упражнения. Найти изображения чисел в двоичной системе счисления по их изображениям 12,6; 8,25; 27; 0,64; 2,34; 7,7 в десятичной системе.

Чтобы перейти от изображения числа в двоичной системе счисления к его изображению в десятичной системе, достаточно в последней записать разложение по степеням 2 и выполнить необходимые операции.

Пример. Пусть изображение некоторого числа в двоичной системе счисления имеет вид 1101. Тогда разложение этого же числа по степеням 2 в десятичной системе будет

$$1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 4 + 1 = 13,$$

т. е.

$$1101_{(2)} = 13_{(10)}.$$

Пример. Пусть изображение некоторого числа в двоичной системе счисления имеет вид 0,101. Тогда разложение этого же числа по степеням 2 в десятичной системе будет

$$1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = \frac{1}{2} + \frac{0}{4} + \frac{1}{8} = 0{,}500 + 0{,}125 = 0{,}625,$$

так что

$$0{,}101_{(2)} = 0{,}625_{(10)}.$$

Пример. Пусть изображение некоторого числа в двоичной системе счисления имеет вид 1011,10101. Тогда разложение этого же числа по степеням 2 в десятичной системе будет

$$1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} +$$
$$+ 0 \cdot 2^{-4} + 1 \cdot 2^{-5} = 8 + 0 + 2 + 1 + \frac{1}{2} + 0 + \frac{1}{8} + 0 +$$
$$+ \frac{1}{32} = 11 + 0{,}500 + 0{,}125 + 0{,}03125 = 11{,}65625,$$

т. е.

$$1011{,}10101_{(2)} = 11{,}65625_{(10)}.$$

Упражнения. Найти изображения чисел в десятичной системе счисления, если их изображения в двоичной системе имеют вид 1111; 1010; 110,01; 1101,1101; 111,111; 0,11; 0,001; 11,011; 101,101; 1100,0011.

Арифметические операции в двоичной системе счисления. Арифметические операции над числами в каждой системе счисления выполняются по соответствующим правилам сложения, вычитания, умножения и деления.

Правила сложения и умножения одноразрядных двоичных чисел имеют вид:

$$0 + 0 = 0;\ 0 + 1 = 1;\ 1 + 0 = 1;\ 1 + 1 = 10;$$
$$0 \cdot 0 = 0;\ 0 \cdot 1 = 0;\ 1 \cdot 0 = 0;\ 1 \cdot 1 = 1.$$

Сложение многоразрядных чисел есть, по сути, приведение подобных членов в сумме разложений этих чисел по степеням 2. Операцию выполняют непосредственно над изображениями складываемых чисел, подписывая их одно под другим аналогично тому, как это делается в десятичной системе счисления.

Пример. Сложить двоичные числа 1101,11 и 11,101. Имеем:

$$\begin{array}{r} 11111 \\ 1101{,}11 \\ +\ \ 11{,}101 \\ \hline 10001{,}011 \end{array}$$

Над первым слагаемым указаны единицы переноса, возникающие при сложении чисел в младших разрядах.

Вычитание многоразрядных чисел также сводится к приведению подобных членов в разложениях уменьшаемого и вычитаемого по степеням 2. Выписывать разложения при вычитании чисел нет необходимости. Операции выполняют непосредственно над изображениями чисел, подписывая их одно под другим аналогично тому, как это делается при вычитании чисел в десятичной системе счисления.

Если в некотором разряде приходится вычитать единицу из нуля, то занимается единица из следующего, старшего разряда. При этом надо помнить, что занимаемая единица соседнего (старшего) разряда равна двум единицам данного разряда.

Пример. Системы счисления:

	двоичная	*десятичная*
Перенос	11 1	11
Уменьшаемое	11001011	303
Вычитаемое	1100101	97
Разность	1100110	206

Если при вычитании двух положительных чисел вычитаемое больше уменьшаемого, то в результате получается отрицательное число. При этом из большего числа вычитают меньшее и результат записывают со знаком «—» (минус).

И в десятичной, и в двоичной системах счисления вычитание можно свести к сложению, используя так называемые *дополнения до единицы* разряда, старшего, чем наиболее старший разряд в уменьшаемом и вычитаемом.

Пусть, например, в десятичной системе счисления необходимо выполнить вычитание $87 - 59$. Так как $100 - 59 = 41$ — дополнение 59 до 100, то эту разность можно представить в виде

$$87 - (100 - 41) = 87 + 41 - 100 = 128 - 100 = 28,$$

т. е. вместо разности $87 - 59$ можно найти сумму $87 + 41$ и в результате отбросить единицу разряда, до которого взято дополнение.

Рассмотрим другой пример:

$$23 - 76 = 23 - (100 - 24) = 23 + 24 - 100 = \\ = 047 - 100 = -53.$$

Если использовать только дополнения и вычитание числа заменить прибавлением его дополнения, то получим

$$87 + 41 = 128;\ 23 + 24 = 047.$$

При этом, если в результате получается положительное (отрицательное) число, то в разряде, до которого взято дополнение, будет 1 (0). После сложения получим дополнение искомого результата до единицы разряда, до которого взято дополнение.

Таким образом, вместо того, чтобы вычитать число, можно прибавить его дополнение, после чего в результате количество единиц в разряде, до которого взято дополнение, следует уменьшить на 1.

Аналогично вычитание можно заменить сложением и в двоичной системе счисления, если вычитаемое заменить его дополнением, для нахождения которого необходимо в изображении числа все нули заменить единицами, а единицы — нулями и к полученному числу добавить 1.

Пусть, например, дано число 11011001. Заменяем в его изображении нули единицами, а единицы — нулями: 00100110. Добавив теперь к полученному числу 1, получим:

$$\begin{array}{r} 00100110 \\ +\quad\quad 1 \\ \hline 00100111 \end{array}$$

Число 00100111 является дополнением заданного числа 11011001 до единицы 9-го разряда.

Пример. Найти разность чисел 11000111 и 10101011 с использованием дополнения в двоичной системе счисления.

Находим дополнение вычитаемого 10101011:

$$\begin{array}{r} 01010100 \\ +\quad\quad 1 \\ \hline 01010101 \end{array}$$

(дополнение взято до 1 в 9-м разряде).

Складываем уменьшаемое и дополнение вычитаемого:

$$\begin{array}{r} 11000111 \\ +\,01010101 \\ \hline 100011100. \end{array}$$

Отбросив 1 в старшем (9-м) разряде, получим

00011100.

Проверим полученный результат, выполнив непосредственное вычитание заданных чисел:

$$\begin{array}{r} 11000111 \\ -\,10101011 \\ \hline 00011100 \end{array}$$

Как видно, результаты совпадают.

Упражнения. В двоичной системе счисления выполнить операции: 1011 — 110; 1101,11 + 111,1; 1010,10— 1100,11; 111,11 + + 11,01. Разность чисел найти двумя способами: непосредственно и с использованием дополнений.

Умножение и деление многоразрядных чисел в двоичной системе счисления выполняются аналогично тому, как это делается в десятичной системе.

Пример. Найти произведения чисел: 1) 101 и 110; 2) 110,11 и 10,01.

Имеем:

$$\begin{array}{r} 101 \\ \times\ 110 \\ \hline 000 \\ +\ 101 \\ 101 \\ \hline 11110 \end{array} \qquad \begin{array}{r} 110{,}11 \\ \times\ 10{,}01 \\ \hline 11011 \\ +\ 00000 \\ 00000 \\ 11011 \\ \hline 1111{,}0011 \end{array}$$

Пример. Найти частное от деления числа $1101{,}11_{(2)}$ на число $111_{(2)}$.

Получаем:

$$\begin{array}{r|l} -\ 1101{,}11 & 111 \\ \hline -\ 111 & 1{,}111\ldots \\ \hline -\ 1101 & \\ \quad 111 & \\ \hline -\ 1101 & \\ \quad 111 & \\ \hline -\ 1100 & \\ \quad 111 & \\ \hline 11 & \end{array}$$

т. е. частное от деления данных чисел равно 1,111... .

Упражнения. 1. Найти произведения чисел в двоичной системе счисления: 11,11 и 100,1; 101 и 100; 11,01 и 110,11.

2. Найти частное от деления чисел в двоичной системе счисления: 11,01 на 11010; 111,011 на 1010; 1111 на 11.

Таким образом, умножение чисел сводится к последовательному сдвигу одного из сомножителей и прибавлению полученного в результате сдвига числа к ранее накопленной сумме.

Так как вычитание представляет собой сложение с дополнением, то деление чисел также сводится к их последовательному сдвигу и сложению.

Следовательно, для выполнения операций сложения, вычитания, умножения и деления чисел достаточно иметь устройство, осуществляющее последовательный сдвиг одного из чисел и сложение чисел.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Что называется системой счисления? 2. В чем заключается основное отличие позиционных и непозиционных систем счисления? 3. Сколько символов используется для изображения чисел в двоичной системе счисления? 4. В чем заключается суть представления числа в двоичной системе счисления? 5. Какое изображение имеет основание системы счисления? 6. Как можно найти изображение целого числа в двоичной системе счисления по его изображению в десятичной системе? 7. Как можно найти изображение дробного числа в двоичной системе счисления по его изображению в десятичной системе? 8. Как складывают числа в двоичной системе счисления? 9. Как вычитают числа в двоичной системе счисления? 10. Как заменить вычитание сложением, используя дополнение вычитаемого? 11. Как выполняют умножение и деление чисел в двоичной системе счисления?

§ 5. ЛОГИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Переменные величины, которые могут принимать только два значения (0 или 1), называются *двоичными*, *логическими* или *булевыми* переменными. Логические переменные будем обозначать большими буквами латинского алфавита: X, Y, Z и т. д.

Введем операции над логическими переменными, которые будем называть *логическими операциями*.

1. Операция логического отрицания переменной X ставит в соответствие новую переменную, обозначаемую $\overline{X}$ и принимающую значение 0, если $X = 1$, и значение 1, если $X = 0$. $\overline{X}$ называется *отрицанием* X.

Операцию логического отрицания можно задать таблицей:

X	$\overline{X}$
0	1
1	0

Эту операцию называют также *инверсией* (или *логическим* «НЕ»).

2. Операция логического умножения ставит в соответствие двум логическим переменным X и Y новую логическую переменную W, принимающую значение 1 только в том случае, если значения обеих переменных X и Y равны единице. При этом пишут $W = X \wedge Y$ или $W = X \,\&\, Y$ (читают «X и Y»). Логическое умножение называется также *конъюнкцией* (или *логическим* «И»).

Операцию логического умножения можно задать таблицей:

X	Y	$X \wedge Y$
0	0	0
0	1	0
1	0	0
1	1	1

3. Операция логического сложения ставит в соответствие двум логическим переменным X и Y новую логическую переменную Z, принимающую значение 0 только в том случае, если значения обеих переменных X и Y равны нулю. При этом пишут $Z = X \vee Y$ (читают «X или Y»). Логическое сложение называется также *дизъюнкцией* (или *логическим* «ИЛИ»).

Операцию логического сложения можно задать таблицей:

X	Y	$X \vee Y$
0	0	0
0	1	1
1	0	1
1	1	1

Можно ввести и другие логические операции (их всего 16). Однако рассмотренных операций достаточно для того, чтобы выразить через них любую логическую функцию двух логических переменных. *Логической* (или *булевой*) *функцией* логических переменных (аргументов) X, Y, Z, ... называют переменную величину, принимающую только два значения (0 или 1) в зависимости от того, какие значения принимают переменные X,Y,Z, ... Таким образом, каждому возможному набору значений аргументов $X,Y,Z,...$ ставится в соответствие одно из значений: 0 или 1.

Логическую функцию можно задать с помощью таблицы, в которой будут перечислены все возможные наборы аргументов и соответствующие значения функции. Например, все возможные логические функции двух аргументов X и Y отражает таблица:

X	Y	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Логическую функцию можно задать также логической формулой, объединяющей логические переменные знаками логических операций. Например, $f(X,Y) = X \wedge Y \vee \overline{X} \wedge \overline{Y}$. Чтобы указать желаемый порядок выполнения логических операций, в формуле (выражении) используют круглые скобки. Например, $f(X, Y) = ((X \wedge Y) \vee (\overline{X} \vee \overline{Y})) \wedge \overline{Y}$. Если скобки отсутствуют, то операции выполняются в следующем порядке: 1) отрицание; 2) конъюнкция; 3) дизъюнкция.

Пример. Вычислить значение функции $f(X, Y)$, заданной выражением $f(X, Y) = \overline{X} \vee Y \wedge X \vee \overline{Y} \vee X \wedge Y$, если $X = 0$, $Y = 0$.

Для вычисления значения заданной функции $f(X, Y)$ при $X = 0$, $Y = 0$ выполняем вначале операции логического отрицания:

$$1)\ \overline{X} = 1;\ 2)\ \overline{Y} = 1,$$

затем — конъюнкции:

$$3)\ Y \wedge X = 0;\ 4)\ X \wedge Y = 0,$$

после этого — дизъюнкции:

$$5)\ \overline{X} \vee Y \wedge X,\ 6)\ \overline{X} \vee Y \wedge X \vee \overline{Y},$$

т. е.

$\overline{X}$	$Y \wedge X$	$\overline{X} \vee Y \wedge X$
1	0	1

,

т. е.

$\overline{X} \vee Y \wedge X$	$\overline{Y}$	$\overline{X} \vee Y \wedge X \vee \overline{Y}$
1	1	1

,

откуда

$$\overline{X} \vee Y \wedge X = 1;$$

откуда

$$\overline{X} \vee Y \wedge X \vee \overline{Y} = 1$$

и, наконец,

$$7)\ \overline{X} \vee Y \wedge X \vee \overline{Y} \vee X \wedge Y,$$

т. е.

$\overline{X} \vee Y \wedge X \vee \overline{Y}$	$X \wedge Y$	$\overline{X} \vee Y \wedge X \vee \overline{Y} \vee X \wedge Y$
1	0	1

.

Следовательно, при $X = 0$, $Y = 0$

$$f(X, Y) = \overline{X} \vee Y \wedge X \vee \overline{Y} \vee X \wedge Y = 1.$$

Упражнения. Вычислить значения функции $f(X, Y)$, заданной выражением $f(X, Y) = \overline{X} \vee Y \wedge X \vee \overline{Y} \vee X \wedge Y$, если: а) $X = 0$, $Y = 1$; б) $X = 1$, $Y = 0$; в) $X = 1$, $Y = 1$.

Любую функцию, заданную таблицей значений, можно представить в виде некоторого логического выражения, содержащего только операции сложения, умножения и отрицания.

Пусть булева функция $f(X, Y, Z)$ трех аргументов X,Y,Z задана таблицей:

X	Y	Z	$f(X, Y, Z)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Необходимо представить данную функцию $f(X,Y,Z)$ в виде некоторого логического выражения.

Возьмем некоторый набор значений аргументов, на котором функция принимает значение 1, например набор 1,0,0. Составим конъюнкцию независимых переменных, поставив знак отрицания над теми из них, значения которых в рассматриваемом наборе равны нулю. В данном примере получим логическое выражение $(X \wedge \bar{Y} \wedge \bar{Z})$, принимающее значение 1 только на наборе значений аргументов 1,0,0 и значение 0 на любом другом наборе значений аргументов. Такие конъюнкции составим для всех наборов, на которых заданная функция принимает значения 1. В рассматриваемом примере это будут конъюнкции $(\bar{X} \wedge \bar{Y} \wedge \bar{Z})$, $(X \wedge \bar{Y} \wedge \bar{Z})$, $(X \wedge Y \wedge \bar{Z})$. Составим теперь дизъюнкцию полученных конъюнкций. В данном примере имеем:

$$(\bar{X} \wedge \bar{Y} \wedge \bar{Z}) \vee (X \wedge \bar{Y} \wedge \bar{Z}) \vee (X \wedge Y \wedge \bar{Z}).$$

Это выражение принимает значение 1 на трех наборах (0,0,0), (1,0,0) и (1,1,0), так как на каждом из них один из членов дизъюнкции равен единице, а остальные — нулю.

Если один из членов дизъюнкции равен единице, то и все выражение принимает значение 1. На любом наборе значений аргументов, отличающемся от указанных, все три конъюнкции, входящие в рассматриваемое выражение, равны нулю, но тогда и все выражение принимает значение 0.

Таким образом, функцию, заданную таблично в рассмотренном примере, можно представить в виде

$$f(X, Y, Z) = (\bar{X} \wedge \bar{Y} \wedge \bar{Z}) \vee \\ \vee (X \wedge \bar{Y} \wedge \bar{Z}) \vee (X \wedge Y \wedge \bar{Z}).$$

Аналогично можно представить и любую другую таблично заданную функцию, отличающуюся от тождественного нуля.

Выражение таблично заданной функции можно получить и другим способом, а именно: рассматривают наборы аргументов, на которых функция принимает значения 0. Для каждого такого набора составляют дизъюнкцию, в которую включают все элементы, взяв со знаком отрицания те из них, которые в данном наборе имеют значения 1. Затем берут конъюнкцию всех образованных дизъюнкций.

Например, для функции f_2, заданной таблицей

X	Y	f_2
0	0	0
0	1	1
1	0	0
1	1	1,

получаем

$$f_2(X, Y) = (X \vee Y) \wedge (\bar{X} \vee Y),$$

а для функции f_6, заданной таблицей

X	Y	f_6
0	0	0
0	1	1
1	0	1
1	1	1,

имеем:

$$f_6(X, Y) = (X \vee Y).$$

Упражнения. Найти аналитическое представление таблично заданных функций:

X	Y	f_1	f_2	f_3	f_4	f_5	f_6
0	0	0	0	1	1	0	0
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	1
1	1	0	1	0	1	1	1

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Какие величины называются логическими переменными? **2.** Как определяется операция отрицания? **3.** Как определяется операция конъюнкции? **4.** Как определяется операция дизъюнкции? **5.** Что называется логической функцией? **6.** Как задают логические функции? **7.** В каком порядке выполняются операции в логических выражениях? **8.** Каким способом можно представить в виде логического выражения таблично заданную функцию?

§ 6. ЭЛЕМЕНТЫ СХЕМОТЕХНИКИ ЭВМ

Существуют различные физические устройства, используя которые можно реализовать булевы логические функции отрицания, дизъюнкции и конъюнкции. Такими устройствами могут быть: электромеханические реле, электронные лампы, полупроводниковые приборы — диоды, транзисторы, интегральные микросхемы.

Рассмотрим, например, электромеханическое реле (рис. 15), состоящее из обмотки *1*, сердечника *2*, якоря *3* и двух групп контактов: нормально замкнутых (размыкающих) *4* и нормально разомкнутых (замыкающих) *5*. При наличии тока в обмотке якорь притягивается к сердечнику, замыкая при этом контакты *5* и размыкая контакты *4*. Если же ток в обмотке якоря отсутствует, то якорь оттягивается от сердечника (под действием, например, пружины), размыкая контакты *5* и замыкая контакты *4*. Устройство, замыкающее или размыкающее электрическую цепь, называется к л ю ч о м. Если по обмотке реле ток идет, то через нормально замкнутые контакты ток не идет; если же по обмотке ток не идет, то через нормально замкнутые контакты ток идет. Наличию тока поставим в соответствие 1, отсутствию — 0. Таким образом, с помощью нормально замкнутого контакта можно осуществить операцию логического отрицания.

На вход схемы (рис. 16) подается сигнал X, а на выходе образуется сигнал $\overline{X}$. Схема, реализующая операцию отрицания, называется *схемой* «НЕ» (или и н в е р т о р о м).

Конъюнкция двух булевых переменных X и Y реализуется последовательным включением двух нормально разомкнутых контактов реле, соответствующих этим переменным (рис. 17). На выходе рассматриваемой схемы сигнал будет только тогда, когда на оба входа X и Y поступают сигналы (идет ток). Схема, реализующая операцию

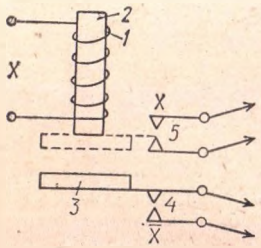


Рис. 15. Электромеханическое реле

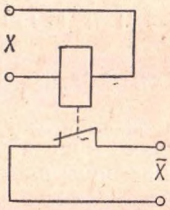


Рис. 16. Схема «НЕ»

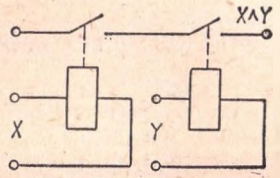


Рис. 17. Схема «X∧Y»

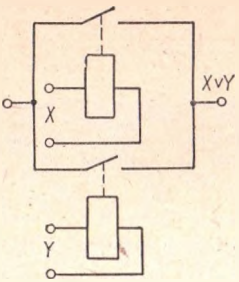
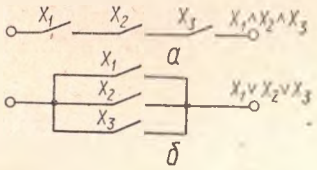


Рис. 18. Схема «XVY»

Рис. 19. Схемы «И» (*а*) и «ИЛИ» (*б*) на три входа



конъюнкции, называется *схемой* «И» (или схемой совпадения).

Дизъюнкция двух булевых переменных реализуется параллельным включением нормально разомкнутых контактов реле, соответствующих этим переменным (рис. 18). На выходе этой схемы сигнал будет тогда, если он будет подан, по крайней мере, на один из ее входов (X и Y). Схема, реализующая операцию дизъюнкции, называется *схемой* «ИЛИ».

Логические элементы «И», «ИЛИ» могут иметь больше двух входов, на которые подаются значения булевых переменных (значения сигналов). Выход у каждого логического элемента один (рис. 19). С помощью устройств, реализующих простейшие логические функции отрицания, дизъюнкции и конъюнкции, можно построить устройство, реализующее любую логическую функцию.

Пусть, например, необходимо составить схему устройства, определяющего знак произведения и частного двух чисел, если знак «+» обозначен через 0, знак «—» — через 1. Работа такого устройства описывается функцией, представленной таблицей:

X	Y	$f(X, Y)$
0	0	0
0	1	1
1	0	1
1	1	0

Составив логическое выражение $f(X, Y)$, получим

$$f(X, Y) = (\overline{X} \wedge Y) \vee (X \wedge \overline{Y}).$$

Если логические операции отрицания, конъюнкции и дизъюнкции изобразить схематически так, как показано на рис. 20, то последовательность операций, которые необ-

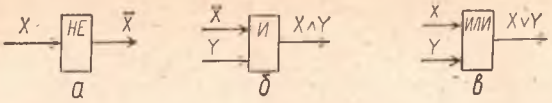


Рис. 20. Схематическое изображение логических элементов «НЕ» (*а*), «И» (*б*) и «ИЛИ» (*в*)

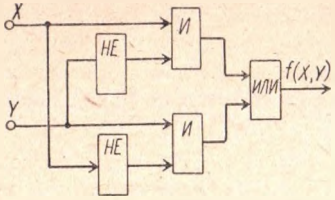


Рис. 21. Схема, определяющая знак произведения и частного двух чисел

ходимо выполнить для вычисления значения рассмотренной функции при заданных значениях аргументов X и Y, схематически можно представить так, как показано на рис. 21. Поскольку значения $X \wedge Y$ и $X \wedge Y$ можно вычислять в любом порядке, в принципе они могут быть вычислены и одновременно.

Из физических элементов, реализующих логические функции «И», «ИЛИ», «НЕ», можно построить сколь угодно сложную вычислительную машину или устройства другого назначения, выполняющие переработку информации, поступающей на их входы, согласно наперед заданным правилам.

Первая автоматическая цифровая вычислительная машина была построена на механических и релейно-контактных элементах. В 1946 г. была собрана вычислительная машина, в которой логические элементы выполнялись на электронных лампах. Ламповые ЭВМ относятся к ЭВМ *первого* поколения.

В конце 50-х — начале 60-х годов ламповые машины первого поколения вытесняются ЭВМ *второго* поколения, в которых логические элементы выполняются на полупроводниках. По сравнению с релейно-контактными электронные схемы обладают намного бóльшим быстродействием. При частом и быстром переключении из одного устойчивого состояния в другое механические ключи значительно уступают электронным по максимально возможной частоте переключений, надежности контакта, сроку службы.

При конструировании современных ЭВМ используют схемы, составленные из совокупностей полупроводниковых элементов — диодов, конденсаторов, резисторов, транзисторов и других деталей, соединенных между собой в определенном порядке на печатной плате металлическими про-

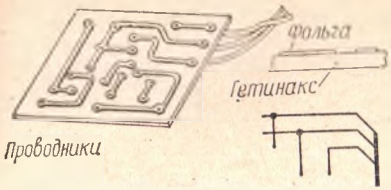


Рис. 22. Печатная плата

Рис. 23. Резисторы (*а*) и их символическое обозначение на схемах (*б*)

водниками (рис. 22). Соединительные проводники используются для беспрепятственного пропускания тока из одного элемента схемы в другой.

Резисторы (рис. 23) применяют для ослабления пропускаемого тока в некоторое число раз. Конденсаторы (рис. 24) могут быть использованы по нескольким назначениям: для накапливания и хранения электрических зарядов, для отделения импульсного тока от постоянного, для отделения коротких импульсов от длительных. Вместе с резистором конденсатор может применяться в качестве своего рода эталона времени: чем больше емкость конденсатора, тем дольше он разряжается через резистор.

В чистом виде полупроводники и полупроводниковые слои используются только как разного рода резисторы и изоляторы. Но небольшое количество специальных примесей вызывает появление в кристаллах полупроводникового вещества свободных носителей зарядов, которые могут перемещаться под действием электрического поля, т. е. создают проводимость. На рис. 25 показаны упрощенная структура полупроводникового диода и его символическое обозначение.

Диод, как и проводник, легко пропускает ток, но только в одном направлении. Его можно рассматривать как выключатель, но с учетом направления тока: если ток идет в одном направлении — выключатель замкнут, если в обратном — разомкнут.

Транзистор открывается и пропускает ток под действием слабого управляющего сигнала. А когда управляющий сигнал исчезает, транзистор вновь закрывается и ведет себя уже как разомкнутый выключатель, т. е. тока не пропускает.

Транзистор имеет эмиттер, базу, коллектор. Эмиттер является источником носителей заряда (электронов), база — управляющим электродом. Сигнал на входе

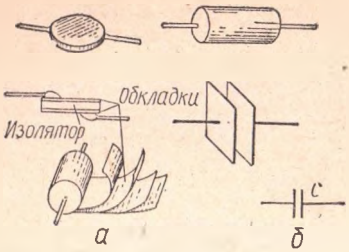


Рис. 24. Конденсаторы (*а*) и их символическое обозначение на схемах (*б*)

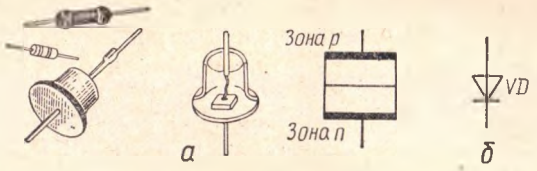


Рис. 25. Полупроводниковые диоды (*а*) и их символическое обозначение на схемах (*б*)

базы воздействует на ток, протекающий через транзистор. Коллектор собирает носители заряда, порождаемые эмиттером. Устройство транзисторов и их символические обозначения показаны на рис. 26.

На рис. 27, *а* изображена схема простейшего *транзисторного* ключа. В управляющей (базовой) цепи включен источник управляющего напряжения X. Если напряжение X мало, то транзистор заперт и ток в цепи очень мал. Соответственно напряжение Y большое. Если напряжение X достаточно велико, то транзистор открыт, в цепи протекает ток и напряжение Y близко к нулю.

Из этого следует, что такой ключ является инвертирующей схемой, т. е. схемой «НЕ», реализующей операцию отрицания (рис. 27, *б*), поскольку увеличение входного напряжения X сопровождается уменьшением выходного напряжения Y и, наоборот, уменьшение входного напряжения X вызывает увеличение выходного напряжения Y. Символическое обозначение транзисторного ключа и его физический аналог показаны на рис. 27, *в* и *г* соответственно.

На рис. 28, 29 изображены схемы «И», «ИЛИ», а на рис. 30, 31 — схемы «И — НЕ», «ИЛИ — НЕ», реализованные на транзисторных ключах (позиционные буквенные обозначения на рис. 28—31 те же, что и на рис. 27).

Простейшие транзисторные ключи составляют основу всей цифровой схемотехники, т. е. правил и принципов разработки состава и конфигурации электронных схем. Эти ключи широко используются как самостоятельно (в качестве прерывателей тока и разного рода коммутаторов),

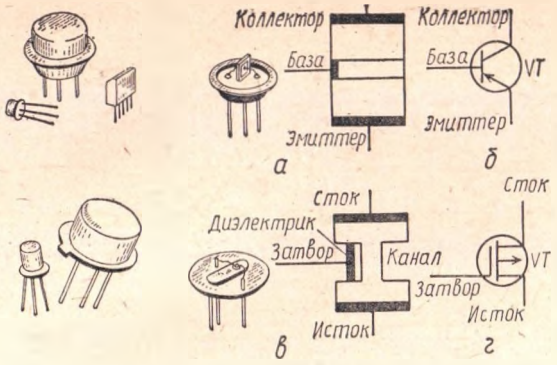


Рис. 26. Устройство биполярного (*а*), полевого (*в*) транзисторов и их соответствующие символические обозначения на схемах (*б*, *г*)

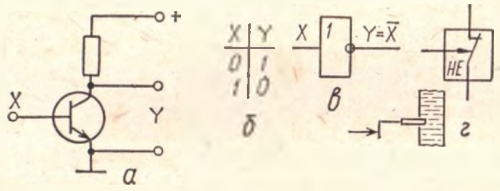


Рис. 27. К пояснению работы транзисторного ключа

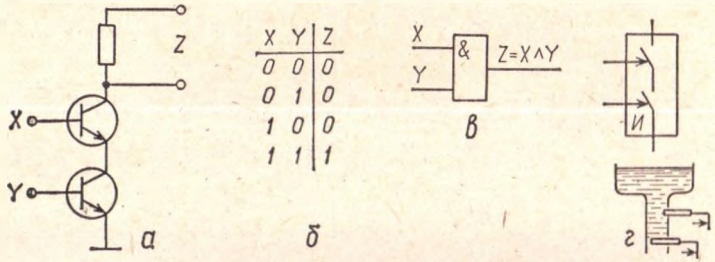


Рис. 28. К рассмотрению схемы «И», реализованной на транзисторных ключах

так и в составе некоторых стандартных функциональных узлов.

Электронные схемы, выполняющие простейшие логические операции, называются *логическими элементами*, или *логическими вентилями*. В схемах, реализующих логические функции, т. е. в логических элементах, логические нули и единицы обычно представлены разными значениями напряжения: напряжением (или уровнем нуля) U_0 и напряжением (или уровнем единицы) U_1 (рис. 32).

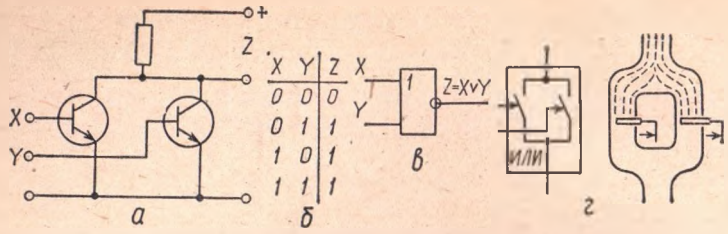


Рис. 29. К рассмотрению схемы «ИЛИ», реализованной на транзисторных ключах

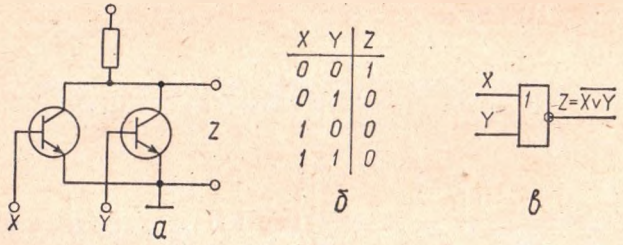


Рис. 30. К рассмотрению схемы «ИЛИ—НЕ», реализованной на транзисторных ключах

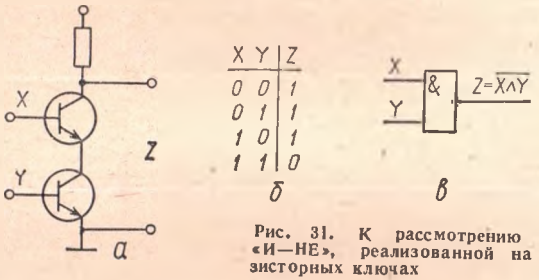


Рис. 31. К рассмотрению схемы «И—НЕ», реализованной на транзисторных ключах

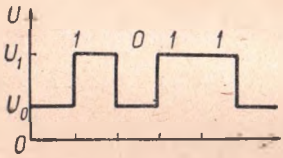


Рис. 32. Уровни нуля и единицы

На рис. 33 показаны диаграммы работы устройств «НЕ», «И», «ИЛИ». Рассмотрим схему, состоящую из двух логических элементов «ИЛИ — НЕ» (рис. 34). Эта схема может быть использована как запоминающий элемент. Если нужно запомнить значение 0, то на вход *О* следует подать сигнал 1. Под действием этого сигнала на выходе Z_1 будет сформирован сигнал $Z_1 = 0$, который поступает на вход второго эле-

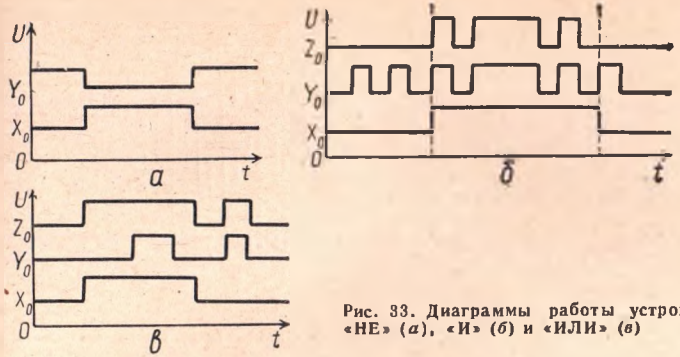


Рис. 33. Диаграммы работы устройств «НЕ» (*а*), «И» (*б*) и «ИЛИ» (*в*)

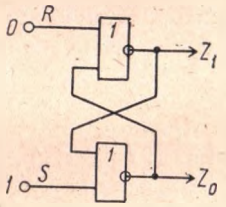
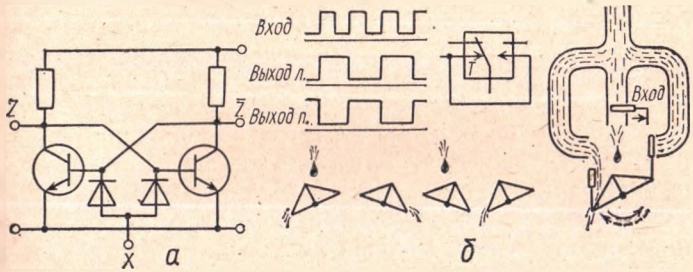


Рис. 34. Схема триггера

Рис. 35. К рассмотрению триггера, работающего в режиме общего входа



мента. Так как теперь на обоих входах второго элемента будут нулевые сигналы, то на его выходе сигнал Z_0 примет значение 1. Этот сигнал поступает на вход первого элемента, на выходе которого устанавливается нулевой сигнал. В таком состоянии схема будет находиться до тех пор, пока на вход *1* не будет подан сигнал 1, в результате чего схема перейдет в состояние $Z_0 = 0$; $Z_1 = 1$. Сигнал $Z_0 = 1$ означает, что последним в схему поступил сигнал на вход *О*. Таким образом, схема «запоминает» какой сигнал поступил последним на вход *О*. Следует иметь в виду, что если на вход *О* поступает сигнал 1, то на *1* подается сигнал 0, и наоборот. Схема рассмотренного типа с двумя устойчивыми состояниями называется т р и г г е р о м и используется для построения запоминающих устройств.

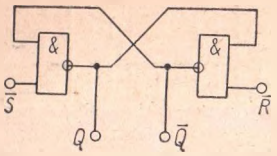


Рис. 36. *RS*-триггер, составленный из двух элементов «И—НЕ»

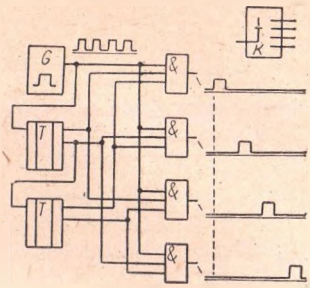


Рис. 37. Коммутатор

Цель управления триггером состоит в том, чтобы с помощью внешних сигналов задавать на одном из его входов то или иное из двух устойчивых состояний либо изменять данное устойчивое состояние на противоположное. Различают два способа (режима) управления триггером: режим *раздельных* входов и режим *общего* входа.

В рассмотренном триггере управляющий сигнал поступал либо на вход *0*, либо на вход *1*, что соответствует режиму раздельных входов. В режиме общего входа (рис. 35,*а*) управляющий сигнал подается одновременно на оба входа триггера, причем каждый очередной сигнал вызывает переход триггера в состояние, противоположное предыдущему. Физический аналог этого триггера отражает рис. 35,*б*.

Триггеры, управляемые в режиме раздельных входов, называются *RS-триггерами*, а триггеры, управляемые в режиме общего входа,— *Т-триггерами*. Существуют и другие типы триггеров.

Оказывается, любой триггер можно составить из совокупности определенным образом соединенных логических элементов. При этом количество логических элементов, а также способы их соединения могут быть различными. Соответственно разными будут и функции, выполняемые триггерами. Поэтому существует много разновидностей триггеров.

На рис. 36 показан *RS*-триггер, составленный из двух элементов «И — НЕ».

Объединив триггеры с логическими элементами, можно получить быстродействующий бесконтактный переключатель — к о м м у т а т о р (рис. 37). Непрерывную очередь импульсов он будет поочередно направлять по разным электрическим цепям, например первый импульс — в один провод, второй — в другой и т. д.

Соединение коммутатора с логическими элементами позволяет создать ш и ф р а т о р, который превращает одно-

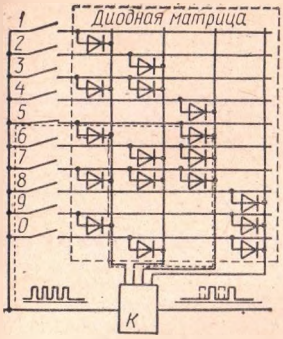


Рис. 38. Шифратор

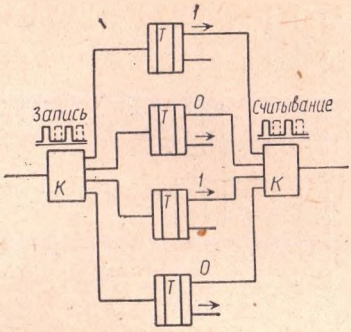


Рис. 39. Регистр

образную последовательность одинаковых импульсов в шифровку из импульсов и пауз, чередующихся в заданном порядке (рис. 38).

Регистр — устройство, предназначенное для приема и запоминания кода одного числа, а также для выполнения определенных операций над этим кодом. Он представляет собой упорядоченную совокупность триггеров со схемой управления входными и выходными сигналами (рис. 39). Разрядность регистра соответствует количеству используемых в нем триггеров. По виду выполняемых операций над кодами различают регистры для *приема*, *передачи* и *сдвига* информации.

Счетчик — устройство, представляющее собой совокупность определенным образом соединенных триггеров и предназначенное для подсчета числа сигналов, поступающих на его вход, и фиксации этого числа в виде кода, хранящегося в триггерах. Счетчики находят широкое применение как в вычислительной технике, так и в различных устройствах автоматики. Они используются для формирования адреса ячеек запоминающих устройств и счета количества циклов выполнения операций.

Дешифратор — это логическая схема, которая в зависимости от заданной последовательности нулей и единиц, т. е. от заданного кода, направляет управляющий сигнал на один из выходов, которому соответствует свой код (рис. 40).

Рассмотренные функциональные устройства представляют собой некоторые совокупности различных электронных схем. Совокупность нескольких взаимосвязанных компонентов (транзисторов, диодов, конденсаторов, резисторов и т. п.), изготовленная одновременно в едином техноло-

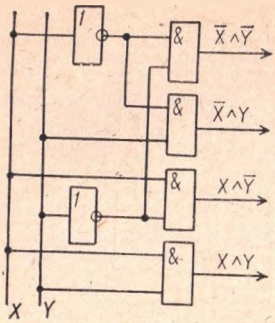


Рис. 40. Логическая схема дешифратора

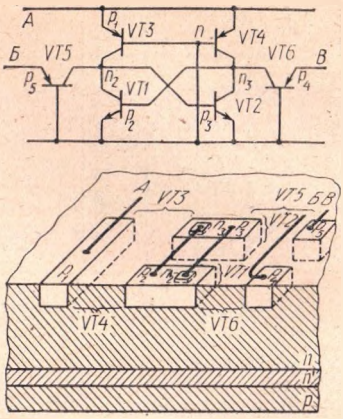


Рис. 41. Фрагмент интегральной микросхемы

гическом цикле на одной и той же несущей конструкции (подложке), называется *интегральной микросхемой* (рис. 41). На рис. 42 показано конструктивное исполнение интегральных микросхем.

Интегральная микросхема предназначена для выполнения определенной функции преобразования информации. Компоненты, входящие в состав интегральной микросхемы, называются ее э л е м е н т а м и, или *интегральными* элементами. Интегральные логические элементы составляют основу (элементную базу) более сложных интегральных микросхем и аппаратуры в целом. Они не могут быть выделены из интегральной микросхемы в качестве самостоятельных изделий. В отличие от интегральных элементов, конструктивно обособленные приборы и детали (транзисторы, резисторы и т. д.) называются *дискретными компонентами*, а электронные узлы и блоки, построенные на их основе,— *дискретными схемами*.

При разработке электронной аппаратуры на основе интегральных микросхем отпадает необходимость в многочисленных паяных соединениях, свойственных дискретным схемам, резко сокращаются размеры и масса аппаратуры благодаря отсутствию корпусов и внешних выводов у каждого отдельного элемента интегральных микросхем, не требуется множество сборочных и монтажных операций.

Интегральные микросхемы изготавливают из полупроводниковых кремниевых пластин по специальной технологии. Использование интегральных микросхем различных типов для построения устройств ЭВМ позволяет резко повысить их надежность и снизить размеры, массу и потреб-

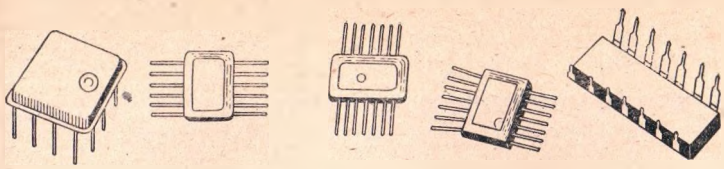


Рис. 42. Конструктивное исполнение интегральных микросхем

ляемую мощность. При этом, чем сложнее логическая схема, сформированная в интегральном элементе, тем выше надежность ЭВМ.

Практически в интегральных микросхемах в качестве логических элементов используются сложные элементы. Широко распространенные интегральные микросхемы малой и средней степеней интеграции имеют в своей основе элементы «И — НЕ» и «И — ИЛИ — НЕ» с разным числом входов.

Интегральная технология послужила основой для создания *третьего* и *последующих* поколений ЭВМ. Эта технология позволяет получить на одной небольшой пластинке (кристалле) полный логический элемент или даже целую схему, причем все соединения между элементами схемы являются неразрывными частями того же кристалла. Интегральная технология дает возможность достичь плотности размещения нескольких сотен тысяч транзисторов на 1 мм^2 кристалла; при этом размеры кристаллов интегральных микросхем лежат в пределах $1{,}5 \times 1{,}5$—6×6 мм^2, а линейные размеры отдельных электродов не превышают 1 мкм.

Функциональная сложность интегральной микросхемы характеризуется *степенью интеграции*, т. е. количеством элементов (чаще всего транзисторов) на одном кристалле. Тенденция к повышению степени интеграции наблюдалась с самого зарождения микроэлектроники. Сначала в отдельных корпусах размещались отдельные логические элементы. Затем стали в одном корпусе размещать несколько логических элементов. После этого простые интегральные микросхемы, расположенные на одном кристалле, стали объединять в сложные, в результате чего появились поначалу *средние*, а затем и *большие* интегральные микросхемы. В основе последних лежит интеграция простых интегральных микросхем.

К средним и большим интегральным микросхемам относятся сумматоры, счетчики, запоминающие устройства, управляющие и арифметико-логические устройства (м и к-

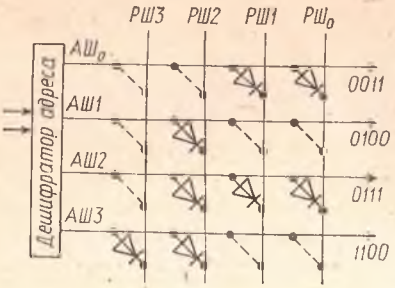


Рис. 43. Схема диодного ПЗУ

ропроцессоры) ЭВМ. Наибольшая степень интеграции свойственна однородным структурам — запоминающим устройствам. В настоящее время выпускаются интегральные микросхемы, в которых степень интеграции достигает сотен тысяч и даже нескольких миллионов элементов на одном кристалле. Такие интегральные микросхемы называются *сверхбольшими*.

Повышение степени интеграции обеспечивает получение новых электронных приборов в виде больших интегральных микросхем, которые функционируют как отдельные устройства и узлы дискретной электронной аппаратуры. Одной из таких микросхем, выполняющих функции арифметического устройства и устройства управления первых ЭВМ, является микропроцессор. Микро-ЭВМ на одном кристалле — вторая ступень больших интегральных микросхем.

Важное место в цифровой аппаратуре и, прежде всего, в ЭВМ занимают системы памяти. Внутренние запоминающие устройства в микро-ЭВМ реализуются в основном на базе интегральных микросхем.

Любое оперативное запоминающее устройство (ОЗУ) состоит из двух основных частей: 1) накопителя; 2) схем управления. В накопителе хранятся данные (двоичные коды). Схемы управления предназначены для ввода и вывода этих данных. Сюда относятся дешифраторы, усилители, регистры, различные ключи, коммутаторы и другие схемы общего назначения.

Накопитель состоит из запоминающих ячеек, каждая из которых хранит один бит информации (0 или 1). Основу запоминающих ячеек составляют триггеры, характерной особенностью которых является наличие двух устойчивых состояний ($Q = 1$ или $Q = 0$) на одном из двух выходов.

Постоянные запоминающие устройства (ПЗУ) строятся на базе диодных элементов. Типичная схема диодного ПЗУ показана на рис. 43. Структура схемы матричная. Строки образуются адресными шинами $АШ_0$, *АШ1*, *АШ2*, *АШ3*, столбцы — разрядными шинами $РШ_0$, *РШ1*, *РШ2*, *РШ3*. Каждая адресная шина хранит свой код, определяемый

тем, с какими разрядными шинами соединена посредством диодов данная адресная шина. На рис. 43 адресная шина $АШ_0$ хранит код 0011, *АШ1* — 0100, *АШ2* — 0111, *АШ3* — 1100. Если подать напряжение, например, на адресную шину *АШ2*, то это напряжение поступит через диоды на разрядные шины *РШ2*, *РШ1*, $РШ_0$, а на разрядной шине *РШ3* напряжение будет равным нулю. Таким образом, при параллельном считывании информации со всех четырех разрядных шин получим код 0111, записанный в выбранной строке.

Все коды, записанные в ПЗУ (как и в ОЗУ), пронумерованы. Номера кодов называются их *адресами*. По адресу кода дешифратор определяет необходимую адресную шину, после чего происходит считывание информации, хранящейся по выбранному адресу.

Принцип работы дешифратора рассмотрим на примере, когда адреса нумеруются с помощью двухразрядного двоичного кода, т. е. имеют номера 00, 01, 10, 11, присвоенные кодам, хранящимся на шинах $АШ_0$, *АШ1*, *АШ2*, *АШ3* соответственно. Для каждого из адресов 00, 01, 10, 11 дешифратор должен подать сигнал 1 только на одну шину, соответствующую выбранному адресу, а на все остальные шины должен быть подан сигнал 0. Таким образом, дешифратор должен работать как устройство с двумя входами (X и Y), на которые подаются двоичные коды адресов, и четырьмя выходами, соответствующими адресным шинам. В таблице показаны логические функции $f_1 = \overline{X} \wedge \overline{Y}$, $f_2 = \overline{X} \wedge Y$, $f_3 = X \wedge \overline{Y}$ и $f_4 = X \wedge Y$, которые необходимо реализовать на каждом из выходов дешифратора:

X	Y	f_1	f_2	f_3	f_4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Одна из возможных реализаций дешифратора на два входа (X и Y) показана на рис. 40. С помощью дешифратора на три входа можно различить восемь различных адресных шин; четырехвходовый дешифратор может выбирать один из $2^4 = 16$ адресов, k-входовый — один из 2^k различных адресов (адресных шин). В связи с этим количество кодов, которые можно хранить в запоминающем устройстве, как правило, есть некоторая степень 2.

Вместо диодов в ПЗУ чаще всего используют транзисторы, работающие в режиме диодов. Использование транзисторов позволяет осуществить и так называемые *полупостоянные* (или *программируемые*) запоминающие устройства (ППЗУ), в которых можно менять записанную информацию. Это дает возможность при необходимости перенастраивать счетно-решающее цифровое устройство на решение новых задач.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Какое устройство называется ключом? 2. С помощью какого устройства можно реализовать электромеханический ключ? 3. С помощью какого устройства можно реализовать электронный ключ? 4. С помощью каких физических устройств можно реализовать логические операции отрицания, дизъюнкции, конъюнкции? Нарисуйте соответствующие схемы. 5. Как различаются значения 0 и 1 в физических устройствах? 6. Как называются схемы, реализующие простейшие логические операции? 7. Сколько входов и выходов могут иметь логические элементы «ИЛИ», «И»? 8. В чем заключаются преимущества электронных схем перед электромеханическими? 9. Какое назначение проводников, резисторов, конденсаторов? 10. Какая полупроводниковая структура называется диодом? 11. Какое основное назначение диода? 12. Что называется транзистором? 13. В чем заключается принцип работы транзистора? 14. Какое основное назначение транзистора? 15. Нарисуйте схему на транзисторах, реализующую логическую функцию «ИЛИ — НЕ». 16. Что называется триггером? 17. Что называется интегральной микросхемой? 18. В чем заключаются преимущества интегральных микросхем перед дискретными схемами? 19. Что называется микропроцессором? 20. Какое назначение дешифратора? 21. Какое назначение регистра? 22. Что называется счетчиком?

Глава 2

ОСНОВЫ ПРОГРАММИРОВАНИЯ

§ 7. ПРОГРАММИРУЕМЫЕ МИКРОКАЛЬКУЛЯТОРЫ. СТРУКТУРА ПАМЯТИ

Целесообразность использования ЭВМ, различных по производительности и назначению, определяется спецификой и сложностью решаемых задач. Однако многие экономические и инженерные задачи достаточно просты или их можно разбить на простые части, решение которых с большой эффективностью обеспечивают ЭВМ малой производительности. Такие сравнительно несложные задачи наиболее оперативно решаются с помощью малогабаритных клавишных ЭВМ — микрокалькуляторов (МК), отличающихся удобствами ввода и вывода информации и не требующих специальной подготовки их пользователей. Эти компактные клавишные вычислительные машины могут выполнять не только обычные арифметические операции, но и достаточно трудоемкие вычисления по встроенным в МК программам, например вычислять значения тригонометрических, логарифмических и других наиболее часто встречающихся функций.

По вычислительным возможностям МК подразделяются на четыре группы: простейшие; инженерные; программируемые; специализированные.

Простейшие МК предназначаются, как правило, для выполнения обычных арифметических операций и их совокупностей. Более совершенные модели этой группы МК обеспечивают и некоторые дополнительные возможности.

Инженерные МК, кроме обычных вычислений, позволяют автоматически выдавать значения основных элементарных функций — показательных, логарифмических, прямых и обратных тригонометрических и некоторых других.

Программируемые МК, кроме указанных возможностей, обеспечивают автоматические вычисления по программам, составленным самими пользователями.

Наибольшие возможности автоматизации решения несложных экономических и инженерных задач связаны с *электронными* программируемыми микрокалькуляторами (ПМК).

Рис. 44. Структурная схема ПМК

Рис. 45. Клавиатурное поле ПМК «Электроника Б3-34»

Микрокалькуляторы всех четырех групп содержат все основные блоки ЭВМ — устройства ввода и вывода информации (УВВ), микропроцессор (арифметико-логическое устройство совместно с устройством управления), запоминающее устройство (ЗУ). Принципиальных отличий между ЭВМ и ПМК (рис. 44) нет.

Устройство ввода (клавиатура) предназначено для занесения в ЗУ исходной информации, необходимой для решения задачи.

Устройство вывода (дисплей или индикатор) служит для вывода из ЗУ промежуточных или окончательных результатов вычислений. Промежуточные результаты выводятся, как правило, для контроля за ходом вычислительного процесса.

Арифметико-логическое устройство (АЛУ) совместно с устройством управления (УУ) предназначено для обработки информации, т. е. выполнения арифметических и логических операций над данными, поступающими из ЗУ, а также для управления функционированием всех остальных устройств МК.

Запоминающее устройство служит для хранения информации.

Принципиальное отличие ПМК от непрограммируемых МК заключается в том, что в последних отсутствует *полупостоянное (программируемое)* запоминающее устройство

(ППЗУ). Поэтому непрограммируемые МК не могут быть настроены на автоматическое решение задач, отличающихся от тех, информация для решения которых заложена в постоянное запоминающее устройство (ПЗУ). Это задачи выполнения арифметических операций сложения, вычитания, умножения, деления, вычисления значений тригонометрических, логарифмических, степенных, показательных и некоторых других функций, указанных на клавиатуре МК.

Микрокалькуляторы с программным управлением благодаря наличию ППЗУ могут быть настроены на автоматическое решение задач по программам, составленным пользователем.

Рассмотрим ПМК «Электроника Б3-34», клавиатурное поле которого показано на рис. 45. На лицевой панели этого ПМК также помещены:

1) тумблер для включения и выключения питания;

2) люминесцентный индикатор (дисплей), на котором высвечиваются числа;

3) переключатель, позволяющий при вычислении значений тригонометрических функций задавать значение аргумента в градусах (положение «Г») или в радианах (положение «Р»). Значения обратных тригонометрических функций также высвечиваются в градусах или в радианах в зависимости от положения этого переключателя.

Ввод информации в ЗУ осуществляется с помощью клавиатуры, посредством которой также запускаются автоматические вычисления по программам, заложенным в ПЗУ и ППЗУ. Выводится информация через световой индикатор (дисплей).

Запоминающее устройство ПМК (рис. 46) состоит из ОЗУ, ПЗУ и ППЗУ.

В ОЗУ информация может меняться в ходе самого вычислительного процесса, выполняемого МК автоматически. Здесь хранятся исходные данные для решения задачи, промежуточные и окончательные результаты вычислений.

В ПЗУ постоянно хранятся программы выполнения арифметических операций и вычисления функций, указанных на клавиатуре. Эти программы вводятся в ПЗУ при изготовлении МК и готовы к использованию каждый раз сразу же после подачи питания.

Для хранения программ, составленных самим пользователем, служит ППЗУ. Эти программы могут выполняться только после того, как они будут введены в ППЗУ с клавиатуры, причем в ходе вычислений информация в ППЗУ не меняется.

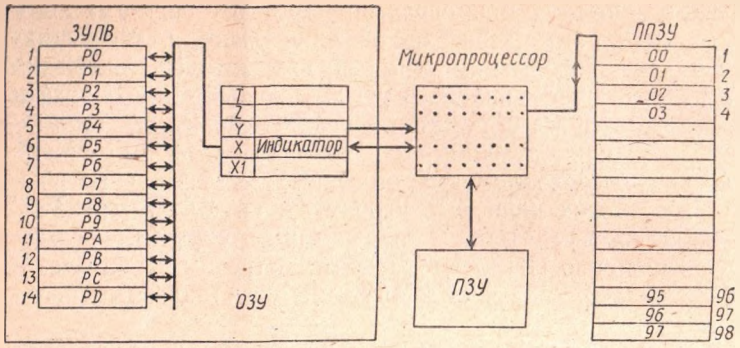


Рис. 46. Структура памяти ПМК

Постоянное и полупостоянное (программируемое) ЗУ предназначены для хранения информации о правилах решения той или иной задачи, вычисления значений тех или иных функций. Другими словами, в ПЗУ и ППЗУ хранятся программы решения задач, т. е. закодированные последовательности операций, которые необходимо выполнить, чтобы получить решение конкретной задачи.

Оперативное ЗУ состоит из двух частей: *стековой* памяти и *адресуемой* памяти. Первая имеет пять регистров, обозначенных как PX1, PX, PY, PZ, PT. Содержимые этих регистров обозначены соответственно как X1, X, Y, Z, T.

Адресуемая память ОЗУ состоит из 14 регистров: P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, PA, PB, PC, PD. Число, записанное в любой из регистров, сохраняется там до тех пор, пока в регистр не будет занесено другое число. Читать число из любого регистра можно сколько угодно раз [подобно тому, как одну и ту же магнитофонную запись можно воспроизводить (читать) сколько угодно раз, но как только записать на ленту новую запись, предыдущая исчезнет].

С регистром PX связан 12-разрядный световой индикатор (дисплей). В режиме автоматической работы ПМК на дисплее высвечивается содержимое регистра PX. Запись информации в любые регистры ОЗУ осуществляется только через регистр PX. Чтобы записать информацию в какой-либо из регистров, ее необходимо предварительно занести в регистр PX. Считывание информации также производится через этот регистр. Чтобы прочитать содержимое любого из регистров, его надо переписать в регистр PX.

При чтении и записи для связи непосредственно с регистром РХ доступен любой из адресуемых регистров. Поэтому адресуемую часть ОЗУ называют *ЗУ с прямым доступом.*

В стековой памяти ОЗУ непосредственно с регистром РХ связаны только регистры РУ и РХ1. Непосредственно из регистра РХ информация может поступить лишь в регистры РУ и РХ1. Наоборот, в регистр РХ информация может подаваться только из регистров РХ1 и РУ. Такое перемещение информации в стековых регистрах обусловлено конструкцией ПМК.

При занесении информации в любой регистр из любого другого регистра информация в последнем сохраняется.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Назовите основные типы МК. **2.** Какие основные блоки содержат МК всех типов? **3.** Для чего предназначено устройство ввода информации? **4.** Для чего предназначено устройство вывода информации? **5.** Как реализованы УВВ в МК? **6.** Для чего предназначено ЗУ? **7.** Для чего предназначен микропроцессор? **8.** Нарисуйте структурную схему ПМК. **9.** Какие типы ЗУ имеются в ПМК? **10.** Какие типы регистров входят в состав ОЗУ ПМК? **11.** В чем заключается принципиальное отличие ПМК от непрограммируемых МК? **12.** В каком из ЗУ ПМК информация никогда не меняется? **13.** В каком из ЗУ ПМК информация может изменяться в ходе автоматически выполняемого вычислительного процесса? **14.** Для чего предназначены ПЗУ и ППЗУ? **15.** Сколько регистров входит в состав адресуемой памяти ПМК? Как обозначаются эти регистры? **16.** Из какого регистра можно перенести информацию в регистры ОЗУ? **17.** В какой регистр вызывается информация из адресуемых регистров? **18.** С каким регистром связан световой индикатор (дисплей) ПМК?

§ 8. НАЗНАЧЕНИЕ КЛАВИШ МИКРОКАЛЬКУЛЯТОРА. ВВОД ЧИСЕЛ

Для управления работой ПМК «Электроника Б3-34» в нем имеется клавиатурное поле из 30 клавиш (см. рис. 45). Каждая клавиша имеет надпись на ней самой и над клавишей (кроме клавиш **F** и **K**), а пять клавиш, образующих нижний ряд, снабжены также и подписями. Это означает, что некоторые клавиши имеют двойное и даже тройное назначение (табл. 1).

При непосредственном нажатии клавиши выполняется операция, обозначенная на клавише. Если же надо выполнить операцию, указанную над клавишей, то следует нажать две клавиши: клавишу с обозначением **F** и клавишу, над которой обозначена нужная операция.

Числа, набираемые на клавиатуре в десятичной системе счисления, заносятся в регистр РХ. При этом числа могут

Таблица 1. Назначение клавиш ПМК «Электроника Б3-34»

Номер по порядку	Обозначение клавиши	Назначение клавиши: при непосредственном нажатии — Операция	Код	при предварительном нажатии клавиш F — Операция	Код	П — Операция	Код	ИП — Операция	Код	К — Операция	Код
1	0	Занесение цифры 0 в РХ	00	Вычисление 10^x	15	Занесение содержимого РХ в Р0	40	Занесение содержимого Р0 в РХ	60	Нет операции (НОП)	54
2	1	Занесение цифры 1 в РХ	01	Вычисление e^x	16	Занесение содержимого РХ в Р1	41	Занесение содержимого Р1 в РХ	61		55
3	2	Занесение цифры 2 в РХ	02	Вычисление $\lg X$	17	Занесение содержимого РХ в Р2	42	Занесение содержимого Р2 в РХ	62		56
4	3	Занесение цифры 3 в РХ	03	Вычисление $\ln X$	18	Занесение содержимого РХ в Р3	43	Занесение содержимого Р3 в РХ	63		30
5	4	Занесение цифры 4 в РХ	04	Вычисление $\arcsin X$	19	Занесение содержимого РХ в Р4	44	Занесение содержимого Р4 в РХ	64		31
6	5	Занесение цифры 5 в РХ	05	Вычисление $\arccos X$	1—	Занесение содержимого РХ в Р5	45	Занесение содержимого Р5 в РХ	65		32
7	6	Занесение цифры 6 в РХ	06	Вычисление $\operatorname{arctg} X$	1L	Занесение содержимого РХ в Р6	46	Занесение содержимого Р6 в РХ	66		33
8	7	Занесение цифры 7 в РХ	07	Вычисление $\sin X$	1[	Занесение содержимого РХ в Р7	47	Занесение содержимого Р7 в РХ	67		34
9	8	Занесение цифры 8 в РХ	08	Вычисление $\cos X$	1Г	Занесение содержимого РХ в Р8	48	Занесение содержимого Р8 в РХ	68		35
10	9	Занесение цифры 9 в РХ	09	Вычисление $\operatorname{tg} X$	1Е	Занесение содержимого РХ в Р9	49	Занесение содержимого Р9 в РХ	69		36
11	,	Занесение десятичной точки в РХ	0—	Передвижение информации в стеке	25	Занесение содержимого РХ в РА	4—	Занесение содержимого РА в РХ	6—		37
12	/—/	Изменение знака числа или знака порядка числа	0L	Переход в режим автоматической работы		Занесение содержимого РХ в РВ	4L	Занесение содержимого РВ в РХ	6L		38
13	ВП	Ввод порядка числа	0[	Переход в режим программирования		Занесение содержимого РХ в РС	4[	Занесение содержимого РС в РХ	6[		39
14	C_x	Гашение содержимого РХ	0Г	Устранение действия префиксной клавиши		Занесение содержимого РХ в PD	4Г	Занесение содержимого PD в РХ	6Г		3—
15	$\overleftarrow{XY}$	Обмен содержимыми РХ и РУ	14	Вычисление X^Y	24		44		64		2—
16	$\uparrow$	Передвижение информации в стеке	0Е	Восстановление содержимого РХ	0		4Е		6Е		3L
17	÷	Деление содержимого РУ на содержимое РХ	13	Вычисление $1/X$	23		43		63		29
18	—	Вычитание из содержимого РУ содержимого РХ	11	Вычисление $\sqrt{X}$	21		41		61		27

Продолжение табл. 1

Номер по порядку	Обозначение клавиши	Назначение клавиши: при непосредственном нажатии — Операция	Код	при предварительном нажатии клавиш F — Операция	Код	П — Операция	Код	ИП — Операция	Код	К — Операция	Код
19	×	Умножение содержимого PY на содержимое PX	12	Вычисление X^2	22		42		62		28
20	+	Сложение содержимого PY с содержимым PX	10	Ввод числа π в PX	20		40		60		26
21	ПП	Переход к подпрограмме; покомандное выполнение программы	53	Организация цикла по содержимому P3	5—					Косвенный переход к подпрограмме по модифицированному адресу	
22	БП	Безусловный переход	5I	Организация цикла по содержимому P2	58					Косвенный безусловный переход по модифицированному адресу	
23	П	Занесение содержимого PX в адресуемые регистры		Организация цикла по содержимому P1	5L					Косвенное занесение по модифицированному адресу содержимого PX	
24	ИП	Занесение содержимого адресуемых регисторов в PX		Организация цикла по содержимому P0	5Г					Косвенный вызов содержимого модифицированного адреса в PX	
25	К	Использование косвенных адресов									
26	С/П	Стоп/Пуск	50	Условный переход по условию $X \neq 0$	57					Косвенный условный переход на модифицированный адрес по условию $X \neq 0$	
27	В/О	Выход из подпрограммы; очистка регистра адресов команд в режиме автоматической работы	52	Условный переход по условию $X \geqslant 0$	59					Косвенный условный переход на модифицированный адрес по условию $X \geqslant 0$	
28	$\overleftarrow{\text{ШГ}}$	Уменьшение на 1 содержимого регистра адресов команд		Условный переход по условию $X = 0$	5E					Косвенный условный переход на модифицированный адрес по условию $X = 0$	
29	$\overrightarrow{\text{ШГ}}$	Увеличение на 1 содержимого регистра адресов команд		Условный переход по условию $X < 0$	5[					Косвенный условный переход на модифицированный адрес по условию $X < 0$	
30	F	Использование функций, написанных над клавишами									

Примечания: 1. Ряд клавишей ПМК «Электроника МК-56» и «Электроника МК-54» обозначен иначе, чем в ПМК «Электроника Б3-34»:

В↑	вместо	↑	$\sin^{-1}$	вместо	arcsin
↔	»	$\overleftarrow{\text{XY}}$	$\cos^{-1}$	»	arccos
X→П	»	$\overrightarrow{\text{П}}$	tg^{-1}	»	arctg
П→X	»	ИП			

2. Назначение клавиш и коды команд в ПМК всех рассматриваемых типов совпадают.

Рис. 47. Структура индикатора ПМК

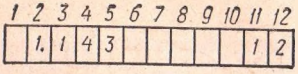


Рис. 48. Показание индикатора ПМК при вводе числа $1{,}143 \cdot 10^{12}$

быть представлены в двух формах: в *естественной*, например 3187,043; 0,8479211; —1,0034; 54387,213, и *в виде произведения* $q \cdot 10^p$ двух сомножителей, первый из которых (q) называется *мантиссой* и представляется в естественной форме. Показатель степени p называется *порядком числа*. Например: $5{,}98 \cdot 10^{24}$; $9{,}1 \cdot 10^{-31}$; $0{,}0054 \cdot 10^{-5}$; $594{,}37 \times \times 10^{16}$. Если при этом $1 \leqslant q < 10$, то такая форма представления чисел называется *стандартизованной*.

Порядок числа — целое число, которое может принимать значения от —99 до +99. Это обусловлено конструкцией МК. Для записи порядка числа отведено два цифровых десятичных разряда и один знаковый разряд. Поэтому число, порядок которого меньше, чем —99 (например, $0{,}01 \cdot 10^{-99}$) МК воспринимается как нуль. Такие числа называют *машинными нулями*.

При попытке ввести в МК число, порядок которого больше, чем +99 (например, $99{,}99 \cdot 10^{99}$), МК выдает сообщение «ERROR», что означает «ОШИБКА». Для изображения чисел в естественной и стандартизованной формах 12 разрядов индикатора МК разделены на две группы (рис. 47): левые девять отведены для изображения числа в естественной форме или мантиссы, если число представляется в виде $q \cdot 10^p$ (при этом левый разряд знаковый: в нем записывается знак числа, если оно отрицательное); правые три — для изображения порядка числа, причем в левом из этих разрядов записывается знак порядка, если он отрицательный.

Для ввода чисел предназначены клавиши МК с обозначениями **0 1 2 3 4 5 6 7 8 9.** Клавишей с обозначением , вводится десятичная точка, которая служит для отделения дробной части числа от целой части. Десятичная точка отдельного разряда индикатора не занимает. Клавиша с обозначением **/–/** меняет знак числа на противоположный. Поэтому при вводе отрицательного числа сначала вводится число без знака, т. е. положительное число, а затем нажатием клавиши **/–/** знак числа меняется на противоположный. Например, если надо ввести число —2,84, то необхо-

Рис. 49. Показание индикатора при вводе числа — 5,87 · 10^{-22}

димо вначале ввести положительное число 2,84, нажав последовательно клавиши 2 , 8 4, а затем изменить знак числа, нажав клавишу /—/.

При вводе числа, представленного в виде $q \cdot 10^p$, сначала вводят мантиссу q. Перед вводом порядка p необходимо нажать клавишу с обозначением **ВП** (ввод порядка). При этом в правых разрядах индикатора высвечиваются цифры 00. После нажатия клавиши **ВП** порядок числа вводят так же, как и мантиссу. Дробной части порядок иметь не может. Например, для введения числа 1,143 · 10^{12} нужно последовательно нажать клавиши **1 , 1 4 3 ВП 1 2.** На индикаторе при этом высвечиваются цифры в таком порядке, как показано на рис. 48.

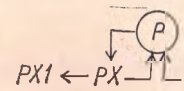
Порядок числа может быть как положительным, так и отрицательным. После нажатия клавиши **ВП** клавишей /—/ можно изменить только знак порядка. Основание 10 при представлении числа в виде $q \cdot 10^p$ подразумевается и в память ПМК не вводится. Например, для ввода числа —5,87 · 10^{-22} необходимо последовательно нажать клавиши **5 , 8 7 /—/ В П 2 2 /—/.** При этом показание индикатора будет таким, как на рис. 49.

Если при наборе числа была допущена ошибка, то регистр РХ очищается (обнуляется) с помощью клавиши C_x, после чего число набирают заново.

Упражнения. Ввести в регистр РХ числа: —0,00785; —2,7 · 10^{-37}; 0,248; —4,548; 2,784 · 10^{2}; 1,34857 · 10^{-3}; —5,432 × × 10^{-10}; —3,874; 10,523; —5,543 · 10^{-7}; 742,3 · 10^{3}.

Выполнение арифметических действий. Если для выполнения операции необходимы два числа, то такая операция называется *двухместной.* Примерами двухместных операций являются сложение, вычитание, умножение, деление и возведение числа X в степень Y.

Результаты всех выполняемых операций помещаются в регистр РХ. Двухместные операции выполняются над содержимым регистров РХ и РУ. При этом содержимое адресуемых регистров не меняется, а в стековых регистрах информация перемещается так:



где через Р обозначен результат операции. Регистры РХ и

РУ называются *операционными*. Следует помнить, что двухместные арифметические операции выполняются на МК так:

$$Y + X \rightarrow \text{РХ (сложение)};\quad Y - X \rightarrow \text{РХ (вычитание)};$$
$$Y \cdot X \rightarrow \text{РХ (умножение)};\quad Y : X \rightarrow \text{РХ (деление)},$$

т. е. при операции вычитания из содержимого регистра РУ вычитается содержимое регистра РХ, а не наоборот; при делении содержимое РУ делится на содержимое РХ, а не наоборот, что обусловлено конструкцией МК.

При выполнении двухместных операций на клавиатуре МК набирают сначала число, которое необходимо занести в регистр РУ. Поскольку при наборе на клавиатуре число заносится в регистр РХ, чтобы перенести это число из регистра РХ в регистр РУ, нужно нажать клавишу с обозначением ↑. При этом информация в стековых регистрах МК переносится так:

$$\text{РХ1}\quad \text{РХ} \rightarrow \text{РУ} \rightarrow \text{РZ} \rightarrow \text{РТ},$$

т. е. содержимое регистра РХ после нажатия клавиши ↑ переходит в регистр РУ и сохраняется также в регистре РХ.

Затем набирают на клавиатуре второе число, которое заносится в регистр РХ, и нажимают клавишу с обозначением необходимой операции. Результат операции помещается в регистр РХ.

Пример. Пусть требуется найти сумму двух чисел: 143,879 и 526,984. Для выполнения данной операции нужно занести одно число в регистр РХ, а другое — в регистр РУ. Для занесения числа 143,879 в регистр РУ набирают это число на клавиатуре МК, а затем нажатием клавиши ↑ перемещают его в регистр РУ. После этого набирают на клавиатуре второе слагаемое (526,984) и нажимают клавишу +. На индикаторе МК высвечивается результат операции — число 670,863.

Порядок выполнения действий и пояснения к ним запишем в виде следующей таблицы:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 4 3 , 8 7 9	Число 143,879 помещается в регистр РХ	143.879
↑	Число 143,879 перемещается в регистр РУ	143.879
5 2 6 , 9 8 4	Число 526,984 помещается в регистр РХ	526.984
+	К содержимому регистра РУ прибавляется содержимое регистра РХ, полученный результат помещается в регистр РХ	670.863

При выполнении операций сложения и умножения порядок занесения чисел в регистры РХ и РУ не является строгим, так как эти операции подчиняются переместительному закону:

$$a + b = b + a;\ a \cdot b = b \cdot a.$$

Нажатие клавиши — предусматривает вычитание из числа, находящегося в регистре РУ, числа, находящегося в регистре РХ. Например, если необходимо вычислить разность 28,354 — 502,987, то нужно уменьшаемое 28,354 поместить в регистр РУ, вычитаемое — в регистр РХ, а затем выполнить операцию вычитания нажатием клавиши —.

При выполнении операции деления делимое всегда помещается в регистр РУ, а делитель — в регистр РХ.

Пример. Пусть требуется найти частное 28,73 : 9,7. Для выполнения данной операции нужно занести делимое 28,73 в регистр РУ, а делитель 9,7 — в регистр РХ. Последовательно нажимая клавиши 2 8 , 7 3 ↑ число 28,73 заносят в регистр РУ. Далее нажатием клавиш 9 , 7 заносят число 9,7 в регистр РХ. После этого нажимают клавишу ÷ . На индикаторе МК высвечивается результат операции — число 2,9618556.

Порядок выполнения действий и пояснения к ним запишем в виде таблицы:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
2 8 , 7 3	Число 28,73 помещается в регистр РХ	28.73
↑	Число 28,73 перемещается в регистр РУ	28.73
9 , 7	Число 9,7 помещается в регистр РХ	9.7
÷	Содержимое регистра РУ делится на содержимое регистра РХ	2.9618556

Для выполнения операции X^Y следует нажать две клавиши: сначала клавишу F, а затем клавишу, над которой нанесена надпись X^Y. При этом должно выполняться требование $X > 0$, так как для вычисления X^Y используется операция логарифмирования: сначала вычисляется $\lg X$, затем — произведение $Y \lg X$, и $10^{Y \lg X} = (10^{\lg X})^Y = X^Y$.

Пример. Найти $2{,}5^{0{,}3}$. Для вычисления значения данного выражения необходимо показатель степени 0,3 занести в регистр РУ, а основание 2,5 — в регистр РХ. Затем следует нажать кла-

вишу **F** и клавишу, над которой нанесена надпись X^Y. Результат — число 1,3163822 высвечивается на индикаторе МК.

Порядок выполнения действий и пояснения к ним занишем в виде следующей таблицы:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
0 , 3	Число 0,3 помещается в регистр РХ	0.3
↑	Число 0,3 перемещается в регистр РУ	0.3
2 , 5	Число 2,5 помещается в регистр РХ	2.5
F X^Y	Содержимое регистра РХ возводится в степень содержимого регистра РУ	1.3163822

Если в процессе решения задачи оказалось, что уменьшаемое, делимое или показатель степени находятся в регистре РХ, а вычитаемое, делитель или основание — в регистре РУ, то перед выполнением операций вычитания, деления или возведения в степень содержимое регистров РХ и РУ надо поменять местами. Для этого предназначена клавиша с обозначением $\overleftarrow{\mathbf{XY}}$.

Если в регистре РХ находится число, не являющееся результатом набора на клавиатуре, то при вводе в регистр РХ нового числа информация в стековых регистрах перемещается так:

$$\text{РХ1} \quad \underset{\downarrow}{\text{РХ}} \rightarrow \text{РУ} \rightarrow \text{РZ} \rightarrow \text{РТ},$$

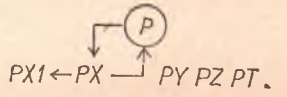
т. е. содержимое регистра РХ автоматически перемещается в регистр РУ. Поэтому использовать клавишу ↑ для занесения содержимого регистра РХ в регистр РУ надо обязательно лишь в случае, если предыдущее число введено с клавиатуры. Во всех остальных случаях для перемещения информации из РХ в РУ нажатие клавиши ↑ необязательно (информация в регистрах стека перемещается автоматически).

Чтобы переписать содержимое регистра РХ1 в регистр РХ, необходимо нажать две клавиши: клавишу **F** и клавишу, над которой нанесена надпись $\mathbf{B_x}$.

Упражнения. Используя ПМК, вычислить: 274,583 — — 295,287; 594,38279 + 8192,3847; 375,41893 · 829,91847; 351,9764: : 295,3269; $5{,}872 \cdot 10^{-2} : 2{,}89 \cdot 10^{3}$; $0{,}38 \cdot 10^{2} \cdot 1{,}54 \cdot 10^{-3}$; $31{,}575 \cdot 10^{-1}$; $2{,}384 \cdot 10^{2}$; $31{,}25 \cdot 10^{-2} - 7{,}54 \cdot 10^{3}$; $1{,}75^{-1}$, ^{25}; $(2{,}374 \cdot 10^{-1})^{-1{,}784 \cdot 10}$; $(-1{,}75)^{2}$; $(-2{,}384)^{-2}$. Воспользовавшись клавишей $\overrightarrow{\mathbf{XY}}$, проверить, сохраняется ли содержимое регистра РУ в МК после выполнения всех вычислений.

Вычисление значений функций. Операцию, выполняемую над одним числом, называют *одноместной*. Для выполнения одноместных операций над содержимым регистра РХ (извлечения квадратного корня, возведения в квадрат, нахождения обратной величины $1/X$, вычисления значений тригонометрических функций $\sin X$, $\cos X$, $\mathrm{tg}\, X$, обратных тригонометрических функций $\arcsin X$, $\arccos X$, $\mathrm{arctg}\, X$, десятичного логарифма $\lg X$, натурального логарифма $\ln X$ — логарифма с основанием e, показательных функций 10^x и e^X) предназначены клавиша **F** и клавиши с соответствующими надписями над ними.

При выполнении одноместных операций содержимое адресуемых регистров не меняется, информация в стековых регистрах перемещается так:



где через P обозначен результат операции.

Для вычисления значения одной из перечисленных функций необходимо занести аргумент в регистр РХ, затем последовательно нажать две клавиши: клавишу **F** и клавишу, над которой указано обозначение вычисляемой функции.

Примеры. 1. Пусть требуется найти $\sqrt{15{,}2}$. Тогда порядок выполнения действий и пояснения к ним отражает следующая таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 5 , 2	Число 15,2 помещается в регистр РХ	15.2
F $\sqrt{}$	Из содержимого регистра РХ извлекается корень квадратный. Результат помещается в регистр РХ	3.8987177

2. Найти $\lg 23{,}3$. Порядок выполнения действий и пояснения к ним дает таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
2 3 , 3	Число 23,3 помещается в регистр РХ	23.3
F lg	Находится десятичный логарифм содержимого регистра РХ. Результат помещается в регистр РХ	1.3673558

3. Вычислить $\frac{1}{28{,}4}$. Порядок выполнения действий и пояснения к ним отражает таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
2 8 , 4	Число 28,4 помещается в регистр РХ	28.4
F 1/X	Находится величина, обратная содержимому регистра РХ. Результат помещается в регистр РХ	3.5211267 —02

4. Пусть требуется вычислить $\log_2 3$. Поскольку ПМК не содержит программы для вычисления значения логарифмической функции по основанию 2, воспользуемся формулой перехода от одного основания логарифма к другому: $\log_2 3 = \frac{\lg 3}{\lg 2}$.

Теперь можно вычислить значения десятичных логарифмов и выполнить операцию деления. Порядок выполнения действий и пояснения к ним дает следующая таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
3	Число 3 помещается в регистр РХ	3
F lg	Находится значение десятичного логарифма числа, находящегося в регистре РХ. Результат помещается в регистр РХ	4.7712121 —01
2	Число 2 помещается в регистр РХ, предыдущее содержимое регистра РХ (0,47712121) автоматически перемещается в регистр РУ	2
F lg	Находится значение десятичного логарифма числа, находящегося в регистре РХ. Результат помещается в регистр РХ	3.0102999 —01
÷	Находится значение частного от деления содержимого регистра РУ на содержимое регистра РХ. Результат помещается в регистр РХ	1.5849623

Если в процессе набора данных на клавиатуре МК ошибочно нажата клавиша F, то для отмены ее действия нужно нажать клавишу, над которой нанесена надпись CF.

Если при вычислении значений тригонометрических функций аргумент задан в градусах, то переключатель «Р — Г» следует перевести в положение «Г». а если — в радианах, то — в положение «Р».

Значения обратных тригонометрических функций будут в градусах, если переключатель «Р — Г» находится в положении «Г», и в радианах, если этот переключатель установлен в положение «Р».

Порядок действий при вычислении, например, значения sin 45° следующий:

1) переключатель «Р — Г» перевести в положение «Г»;

2) нажать последовательно клавиши **4 5 F sin**. Результат в виде 7.071068—01 высвечивается на индикаторе МК.

Если при вычислении значений тригонометрических и обратных тригонометрических функций аргумент задан в градусах и минутах, то минуты необходимо перевести в градусы, учитывая, что $1' = \left(\frac{1}{60}\right)^\circ$.

Пример. Вычислить значение cos 24°12′. Порядок выполнения действий и пояснения к ним отражает следующая таблица (переключатель «Р — Г» — в положении «Г»):

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 2	Число 12 помещается в регистр РХ	12
↑	Число 12 перемещается в регистр РУ	12
6 0	Число 60 помещается в регистр РХ	60
÷	Содержимое регистра РУ делится на содержимое регистра РХ. Результат помещается в регистр РХ	2 —01
2 4	Число 24 помещается в регистр РХ, при этом предыдущее содержимое (0,2) регистра РХ автоматически перемещается в регистр РУ	24
+	К содержимому регистра РУ добавляется содержимое регистра РХ. Результат помещается в регистр РХ	24.2
F cos	Находится значение косинуса числа, находящегося в регистре РХ. Результат помещается в регистр РХ	9.1212612 —01

Последовательным нажатием клавиш **1 2** ↑ **6 0** ÷ выполняют деление числа 12 на 60, т. е. переводят 12′ в градусы; нажатием клавиш **2 4** + находят сумму 24° и (12/60)°, а нажатием клавиш **F cos** вычисляют значение косинуса.

Если при вычислении значения функции значение аргумента в область определения функции не входит, то МК выдает сообщение «ERROR».

Области определения функций, вычисляемых МК, следующие:

Функция	Область допустимых значений аргумента X
$\sqrt{X}$	$X \geqslant 0$
$1/X$	$X \neq 0$
X^2	Произвольные значения
$\sin X$	То же
$\cos X$	»
$\operatorname{tg} X$	$X \neq \frac{\pi}{2} + \pi n,\ n \in Z$
$\arcsin X$	$\lvert X \rvert \leqslant 1$
$\arccos X$	$\lvert X \rvert \leqslant 1$
$\operatorname{arctg} X$	Произвольные значения
e^X	То же
10^X	»
$\ln X$	$X > 0$
$\lg X$	$X > 0$

Нажатием клавиш **F π** осуществляется вызов в регистр РХ из ПЗУ числа 3,1415926 — приближенного значения числа π.

Упражнения. Используя ПМК, вычислить:

1. $\sin 60°$; $\sin 45°$; $\sin \pi/2$; $\sin 23° 15'$; $\sin 1{,}7$; $\sin 30°$; $\cos 60°$; $\cos 37° 30'$; $\cos 90°$; $\cos \pi/4$; $\cos 1{,}5$; $\cos 30°$; $\operatorname{tg} 60°$; $\operatorname{tg} 45°$; $\operatorname{tg} 30°$; $\operatorname{tg} \pi/4$; $\operatorname{tg} 27$; $\operatorname{tg} 13° 15'$; $\operatorname{tg} \pi/2$.

2. $\lg 2$; $\lg 3$; $\lg 20$; $\lg 30$; $\ln 2$; $\ln 3$; $\ln 20$; $\ln 30$; $\log_2 25$; $\log_3 18$; $\sqrt{27}$; $\sqrt{30}$; $\sqrt{25{,}01}$; $\sqrt{99{,}99}$; $(-1{,}75)^2$; 25^2; $(-0{,}024)^2$; $0{,}492^2$.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Сколько клавиш имеется в клавиатурном поле ПМК «Электроника Б3-34»? **2.** В какой регистр заносится число, набираемое на клавиатуре МК? **3.** Какая форма представления чисел называется стандартизованной? **4.** Как вводятся в МК отрицательные числа? **5.** Как вводятся в МК числа, представленные в стандартизованной форме? **6.** Как выполняются на ПМК одноместные операции, обозначения которых указаны над клавишами? **7.** Какие операции называются одноместными? **8.** Над содержимым какого регистра МК выполняются одноместные операции? **9.** Для чего предназначена клавиша F? **10.** Для чего предназначен переключатель «Р — Г»? **11.** Для чего предназначена клавиша ↑? **12.** Когда используется клавиша $\overleftrightarrow{XY}$? **13.** Как исправить число, неправильно набранное на клавиатуре МК? **14.** Какие операции называются двухместными? **15.** Над содержимым каких регистров МК выпол-

няются двухместные операции? 16. Какие регистры называются операционными? 17. Как выполняются на МК операции вычитания, деления и возведения в степень?

§ 9. ВЫЧИСЛЕНИЕ ЗНАЧЕНИЙ АЛГЕБРАИЧЕСКИХ ВЫРАЖЕНИЙ

При определении значений некоторых алгебраических выражений необходимо выполнять вычисления, в которых первым компонентом следующего действия является результат предыдущего.

Пример. Требуется найти значение выражения

$$(9{,}15 \cdot 10^{-2} + 8{,}09 \cdot 10^{3}) \cdot 0{,}095.$$

Порядок выполнения действий и пояснения к ним дает таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
9 , 1 5 ВП 2 /—/	$9{,}15 \cdot 10^{-2} \to$ РХ	9.15 —02
↑	$9{,}15 \cdot 10^{-2} \to$ РУ	9.15 —02
8 , 0 9 ВП 3	$8{,}09 \cdot 10^{3} \to$ РХ	8.09 03
+	К содержимому регистра РУ добавляется содержимое регистра РХ. Результат помещается в регистр РХ	8090.0915
0 , 0 9 5	$0{,}095 \to$ РХ. Предыдущее содержимое (8090, 0915) регистра РХ автоматически перемещается в регистр РУ	0.095
×	Содержимое регистра РУ умножается на содержимое регистра РХ. Результат помещается в регистр РХ	768.55869

Для того чтобы вычислить значение заданного выражения, необходимо последовательно выполнить указанные в таблице действия по «старшинству» и согласно правилам выполнения действий в выражениях, содержащих скобки. Для выполнения сложения (действия, указанного в скобках) одно из чисел, например $9{,}15 \cdot 10^{-2}$, помещают в регистр РУ, второе — в регистр РХ. Нажатием клавиши + производится сложение чисел. Результат (8090,0915) помещается в регистр РХ и высвечивается на индикаторе МК.

Теперь следует выполнить операцию умножения, для чего один из сомножителей (8090,0915) нужно переместить в регистр РУ, а другой (0,095) — в регистр РХ (или наоборот). В этом случае для перемещения числа из регистра РХ в РУ нет необходимости

нажимать клавишу ↑, а достаточно поместить новое число в РХ. При этом предыдущее содержимое регистра РХ автоматически заносится в РУ. Поскольку число 8090,0915 является результатом выполненной операции сложения, для автоматического перемещения его в регистр РУ достаточно ввести в РХ с клавиатуры другое число, в данном случае 0,095. Теперь остается выполнить операцию умножения нажатием клавиши ×. Результат 768,55869 помещается в регистр РХ.

Пример. Вычислить значение выражения

$$\left(\frac{9{,}673 \cdot 834 \cdot 10^{-2}}{0{,}876} + 12{,}372 \cdot 10^{2} - 17{,}268\right) : 3{,}76 \cdot 10^{3} - 28{,}615 \cdot 10^{-2}.$$

Нажав последовательно клавиши: **9, 6 7 3 ↑ 8 3 4 ВП 2 /—/ × 0,876 ÷ 1 2 , 3 7 2 ВП 2 + 1 7 , 2 6 8 — 3,7 6 ВП 3 ÷ 2 8, 6 1 5 ВП 2 /—/—** получим 2861,1511.

Упражнения. Вычислить значения выражений:

1. $\dfrac{(623{,}54 - 101{,}848 + 19{,}0341) \cdot 12{,}472}{26{,}218}$.
2. $\dfrac{(12{,}934 \cdot 10^{2} + 631 \cdot 10^{-1} - 19{,}25 \cdot 10^{-2})\,(-72{,}30_{1})}{26{,}107 \cdot 10^{4}}$.
3. $(12{,}534 + 613{,}2 : 15{,}4)\,(-3{,}279) : 15{,}407 - 108{,}543$.
4. $\left(\dfrac{10{,}987 \cdot 10^{-3}}{5{,}314 \cdot 10^{-2}} + 12{,}762 - 17{,}462 \cdot 10^{2}\right) : 7{,}32 \cdot 10^{3}$.

Пример. Пусть требуется вычислить $\dfrac{8{,}11}{8{,}304 - 3{,}752}$. Это выражение можно переписать в виде $8{,}11 : (8{,}304 - 3{,}752)$.

Согласно правилам выполнения действий вычислим сначала выражение в скобках. Для этого число 8,304 занесем в регистр РУ, число 3,752 — в регистр РХ и, нажав клавишу —, выполним операцию вычитания. Результат 4,552 заносится в регистр РХ.

В регистр РХ занесем число 8,11; при этом предыдущее содержимое этого регистра (число 4,552) автоматически переместится в регистр РУ. Так как делитель находится в регистре РУ, а делимое 8,11 — в регистре РХ, необходимо поменять местами содержимое регистров РХ и РУ, для чего воспользуемся клавишей $\overleftrightarrow{XY}$. После этого можно выполнить деление, нажав клавишу ÷. В регистре РХ будет искомый результат 1,7816344, который и высвечивается на индикаторе МК.

Порядок выполнения действий и пояснения к ним отражает следующая таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
8 , 3 0 4	8,304 → РХ	8.304
↑	8,304 → РУ	8.304
. 3 , 7 5 2	3,752 → РХ	3.752
—	Из содержимого регистра РУ вычитается содержимое регистра РХ. Результат помещается в регистр РХ	4.552

Продолжение табл

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
8., 1 1	8,11 → РХ. При этом предыдущее содержимое (4,552) регистра РХ автоматически перемещается в регистр РУ	8.11
← XY →	Содержимые регистров РХ и РУ меняются местами	4.552
÷	Содержимое регистра РУ делится на содержимое регистра РХ. Результат помещается в регистр РХ	1.7816344

Пример. Пусть требуется вычислить значение выражения

$$\frac{257{,}384 - 193{,}285}{921{,}849 - 388{,}456}.$$

Вычислим сначала, например, значение знаменателя. Для этого вводим с клавиатуры в регистр РХ число 921,849 и нажатием клавиши ↑ заносим его из регистра РХ в регистр РУ. Затем вводим с клавиатуры в регистр РХ число 388,456. Нажав клавишу —, в регистре РХ получим значение знаменателя 533,393. Для вычисления значения числителя необходимо использовать оба операционных регистра (РХ и РУ). Следовательно, хранить значение знаменателя ни в одном из этих регистров нельзя. Возникает необходимость где-то записать значение знаменателя. Можно было бы это сделать и на бумаге, а затем (после вычисления значения числителя) ввести в регистр РХ полученное ранее значение знаменателя и продолжить вычисления.

Чтобы не делать запись промежуточных результатов на бумаге, используют регистры ОЗУ, в частности адресуемые регистры. Число, находящееся в регистре РХ, можно занести в любой из адресуемых регистров. Для этого надо нажать две клавиши: клавишу с обозначением П (память) и клавишу с номером адресуемого регистра, в который решено записать на хранение содержимое регистра РХ. Для того чтобы переписать содержимое любого из адресуемых регистров в регистр РХ, необходимо нажать две клавиши: клавишу с обозначением **ИП** и клавишу с номером регистра, из которого нужно перенести информацию.

Пусть для хранения значения знаменателя выбран адресуемый регистр Р2. Тогда после получения значения знаменателя 533,393 нажимаем клавиши **П** и **2**. Тем самым число 533,393 будет занесено на хранение в регистр Р2. После этого с клавиатуры вводим в регистр РХ число 257,384 и переносим его в регистр РУ нажатием клавиши ↑. Затем с клавиатуры вводим в регистр РХ число 193,285 и, нажимая клавишу —, в регистре РХ получаем значение числителя 64,099. Далее переносим из регистра Р2 в регистр РХ значение знаменателя, нажав клавиши **ИП** и **2**. После этого можно выполнить операцию деления, нажав клавишу ÷. В регистре РХ получим искомый результат $1{,}2017218 \cdot 10^{-1}$.

Порядок выполнения действий и пояснения к ним дает таблица на с. 74.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
9 2 1 , 8 4 9	921,849 → PX	921.849
↑	921,849 → PY	921.849
3 8 8 , 4 5 6	388,456 → PX	388.456
—	Из содержимого регистра PY вычитается содержимое регистра PX. Результат помещается в регистр PX	533.393
П2	Содержимое (533,393) регистра PX помещается на хранение в адресуемый регистр P2	533.393
2 5 7 , 3 8 4	257,384 → PX	257.384
↑	257,384 → PY	257.384
1 9 3 , 2 8 5	193,285 → PX	193.285
—	Из содержимого регистра PY вычитается содержимое регистра PX. Результат помещается в регистр PX	64.099
ИП2	Из регистра P2 в регистр PX переписывается значение 533,393. При этом предыдущее содержимое регистра PX автоматически перемещается в регистр PY	533.393
÷	Содержимое регистра PY делится на содержимое регистра PX. Результат помещается в регистр PX	1.2017218 —01

Пример. Вычислить значение выражения

$$\frac{34{,}042 \cdot 25{,}2034 + 24{,}0991 \cdot 29{,}0099}{\sqrt{48{,}349 - 31{,}5446} - 15{,}202^2}$$

Порядок выполнения действий и пояснения к ним отражает следующая таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
4 8 , 3 4 9	Число 48,349 помещается в регистр PX	48.349
↑	Число 48,349 перемещается в регистр PY	48.349
3 1 , 5 4 4 6	Число 31,5446 помещается в регистр PX	31.5446

Продолжение табл.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
—	Из содержимого регистра PY вычитается содержимое регистра PX. Результат помещается в регистр PX	16.8044
F $\sqrt{}$	Из числа, находящегося в регистре PX, извлекается корень квадратный. Результат помещается в регистр PX	4.099317
1 5 , 2 0 2	Число 15,202 помещается в регистр PX, предыдущее содержимое этого регистра автоматически перемещается в регистр PY	15.202
FX^2	Число 15,202, т. е. содержимое регистра PX, возводится в квадрат. Результат помещается в регистр PX	231.1008
—	Из содержимого (4,099317) регистра PY вычитается содержимое (231,1008) регистра PX. Результат помещается в регистр PX	—227.00148
П2	Содержимое регистра PX помещается в адресуемый регистр P2 для хранения	—227.00148
3 4 , 0 4 2	Число 34,042 помещается в регистр PX	34.042
↑	Число 34,042 помещается в регистр PY	34.042
2 5 , 2 0 3 4	Число 25,2034 помещается в регистр PX	25.2034
×	Содержимое регистра PX умножается на содержимое регистра PY. Результат помещается в регистр PX	857.97414
П 1	Произведение чисел 34,042 × ×25,2034 = 857,97414 помещается на хранение в регистр P1	857.97414
2 4 , 0 9 9 1	Число 24,0991 помещается в регистр PX	24.0991
↑	Число 24,0991 перемещается в регистр PY	24.0991
2 9 , 0 0 9 9	Число 29,0099 помещается в регистр PX	29.0099
×	Содержимые регистров PX и PY перемножаются. Результат помещается в регистр PX	699.11248

Продолжение табл.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
ИП 1	В регистр РХ вызывается содержимое регистра Р1, прежнее содержимое регистра РХ автоматически перемещается в регистр РУ	857.97414
+	Содержимые регистров РХ и РУ складываются. Результат помещается в регистр РХ	1557.0866
ИП 2	Содержимое регистра Р2 вызывается в регистр РХ, при этом предыдущее содержимое регистра РХ автоматически перемещается в регистр РУ	—227.00148
÷	Содержимое регистра РУ делится на содержимое регистра РХ. Результат помещается в регистр РХ	—6.8593676

Пример. Вычислить, какую скорость приобретет корпус ракеты массой $M = 0{,}27$ т, израсходовав $m = 12$ т горючего при скорости его истечения из ракеты $u = 2{,}85$ км/с.

Задачу решаем по формуле Циолковского

$$v = 2{,}3026u \lg (1 + m/M).$$

Порядок выполнения действий и пояснения к ним дает таблица:

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 2	Число 12 помещается в регистр РХ	12
↑	Число 12 перемещается в регистр РУ	12
0 , 2 7	Число 0,27 помещается в регистр РХ	0.27
÷	Содержимое (12) регистра РУ делится на содержимое (0,75) регистра РХ. Результат помещается в регистр РХ	44.444444
1	Число 1 помещается в регистр РХ. Предыдущее содержимое (44,444444) этого регистра автоматически перемещается в регистр РУ	1

Продолжение табл.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
+	К содержимому (44.444444) регистра РУ прибавляется содержимое (1) регистра РХ. Результат помещается в регистр РХ	45.444444
F lg	Вычисляется значение десятичного логарифма числа, находящегося в регистре РХ. Результат помещается в регистр РХ	1.6574807
2 , 8 5	Число 2,85 помещается в регистр РХ. Предыдущее содержимое (1,6574807) этого регистра автоматически перемещается в регистр РУ	2.85
×	Содержимое (1,6574807) регистра РУ умножается на содержимое (2,85) регистра РХ. Результат помещается в регистр РХ	4.7238199
2 , 3 0 2 6	Число 2,3026 помещается в регистр РХ. Предыдущее содержимое (4,7238199) этого регистра автоматически перемещается в регистр РУ	2.3026
×	Содержимое (4,7238199) регистра РУ умножается на содержимое (2,3026) регистра РХ. Результат помещается в регистр РХ	10,877068

Пример. Вычислить значение выражения

$$\frac{\sin(3 \cdot 1{,}24367) - \sqrt[3]{12{,}845}}{\frac{1}{1{,}2348^2} + \pi - \sqrt{25{,}48}}.$$

Порядок выполнения действий и пояснения к ним отражает следующая таблица (переключатель «Р — Г» — в положении «Р»):

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 , 2 3 4 8	Число 1,2348 помещается в регистр РХ	1.2348
F X^2	Число 1,2348 возводится в квадрат. Результат помещается в регистр РХ	1.524731

Продолжение табл.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
F 1/X	Вычисляется $1/1{,}2348^2$. Результат помещается в регистр РХ	6.5585339 —01
F π	В регистр РХ вызывается число $\pi = 3{,}1415926$, предыдущее содержимое этого регистра автоматически перемещается в регистр РУ	3.1415926
+	Складывается содержимое регистров РХ и РУ. Результат помещается в регистр РУ	3.797446
2 5 , 4 8	Число 25,48 помещается в регистр РХ, предыдущее содержимое (3,797446) этого регистра автоматически перемещается в регистр РУ	25.48
F √	Из числа 25,48 извлекается корень квадратный. Результат помещается в регистр РХ	5.0467811
—	Из содержимого (3,797446) регистра РУ вычитается содержимое (5,0467811) регистра РХ. Результат помещается в регистр РХ	—1.2493351
П 2	Содержимое регистра РХ перемещается на хранение в регистр Р2	—1.2493351
1 , 2 4 3 6 7	Число 1,24367 помещается в регистр РХ	1.24367
↑	Число 1,24367 перемещается в регистр РУ	1.24367
3	Число 3 помещается в регистр РХ	3
×	Число 1,24367 умножается на 3. Результат помещается в регистр РХ	3.73101
F sin	Вычисляется sin (3 · 1,24367). Результат помещается в регистр РХ	—5.5587685 —01
П 1	Значение sin (3 · 1,24367) помещается для хранения в регистр Р1	—5.5587685 —01
1	Число 1 помещается в регистр РХ	1
↑	Число 1 перемещается в регистр РУ	1
3	Число 3 помещается в регистр РХ	3
÷	Вычисляется 1 : 3. Результат помещается в регистр РХ	3.3333333 —01

Продолжение табл.

Порядок нажатия клавиш МК	Пояснение	Показание индикатора МК
1 2 , 8 4 5	Число 12,845 помещается в регистр РХ, предыдущее содержимое $(3{,}3333333 \cdot 10^{-1})$ этого регистра автоматически перемещается в регистр РУ	12.845
F X^Y	Вычисляется значение $\sqrt[3]{12.845}$. Результат помещается в регистр РХ	2.3419522
ИП 1	Содержимое регистра Р1 вызывается в регистр РХ, предыдущее содержимое (2,3419522) этого регистра автоматически перемещается в регистр РУ	—5.5587685 —01
$\overleftarrow{XY}$ $\overrightarrow{\ }$	Содержимые регистров РХ и РУ меняются местами	2.3419522
—	Находится разность $-5{,}5587685 \times 10^{-1} - 2{,}3419522$. Результат помещается в регистр РХ	—2.897829
ИП 2	Содержимое (значение знаменателя дроби) регистра Р2 вызывается в регистр РХ, предыдущее содержимое (значение числителя дроби) этого регистра автоматически перемещается в регистр РУ	—1.2493351
÷	Содержимое регистра РУ делится на содержимое регистра РХ. Результат помещается в регистр РХ	2.3194969

Упражнения. 1. Ввести числа в указанные регистры: —0,345 в Р5; 15,901 в Р0; 7,329 в РD; —16,25 в Р9. Прочитав содержимые всех регистров, убедиться в том, что после чтения они сохраняются, для чего прочитать их еще раз.

2. Вычислить значение выражения

$$5{,}92 \cdot 3{,}0375 + 21{,}52 \cdot 0{,}561 - \frac{7{,}891}{1{,}312} + \frac{25{,}443}{48{,}57},$$

используя для хранения промежуточных результатов адресуемые регистры.

3. Вычислить значения выражений:

а) $\dfrac{18{,}23\,(2{,}4899 + 237{,}998)}{7{,}35 \cdot 1{,}0341 - 7{,}82735}$;

б) $\dfrac{12{,}8723 + 12{,}7852 : 5{,}9321}{\sqrt{11{,}027 + 12{,}872} - 15{,}238}$;

в) $\dfrac{25{,}844\,(34{,}0972 - 29{,}976) + 46{,}9875}{\sqrt{102{,}997 - 54{,}9223}}$;

г) $\dfrac{44{,}368 \cdot 9{,}9832 - 12{,}8 : 21{,}3356}{(12{,}303 - 9{,}9082)^2 + 15{,}8067}$;

д) $\dfrac{0{,}3 \cdot 2{,}1248 - \cos 2{,}1248}{\sqrt{\pi + 1 - 1{,}357^3}}$;

е) $\dfrac{10^{2{,}4} + 2 \operatorname{tg} 13{,}876}{5{,}2416^2 + \pi}$;

ж) $\dfrac{3 \cdot 10{,}975}{4\sqrt{11{,}973} + \cos(5 - 2{,}844)}$.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как определяется порядок выполнения операций над числами? 2. Как записать число в какой-либо адресуемый регистр МК? 3. Как прочитать число из какого-либо адресуемого регистра МК?

§ 10. ЛИНЕЙНЫЕ ПРОГРАММЫ

Пусть требуется вычислить значение функции

$$Y = (\lg X + X + 3)^2 / (X^2 + 1)$$

при условии, что значение аргумента X предварительно записано в адресуемом регистре P5. Вычислим сначала значение знаменателя. Для этого вызываем значение X в регистр PX из адресуемого регистра P5, нажимая клавиши **ИП** и **5**. Затем возведем содержимое регистра PX в квадрат, нажав клавиши **F** и $\mathbf{X^2}$. После этого вводим с клавиатуры в регистр PX число 1. Значение X^2 при этом переместится из регистра PX в регистр PY. Нажимая далее клавишу **+**, получаем в регистре PX значение $(X^2 + 1)$.

Поскольку для вычисления числителя необходимо выполнять двухместные операции, промежуточное значение $(X^2 + 1)$, находящееся у регистре PX, занесем на временное хранение в один из адресуемых регистров, например в P0. Затем вычисляем значение числителя. Для этого вновь вызываем значение X из регистра P5 в регистр PX, нажимая клавиши **ИП** и **5**. Далее, нажав клавишу **↑**, занесем значение X из регистра PX в регистр PY; при этом значение X остается и в регистре PX. После этого вычислим значение $\lg X$, нажав клавиши **F** и **lg**. Значение $\lg X$ будет в регистре PX. Содержимое регистра PY при этом не изменяется, и занесенное в него ранее значение X сохраняется. Нажав клавишу **+**, в регистре PX получим значение $\lg X + X$. Введем теперь с клавиатуры МК

в регистр РХ число 3. Значение $\lg X + X$ при этом переместится в регистр РУ. После нажатия клавиши **+** в регистре РХ получим значение $(\lg X + X + 3)$, а нажав клавиши **F** и **X^2** — значение числителя $(\lg X + X + 3)^2$.

Введем теперь в регистр РХ из регистра Р0 хранящееся в Р0 значение знаменателя $(X^2 + 1)$. Для этого нажимаем клавиши **ИП** и **0**. Предыдущее содержимое регистра РХ, т. е. значение $(\lg X + X + 3)^2$ при этом автоматически переместится в регистр РУ. Нажав далее клавишу **÷**, в регистре РХ получим искомый результат — значение функции Y.

Описанная последовательность операций, которые необходимо выполнить на МК, чтобы решить задачу, называется *программой* ее решения на МК, а указание о выполнении каждой отдельной операции — *командой* для МК.

Для большей наглядности программу решения рассмотренной задачи перепишем в виде таблицы:

Номер команды	Команда	Примечание
1	ИП5	$X \to$ РХ (занесение значения X в РХ из Р5)
2	FX²	$X^2 \to$ РХ (возведение значения X в квадрат и занесение результата в РХ)
3	1	$1 \to$ РХ, $X^2 \to$ РУ (занесение 1 в РХ, X^2 в РУ)
4	+	$1 + X^2 \to$ РХ (получение значения $1 + X^2$ и занесение его в РХ)
5	П0	$1 + X^2 \to$ Р0 (занесение значения $1 + X^2$ в Р0 на временное хранение)
6	ИП5	$X \to$ РХ (занесение значения X в РХ из Р5)
7	↑	$X \to$ РУ (занесение значения X в РУ и РХ)
8	Flg	$\lg X \to$ РХ (получение значения $\lg X$ и занесение его в РХ)
9	+	$\lg X + X \to$ РХ (получение значения $\lg X + X$ и занесение его в РХ)
10	3	$3 \to$ РХ, $\lg X + X \to$ РУ (занесение 3 в РХ и $\lg X + X$ в РУ из РХ)
11	+	$\lg X + X + 3 \to$ РХ (получение значения $\lg X + X + 3$ и занесение его в РХ)
12	FX²	$(\lg X + X + 3)^2 \to$ РХ (получение значения $(\lg X + X + 3)^2$ и занесение его в РХ)
13	ИП0	$X^2 + 1 \to$ РХ, $(\lg X + X + 3)^2 \to$ РУ [занесение значения $X^2 + 1$ в РХ из Р0 и $(\lg X + X + 3)^2$ в РУ из РХ]
14	÷	$(\lg X + X + 3)^2/(X^2 + 1) \to$ РХ (получение окончательного результата и занесение его в РХ)

Примечания записывают в произвольной форме, однако по возможности наиболее лаконично. Словесные пояснения

(записываемые в скобках) в примечаниях опускают, оставляя, как правило, только символическое описание результата, полученного после выполнения каждой команды.

Иногда при записи программ записывают только команды. При такой записи рассмотренная программа будет иметь вид:

ИП5, FX², 1, +, П0, ИП5, ↑, F lg, +, 3, +,
FX², ИП0, ÷

Такая запись программ часто оказывается неудобной, особенно в тех случаях, если структура программ достаточно сложна.

Записав в регистр Р5 некоторое число (например, 1,783) и выполнив все 14 указанных в программе операций, получим искомое значение $(\lg 1{,}783 + 1{,}783 + 3) / (1{,}783^2 + 1)$. Занеся в регистр Р5 другое число и выполнив те же операции, найдем новое значение рассмотренной функции. По данной программе можно вычислить сколько угодно значений функции $Y = (\lg X + X + 3)^2 / (X^2 + 1)$ для задаваемых значений аргумента, не задумываясь каждый раз над тем, какие операции и в какой последовательности нужно выполнять.

Программы, в которых все команды выполняются строго в том порядке, в каком они записаны, называются *линейными*. Рассмотренная программа линейна.

Пример. Составить программу для вычисления пути s, пройденного телом при равноускоренном движении с начальной скоростью v_0, ускорением a за время t.

Длина пути при равноускоренном движении определяется выражением

$$s = v_0 t + at^2/2.$$

Составим программу решения задачи при условии, что значения начальной скорости v_0 будут храниться в адресуемом регистре Р1, значения ускорения a — в Р2, значения времени t — в Р3:

Номер команды	Команда	Примечание
1	ИП3	$t \rightarrow$ РХ (занесение значения t в РХ из Р3)
2	FX²	$t^2 \rightarrow$ РХ (возведение значения t в квадрат и занесение результата в РХ)
3	ИП2	$a \rightarrow$ РХ, $t^2 \rightarrow$ РУ (занесение значения a в РХ из Р2, перемещение значения t^2 в РУ)
4	×	$t^2a \rightarrow$ РХ (получение значения t^2a и занесение его в РХ)
5	2	$2 \rightarrow$ РХ, $t^2a \rightarrow$ РУ (занесение числа 2 в РХ, перемещение значения t^2a в РУ)

Продолжение табл.

Номер команды	Команда	Примечание
6	÷	$t^2a/2 \rightarrow$ РХ (получение значения $t^2a/2$ и занесение его в РХ)
7	П0	$t^2a/2 \rightarrow$ Р0 (занесение значения $t^2a/2$ в Р0 на временное хранение)
8	ИП1	$v_0 \rightarrow$ РХ (занесение значения v_0 в РХ из Р1)
9	ИП3	$t \rightarrow$ РХ, $v_0 \rightarrow$ РУ (занесение значения t в РХ из Р3 и перемещение значения v_0 в РУ из РХ)
10	×	$v_0t \rightarrow$ РХ (получение значения v_0t и занесение его в РХ)
11	ИП0	$t^2a/2 \rightarrow$ РХ, $v_0t \rightarrow$ РУ (занесение значения $t^2a/2$ в РХ из Р0 и перемещение значения v_0t в РУ из РХ)
12	+	$v_0t + t^2a/2 \rightarrow$ РХ (получение окончательного результата и занесение его в РХ)

Упражнения. **1.** Составить программу для вычисления значения функции

$$Y = \frac{5{,}7\,X - \sin X}{X^2 + \sqrt{\pi} + 2}$$

при условии, что значение X предварительно занесено в регистр Р2. Вычислить по программе значения Y для значений аргумента X, равных 1,0485; —2,783; 0,003847; $-2{,}7 \cdot 10^{-2}$; $2{,}31485 \cdot 10^{-3}$; $0{,}27485 \cdot 10^{2}$.

2. Составить программу для вычисления длины c стороны треугольника по заданным значениям длин двух других его сторон a и b и угла α между ними, используя выражение

$$c = \sqrt{a^2 + b^2 - 2ab\cos\alpha}$$

(теорема косинусов), если значения a и b предварительно занесены в регистры Р1 и Р2 соответственно, а угла α (в градусах) — в регистр Р0. По составленной программе вычислить значения c, если a, b и α имеют значения:

$a = 2$, $b = 3$, $\alpha = 90°$; $a = 8{,}5$, $b = 7{,}34$, $\alpha = 140°$;
$a = 7$, $b = 4{,}5$, $\alpha = 120°$; $a = 15$, $b = 27$, $a = 30°$;
$a = 2{,}75$, $b = 3{,}48$, $\alpha = 45°$; $a = 1{,}8$, $b = 4{,}3$, $\alpha = 60°$.

3. Составить программу для вычисления работы A, выполняемой силой F на пути s, если угол между направлением силы и направлением перемещения равен α. Вычислить по программе значения $A = F\,s\cos\alpha$ для значений $F = 1{,}14$ Н, $s = 24{,}89$ м, $\alpha = 15$, 28, 35, 50, 62°.

4. Составить программу для вычисления высоты H, на которой будет лететь самолет, если давление атмосферы на этой высоте p, а у поверхности земли давление нормальное (p_0). Вычислить высоту по формуле

$$H = 1{,}8 \cdot 10^4 \lg(p_0/p)$$

для разных значений p.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Что называется программой решения задачи на МК? 2. Что называется командой для МК? 3. Какая программа называется линейной?

§ 11. ВЫЧИСЛЕНИЯ В АВТОМАТИЧЕСКОМ РЕЖИМЕ

При многократных вычислениях по одной и той же программе МК можно настроить на автоматическое выполнение вычислений. Для этого необходимо:

1) ввести все команды программы в полупостоянное запоминающее устройство (ППЗУ);

2) ввести исходные данные в оперативное запоминающее устройство (ОЗУ);

3) осуществить запуск автоматического выполнения команд программы.

После окончания вычислений МК можно перенастроить на решение задач другого типа, для чего необходимо ввести в ППЗУ новую программу. В ПМК «Электроника Б3-34» ППЗУ состоит из 98 запоминающих ячеек. Ячейки пронумерованы от 00 до 97. Номера ячеек называются *адресами* содержимого этих ячеек. В каждой ячейке можно записать двухсимвольный код одной операции (команды) или двухцифровой номер другой ячейки.

В табл. 1 приведены коды команд МК. Таблица кодов используется при проверке правильности ввода команд в ППЗУ. По коду, занесенному в ту или иную ячейку ППЗУ, можно восстановить операцию, имеющую такой код.

При вводе команд программы нет необходимости знать коды команд. МК автоматически вырабатывает код команды, набираемой на клавиатуре.

Прежде чем занести первую команду программы в ППЗУ, необходимо адрес, по которому записывается код первой команды, занести в *регистр адресов команд* (РАК). Этот регистр называется еще *счетчиком адресов команд*, или *программным счетчиком*.

Для занесения требуемого адреса в РАК нужно нажать три клавиши: клавишу с обозначением **БП** и две клавиши, на которых нанесены цифры, изображающие нужный адрес, например **БП 0 0** , **БП 0 7** , **БП 3 5**.

Для ввода информации в ППЗУ следует перейти в так называемый *режим программирования*. В этом режиме МК настраивают на автоматическое выполнение вычислений по программе, т. е. записывают в ППЗУ команды программы.

Никакие программы, хранящиеся в ППЗУ и в ПЗУ, в режиме программирования выполняться не могут.

Для перехода в режим программирования необходимо нажать две клавиши: клавишу с обозначением **F** и клавишу, над которой нанесена надпись **ПРГ**. Затем надо набрать программу один раз, т. е. нажать клавиши точно в той последовательности, как и при вычислениях вручную. Таким образом, информация в ППЗУ вводится с клавиатуры. При этом МК не выполняет операции, а только кодирует их и записывает коды операций в ячейки ППЗУ.

При переходе в режим программирования индикатор МК отключается от операционного регистра РХ и используется для визуального контроля ввода программы. В этом режиме на индикаторе высвечивается не более восьми цифр. Две правые из них изображают содержимое РАК, т. е. номер ячейки ППЗУ, куда будет заноситься информация. Два символа слева отображают код последней записанной в ППЗУ команды, адрес которой на единицу меньше, чем содержимое РАК.

Следующие вправо две пары символов изображают коды предыдущих команд, адреса которых на две и три единицы меньшие, чем содержимое РАК.

Пусть, например, в режиме программирования показание индикатора следующее:

	2	2		1	0		6	2		1	6

Число 16 изображает здесь содержимое РАК и показывает, что код вновь записываемой команды будет помещен в ячейку ППЗУ с номером 16. Число 22 — это код команды с адресом 15, число 10 — код команды с адресом 14, а число 62 — код команды с адресом 13.

После того, как программа будет полностью записана в ППЗУ, нужно выйти из режима программирования. Для этого достаточно нажать две клавиши: клавишу с обозначением **F** и клавишу, над которой нанесена надпись **АВТ**. Теперь ПМК может автоматически выполнять записанную программу. Режим, в котором МК автоматически выполняет программы, хранящиеся в ПЗУ или в ППЗУ, называется *режимом автоматической работы*, или *автоматическим режимом*. Занесение исходных данных в ОЗУ производится в режиме автоматической работы.

Перед началом работы в автоматическом режиме в РАК заносят адрес команды, с которой следует начать счет. Для

проверки содержимого РАК необходимо перевести ПМК в режим программирования.

Запуск автоматических вычислений по программе осуществляется в режиме автоматической работы нажатием клавиши с обозначением **С/П** (Стоп/Пуск). Нажатием этой же клавиши можно прервать автоматическое выполнение команд программы.

После окончания вычислений МК можно перенастроить на решение задач другого типа, для чего надо ввести в ППЗУ новую программу. Если предыдущую программу желательно сохранить, то вновь вводимую программу следует ввести в свободные ячейки, начиная ввод, например, с адреса, следующего за адресом последней команды сохраняемой программы.

Таким образом, для ввода программы в ППЗУ необходимо:

1. Занести в РАК адрес первой команды программы. Для этого в режиме автоматической работы надо нажать клавишу **БП** и две клавиши, на которых обозначены цифры нужного номера ячейки. Номер ячейки обязательно должен изображаться двухразрядным числом. Нулевой адрес в РАК можно занести также нажатием клавиши **В/О**.

2. Перевести ПМК в режим программирования, для чего нажать клавиши **F** и **ПРГ**.

3. Ввести программу, нажимая клавиши точно в той же последовательности, в какой их нажимают при ручном управлении вычислениями.

4. После ввода команды, в результате выполнения которой достигается окончательный результат вычислений, ввести еще одну дополнительную команду «Стоп», для чего нажать клавишу **С/П**. Эта команда прекращает работу процессора, хотя в РАК и находится уже адрес следующей команды.

5. Перевести ПМК в режим автоматической работы, для чего нажать клавиши **F** и **АВТ**.

Для запуска автоматических вычислений по программе необходимо:

1. Занести в РАК адрес команды, с которой следует начать вычисления, для чего в режиме автоматической работы нажать клавишу **БП** и две клавиши для набора адреса указанной команды. Если начальный адрес нулевой, то можно нажать также клавишу **В/О**.

2. Занести исходные данные в регистры ОЗУ, отведенные для хранения этих данных.

3. Нажать клавишу **С/П**.

При работе МК в автоматическом режиме наблюдается характерное помигивание индикатора, что свидетельствует о быстром изменении содержимого регистра РХ.

Порядок работы на ПМК при подготовке и запуске программ отражает таблица:

Номер по пор.	Этап решения задачи	Режим работы	Команды	
			подготовительные	основные
1	Составление программы на бланке	—	—	—
2	Ввод программы	Программирование	БП□□ FПРГ	Ввод программы в ППЗУ
3	Ввод исходных данных	Автоматический	FABT	Ввод значений исходных данных в оперативную память МК
4	Отладка программы	То же	БП□□	ПП, ПП, ПП, ...
5	Вычисления	»	То же	С/П

Прежде чем приступить к составлению программы, необходимо определить, в каких регистрах ОЗУ будут храниться исходные данные, т. е. значения переменных и постоянных величин, используемых в процессе вычислений, а также промежуточные результаты. Этот процесс называется *распределением памяти*. Необходимость в нем возникает в связи с тем, что в процессе вычислений одно и то же значение X может понадобиться многократно. Чтобы не вводить это значение каждый раз с клавиатуры, целесообразно (особенно, если значение X изображается многоразрядным числом) записать его в некоторый адресуемый регистр. Затем при необходимости это число можно вызвать из адресуемого регистра в регистр РХ, нажимая при этом только две клавиши: клавишу **ИП** и клавишу с номером регистра, в котором хранится данное значение X. Если, кроме того, вычисления проводятся для ряда значений переменной, то хранение этих значений в одном из адресуемых регистров дает возможность проводить вычисления по одной и той же программе, не меняя в ней ни одной команды. Последовательность нажимаемых клавиш при этом всегда остается одной и той же. Менять надо только исходные данные, т. е. содержимое адресуемых регистров, выделенных для хранения исходных данных.

Регистры ОЗУ, отводимые для хранения промежуточных результатов, называются *рабочими*. Чаще всего это операционные регистры стека, но могут быть и другие регистры стека либо адресуемые регистры. Рабочие регистры часто назначают уже в ходе составления программы. При этом, если некоторый промежуточный результат уже использован и в дальнейших вычислениях не понадобится, регистр, отведенный для его временного хранения, целесообразно использовать для хранения других промежуточных или окончательных результатов.

Пример. Пусть требуется рассчитать зависимость тока I в цепи, напряжения U и мощности P потребителя по заданным э. д. с. E, сопротивлению потребителя R (рис. 50) для значений

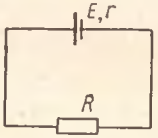


Рис. 50. К рассматриваемому примеру

$E = 100$ В, $r = 10$ Ом, если R принимает значения: 0, 5, 10, 15, 20, 25 Ом.

Значения I, U, R определяются выражениями

$$I = \frac{E}{R + r};\ U = IR;\ P = UI.$$

Составим программу вычисления этих значений. Для хранения значений E, r и R отведем адресуемые регистры P0, P1, P2 соответственно. Найденные в процессе вычисления значения I, U, P поместим в адресуемые регистры P3, P4, P5.

Программу вычислений запишем в виде таблицы, в которой вместо номеров команд будем указывать их адреса. Кроме того, добавим графу, в которой будем записывать коды команд. Поскольку в ППЗУ нет программ, которые желательно сохранить, команды вводимой программы начнем записывать с адреса 00. В итоге получим следующую таблицу:

Адрес команды	Команда	Код команды	Примечание	
00	ИП2	62	$R \to$ PX	$I = \frac{E}{r+R}$
01	ИП1	61	$r \to$ PX, $R \to$ PY	
02	+	10	$r + R \to$ PX	
03	ИП0	60	$E \to$ PX, $r + R \to$ PY	
04	$\overleftarrow{\text{XY}}$	14	$r + R \to$ PX, $E \to$ PY	
05	$\div$	13	$E/(r + R) \to$ PY	
06	П3	43	$E/(r + R) \to$ P3	
07	ИП2	62	$R \to$ PX, $I \to$ PY	$U = IR$
08	$\times$	12	$IR \to$ PX	
09	П4	44	$IR \to$ P4	

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание
10	ИП3	63	$I \to$ PX, $U \to$ PY
11	×	12	$UI \to$ PX $\}$ $P = UI$
12	П5	45	$UI \to$ P5
13	С/П	50	Стоп

Введем теперь программу в ППЗУ, начиная с адреса 00. Занесем в РАК адрес первой команды программы, нажав клавиши **БП 0 0.** Далее переходим в режим программирования, нажимая клавиши **F** и **ПРГ.** Показание индикатора МК при этом будет следующее:

			00

т. е. в РАК занесен адрес 00. Введем по этому адресу код первой команды, нажав клавиши **ИП** и **2.** Показание индикатора МК будет:

62			01

Таким образом, по адресу 00 введен код 62 (код команды ИП2), а содержимое РАК увеличено на 1. Введем по адресу 01 код второй команды, нажав клавиши **ИП** и **1.** Показание индикатора МК будет:

61	62		02

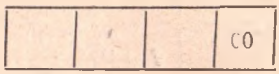
где 61 — код команды ИП1, введенной по адресу 01, а 62 — код команды ИП2, введенной по адресу 00. Введем по адресу 02 код следующей команды, нажав клавишу +. Показание индикатора МК будет:

10	61	62	03

Продолжив этот процесс, введем все остальные команды программы: ИП0; $\overleftarrow{XY}$; ÷; П3; ИП2; ×; П4; ИП3; ×; П5; С/П. После занесения команды по адресу 13 в РАК появляется адрес 14, однако по этому адресу никакую информацию заносить в рассматриваемом примере не требуется.

Переведем ПМК в режим автоматической работы, нажав клавиши **F** и **АВТ.** Однако программа к запуску еще не готова, так как в РАК находится адрес 14, а вычисления надо начать с выполнения команды ИП2, код которой хранится по адресу 00. Поэтому занесем в РАК адрес начальной команды, нажав клавиши **БП 0 0.**

Чтобы проверить, находится ли в РАК адрес 00, переведем МК в режим программирования, нажав клавиши **F** и **ПРГ**. При этом показание индикатора МК должно быть следующим:



Это означает, что в РАК занесен адрес 00. В случае, если показание индикатора будет иным, необходимо перейти в режим автоматической работы МК, нажав клавиши **F АВТ**, и повторить ввод в РАК адреса начальной команды программы.

При правильном показании индикатора МК переводят в режим автоматической работы нажатием клавиш **F АВТ** и вводят в ОЗУ исходные данные. В рассматриваемом примере для этого необходимо ввести в регистры Р0, Р1, Р2 значения $E = 100$, $r = 10$, $R = 0$. После нажатия клавиши **С/П** МК автоматически выполняет все команды программы без какого-либо вмешательства извне.

Когда вычисления закончатся, нужно вручную вывести на индикатор полученные значения $I = 10$, $U = 0$, $P = 0$ из отведенных для их хранения регистров Р3, Р4, Р5.

Выполним вычисления по программе, но уже для значения $R = 5$ Ом, для чего опять занесем в РАК адрес начальной команды программы, нажав клавиши **БП 0 0**; введем число 5 в регистр Р2 и нажмем клавишу **С/П**. В результате получим значения $I = 6{,}7$ А, $U = 33{,}3$ В, $P = 222{,}2$ Вт.

Продолжив вычисления на МК, для остальных значений R найдем:

R,	Ом	10	15	20	25
I,	А	5	4	3,3	2,9
U,	В	50	60	66,7	71,4
P,	Вт	250	240	222,2	204,1

Анализируя полученные результаты, можно сделать следующие выводы:

1) с увеличением сопротивления R ток в цепи уменьшается, а напряжение увеличивается;

2) мощность потребителя принимает наибольшее значение при $R = 10$ Ом, т. е. при $R = r$.

Упражнения. 1. Составить программы и выполнить по ним автоматические вычисления на МК значений заданных функций при заданных значениях аргументов, предполагая, что значение аргумента хранится в адресуемом регистре Р1:

а) $Y = \lg(X + 1) + \sqrt{X}$ для $X \geqslant 0$ на отрезке [0, 1];

б) $Y = 10^X + \dfrac{X^2 + \pi}{4}$ » $X \in R$ » [0, 1];

в) $Y = \dfrac{\operatorname{tg}(X/2) + 1}{X^3}$ » $X \neq \pi k,\ k \in z$ » [1, 2];

г) $Y = \sin\sqrt{X} + 3/X$ » $X > 0$ » [2, 3];

д) $Y = \dfrac{\cos X + 1}{X^2 + 1}$ » $X \in R$ » [0, 1].

2. Составить программу для определения смещения X, скорости v и ускорения a в любой момент времени t тела заданной массы

m при гармонических колебаниях под действием пружины с жесткостью k, если амплитуда колебаний X_m. Вычислить X, v, a при $m = 0{,}5$ кг, $k = 8$ Н/м, $X_m = 0{,}1$ м для моментов времени $t = 0$; 0,5; 1; 1,5; 2; 2,5; 3 с. Вычисления произвести по формулам

$$X = X_m \cos \omega t; \quad v = -X_m \sqrt{k/m} \sin \omega t; \quad a = -kX/m.$$

3. На горизонтальной платформе шахтной клети стоит человек массой $M = 62{,}5$ кг. Определить давление Q человека на платформу: а) при ее подъеме с ускорением 3 м/с²; б) при спуске с ускорением 3 м/с²; в) при равномерном подъеме и спуске; г) при спуске с ускорением 7,6 м/с². Вычисления произвести по формуле

$$Q = M(g + a),$$

где $g \approx 9{,}8$ м/с² — ускорение свободного падения; a — ускорение движения платформы.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как настраивается МК на автоматическое выполнение вычислений по программе? 6. Как записать новую программу в ППЗУ? 3. Что необходимо сделать для запуска автоматических вычислений на МК по программе, хранящейся в ППЗУ? 4. Что называется адресом команды? 5. Куда поступают адреса команд на МК при автоматическом выполнении программы? 6. Как перейти на МК в режим программирования? 7. Как перейти на МК в режим автоматической работы? 8. Как занести в РАК необходимый адрес? 9. Почему в программе вычислений на МК необходима команда «Стоп»? 10. Какую информацию высвечивает индикатор МК в режиме программирования? 11. Как проверить содержимое РАК? 12. Почему нецелесообразно хранить в ячейках ППЗУ многоразрядные числа? 13. Что называется распределением памяти?

§ 12. ОТЛАДКА ПРОГРАММ

При составлении программы и при ее вводе в ПМК возможно появление ошибок. Процесс их нахождения и устранения при работе по программе называется *отладкой программы*. Этот процесс — обязательный этап работы по подготовке программ к эксплуатации на больших и малых ЭВМ, в том числе и на ПМК.

Для отладки программы берут простой, но и достаточно общий контрольный пример, исходные данные которого подбирают так, чтобы результат вычислений можно было легко получить с помощью ручных (не автоматических) вычислений или даже устно. Вся программа расчленяется на отдельные участки, каждый из них проверяется (отлаживается) отдельно. Для проверки правильности работы некоторого участка в программу после последней команды данного участка временно заносят команду С/П («Стоп»). Затем переходят в автоматический режим работы МК,

заносят в РАК адрес первой команды участка и производят запуск вычислений в автоматическом режиме.

Если после окончания вычислений по командам данного участка программы все результаты, полученные автоматически, совпадут с наперед найденными контрольными результатами, то это означает, что на данном участке программы неправильных команд нет. После этого восстанавливают команду, вместо которой была занесена команда С/П. Для этого достаточно:

1) перейти в режим программирования, нажав клавиши **F** и **ПРГ**;

2) установить в РАК адрес, куда временно была занесена команда С/П, нажав клавишу $\overleftarrow{\text{ШГ}}$;

3) восстановить по этому адресу необходимую команду программы, для чего нажать клавишу с обозначением нужной операции.

Для увеличения адреса команды на 1 в режиме программирования следует нажать клавишу с обозначением $\overrightarrow{\text{ШГ}}$, а для уменьшения — $\overleftarrow{\text{ШГ}}$. Соответствующим количеством нажатий клавиш $\overrightarrow{\text{ШГ}}$ и $\overleftarrow{\text{ШГ}}$ можно занести в РАК любой адрес от 00 до 97.

Пусть, например, требуется проверить правильность участка программы:

11 ИП2	13 ÷
12 ИП1	14 F$\sqrt{-}$

Временно занесем команду С/П по адресу 15. Для этого в автоматическом режиме работы ПМК заносим в РАК адрес 15, нажимая клавиши **БП 1 5**. Затем переходим в режим программирования, нажав клавиши **F** и **ПРГ**. При этом показание индикатора МК будет следующим:

21	13	61	15

где 21, 13 и 61 — коды команд с адресами 14, 13 и 12 соответственно.

Далее нажимаем клавишу **С/П**. При этом показание индикатора МК будет:

50	21	13	16

Затем переходим в режим автоматической работы МК, нажав клавиши **F** и **АВТ.** Заносим в РАК адрес 11, нажимая клавиши **БП 1 1.** Нажав клавишу **С/П,** переходим к вычислениям по командам данного участка программы. Если результат автоматических вычислений совпадает с контрольным, то восстанавливаем команду с адресом 15. Для этого переходим в режим программирования, нажимая клавиши **F** и **ПРГ.** Показание индикатора МК будет следующим:

50	21	13	16

Уменьшим содержимое РАК на 1, нажав клавишу **ШГ.** Показание индикатора МК будет:

21	13	61	15

Восстанавливаем команду с адресом 15, нажимая клавиши **F sin.** Показание индикатора МК будет следующим:

1[	21	13	16

Если при реализации некоторого участка программы обнаруживают неправильные результаты, то такой участок расчленяют на более мелкие, каждый из которых проверяют точно так же, как и раньше.

Небольшие участки проверяют покомандно (пошагово), причем результат выполнения каждой команды сравнивают с наперед найденным контрольным результатом. Если после выполнения некоторой команды получен неправильный результат, то такая команда подлежит исправлению. Для этого необходимо:

1) перейти в режим программирования, нажав клавиши **F** и **ПРГ**;

2) занести в РАК адрес исправляемой команды, нажав клавишу **ШГ**;

3) по этому адресу занести необходимую информацию, для чего нажать клавишу с обозначением нужной операции:

После этого отладка программы продолжается аналогично описанной (возможно, с начала участка, на котором вносились исправления).

Для покомандной отладки программы необходимо:

1. Ввести программу в ППЗУ (в режиме программирования).

2. Перейти в автоматический режим работы, нажав клавиши **F** и **АВТ**.

3. Подобрать контрольный пример и определить контрольные результаты.

4. Ввести в адресуемые регистры МК исходные данные контрольного примера.

5. Занести в РАК начальный адрес программы (или начальный адрес участка, который надо проверить).

6. Проверить все команды программы, последовательно нажимая клавишу **ПП**.

При каждом нажатии клавиши **ПП** происходит выполнение команды, адрес которой находится в РАК. Результат выполнения команды высвечивается на индикаторе МК. Этот результат необходимо сравнить с контрольным. После выполнения команды содержимое РАК увеличивается на 1, тем самым подготавливается к выполнению следующая команда. Поэтому, если после выполнения очередной команды результат будет неправильным, необходимо:

1) перейти в режим программирования;

2) нажать клавишу **ШГ**;

3) набрать правильную команду по полученному адресу;

4) начать отладку участка или всей программы сначала (с п. 5).

Пример. Составить и отладить программу для вычисления значений функции $Y = \cos X^2 + X/3$.

Пусть, например, для хранения значений X отведен регистр Р1. Тогда программа принимает следующий вид:

Номер команды	Команда	Код команды	Примечание
00	ИП1	61	$X \to$ РХ
01	FX²	22	$X^2 \to$ РХ
02	F cos	1Г	$\cos X^2 \to$ РХ
03	П3	43	$\cos X^2 \to$ Р3
04	ИП1	61	$X \to$ РХ
05	3	03	$3 \to$ РХ, $X \to$ РУ
06	÷	13	$X/3 \to$ РХ
07	ИП3	63	$\cos X^2 \to$ РХ, $X/3 \to$ РУ
08	+	10	$\cos X^2 + X/3 \to$ РХ
09	С/П	50	Стоп

Введем программу в ППЗУ и перейдем в автоматический режим работы МК. Контрольные вычисления выполним для значения $X = 2$ (см. последнюю графу следующей таблицы). В регистр Р1 занесем значение $X = 2$, а в РАК — адрес 00.

После первого нажатия клавиши **ПП** на индикаторе МК должен появиться результат выполнения первой команды (ИП1), а именно: содержимое регистра Р1, т. е. число 2. Опять нажимаем клавишу **ПП**, на индикаторе должен появиться результат выполнения второй команды (FX^2), т. е. число 4. Параллельно нужно сопоставлять полученный на МК результат с контрольным. Если эти результаты совпадают, то выполняется следующая команда (нажатием клавиши **ПП**), результат которой проверяется аналогично предыдущему.

При пошаговой проверке программы и $X = 2$ получаем:

Номер команды	Команда	Код команды	Примечание	Номер нажатия клавиши ПП	Показание индикатора МК	Контрольный результат
00	ИП1	61	$X \rightarrow РХ$	1	2	2
01	FX^2	22	$X^2 \rightarrow РХ$	2	4	4
02	F cos	1Г	$\cos X^2 \rightarrow РХ$	3	—6.5364363—01	—0.65364363
03	П3	43	$\cos X^2 \rightarrow Р3$	4	—6.5364363—01	—0.65364363
04	ИП1	61	$X \rightarrow РХ$	5	2	2
05	3	03	$3 \rightarrow РХ$	6	3	3
06	÷	13	$X/3 \rightarrow РХ$	7	6.6666666—01	0.66666666
07	ИП3	63	$\cos X^2 \rightarrow РХ$, $X/3 \rightarrow РУ$	8	—6.5364363—01	—0.65364363
08	+	10	$\cos X^2 + X/3 \rightarrow РХ$	9	1.302303—02	0.01302303
09	С/П	50	Стоп			

Для того чтобы узнать, какую команду будет выполнять МК при пошаговой проверке программы (сколько раз уже нажата клавиша **ПП**), нужно перейти в режим программирования; РАК и покажет адрес следующей команды.

При несовпадении результатов МК переводят в режим программирования, нажимая клавиши **F** и **ПРГ**. Затем устанавливают в РАК адрес команды, по которой получен неправильный результат, для чего достаточно нажать клавишу **ШГ**. Неправильную команду исправляют, набирая правильную.

После этого МК переводят в режим автоматической работы, нажимая клавиши **F АВТ**, и продолжают отладку программы (возможно, начиная с самого начала в зависимости от создавшейся ситуации).

Так выполняя в пошаговом режиме все команды программы и каждый раз анализируя получаемые результаты, можно выявить неправильные команды или же убедиться в правильности всех команд, т. е. программы в целом.

Упражнение. В регистре Р1 находится значение аргумента X, при котором требуется выполнить автоматические вычисления по формуле

$$Y = \frac{\sin X^2 + 2X}{X^2 + 5}.$$

В ячейки ППЗУ, начиная с адреса 00, занесены коды команд:

Номер команды	Код	Номер команды	Код	Номер команды	Код
00	61	06	02	12	05
01	22	07	12	13	10
02	42	08	63	14	63
03	1[	09	10	15	14
04	43	10	43	16	13
05	61	11	62	17	50

Используя эту таблицу, по кодам восстановить команды и программу в целом. Правильно ли составлена программа?

Выполнить программу в покомандном режиме, используя клавишу **ПП**.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Что называется отладкой программы? 2. Как проверяется правильность отдельного участка программы? 3. Как восстанавливается команда, вместо которой временно была занесена команда «Стоп»? 4. Как исправить команду, если обнаружено, что после ее выполнения получается неправильный результат? 5. Что происходит в результате нажатия клавиши **ПП** в автоматическом режиме работы ПМК? 6. Как по коду операции определить саму операцию? 7. Как реализуется покомандное выполнение программы на МК? 8. В каком режиме работы МК производится отладка программы? 9. Для чего предназначены клавиши $\overrightarrow{\text{ШГ}}$ и $\overleftarrow{\text{ШГ}}$ на МК?

§ 13. АЛГОРИТМЫ

Программы могут быть составлены не только как последовательности команд для МК. Это могут быть наборы указаний, рассчитанные и на другого исполнителя (например, человека, ЭВМ, систему «человек — машина», некоторый автомат, робот).

Любой конечный набор предписаний, выполнение которых одно за другим через конечное число шагов приводит к решению некоторой задачи, называется *алгоритмом* (или программой) решения этой задачи. Каждый алгоритм — это правило, указывающее действия, в результате выполнения

цепочки которых от исходных данных приходят к искомому результату. Такая цепочка действий называется *алгоритмическим процессом*, а каждое действие — его *шагом*. Каждый алгоритм предполагает наличие некоторых исходных данных и приводит к получению определенного искомого результата.

Из курса алгебры, например, хорошо известны алгоритмы (правила) решения квадратных уравнений, систем линейных уравнений, раскрытия скобок и приведения подобных членов в буквенных выражениях и т. п. Но алгоритмы широко распространены и в других науках. Если сформировать все правила, употребляемые опытным переводчиком для переводов, скажем, с английского языка на русский, то получим не что иное, как алгоритм англо-русского перевода. Если элементарные правила шахматной игры дополнить системой стратегических правил, позволяющих в каждой позиции находить единственный наилучший (с точки зрения данной системы правил) ход, то получится алгоритм игры в шахматы.

Теоретически чуть ли не всякий вид умственной деятельности человека может быть сведен к выполнению того или иного алгоритма. Но практически найти правила, определяющие эти алгоритмы,— очень сложная и трудоемкая задача.

В повседневной жизни приходится сталкиваться с различными алгоритмами, не связанными с вычислительными процессами, но определяющими ту или иную последовательность действий самой разнообразной природы. Например, последовательность действий, которые необходимо выполнить, чтобы позвонить по междугородному телефону, изложенная в инструкции, прилагаемой к телефону, представляет собой алгоритм. Точно выполняя указания инструкции, абонент может позвонить в нужный ему пункт.

Последовательности действий, которые необходимо выполнить, чтобы разобрать или собрать некоторый механизм, представляют собой алгоритмы разборки и сборки этого механизма. Рецепты для приготовления пищи, лекарств, инструкции по вводу в действие электро- или радиоприборов и многие другие последовательности предписаний или рекомендаций являются примерами алгоритмов. Можно говорить об алгоритме изготовления автомобиля, алгоритме управления полетом космического корабля и др. Правила выполнения операций сложения, вычитания, умножения, деления чисел представляют собой примеры численных алгоритмов.

Использование алгоритмов дает возможность автоматизировать многие производственные процессы, организовывать и направлять деятельность человека, переложить многие управляющие и контролирующие функции на автоматы в самых различных областях человеческой деятельности.

Алгоритмы должны обладать свойствами:

1. Массовости. Это означает, что алгоритм должен быть применим ко всем задачам рассматриваемого типа, при любых исходных данных, а не только к какому-то ее отдельному варианту. Например, алгоритм нахождения общего знаменателя двух дробей применим к любым двум дробям; алгоритм вычисления площади трапеции по заданным основаниям и высоте приводит к решению задачи, какая бы ни была трапеция.

2. Определенности (детерминированности). Любое указание алгоритма должно быть строго определено и не допускать неоднозначного толкования. Строго определен должен быть и порядок выполнения указаний алгоритма. Например, указание «Сравнить числа a и b» может быть истолковано по-разному и не является определенным. Такие указания в алгоритмах недопустимы. Указание же «Если число a меньше числа b, то переменной u присвоить значение a, в противном случае переменной u присвоить значение b» не допускает неоднозначного толкования и вполне может быть указанием некоторого алгоритма.

3. Результативности. Алгоритм всегда должен приводить к результату через конечное, хотя, возможно, и очень большое число шагов (действий).

4. Формальности. Любой исполнитель, способный воспринимать и выполнять указания алгоритма (возможно, даже не понимая их смысла), правильно выполнит весь алгоритм. Например, вычислительная машина правильно решает задачи по заданным ей алгоритмам (программам), хотя, безусловно, суть задач машина понимать не может.

5. Дискретности. Это означает, что всякий процесс, определяемый алгоритмом, должен иметь дискретный (прерывистый) характер, т. е. представлять собой последовательность выполняемых один за другим отдельных законченных шагов.

Не всякая последовательность инструкций удовлетворяет перечисленным требованиям, и поэтому не всякую последовательность инструкций можно называть алгоритмом. Например, правило «Во время движения по тротуару придерживайся правой стороны», хотя и является предпи-

санием, но имеет непрерывный характер и потому не относится к алгоритмам.

Существуют различные способы описания алгоритмов — *словесные*, *словесно-формульные*, *графические*, в виде *последовательности кодов* из некоторого конечного набора кодов и другие в зависимости от того, на какого исполнителя ориентирован алгоритм. Примером словесного описания алгоритмов есть рецепты приготовления пищи, инструкции по приему лекарств.

В словесной форме можно представить и математические правила, например, известный алгоритм нахождения наибольшего общего делителя двух чисел (алгоритм Евклида), правила выполнения арифметических действий, вычисления корней квадратного трехчлена, алгоритм нахождения квадратного корня из положительного числа, логарифма произведения чисел по известным логарифмам сомножителей, длины окружности, площади треугольника.

Алгоритмы вычислительных процессов можно задавать и в виде *формул*. Например, алгоритм вычисления суммы членов бесконечно убывающей геометрической прогрессии можно представить в виде $s = \frac{b_1}{1-q}$, а алгоритм нахождения логарифма произведения чисел X и Y по известным логарифмам сомножителей — в виде

$$\lg (XY) = \lg X + \lg Y.$$

При записи алгоритмов часто комбинируют словесную и формульную формы записи указаний. Например, алгоритм нахождения действительных корней квадратного трехчлена $X^2 + pX + q$ можно представить так:

1. Найти $(-p/2)^2 - q$. Перейти к выполнению указания 2.
2. Проверить, выполняется ли условие $(-p/2)^2 - q \geqslant 0$. Если выполняется, то перейти к выполнению указания 5, если нет,— к выполнению указания 3.
3. Ответ представить в виде «Действительных корней не существует». Перейти к выполнению указания 4.
4. Вычисления прекратить.
5. Найти

$$X_1 = -\frac{p}{2} + \sqrt{\left(-\frac{p}{2}\right)^2 - q};$$

$$X_2 = -\frac{p}{2} - \sqrt{\left(-\frac{p}{2}\right)^2 - q}.$$

Перейти к выполнению указания 4.

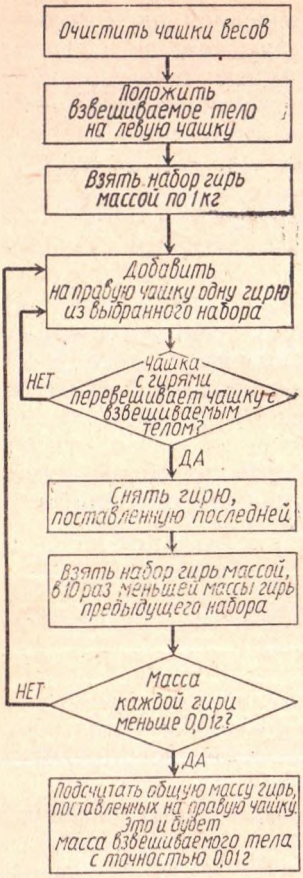


Рис. 51. Алгоритм взвешивания тела

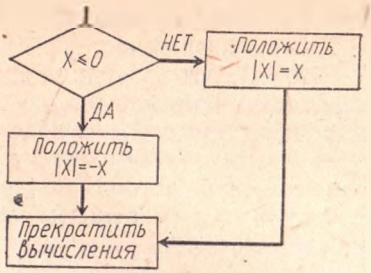


Рис. 52. Алгоритм вычисления модуля числа X

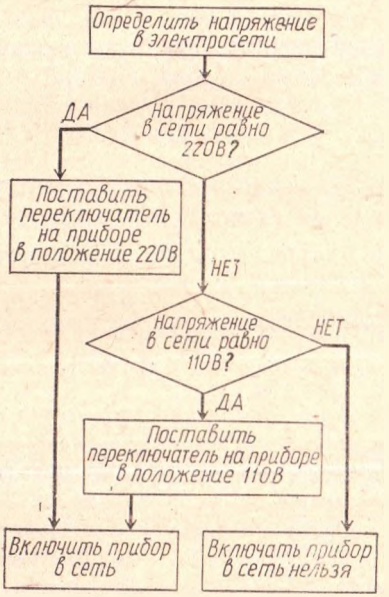


Рис. 53. Алгоритм включения электроприбора в сеть

На рис. 51 изображен алгоритм взвешивания тела, масса которого не превышает 10 кг. При этом используются наборы по 10 гирь такие, что в первом наборе содержатся гири массой по 1 кг, а в каждом следующем их масса в 10 раз меньше, чем в предыдущем. Гири разрешается ставить только на одну чашку весов, а взвешивание требуется выполнить с точностью до 0,01 г.

Иногда алгоритм представляют, комбинируя словесную, формульную и графическую формы. Например, алго-

ритм вычисления модуля числа X можно представить так, как показано на рис. 52.

На рис. 53 показан алгоритм включения электроприбора в сеть.

Упражнения. 1. Записать в словесной форме алгоритм сложения двух многоразрядных чисел.

2. Записать в словесной форме алгоритм вычисления площади треугольника по заданным длинам его сторон.

3. В словесно-формульной форме записать алгоритм определения расстояния между двумя точками по заданным координатам этих точек.

4. В словесно-формульной форме записать алгоритм решения одного линейного уравнения с одним неизвестным по заданным коэффициентам уравнения.

5. Представить алгоритм определения большего из двух заданных чисел, комбинируя словесную, формульную и графическую формы.

6. Представить алгоритм определения знака произведения по знакам сомножителей, комбинируя словесную, формульную и графическую формы.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Что называется алгоритмом? 2. Приведите примеры алгоритмов. 3. Какими свойствами должны обладать алгоритмы? 4. Всякую ли последовательность инструкций можно называть алгоритмом? 5. Назовите основные формы описания алгоритмов.

§ 14. АЛГОРИТМИЧЕСКИЕ ЯЗЫКИ И СХЕМЫ АЛГОРИТМОВ

Алгоритм решения одной и той же задачи может быть описан разными средствами. Различные совокупности средств для описания алгоритмов называются *алгоритмическими языками*, или *языками программирования*. Другими словами, алгоритмический язык — это совокупность определенных символов и правил, устанавливающих, как с помощью этих символов описывать алгоритмы. Примером алгоритмического языка может быть система команд для описания алгоритмов решения задач на МК.

Переход от описания алгоритма решения некоторой задачи на одном алгоритмическом языке к описанию алгоритма решения той же задачи на другом алгоритмическом языке называется *переводом* описания алгоритма с одного алгоритмического языка на другой, или *трансляцией*.

Перед описанием алгоритмов вычислительных процессов на том или ином алгоритмическом языке для более наглядного представления логической структуры вычислительного процесса алгоритмы представляют в графической

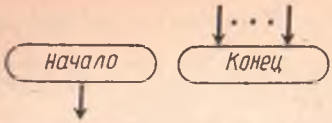


Рис. 54. Изображение операторов «Начало» и «Конец» на схемах алгоритмов

форме. Для этого выделяют некоторые подзадачи основной задачи, последовательность решения которых на схемах указывают стрелками.

Впредь, прежде чем составлять программу решения задачи для ПМК, схему алгоритма и его запись будем представлять в словесно-формульной форме, записывая в каждом пункте словесно-формульного описания алгоритма отдельную подзадачу. Затем каждый пункт этого описания будем переводить на язык МК, т. е. составлять программы, по которым можно решать подзадачи, сформулированные в отдельных пунктах алгоритма. Набор таких программ и будет программой решения задачи в целом.

Словесно-формульная форма описания алгоритмов решения задач называется *учебным алгоритмическим языком* (УАЯ). Этот язык весьма близок к реально используемым при решении задач на ЭВМ алгоритмическим языкам.

Указание о выполнении некоторого действия называется *оператором*. В УАЯ используется восемь операторов. Поясним назначение каждого из них.

Оператор начала алгоритма на УАЯ записывается в виде

МО: Начало;

Этот оператор нацеливает исполнителя на то, что выполнение указаний алгоритма необходимо начать с оператора, следующего за данным. МО — так называемая *метка* (обозначение, имя) оператора. Метки могут быть поставлены перед всеми операторами. Метка отделяется от оператора двоеточием. Необходимо помнить правила образования меток: первым символом метки должна быть буква, далее могут следовать буквы либо цифры. В конце любого оператора должен стоять символ «;» (точка с запятой) и никакой другой. Символ «;» служит признаком того, что описание оператора закончено.

На схемах алгоритмов оператор Начало; изображают в овале (рис. 54), от которого исходит стрелка, указывающая на операцию, к выполнению которой необходимо перейти. К символу «Начало» не подходит ни одна стрелка на схеме, так как нет предшествующих операций, после выполнения которых нужно перейти к операции «Начало».

Оператор конца алгоритма на УАЯ записывается в виде

Конец;

Этот оператор указывает исполнителю на то, что дальнейшие действия необходимо прекратить. На схемах алгоритмов оператор Конец; также изображают в овале (см. рис. 54), от которого не выходит ни одна стрелка, поскольку нет операций, к выполнению которых нужно перейти после операции «Конец». К символу «Конец» на схеме может подходить несколько стрелок от других ее элементов (символов), выполнением функций которых заканчивается алгоритм.

Описывая алгоритмы на УАЯ, необходимо придерживаться определенных правил. Допускается использование любых символов, обычно употребляемых в математике, т. е. больших и малых букв русского, латинского и греческого алфавитов, арабских цифр 0, 1, 2, ..., 9, знаков математических операций «+» (сложение), «—» (вычитание), «/» (деление), «×» (умножение), знаков соотношений «<» «≤», «=», «≠», «>», «≥», разделительных знаков «,» «.», «;», «:», «(», «)» и т. д. При записи чисел целая часть от дробной отделяется точкой, так как символ «,» используется для отделения элементов один от другого при их перечислении. Знак умножения иногда изображается символом·).

Переменная — это величина, принимающая на данном этапе вычислений определенное значение. Обозначение (наименование) переменной величины называется *идентификатором* (именем) этой переменной. Идентификатор может быть составлен из одного или нескольких символов, среди которых допускаются только буквы и цифры, причем первым символом обязательно должна быть буква (например, *A1*, *AB*, *C01*).

Выражение — это текст, задающий правило вычисления одного числового или логического значения. Если вычисляемое значение числовое, то выражение называется *арифметическим*, а если логическое, то — *логическим*. Например: $(A1 + B1) \times C2$, $X^2 + Y^2 - 1$, -3.5, $-X$.

Оператор ввода на УАЯ имеет вид

Ввести (*X*, *Y*, *Z*, ...);

Здесь *X*, *Y*, *Z*, ... — переменные, значения которых необходимо ввести в ячейки ОЗУ, отведенные для хранения этих значений. Переменные *X*,*Y*,*Z*,... называются *элемен-*

тами списка ввода и обязательно заключаются в скобки, даже если в списке один элемент. Друг от друга в списке элементы отделяются запятыми. Количество элементов в списке ввода может быть сколь угодно большим, но конечным (не превышающим объема ОЗУ). Если, например, при распределении памяти в МК для хранения значения переменной X отведен регистр Р2, а переменной h — регистр Р5 и если программу вычислений необходимо реализовать для варианта, когда $X = 1{,}785$ и $h = 0{,}015$, то, выполняя указание

Ввести (X, h);

в регистр Р2 надо ввести число 1,785, а в регистр Р5 — число 0,015.

Оператор вывода на УАЯ записывается в виде

Вывести (U, V, W, ...);

Здесь U, V, W, ... — переменные, значения которых необходимо вывести из ячеек ОЗУ, отведенных для их хранения в процессе распределения памяти. Список элементов вывода, как и элементов ввода, обязательно заключается в скобки, даже если в нем один элемент; один от другого в списке элементы также отделяются запятыми. Количество элементов в списке вывода может быть как угодно большим. В отличие от списка элементов ввода, в список элементов вывода, кроме обозначений переменных величин, могут входить так называемые *символьно-строчные константы*, т. е. некоторые наборы символов, взятые в кавычки. При выводе эти наборы символов просто воспроизводятся на бумаге. Если, например, для хранения значений переменной X в МК отведен регистр Р2, а переменной Y — регистр Р3 и если в Р2 хранится число 0,75, а в Р3 — число — 1,589, то, выполняя указание

Вывести ('$X =$', X, '$Y =$', Y);

исполнитель должен записать на бумаге $X = 0{,}75$ и $Y = -1{,}589$.

Выполнение операций ввода и вывода в автоматическом режиме на ПМК «Электроника Б3-34» не предусмотрено. При работе на данном ПМК управление вводом и выводом информации осуществляют вручную с клавиатуры. На схемах алгоритмов операторы ввода и вывода изображают в виде параллелограммов (рис. 55), от каждого из которых исходит одна стрелка, указывающая на операцию, к выполнению которой необходимо перейти. К каждому из дан-

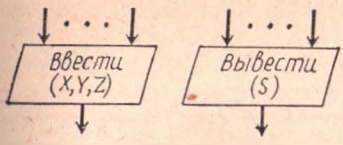


Рис. 55. Изображение операторов «Ввод» и «Вывод» на схемах алгоритмов

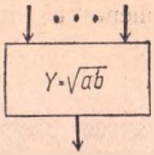


Рис. 56. Изображение вычислительных операций на схемах алгоритмов

ных символов может подходить несколько стрелок от других элементов (символов) схемы, после выполнения функций которых следует перейти к выполнению функций рассматриваемого символа.

Оператор присваивания (арифметический оператор) на УАЯ имеет вид

$$u = A;$$

где u — обозначение переменной величины; A — некоторое выражение. Выполняя указание

$$u = A;$$

исполнитель сначала должен вычислить значение выражения A, записанного справа от знака равенства, затем — присвоить это значение переменной u, т. е. записать его в ту ячейку ОЗУ, которая отведена для хранения значений переменной u.

Если, например, для хранения значений переменной X в МК отведен регистр Р2, а переменной Y — регистр Р3 и если в Р2 хранится число — 2,84, в Р3 — число 3,48, а оператор присваивания

$$Y = Y + X/2;$$

то, вычислив значение арифметического выражения справа от знака равенства и учтя, что на момент вычисления этого значения переменная X имеет значение —2,84, а переменная Y — значение 3,48, получим

$$3{,}48 + (-2{,}84) / 2 = 2{,}06.$$

Оператор присваивания предписывает в данном случае полученное значение 2,06 присвоить переменной Y, т. е. записать его в регистр Р3, отведенный для хранения значений переменной Y. В результате содержимое регистра Р2 останется без изменений, а в регистр Р3 будет записано число 2,06 — новое значение Y вместо предыдущего 3,48.

В операторе присваивания запись вида

$$u = A;$$

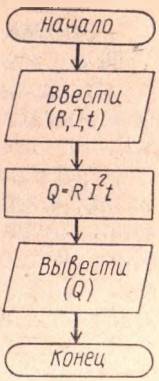
не является равенством, в котором к его обеим частям можно прибавить или отнять одно и то же выражение, умножить обе части равенства на одно и то же число. Ни один символ в операторе присваивания не может быть изменен. Указания этого оператора должны выполняться в строгом соответствии с его определением без каких-либо упрощений или преобразований.

На схемах алгоритмов вычислительные операции показывают в прямоугольниках (рис. 56), из которых исходят (по одной) стрелки, указывающие на операции, к выполнению которых необходимо перейти. К этим прямоугольникам может подходить несколько стрелок от других элементов (символов) схемы, после выполнения функций которых следует перейти к выполнению функций рассматриваемого символа.

Составление схемы алгоритма должно предшествовать его описанию на том или ином алгоритмическом языке. Не существует правил, по которым определяются подзадачи, выделяемые в отдельные элементы схемы. Отдельный элемент в схеме в свою очередь может быть расчленен на подэлементы, расположенные в определенной последовательности, так что алгоритм решения подзадачи, выделенной в отдельный элемент схемы, может быть довольно сложным. Внутри символа, отображающего те или иные операции на схеме, записи могут быть произвольной формы, лишь бы ясна была постановка задачи, выделенной в данном элементе.

В отдельные элементы схемы могут выделяться и довольно сложные задачи, и задачи, которые можно решить одной командой для МК. Однако при чрезмерной детализации алгоритма, выделении в отдельные элементы слишком мелких задач теряются все преимущества схем, так как в слишком детализованной схеме ориентироваться ничуть не проще, чем в программе решения задачи на МК. Такие слишком детализованные схемы не делают более наглядной логическую структуру вычислительного процесса и не облегчают его алгоритмизацию. Слишком крупные элементы также не облегчают алгоритмизацию решения задачи, поскольку задача при этом плохо расчленяется на подзадачи, программирование которых затем уже не представляет затруднений. Таким образом, схема алгоритма

Рис. 57. Схема алгоритма вычисления количества теплоты Q



есть более или менее подробный графически представленный план решения задачи.

С помощью рассмотренных операторов на УАЯ можно описывать произвольные линейные алгоритмы (программы) вычислительных процессов. Например, алгоритм вычисления количества теплоты Q, выделяемого в цепи сопротивлением R при протекании тока I за время t, на УАЯ может быть таким:

Оператор (команда) на УАЯ	Примечание
М0: Начало;	Начать работу
М1: Ввести (R, I, t);	Переменным R, I, t присвоить конкретные значения, для которых нужно провести вычисления
М2: $Q = R \times I^2 \times t$;	Вычислить искомый результат — количество выделяемой теплоты
М3: Вывести (Q);	Найденный результат вывести на дисплей или на бумагу
М4: Конец;	Работу по данной программе прекратить

Схема данного алгоритма показана на рис. 57.

Алгоритм вычисления площади трапеции по заданным значениям длин ее диагоналей и угла между ними на УАЯ может быть таким:

М0 : Начало;
М1 : Ввести (d_1, d_2, α);
М2 : $S = (1/2)\, d_1 \times d_2 \times \sin\alpha$;
М3 : Вывести (S);
М4 : Конец;

Алгоритм вычисления значения функции $Y = (\lg X)^2 + X/2$ для заданного значения аргумента X на УАЯ может быть таким:

М : Начало;
М0 : Ввести (X);
М1 : $Y = (\lg X)^2 + X/2$;
М2 : Вывести (X, Y);
М3 : Конец;

Переведем теперь этот алгоритм, написанный на УАЯ, на язык МК. Для хранения значений переменной величины

X отведем адресуемый регистр P1. Будем транслировать на язык МК каждый отдельный пункт алгоритма.

Первый пункт

М: Начало;

Этому пункту не ставятся в соответствие никакие команды МК.

Второй пункт

М0: Ввести (X);

Этому пункту также не ставятся в соответствие никакие команды МК. Ввод заданного значения переменной X осуществляется вручную после того, как программа для МК будет введена в ППЗУ.

Третий пункт

М1: $Y = (\lg X)^2 + X/2$:

Поскольку ППЗУ свободно, команды программы, соответствующие этому пункту алгоритма, можно вводить в ППЗУ, начиная с адреса 00. Учтя распределение памяти, получим следующую программу, по которой выполняется данный пункт алгоритма:

Адрес команды	Команда	Код команды	Примечание
00	ИП1	61	$X \to$ PX
01	F lg	17	$\lg X \to$ PX
02	FX²	22	$(\lg X)^2 \to$ PX
03	П2	42	$(\lg X)^2 \to$ P2
04	ИП1	61	$X \to$ PX
05	2	02	$2 \to$ PX, $X \to$ PY
06	÷	13	$X/2 \to$ PX
07	ИП2	62	$(\lg X)^2 \to$ PX, $X/2 \to$ PY
08	+	10	$(\lg X)^2 + X/2 \to$ PX

Четвертый пункт

М2: Вывести (X, Y);

реализуется вручную.

Пятый пункт

М3: Конец;

Этому пункту соответствует команда:

09	С/П	50	Останов для вывода значений Y и X

Пусть теперь необходимо построить таблицу значений функции $Y = (\lg X)^2 + X/2$ на участке [0,1; 3,5]. Выбрав равноотстоящие значения абсцисс X, для каждого из них на МК вычисляем значение ординаты Y по одной и той же

программе. При этом каждое последующее значение абсциссы будет отличаться от предыдущего на одно и то же число h, называемое *шагом изменения аргумента*. Пусть для рассматриваемого примера $h = 0{,}2$. Тогда, если $X_1 = 0{,}1$, то $X_2 = X_1 + h = 0{,}1 + 0{,}2 = 0{,}3$. Для каждого выбранного таким образом значения X будем находить соответствующее значение Y, причем каждый раз перед началом вычислений по программе в адресуемый регистр P1, отведенный для хранения значений X, будем помещать очередное значение абсциссы. Кроме того, в РАК будем заносить начальный адрес программы, нажимая клавиши **БП 0 0**. Только после этого можно осуществить запуск программы, нажав клавишу **С/П**.

Упражнения. Составить схемы, написать на УАЯ и перевести на язык ПМК алгоритмы решения следующих задач:

1. По включенным параллельно в электрическую цепь четырем сопротивлениям R_1, R_2, R_3, R_4 определить общее сопротивление цепи по формуле

$$R = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}}.$$

2. Световой луч падает в воздухе на поверхность некоторой среды. Найти показатель преломления среды, если заданы угол падения α и угол преломления γ. Показатель преломления рассчитать по формуле

$$n = \sin \alpha / \sin \gamma.$$

3. На мачте висит лампа с заданной силой света I. Какова освещенность почвы в точке, удаленной на расстояние r от лампы, если угол падения лучей в эту точку равен α? Освещенность вычислить по формуле

$$E = \frac{I}{r^2} \cos \alpha.$$

4. По заданным первому члену арифметической прогрессии a_1, разности d и числу n ее членов найти сумму всех членов прогрессии, используя формулу

$$s = \frac{2a_1 + (n-1)\,d}{2}\, n.$$

5. По заданным числу сторон n правильного многоугольника и радиусу R описанной окружности рассчитать площадь многоугольника, используя формулу

$$S = \frac{1}{2}\, nR^2 \sin \frac{2\pi}{n}.$$

6. По заданным стороне a и трем углам A, B, C треугольника определить его площадь, используя формулу

$$S = \frac{a^2 \sin B \sin C}{2 \sin A}.$$

7. Составить схемы, написать на УАЯ и перевести на язык ПМК алгоритмы вычисления значений заданных функций при заданных значениях аргумента:

а) $Y = (\sin X + 3\cos X)/(1 + X^4)$;

б) $Y = \operatorname{tg} X/(1 + \lg^2 X)$;

в) $Y = (X^2 + 3X + 7)/(X^2 - 2X + 9)$.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Что называется алгоритмическим языком? 2. Что называется трансляцией с одного алгоритмического языка на другой? 3. Какие символы используются при описании алгоритмов на УАЯ? 4. Назовите правила образования идентификаторов. 5. Что называется оператором? 6. Назовите основные операторы УАЯ. 7. Для чего используется символ «;»? 8. Что называется меткой оператора? 9. Для чего используется символ «:»? 10. Назовите правила образования меток. 11. Можно ли ставить метки перед всеми операторами? 12. Каким символом в УАЯ обозначена операция деления? 13. Что называется схемой алгоритма? 14. Как на схемах алгоритмов изображают операторы «Начало» и «Конец»? 15. Сколько стрелок может входить в символ «Начало» на схеме алгоритма? 16. Сколько стрелок может входить в символ «Конец» на схеме алгоритма? 17. Сколько стрелок может выходить из символа «Конец» на схеме алгоритма? 18. В каком виде на УАЯ записывают оператор ввода? 19. Сколько элементов можно записать в списке элементов ввода? 20. В каком виде на УАЯ записывают оператор вывода? 21. Какие элементы могут быть включены в список элементов вывода? 22. Как на схемах алгоритмов изображают операции ввода — вывода? 23. Какой вид имеет на УАЯ оператор присваивания? 24. Обладает ли оператор присваивания свойствами обычных равенств? 25. Как на схемах алгоритмов изображают вычислительные операции? 26. По каким правилам определяются подзадачи, выделяемые в отдельные элементы схем алгоритмов? 27. В какой форме должны быть выполнены записи внутри символов, обозначающих на схемах алгоритмов отдельные операции? 28. Можно ли в отдельный элемент на схеме алгоритма выделить операцию, которую можно выполнить на ПМК?

§ 15. ОПЕРАТОРЫ УСЛОВНОГО И БЕЗУСЛОВНОГО ПЕРЕХОДОВ. ЦИКЛИЧЕСКИЕ ВЫЧИСЛИТЕЛЬНЫЕ ПРОЦЕССЫ

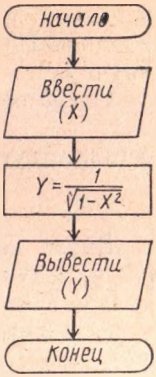
Пусть требуется в координатах m/m_0 и v/c построить график зависимости массы тела от скорости. Эта зависимость задается формулой

$$\frac{m}{m_0} = \frac{1}{\sqrt{1 - (v/c)^2}}.$$

Обозначив $m/m_0 = Y$, $v/c = X$, получим

$$Y = \frac{1}{\sqrt{1 - X^2}}.$$

Рис. 58. Схема алгоритма вычисления значения функции $Y = 1/\sqrt{1 - X^2}$ при заданном значении X



Для построения графика выберем на осях равноотстоящие значения, начиная от $X = 0$, через шаг $h = 0{,}1$ до $X = 1$. Затем вычислим для всех этих значений X соответствующие значения Y.

Схема алгоритма вычисления значения Y при заданном значении X показана на рис. 58. На УАЯ этот алгоритм можно представить в виде:

M0 : Начало;
Ввести (X);
M1 : $Y = 1/\sqrt{1 - X^2}$;
Вывести (Y);
Конец;

Переведем алгоритм на язык ПМК. Операторам

M0 : Начало;
Ввести (X);

на языке ПМК не ставятся в соответствие никакие команды (ввод исходных данных в ОЗУ ПМК осуществляется вручную). Оператор

$$M1 : Y = 1/\sqrt{1 - X^2};$$

при условии, что значение аргумента X хранится в адресуемом регистре Р1, можно реализовать на ПМК с помощью следующих команд:

Адрес команды	Команда	Код команды	Примечание
00	ИП1	61	$X \to$ РХ
01	F X^2	22	$X^2 \to$ РХ
02	1	01	$1 \to$ РХ, $X^2 \to$ РУ
03	$\overleftarrow{\text{XY}}$	14	$X^2 \to$ РХ, $1 \to$ РУ
04	$\longrightarrow$	11	$1 - X^2 \to$ РХ
05	F $\sqrt{\ }$	21	$\sqrt{1 - X^2} \to$ РХ
06	F1/X	23	$1/\sqrt{1 - X^2} \to$ РХ

Оператору

Вывести (Y);

в данной программе не ставятся в соответствие никакие команды ПМК (вывод значений Y на бумагу производится вручную).

Оператору

Конец;

соответствует команда:

07 | С/П | 50 | Стоп

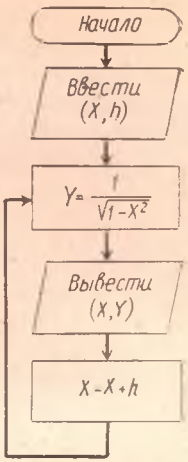
Вводя каждый раз значение X в адресуемый регистр Р1 и запустив вычисления по программе (с адреса 00), получим все необходимые значения Y. При этом каждое новое значение аргумента вычисляется (увеличивается на шаг h по сравнению с предыдущим значением) и вводится в регистр Р1 вручную. Эти операции на ПМК можно выполнять автоматически. Для хранения значений шага h отведем, например, адресуемый регистр Р0. Начав с адреса 08, допишем в программу следующие команды:

Адрес команды	Команда	Код команды	Примечание	Соответствующий пункт алгоритма
08	ИП1	61	$X \to$ РХ	$X = X + h;$
09	ИП0	60	$h \to$ РХ, $X \to$ РУ	
10	+	10	$X + h \to$ РХ	
11	П1	41	$X + h \to$ Р1	
12	БП	51	Занесение в РАК начального адреса программы 00 и переход на выполнение команды с адресом 00	Перейти к М1;
13	00	00		

Здесь команда с адресом 08 вызывает из регистра Р1 значение X в операционный регистр РХ, а команда с адресом 09 — из регистра Р0 в регистр РХ значение h. При этом значение X из регистра РХ перемещается в регистр РУ. По команде с адресом 10 содержимые операционных регистров РХ и РУ складываются, причем результат $X + h$ помещается в регистр РХ. По команде с адресом 11 содержимое регистра РХ пересылается в регистр Р1, отведенный для хранения значений переменной X. Таким образом, в регистр Р1 записывается новое значение переменной X взамен хранившегося там предыдущего значения той же переменной.

Команда, хранящаяся по адресам 12 и 13, заносит в РАК адрес 00, т. е. сразу после команды, хранящейся по адресам 12 и 13, будет выполняться команда, хранящаяся по адресу 00. Команда, хранящаяся по адресам 12 и 13,

Рис. 59. Последовательность вычисления значений функции $Y = 1/\sqrt{1 - X^2}$



позволяет нарушить естественный порядок выполнения команд, т. е. порядок, в котором записаны команды. Такого вида команды дают возможность осуществить переход на выполнение любой команды программы непосредственно после выполнения любой другой команды и называются *командами безусловного перехода*.

В УАЯ команды безусловного перехода реализуются оператором безусловного перехода, который имеет вид

Перейти к М;

где М — метка оператора, к выполнению которого необходимо перейти.

На практике метки ставят только перед операторами, на которые осуществляются переходы с нарушением естественного порядка выполнения операторов. При записи команд программы для МК роль меток играют адреса команд. Каждая команда может быть выделена среди других по ее адресу.

На схемах алгоритмов безусловный переход изображают стрелкой в направлении операции, к выполнению которой надо перейти (рис. 59).

Используя операторы УАЯ, эту последовательность действий можно записать так:

М0: Начало;
Ввести (X, h);
М1: $Y = 1/\sqrt{1 - X^2}$;
Вывести (X, Y);
$X = X + h$;
Перейти к М1;

При этом количество вычисленных значений функции Y контролируется вручную.

Табулируя функцию, необходимо возвращаться к началу вычисления лишь в том случае, если значение X не выходит за пределы промежутка $[0, 1[$. Фактически последовательность выполнения команд зависит от того, выполняется ли условие $X < 1$ или нет. Операторы, позволяющие выбирать тот или иной путь продолжения вычислений

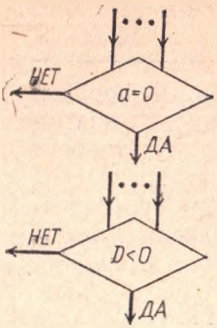


Рис. 60. Изображение логических символов на схемах алгоритмов

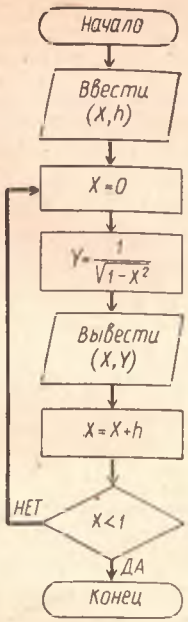


Рис. 61. Схема алгоритма расчета зависимости массы тела от скорости

в зависимости от того, выполняется или нет некоторое условие, называются *операторами условного перехода*, или *логическими операторами*. В УАЯ оператор условного перехода записывают в виде

Если $A < B$, то перейти к М;

где A и B — некоторые арифметические выражения. Выполняя такое указание, исполнитель должен перейти на оператор с меткой М лишь в случае, когда условие, записанное после слова «Если», выполняется. Если же это условие не выполняется, естественный порядок выполнения указаний не нарушается. В операторе условного перехода можно проверить любое из соотношений вида $A < B$, $A \leqslant B$, $A > B$, $A \geqslant B$, $A = B$, $A \neq B$.

Операции проверки некоторых условий объединяют в логические символы, которые на схемах алгоритмов изображают в виде ромбов (рис. 60). Из логического символа выходят две стрелки. Стрелка с надписью «ДА» указывает на операцию, к выполнению которой необходимо перейти, если условие, проверяемое в логическом символе, выполняется. Стрелка с надписью «НЕТ» указывает на операцию, к выполнению которой следует перейти в случае, если про-

веряемое условие не выполняется. К логическому символу может подходить несколько стрелок от символов, обозначающих операции, после выполнения которых нужно перейти к проверке условия, указанного в данном символе.

На рис. 61 показана схема алгоритма рассмотренной ранее задачи. На УАЯ этот алгоритм следует записать так:

М0 : Начало;
Ввести (X, h);
М1 : $Y = 1/\sqrt{1 - X^2}$;
Вывести (X, Y);
$X = X + h$;
Если $X < 1$, то перейти к М1;
Конец;

Данный вариант алгоритма удовлетворяет всем требованиям, предъявляемым к алгоритмам: через конечное число шагов переменная X (при $h > 0$) достигнет значения 1, после чего условие $X < 1$ выполняться уже не будет. В соответствии с оператором условного перехода управление вычислительным процессом теперь будет передано не на оператор с меткой М1, а на оператор Конец;.

Очевидно, запись алгоритма на УАЯ гораздо нагляднее и понятнее, чем запись того же алгоритма на языке МК.

В программе для МК отсутствуют команды, реализующие операторы М0: Начало; и Ввести (X, h);. Эти команды алгоритма на ПМК реализуются вручную: начальное значение $X = 0$ и шаг $h = 0{,}1$ вводятся соответственно в регистры Р1 и Р0 с клавиатуры. Оператор

$$\text{М1 : } Y = 1/\sqrt{1 - X^2};$$

реализуется выполнением команд, хранящихся по адресам 00 — 06; оператор

Вывести (X, Y);

выполняется вручную (после выполнения команды С/П с адресом 07). Этой командой прекращаются автоматические вычисления и создается возможность записи значений величин X и Y. Для продолжения вычислений необходимо нажать клавишу С/П (Пуск). Команды, хранящиеся по адресам 08 — 11, реализуют указание $X = X + h$;, т. е. заменяют текущее значение X новым, отличающимся от предыдущего на шаг h. Оператор условного перехода

Если $X < 1$, то перейти к М1;

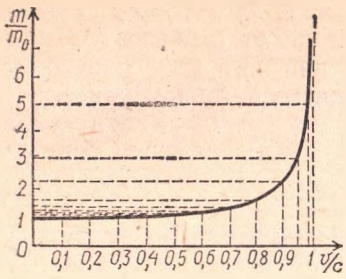


Рис. 62. График функции $Y = 1/\sqrt{1 - X^2}$

в рассматриваемом примере реализуется полуавтоматически. Исполнитель проверяет визуально, все ли необходимые значения Y уже вычислены. Если не все, то после очередного вывода значений X и Y он нажимает клавишу **С/П**, запуская очередной раз повторение вычислений.

Выполнив вычисления по описанной программе, получим следующую таблицу значений X и Y:

X	0	0,1	0,2	0,3	0,4
Y	1	1,0050378	1,0206207	1,0482846	1,0910894

X	0,5	0,6	0,7	0,8	0,9
Y	1,1547005	1,25	1,40028	1,6666666	2,2941573

График функции $Y = 1/\sqrt{1 - X^2}$ изображен на рис. 62.

Алгоритмы, в которых многократно выполняются вычисления по одним и тем же указаниям для различных значений исходных данных, называются *циклическими*. Повторяющиеся операторы носят название *операторов цикла*. В рассмотренном примере операторами цикла будут операторы, начиная от оператора с меткой М1 и кончая оператором условного перехода. Эти четыре оператора выполняются 10 раз при значениях $X = 0; 0{,}1; \ldots; 0{,}9$. Таким образом, хотя в алгоритме и записано всего семь операторов, фактически при решении поставленной задачи выполняется 47 указаний. Количеством повторений операторов цикла управляет оператор условного перехода.

Упражнения. 1. Составить схемы и записать на УАЯ алгоритмы:

а) вычисления всех значений квадратного тричлена $Y = -X^2 + 2X - 3$ для значений аргумента X, изменяющихся от -2 до $+2$ через шаг $h = 0{,}1$;

б) вычисления всех значений функции $Y = \sin X + \cos X$ для значений аргумента X, изменяющихся от -1 до $+3$ с шагом $h = 0{,}2$;

в) вычисления корней квадратного уравнения $X^2 + 2pX + q = 0$;

г) нахождения наименьшего из трех чисел a, b, c;

д) вычисления суммы положительных чисел из заданных трех чисел a, b, c;

е) вычисления суммы всех значений функции $Y = \sin X^2 + \cos X^2$, найденных для значений аргумента X, изменяющихся от 0 до 2 с шагом $h = 0{,}2$.

2. Построить траекторию движения тела, брошенного под углом 63°30′ к горизонту, начальная скорость v которого имеет составляющие $v_x = 22{,}4$ м/с, $v_y = 44{,}8$ м/с. Ускорение свободного падения $g = 9{,}8$ м/с^2.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как записывается на УАЯ оператор безусловного перехода? 2. Как изображается безусловный переход на схемах алгоритмов? 3. Как записывается на УАЯ оператор условного перехода? 4. Как изображаются логические символы на схемах алгоритмов? 5. Нарушается ли естественный порядок выполнения операторов, если условие, указанное в операторе условного перехода, не выполняется? 6. Какие алгоритмы называются циклическими? 7. С помощью какого оператора осуществляется управление количеством повторений операторов цикла? 8. Можно ли оператор условного перехода записать в виде Если $A \leqslant B$, то перейти к М; ? 9. Какие соотношения можно записать в операторе условного перехода после слова «Если»?

§ 16. РАЗВЕТВЛЯЮЩИЕСЯ ВЫЧИСЛИТЕЛЬНЫЕ ПРОЦЕССЫ

Вычислительные процессы, в которых вычисления проводятся различными путями в зависимости от того, выполняется или нет некоторое условие, называются *разветвляющимися*. Алгоритмы и программы, описывающие такие процессы, также называются *разветвляющимися*.

Выбор пути, по которому должен продолжаться вычислительный процесс, иногда называют *принятием решения*. Разветвление вычислительного процесса, т. е. выбор одного из возможных путей продолжения вычислений, осуществляется с помощью операторов условного перехода (логических операторов). Примером разветвления вычислительного процесса есть принятие решения о том, возвращаться ли на повторение вычислений при табулировании функции после наращивания ее аргумента или нет.

В ПМК «Электроника Б3-34» имеется набор команд условного перехода

$$FX < 0;\ FX \geqslant 0;\ FX = 0;\ FX \neq 0,$$

с помощью которых МК автоматически может выбрать тот или иной путь продолжения вычислений. Согласно этим

командам содержимое операционного регистра РХ сравнивается с нулем по одному из условий: $X < 0$, $X \geqslant 0$, $X = 0$, $X \neq 0$. Если при этом указанное условие не выполняется, то управление передается команде, адрес которой указан вслед за командой о проверке условия.

Если же проверяемое условие выполняется, то управление передается команде, адрес которой на две единицы больше, чем адрес команды о проверке условия. После этого выполнение команд продолжается в естественном порядке, который может быть нарушен только по команде безусловного перехода или же по команде условного перехода при невыполнении проверяемого условия.

Схематически выбор одного из двух возможных путей продолжения вычислений по команде FX = 0 отражает таблица:

Адрес команды	Команда	Примечание
24	FX = 0 — ДА	Команда о проверке условия $X = 0$
	НЕТ ↓	
25	08	Адрес, по которому необходимо перейти при невыполнении условия $X = 0$
→26	----------	Команда, на которую осуществляется переход при выполнении условия $X = 0$

Для иллюстрации сказанного рассмотрим следующий фрагмент программы:

```
15. ИП1        25.  — FX < 0 ——
16. С/П                НЕТ          ДА
--------       26.  →15
24. ИП1              →27. FX²
                     28. С/П
```

По адресу 25 записана команда условного перехода по условию $X < 0$ (FX < 0). Эта команда предписывает перейти к выполнению команды с адресом 15, если содержимое регистра РХ не удовлетворяет условию $X < 0$. Если же условие $X < 0$ выполняется, то далее выполняется команда с адресом 27.

Для проверки выполнения на ПМК данного фрагмента программы занесем в ППЗУ команды, начиная с адреса 15:

1. На счетчике адресов команд установим 15, используя команду безусловного перехода. Для этого в режиме автоматической работы ПМК нажимаем клавиши **БП 1 5.**

2. Переход в режим программирования осуществляем нажатием клавиш **F ПРГ.** Показание индикатора

1	2	3	4	5	6	7	8	9	10	11	12
	0	0		0	0		0	0		1	5

свидетельствует о том, что в счетчик адресов команд занесен адрес 15.

Теперь, начиная с адреса 15, последовательно заносим в МК команды:

Адрес команды	Команда	Показание индикатора МК
15	ИП1	6 1 0 0 0 0 1 6
16	С/П	5 0 6 1 0 0 1 7

3. С помощью команды безусловного перехода (в автоматическом режиме) или клавиши **ШГ** (в режиме программирования) заносим в РАК 24 и затем заносим команды:

Адрес команды	Команда	Показание индикатора МК
24	ИП1	6 1 0 0 0 0 2 5
25	FX < 0	5 [6 1 0 0 2 6
26	15	1 5 5 [6 1 2 7
27	FX²	2 2 1 5 5 [2 8
28	С/П	5 0 2 2 1 5 2 9

4. Переход в режим автоматической работы МК осуществляем нажатием клавиш **F АВТ.**

5. Заносим некоторое число, например $X = -3$, в адресуемый регистр Р1.

6. Нажатием клавиш **БП 2 4** заносим в РАК адрес команды 24.

7. Запускаем программу. Через несколько секунд на индикаторе МК должен появиться результат — число 9.

Анализируя программу и исходные данные, видим, что действительно в регистр Р1 было занесено отрицательное число (—3). В этом случае ПМК по команде 25 (FX < 0) должен был:

1) перейти на команду с адресом 27, так как проверяемое условие удовлетворяется;

2) по команде 27 возвести число, находящееся в регистре РХ, в квадрат, т. е. выполнить операцию $(-3)^2 = 9$;

3) остановиться (по команде 28), что он и выполнил.

Нажав клавиши **F ПРГ**, можно убедиться в том, что в РАК находится число 29. Это означает, что последней была выполнена команда с адресом 28.

Можно выполнить теперь те же операции, начиная с п. 5, но в регистр Р1 занести положительное число, например 4. При запуске программы через некоторое время на индикаторе МК должен появиться результат 4, который означает, что условие, проверяемое по команде 25, на этот раз не выполнилось. После команды с адресом 25 выполнялась команда с адресом 15, вызывающая число 4 в регистр РХ из регистра Р1, затем — команда С/П, хранящаяся по адресу 16. В этом можно убедиться, перейдя в режим программирования.

Порядок передачи управления вычислениями в зависимости от выполнения проверяемого условия можно проверить, выполняя программу покомандно.

Рассмотрим еще один фрагмент программы:

```
12. ИП1                    ┌→ 15. 2
13. ┌— FX = 0 ——┐ ДА        16. С/П
    │ НЕТ                  17. 1
14. └→ 17                  18. С/П
```

Занесем, начиная с адреса 12, данный фрагмент программы в ППЗУ, затем в режиме автоматической работы занесем некоторое число, например $X = 0$, в адресуемый регистр Р1. Запустим программу с команды, имеющей адрес 12. Через несколько секунд на индикаторе МК должно

появиться число 2, свидетельствующее о том, что управление было передано на команду, хранящуюся по адресу 15. Занесем далее в регистр Р1 число — 0,5 и опять запустим программу с команды, имеющей адрес 12. Теперь после остановки МК на его индикаторе должно появиться число 1, означающее, что управление было передано не на команду, хранящуюся по адресу 15, а на команду, хранящуюся по адресу 17. В этом же можно убедиться, выполнив программу покомандно.

Для этого занесем в Р1 некоторое, отличающееся от нуля число (например, —5), и нажмем клавишу **ПП**. Из регистра Р1 в регистр РХ вызывается число —5, т. е. выполняется команда с адресом 12. Нажмем далее еще раз клавишу **ПП**. При этом МК должен перейти на команду с адресом 17, так как условие $X = 0$ не выполняется. Нажав клавиши **F ПРГ**, убеждаемся в том, что содержимое РАК действительно равно 17.

Занесем теперь в Р1 нуль и, начиная с команды, имеющей адрес 12, будем поочередно выполнять команды программы, нажимая клавишу **ПП**. После первого нажатия этой клавиши выполнится команда с адресом 12, которая вызовет из регистра Р1 в регистр РХ число 0. Поскольку теперь условие $X = 0$ выполняется, следующая команда FX = 0 с адресом 13 должна передать управление не команде, хранящейся по адресу 17, как прежде, а команде, хранящейся по адресу 15. Нажав клавишу **ПП** и затем клавиши **F ПРГ**, убеждаемся в том, что содержимое РАК действительно равно 15.

Пример. Составим программу для вычисления общего сопротивления двух резисторов с учетом вида их соединения (параллельное или последовательное).

Общее сопротивление двух резисторов с сопротивлениями R_1 и R_2, соединенных параллельно, вычисляется по формуле

$$R = \frac{1}{1/R_1 + 1/R_2},$$

а последовательно — по формуле

$$R = R_1 + R_2,$$

т. е. вычисление общего сопротивления может производиться по двум различным выражениям в зависимости от вида соединения резисторов. Признаком вида соединения будем считать значение величины

$$a = \begin{cases} 0 \text{ при параллельном соединении;} \\ 1 \text{ » последовательном » .} \end{cases}$$

Графическая схема алгоритма решения задачи показана на рис. 63. Опишем алгоритм ее решения на УАЯ:

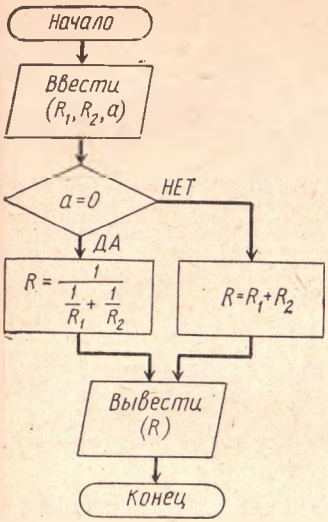


Рис. 63. Схема алгоритма вычисления общего сопротивления двух резисторов

M0: Начало;
Ввести (R_1, R_2, a);
Если $a=0$, то перейти к M1;
$R=R_1+R_2$;
M2: Вывести ('R=' ,R);
Конец;
M1: $R=1/(1/R_1+1/R_2)$;
Перейти к M2;

Составим программу решения задачи на ПМК, распределив память так: для хранения значений признака вида соединения резисторов (величина a) отведем адресуемый регистр Р1, для R_1 — Р2, для R_2 — Р3. При необходимости рабочими будем считать регистры Р3, Р4 и т. д. Будем транслировать на язык МК каждый отдельный пункт алгоритма.

Первый пункт:
Начало;

Этому пункту не ставятся в соответствие никакие команды МК.

Второй пункт:
Ввести (R_1, R_2, a);

Данному пункту также не ставятся в соответствие никакие команды МК.

Ввод значений R_1, R_2 и a осуществляется вручную непосредственно перед запуском автоматических вычислений на МК по программе.

Третий пункт:
Если $a=0$, то перейти к M1;

реализуется командами условного и безусловного переходов:

Адрес команды	Команда	Код команды	Примечание
00	ИП1	61	$a \rightarrow$ РХ
01	FX=0 (НЕТ / ДА)	5E	Команда условного перехода
02	→05	05	Адрес команды, на которую необходимо перейти при невыполнении условия $a=0$
→03	БП	51	Команда, на которую осуществляется переход при выполнении условия $a=0$

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание
04	09	09	Адрес перехода для команды безусловного перехода

При $a = 0$ (резисторы соединены параллельно) управление вычислениями будет передано команде, хранящейся по адресу 03, которая в свою очередь передает управление команде 09 (с нее начинается программа, соответствующая оператору с меткой М1). Если же $a \neq 0$ (резисторы соединены последовательно), то управление вычислениями будет передано команде с адресом 05, т. е., начиная с адреса 05, можно продолжать команды программы, реализующие четвертый пункт алгоритма:

$$R = R_1 + R_2;$$

Имеем:

Адрес команды	Команда	Код команды	Примечание
05	ИП2	62	$R_1 \to$ РХ
06	ИП3	63	$R_2 \to$ РХ, $R_1 \to$ РУ
07	+	10	$R_1 + R_2 \to$ РХ

Пятый пункт

М2: Вывести ('$R =$', R);

реализуется вручную.

Оператору

Конец;

соответствует команда

08 | С/П | 50 | Стоп

Теперь определился и адрес 09 первой команды программы, соответствующей оператору с меткой М1, и можно указать адрес перехода к команде, хранящейся по адресам 03 и 04, т. е. занести 09 по адресу 04.

Шестой пункт

М1 : $R = 1/(1/R_1 + 1/R_2)$;

реализуется командами:

Адрес команды	Команда	Код команды	Примечание
09	ИП2	62	$R_1 \to$ РХ
10	F1/X	23	$1/R_1 \to$ РХ
11	ИП3	63	$R_2 \to$ РХ, $1/R_1 \to$ РУ
12	F1/X	23	$1/R_2 \to$ РХ
13	+	10	$1/R_1 + 1/R_2 \to$ РХ
14	F1/X	23	$1/(1/R_1 + 1/R_2) \to$ РХ

Седьмой пункт
Перейти к М2;
реализуется командой:

Адрес команды	Команда	Код команды	Примечание
15	БП	51	Команда безусловного перехода
16	08	08	Адрес перехода

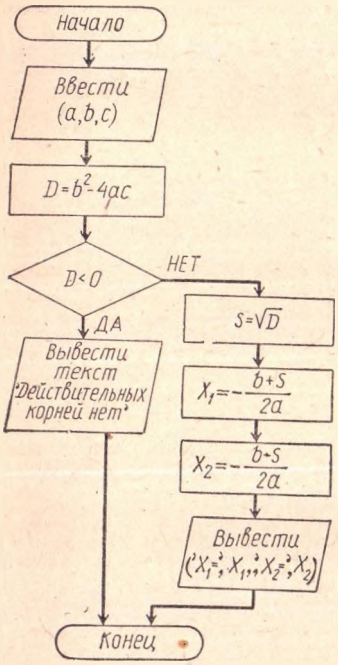


Рис. 64. Схема алгоритма решения квадратного уравнения

Соединив программы, соответствующие отдельным пунктам алгоритма, получим единую программу для МК, по которой реализуется рассмотренный алгоритм.

Пример. Рассмотрим задачу определения корней квадратного уравнения $aX^2 + bX + c = 0$, $a \neq 0$. Это уравнение имеет действительные корни, если дискриминант $D = b^2 - 4ac$ неотрицателен. При этом корни уравнения вычисляются по формулам

$$X_1 = (-b - \sqrt{D}) / (2a); \quad X_2 = (-b + \sqrt{D}) / (2a).$$

Если же дискриминант D отрицателен, то действительных корней не существует.

Графическая схема алгоритма решения задачи показана на рис. 64.

Опишем алгоритм решения задачи на УАЯ:

М0: Начало;
Ввести (a, b, c);
$D = b^2 - 4ac$;
Если $D < 0$, то перейти к М1;
$X_1 = (-b - \sqrt{D})/(2a)$;
$X_2 = (-b + \sqrt{D})/(2a)$;
Вывести $('X_1 = ', X_1, ', X_2 = ', X_2)$;
Перейти к М2;
М1 : Вывести ('Действительных корней нет');
М2 : Конец;

Перейдем теперь к составлению программы решения рассматриваемой задачи на МК. Распределим память: для хранения значений коэффициента a отведем адресуемый регистр Р1, для b — Р2, для c — Р3, для D — Р0, для X_1 — Р4 и для X_2 — Р5. При необходимости рабочими будем считать регистры Р6, Р7 и т. д. Поскольку

ку автоматические ввод и вывод информации на МК не предусмотрены, эти операции будем выполнять вручную.

Чтобы определить, какая из ситуаций имела место при решении задачи ($D < 0$ или $D \geqslant 0$), условимся в первом случае перед остановкой МК по программе вводить в регистр РХ число 1, а во втором — 2. Таким образом, если при остановке МК на индикаторе будет высвечено число 1, это будет означать, что под данным уравнением следует записать текст «Действительных корней нет». Если же при остановке МК на индикаторе будет высвечено число 2, это будет означать, что под данным уравнением необходимо записать текст «$X_1 =$ », затем вызвать в регистр РХ из регистра Р4 хранящееся там число и записать его вслед за этим текстом. Далее нужно записать текст «, $X_2 =$ », после чего вызвать из регистра Р5 хранящееся там число и записать его вслед за указанным текстом. Если в регистрах Р4 и Р5 при этом хранятся, например, числа 2 и —1, то в результате должна быть запись:

$$X_1 = 2,\ X_2 = -1.$$

Будем транслировать на язык МК каждый отдельный пункт алгоритма.

Первый пункт:

Начало;

Этому пункту не ставятся в соответствие никакие команды МК.

Второй пункт:

Ввести (a, b, c);

Данному пункту также не ставятся в соответствие никакие команды МК. Ввод значений коэффициентов a, b, c осуществляется вручную непосредственно перед запуском автоматических вычислений по программе.

Третий пункт:

$D = b^2 - 4ac$;

Поскольку ППЗУ свободно, команды программы, соответствующей этому пункту алгоритма, можно вводить в ППЗУ, начиная с адреса 00. Учитывая распределение памяти, получим следующую программу, по которой выполняется указание данного пункта алгоритма:

Адрес команды	Команда	Код команды	Примечание
00	ИП1	61	$a \to$ PX
01	ИП3	63	$c \to$ PX, $a \to$ PY
02	×	12	$ac \to$ PX
03	4	04	$4 \to$ PX, $ac \to$ PY
04	×	12	$4ac \to$ PX
05	ИП2	62	$b \to$ PX, $4ac \to$ PY
06	FX^2	22	$b^2 \to$ PX, $4ac \to$ PY
07	$\overrightarrow{XY}$	14	$4ac \to$ PX, $b^2 \to$ PY
08	$\overleftarrow{-}$	11	$b^2 - 4ac \to$ PX
09	П0	40	$b^2 - 4ac \to$ P0

Четвертый пункт:

Если $D < 0$, то перейти к М1;

Этот пункт реализуется командами условного и безусловного переходов. Так как после выполнения команды с адресом 09 значение D еще находится в регистре РХ, то, начиная с адреса 10, можно записать:

Адрес команды	Команда	Код команды	Примечание
10	FX < 0	5[	Команда условного перехода
11	14	14	Адрес команды, на которую необходимо перейти при невыполнении условия $D < 0$
12	БП	51	Команда, на которую осуществляется переход при выполнении условия $D < 0$
13	36	36	Адрес перехода для команды безусловного перехода

При $X \geqslant 0$ (т. е. при $D = b^2 - 4ac \geqslant 0$) управление вычислениями будет передано команде, хранящейся по адресу 14. Если же окажется, что $X < 0$ (т. е. $D = b^2 - 4ac < 0$), то управление будет передано команде с адресом 12, которая в свою очередь передаст управление команде, с которой начинается программа, соответствующая оператору с меткой М1.

Пятый пункт:

Начиная с адреса 14, можно располагать следующие команды программы, реализующей указание алгоритма

$$X_1 = (-b - \sqrt{D})/(2a);$$

Получим:

Адрес команды	Команда	Код команды	Примечание
14	ИП0	60	$D \to$ РХ
15	F$\sqrt{\ }$	21	$\sqrt{D} \to$ РХ
16	ИП2	62	$b \to$ РХ, $\sqrt{D} \to$ РУ
17	+	10	$b + \sqrt{D} \to$ РХ
18	/—/	0[	$-b - \sqrt{D} \to$ РХ
19	ИП1	61	$a \to$ РХ, $-b - \sqrt{D} \to$ РУ
20	÷	13	$(-b - \sqrt{D})/a \to$ РХ
21	2	02	$2 \to$ РХ, $(-b - \sqrt{D})/a \to$ РУ
22	÷	13	$(-b - \sqrt{D})/(2a) \to$ РХ
23	П4	44	$(-b - \sqrt{D})/(2a) \to$ Р4

Шестой пункт:

Начиная с адреса 24, можно далее располагать команды программы, реализующей указание алгоритма

$$X_2 = (-b + \sqrt{D})/(2a);$$

Получим:

Адрес команды	Команда	Код команды	Примечание
24	ИП0	60	$D \to$ PX
25	F$\sqrt{}$	21	$\sqrt{D} \to$ PX
26	ИП2	62	$b \to$ PX, $\sqrt{D} \to$ PY
27	—	11	$-b + \sqrt{D} \to$ PX
28	ИП1	61	$a \to$ PX, $-b + \sqrt{D} \to$ PY
29	÷	13	$(-b + \sqrt{D})/a \to$ PX
30	2	02	$(2 \to$ PX, $(-b + \sqrt{D}/a \to$ PY
31	÷	13	$(-b + \sqrt{D})/(2a) \to$ PX
32	П5	45	$(-b + \sqrt{D})/(2a) \to$ P5

Седьмой пункт:

Вывести ('$X_1 =$', X_1,' $X_2 =$ ', X_2);

реализуется вручную. Команда же, соответствующая этому пункту алгоритма, лишь вводит в регистр PX число 2:

Адрес команды	Команда	Код команды	Примечание
33	2	02	$2 \to$ PX

Восьмой пункт:

Перейти к М2;

реализуется командой:

Адрес команды	Команда	Код команды	Примечание
34	БП	51	Команда безусловного перехода
35	37	37	Адрес перехода

Девятый пункт:

М1: Вывести ('Действительных корней нет');

Как и предыдущий оператор вывода, данный оператор реализуется вручную. Команда же, соответствующая этому пункту алгоритма, лишь вводит в регистр PX число 1:

Адрес команды	Команда	Код команды	Примечание
36	1	01	1 → РХ

Только теперь определился адрес первой команды программы, соответствующей оператору с меткой М1, и можно указать адрес перехода в команде, хранящейся по адресам 12, 13, т. е. записать 36 по адресу 13.

Десятому пункту:

М2: Конец;

соответствует команда:

Адрес команды	Команда	Код команды	Примечание
37	С/П	50	Стоп

Теперь определился и адрес 37 первой команды программы, соответствующей оператору с меткой М2, и можно указать адрес перехода к команде, хранящейся по адресам 34, 35, т. е. занести 37 по адресу 35.

Введя в ППЗУ все полученные программы, соответствующие отдельным пунктам алгоритма, получим единую программу для МК, по которой реализуется рассмотренный алгоритм решения квадратного уравнения.

Упражнения. 1. Используя составленную программу решения квадратного уравнения, в автоматическом режиме найти корни квадратных уравнений:

а) $X^2 - 5X + 6 = 0$; б) $X^2 - 4 = 0$; в) $X^2 + 7 = 0$;

г) $2{,}784X^2 - 25{,}385X + 11{,}784 = 0$;

д) $0{,}894356X^2 - 4{,}178524X + 5{,}989771 = 0$.

2. Переделать упомянутые в предыдущем упражнении алгоритм и программу так, чтобы $\sqrt{D}$ и $2a$ вычислялись только один раз, а затем использовались их найденные значения.

3. Составить схему алгоритма вычисления значений следующих функций:

$$\text{а) } Y = \begin{cases} -X^2 \text{ при } X < 0; \\ X + 0{,}1 \text{ » } 0 \leqslant X < 0{,}5; \\ (X - 0{,}5)^2 + 0{,}6 \text{ » } X \geqslant 0{,}5, \end{cases}$$

если $X_1 = -12{,}876$; $X_2 = 0{,}2497$; $X_3 = 2{,}50434$;

$$\text{б) } Y = \begin{cases} X^3 \text{ при } X \leqslant 0; \\ X - 3 \text{ » } 0 < X \leqslant 1; \\ \dfrac{1}{X^3} - \dfrac{4}{X} \text{ » } X > 1, \end{cases}$$

если $X_1 = -2{,}2874$; $X_2 = 0{,}2874$; $X_3 = 15{,}8072$;

в) $$Y=\begin{cases}\dfrac{1}{X^3}+5 \text{ при } X\leqslant -1;\\ X+3 \text{ » } -1<X\leqslant 1;\\ \dfrac{2X+1}{X^3+5} \text{ » } X>1,\end{cases}$$

если $X_1=-4{,}8726$; $X_2=0{,}5874$; $X_3=9{,}2147$.

4. Составить схему алгоритма и программу для вычисления на МК общей емкости двух конденсаторов при их параллельном или последовательном соединении, используя формулы

$$C=\begin{cases}C_1+C_2 \text{ при параллельном соединении;}\\ \dfrac{C_1C_2}{C_1+C_2} \text{ » последовательном » .}\end{cases}$$

5. Составить схему алгоритма и программу вычисления на МК скорости v вылетающих из металла электронов при фотоэффекте по заданной работе выхода A и длине волны λ падающего света. Скорость v вычисляется по формуле

$$v=\sqrt{\frac{2\,(hc/\lambda-A)}{m}},$$

где c, h, m — постоянные. Фотоэффект происходит в том случае, когда подкоренное выражение $2\,(hc/\lambda-A)/m\geqslant 0$; в противном случае фотоэффекта нет.

6. Составить схему алгоритма и программу вычисления на МК увеличения линзы с фокусным расстоянием F, если предмет находится на расстоянии d от нее. Воспользоваться следующими формулами:

$$\Gamma=\frac{f}{d};\ f=1/X;\ X=\frac{1}{F}-\frac{1}{d}.$$

Если $\Gamma>0$, то изображение действительное; при $\Gamma=0$ изображение в бесконечности, а при $\Gamma<0$ — мнимое. Выполнить вычисления для линзы с $F=0{,}5$ м, если $d=0{,}2$; 0,5; 0,7; 1 и 1,2 м. Сделать выводы о размере и виде изображения для каждого из рассматриваемых случаев.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Какие вычислительные процессы называют разветвляющимися? 2. Как осуществляется выбор одного из возможных путей продолжения вычислительного процесса? 3. Какие условия можно проверять с помощью команд ПМК? 4. На какую команду передается управление вычислениями в ПМК, если проверяемое условие не выполняется? 5. На какую команду передается управление вычислениями в ПМК, если проверяемое условие выполняется? 6. Как определить адрес, по которому следует располагать в ППЗУ первую команду программы, реализующей очередной пункт алгоритма? 7. Как получить программу, соответствующую всему алгоритму вычислений, имея программы, соответствующие его отдельным пунктам? 8. Как на ПМК можно определить, какая из нескольких возможных ситуаций была при конкретной реализации алгоритма?

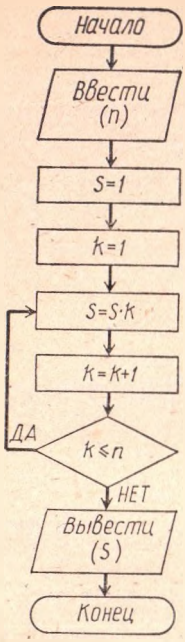


Рис. 65. Схема алгоритма вычисления $Y = n!$

§ 17. АВТОМАТИЧЕСКОЕ ВЫПОЛНЕНИЕ ЦИКЛИЧЕСКИХ ПРОЦЕССОВ

Наряду с задачами табулирования функций существуют и другие алгоритмы циклических вычислительных процессов, которые на ПМК можно осуществить в автоматическом режиме. Примером такого алгоритма может быть алгоритм вычисления произведения n натуральных чисел $1 \cdot 2 \times \times 3 \cdot \ldots \cdot n$, обозначаемого $n!$ (читается n-факториал). Графическая схема решения этой задачи показана на рис. 65. На УАЯ алгоритм вычисления $n!$ можно записать в виде:

М0 : Начало;
Ввести (n);
$S = 1$;
$k = 1$;
М1 : $S = S \cdot k$;
$k = k + 1$;
Если $k \leqslant n$, то перейти к М1;
Вывести (S);
Конец М0;

Распределим память ПМК так: $n \rightarrow$ Р0, $S \rightarrow$ Р1, $k \rightarrow$ Р2 и переведем алгоритм с УАЯ на язык МК. В соответствие операторам М0: Начало; и Ввести (n); в программе для МК не ставятся никакие команды. Оператору $S = 1$; соответствуют команды:

Адрес команды	Команда	Код команды	Примечание
00	1	01	$1 \rightarrow$ РХ
01	П1	41	$1 \rightarrow$ Р1

Оператору $k = 1$; ставится в соответствие команда:

Адрес команды	Команда	Код команды	Примечание
02	П2	42	$1 \rightarrow$ Р2

Оператору М1: $S = S \cdot k$; ставятся в соответствие команды:

Адрес команды	Команда	Код команды	Примечание
03	ИП1	61	$S \to$ РХ
04	ИП2	62	$k \to$ РХ, $S \to$ РУ
05	×	12	$S \cdot k \to$ РХ
06	П1	41	$S \cdot k \to$ Р1

Оператору $k = k + 1$; ставятся в соответствие команды:

Адрес команды	Команда	Код команды	Примечание
07	ИП2	62	$k \to$ РХ
08	1	01	$1 \to$ РХ, $k \to$ РУ
09	+	10	$k + 1 \to$ РХ
10	П2	42	$k + 1 \to$ Р2

Оператору Если $k \leqslant n$, то перейти к М1; ставятся в соответствие команды:

Адрес команды	Команда	Код команды	Примечание
11	ИП0	60	$n \to$ РХ
12	ИП2	62	$k \to$ РХ, $n \to$ РУ
13	—	11	$n - k \to$ РХ
14	FX $\geqslant$ 0	59	Если $n - k \geqslant 0$, то на 16, иначе на 18
15	18	18	Адрес перехода
16	БП	51	Команда безусловного перехода
17	03	03	Адрес перехода

Операторам Вывести (S); и Конец; ставятся в соответствие команды:

Адрес команды	Команда	Код команды	Примечание
18	ИП1	61	$S \to$ РХ Вызов значения S в РХ для чтения
19	С/П	50	Стоп

Объединив программы, реализующие каждый отдельный оператор алгоритма, в одну, получим программу, реализующую алгоритм решения задачи в целом.

Занесем теперь программу в ППЗУ и реализуем ее, например, для значения $n = 5$. Введя число 5 в регистр Р0 и запустив вычисления на МК в автоматическом режиме с команды, хранящейся по адресу 00, после окончания вычислений получим $5! = 120$.

Упражнения. **1.** Составить схему и записать на УАЯ алгоритмы вычисления значений следующих функций:

а) $Y = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{20^2}$;

б) $S = 1^2 + 2^2 + 3^2 + 4^2 + \cdots + 100^2$;

в) $S = \sum_{k=1}^{20} \frac{k}{k+1} = \frac{1}{2} + \frac{2}{3} + \frac{3}{4} + \frac{4}{5} + \cdots + \frac{18}{19} + \frac{19}{20} + \frac{20}{21}$;

г) $S = \sum_{k=1}^{30} \frac{1}{k^3} = \frac{1}{1^3} + \frac{1}{2^3} + \frac{1}{3^3} + \frac{1}{4^3} + \cdots + \frac{1}{29^3} + \frac{1}{30^3}$;

д) $S = \sum_{k=1}^{10} \frac{1}{k!} = \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \frac{1}{1 \cdot 2 \cdot 3 \cdot 4} + \cdots$

$\cdots + \frac{1}{1 \cdot 2 \cdot 3 \cdot \ldots \cdot 10}$.

2. Составить схему алгоритма и программу подсчета суммы членов арифметической прогрессии

$$S = a + (a + d) + (a + 2d) + \ldots + [a + (n - 1)\, d].$$

Формулу

$$S_n = \frac{1}{2}(a_1 + a_n)\, n$$

не использовать.

3. Составить схему алгоритма и программу подсчета суммы n членов геометрической прогрессии

$$S = a + aq + aq^2 + aq^3 + \cdots + aq^{n-1}.$$

Формулу

$$S_n = \frac{a_1 (q^n - 1)}{q - 1}$$

не использовать.

§ 18. ПОДПРОГРАММЫ

Если в процессе вычислений требуется несколько раз повторять их по одной и той же формуле, то, чтобы не составлять каждый раз одну и ту же программу вычисле-

ний, ее пишут один раз и оформляют в виде так называемой *подпрограммы*. При необходимости провести вычисления по формуле, для которой написана эта подпрограмма, достаточно дать команду обращения к ней. Обращение к подпрограмме на ПМК «Электроника Б3-34» осуществляется командой ПП, после которой указывается адрес первой команды подпрограммы.

Единственное отличие написания подпрограммы от написания программы заключается в том, что подпрограмма должна заканчиваться командой В/О (возврат в программу), а не командой С/П (Стоп). Программа, в которой другая программа используется как подпрограмма, называется *вызывающей программой*. Подпрограмма в свою очередь может обращаться к другой подпрограмме. Глубина вложения подпрограммы в подпрограмму при программировании на ПМК «Электроника Б3-34» не должна превышать пяти, что обусловлено конструкцией МК.

Текст подпрограммы, как правило, размещают после текста вызывающей программы. Распределение памяти при написании подпрограммы осуществляется так же, как и при написании программ. Однако в вызывающей программе необходимо предусмотреть, чтобы исходные данные для работы по подпрограмме были занесены в отведенные для них регистры перед обращением к подпрограмме. Результат, полученный после выполнения команд подпрограммы, также должен быть помещен в отведенный для него регистр, откуда его можно переписать для дальнейшего использования в вызывающей программе.

При описании на УАЯ подпрограмма должна начинаться заглавием вида

F: Подпрограмма;

где F — метка первого оператора подпрограммы. Заканчиваться подпрограмма должна оператором вида

Конец F;

Обращение к подпрограмме осуществляется с помощью оператора вида

Выполнить F;

Пример. Составить для ПМК программу вычисления значений функции

$$Y = \frac{aX^3 + bX + c}{[a(2X-1)^2 + b(2X-1) + c]^2 + 3} + \\ + a(X^2+5)^2 + b(X^2+5) + c$$

при заданном значении аргумента X.

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
00	ИП0	60	$X \to$ PX	$S = 2 \cdot X - 1;$ $Z = S;$
01	2	02	$2 \to$ PX, $X \to$ PY	
02	×	12	$2X \to$ PX	
03	1	01	$1 \to$ PX, $2X \to$ PY	
04	—	11	$2X - 1 \to$ PX	
05	ПП	53	Обращение к подпрограмме $W = aZ^2 + bZ + c$	Выполнить F;
06	26	26	Адрес первой команды подпрограммы	
07	FX²	22	$[a(2X-1)^2 + b(2X-1) + c]^2 \to$ PX	$r_1 = W^2 + 3;$
08	3	03	$3 \to$ PX, $[a(2X-1)^2 + b(2X-1) + c]^2 \to$ PY	
09	+	10	$[a(2X-1)^2 + b(2X-1) + c]^2 + 3 \to$ PX	
10	П4	44	$[a(2X-1)^2 + b(2X-1) + c]^2 + 3 \to$ P4	
11	ИП0	60	$X \to$ PX	$Z = X;$
12	ПП	53	Обращение к подпрограмме $W = aZ^2 + bZ + c$	Выполнить F;
13	26	26	Адрес первой команды подпрограммы	
14	ИП4	64	$[a(2X-1)^2 + b(2X-1) + c]^2 + 3 \to$ PX; $aX^2 + bX + c \to$ PY	
15	÷	13	$\frac{aX^2 + bX + c}{[a(2X-1)^2 + b(2X-1) + c]^2 + 3} \to$ PX	$r_1 = W / r_1;$
16	П4	44	$\frac{aX^2 + bX + c}{[a(2X-1)^2 + b(2X-1) + c]^2 + 3} \to$ P4	
17	ИП0	60	$X \to$ PX	$t = X^2 + 5;$ $Z = t;$
18	FX²	22	$X^2 \to$ PX	
19	5	05	$5 \to$ PX, $X^2 \to$ PY	
20	+	10	$X^2 + 5 \to$ PX	
21	ПП	53	Обращение к подпрограмме $W = aZ^2 + bZ + c$	Выполнить F;
22	26	26	Адрес первой команды подпрограммы	
23	ИП4	64	$\frac{aX^2 + bX + c}{[a(2X-1)^2 + b(2X-1) + c]^2 + 3} \to$ PX, $a(X^2+5)^2 + b(X^2+5) + c \to$ PY	$Y = r_1 + W;$
24	+	10	$\frac{aX^2 + bX + c}{[a(2X-1)^2 + b(2X-1) + c]^2 + 3} +$ $+ a(X^2+5)^2 + b(X^2+5) + c \to$ PX	
25	С/П	50	Стоп	Вывести (Y); Конец А;

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
26	П5	45	$Z \to$ P5	
27	FX²	22	$Z^2 \to$ PX	
28	ИП1	61	$a \to$ PX, $Z^2 \to$ PY	
29	×	12	$aZ^2 \to$ PX	
30	П6	46	$aZ^2 \to$ P6	
31	ИП5	65	$Z \to$ PX	F: Подпрограмма;
32	ИП2	62	$b \to$ PX, $Z \to$ PY	$W = aZ^2 + bZ + c$;
33	×	12	$bZ \to$ PX	
34	ИП6	66	$aZ^2 \to$ PX, $bZ \to$ PY	
35	+	10	$aZ^2 + bZ \to$ PX	
36	ИП3	63	$c \to$ PX, $aZ^2 + bZ \to$ PY	
37	+	10	$aZ^2 + bZ + c \to$ PX	
38	В/О	52	Выход из подпрограммы	Конец F;

Здесь для вычисления значения Y три раза используется фактически одна и та же формула $W = aZ^2 + bZ + c$, только один раз вместо Z следует иметь в виду X, другой — $(2X - 1)$, третий — $(X^2 + 5)$. Поэтому вычисления по формуле $W = aZ^2 + bZ + c$ удобно оформить подпрограммой.

На УАЯ эту подпрограмму можно представить в виде:

```
F: Подпрограмма;
W = aZ² + bZ + c;
Конец F;
```

Весь алгоритм вычисления значения заданной функции W при заданном значении ее аргумента Z можно записать так:

```
A : Начало;                    t = X² + 5;
    Ввести (a, b, c, X);       Z = t;
    S = 2 · X — 1;             Выполнить F;
    Z = S;                     Y = r1 + W;
    Выполнить F;               Вывести (Y);
    r1 = W² + 3;               Конец A;
    Z = X;                 F : Подпрограмма;
    Выполнить F;               W = aZ² + bZ + c;
    r1 = W/r1;                 Конец F;
```

Распределим память МК следующим образом: $X \to$ P0, $a \to$ $\to$ P1, $b \to$ P2, $c \to$ P3. Регистр P4 используем как рабочий в программе, регистры P5, P6 — как рабочие в подпрограмме. Условимся перед обращением к подпрограмме значение Z записывать в регистр PX. В этот же регистр будем записывать результат работы по подпрограмме. В вызывающей программе при необходимости будем переписывать результат работы по подпрограмме из регистра PX в другие регистры.

Будем транслировать алгоритм, записанный на УАЯ, на язык ПМК. После соединения всех программ, полученных при трансляции каждого оператора алгоритма на язык ПМК, окончательную программу вычисления значений заданной функции представим в виде, показанном на с. 134—136.

Подпрограмма занимает адреса ППЗУ, начиная от 26 и кончая 38, т. е. 13 ячеек памяти. Обращения к подпрограмме занимают шесть ячеек: 05, 06, 12, 13 и 21, 22. Если вычисления по формуле $W = aZ^2 + bZ + c$ в подпрограмму не оформлять, то вместо двух команд обращения к подпрограмме в данном примере пришлось бы записать 12 команд самой подпрограммы (опустив команду В/О). Весьма эффективно использование подпрограммы, если обращений к ней много и сама подпрограмма занимает много ячеек памяти.

Связь вызывающей программы с подпрограммой и обратная связь показаны на рис. 66.

Подпрограммы удобно использовать не только тогда, когда одна и та же формула встречается многократно. Иногда весь вычислительный процесс разбивают на отдельные части и для каждой из них пишут соответствующую подпрограмму. Программа же лишь координирует работу

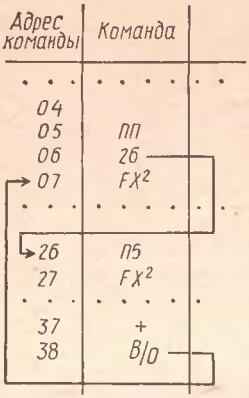


Рис. 66. Связь подпрограммы с вызывающей программой

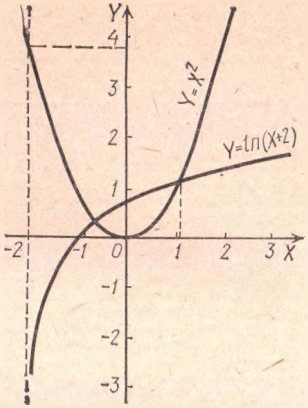


Рис. 67. Графическое решение уравнения $X^2 - \ln(X+2) = 0$

подпрограмм, устанавливая порядок их выполнения. В виде подпрограмм удобно оформить также программы вычисления значений часто встречающихся функций, чтобы не писать каждый раз эти программы заново. Для вычисления значений функций $\sin X$, $\cos X$, $\operatorname{tg} X$, $\arcsin X$, $\arccos X$, $\operatorname{arctg} X$, e^X, $\lg X$, $\ln X$, 10^X, $\sqrt{X}$, $1/X$, X^2, X^Y такие (стандартные) подпрограммы занесены в ПЗУ МК и хранятся там постоянно.

Пример. Найти с точностью до 0,0001 корень уравнения $X^2 - \ln(X+2) = 0$, находящийся на промежутке $[0, 2]$ (рис. 67).

Точные методы для решения такого уравнения неприменимы: нет формулы, по которой можно было бы найти X из данного уравнения.

Для приближенного определения корня уравнения воспользуемся методом деления отрезка пополам. Этот метод заключается в следующем. Промежуток, на котором находится корень уравнения, делят на два равных промежутка. Тот из них, на концах которого функция $X^2 - \ln(X+2)$ принимает значения разных знаков, опять делят на две равные части. Оставляя каждый раз промежуток, на концах которого заданная функция принимает значения разных знаков, процесс деления продолжают до тех пор, пока длина промежутка станет не больше, чем точность, с которой требуется определить корень уравнения.

Алгоритм приближенного нахождения с точностью до $\varepsilon > 0$ корня уравнения $f(X) = 0$, находящегося на промежутке $[a, b]$, на УАЯ можно записать так:

М0 : Начало;
Ввести (a, b, ε); $U = a$; $V = b$; $Z = a$;
Выполнить F; $W = R$;

M1: Если $|V - U| < \varepsilon$, то перейти к M3;
$X = (U + V)/2$;
$Z = X$; Выполнить F;
Если $W \cdot R < 0$, то перейти к M2;
$U = X$;
Перейти к M1;
M2: $V = X$;
Перейти к M1;
M3: $X = (U + V)/2$;
Вывести (X);
Конец;
F: Подпрограмма;
$R = Z^2 - \ln(Z + 2)$;
Конец;

Распределим память МК следующим образом: $a \to$ P0, $b \to$ P1, $U \to$ P2, $V \to$ P3, $\varepsilon \to$ P4, $X \to$ P5 и $W \to$ P6. Перед обращением к подпрограмме значение Z будем заносить в регистр РХ. В этот же регистр будем записывать результат вычислений по подпрограмме.

Транслируя алгоритм с УАЯ на язык ПМК, получаем:

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
00 01	ИП0 П2	60 42	$a \to$ РХ $a \to$ P2	$U = a$;
02 03	ИП1 П3	61 43	$b \to$ РХ $b \to$ P3	$V = b$;
04	ИП2	62	$a \to$ РХ	$Z = a$;
05 06	ПП 50	53 50	Обращение к подпрограмме вычисления $R = Z^2 - \ln(Z + 2)$ Адрес первой команды подпрограммы	Выполнить F;
07	П6	46	$f(a) \to$ P6	$W = R$;
08 09 10 11 12	ИП3 ИП2 — FX < 0 14	63 62 11 5[} 14 }	$V \to$ РХ $U \to$ РХ, $V \to$ РУ $V - U \to$ РХ Если $V - U < 0$, то на 13; иначе на 14	

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
13	⟩ /—/	0L	$\lvert V-U\rvert \to$ PX. Изменение знака разности $V-U$, если $V-U<0$	M1: Если $\lvert V-U\rvert<\varepsilon$, то перейти к M3;
14	ИП4	64	$\varepsilon \to$ PX, $\lvert V-U\rvert \to$ PY	
15	—	11	$\lvert V-U\rvert-\varepsilon \to$ PX	
16	FX <0	5[ )	Если $\lvert V-U\rvert-\varepsilon<0$,	
17	20	20 ∫	то на 18; иначе на 20	
18	БП	51	Безусловный переход	
19	43	43	Адрес перехода	
20	ИП2	62	$U \to$ PX	
21	ИП3	63	$V \to$ PX, $U \to$ PY	
22	+	10	$U+V \to$ PX	
23	2	02	$2 \to$ PX, $U+V \to$ PY	$X=(U+V)/2$;
24	÷	13	$(U+V)/2 \to$ PX	
25	П5	45	$(U+V)/2 \to$ P2	
26	ИП5	65	$X \to$ PX	$Z=X$;
27	ПП	53	Обращение к подпрограмме вычисления $Z^2-\ln(Z+2)$	Выполнить F;
28	50	50	Адрес первой команды подпрограммы	
29	ИП6	66	$f(a) \to$ PX, $X^2-\ln(X+2) \to$ PY	
30	×	12	$f(a)[X^2-\ln(X+2)] \to$ PX	
31	FX <0	59	Если $f(a)[X^2-\ln(X+2)]<0$, то перейти на 33	Если $W \cdot R<0$, то перейти к M2;
32	35	35	Адрес перехода при невыполнении условия $W \cdot R<0$	
33	БП	51	Безусловный переход	
34	39	39	Адрес перехода	
35	ИП5	65	$X \to$ PX	$U=X$;
36	П2	42	$X \to$ P2	

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
37 38	БП 08	51 08	Безусловный переход Адрес перехода	Перейти к М1;
39 40	ИП5 П3	65 43	$X \to$ РХ $X \to$ Р3	М2: $V = X$;
41 42	БП 08	51 08	Безусловный переход Адрес перехода	Перейти к М1;
43 44 45 46 47 48	ИП2 ИП3 + 2 ÷ П5	62 63 10 02 13 45	$U \to$ РХ $V \to$ РХ, $U \to$ РУ $U + V \to$ РХ $2 \to$ РХ, $U + V \to$ РУ $(U + V)/2 \to$ РХ $(U + V)/2 \to$ Р5	М3: $X = (U + V)/2$;
49	С/П	50	Стоп	Вывести (X); Конец;
50 51 52 53 54 55 56 57 58 59	П7 2 + F ln П8 ИП7 FX² ИП8 — В/О	47 02 10 18 48 67 22 68 11 52	$Z \to$ Р7 $2 \to$ РХ, $Z \to$ РУ $Z + 2 \to$ РХ $\ln(Z + 2) \to$ РХ $\ln(Z + 2) \to$ Р8 $Z \to$ РХ $Z^2 \to$ РХ $\ln(Z + 2) \to$ РХ, $Z^2 \to$ РУ $Z^2 - \ln(Z + 2) \to$ РХ Выход из подпрограммы	F: Подпрограмма; $R = Z^2 - \ln(Z+2)$; Конец;

Положив $\varepsilon = 0{,}001$, получим $X \approx 1{,}057098$. При этом $\ln(X + 2) \approx 1{,}117642$, $X^2 \approx 1{,}117456$. Если $\varepsilon = 0{,}000001$, то $X \approx 1{,}057101$, а $\ln(X + 2) = \ln 3{,}057101 = 1{,}117464$, $X^2 = 1{,}057101^2 = 1{,}117462$.

Удобным для приближенного решения уравнений вида $f(X) = 0$ является *метод итераций*. Используя этот метод, уравнения вида $f(X) = 0$ представляют как $X = \varphi(X)$, причем $\varphi(X)$ подбирают так, чтобы выполнялось условие $|\varphi'(X)| \leqslant q < 1$ для всех X из промежутка

$[a, b]$, на котором находится корень уравнения. За начальное приближение берут произвольное значение X_0 из промежутка $[a, b]$ и вычисляют значение

$$X_1 = \varphi(X_0).$$

Если при этом окажется, что $X_1 \approx X_0$, то X_1 и будет решением уравнения. Если же X_1 заметно отличается от X_0, то находят значение

$$X_2 = \varphi(X_1).$$

Этот процесс продолжают до тех пор, пока подставляемое в правую часть уравнения $X = \varphi(X)$ значение аргумента X и получаемое значение функции $\varphi(X)$ не совпадут с точностью до заданной погрешности.

Уравнение $f(X) = 0$ в виде $X = \varphi(X)$ можно представить, например, как

$$X = X - Mf(X).$$

Постоянную M при этом подбирают так, чтобы выполнялось условие

$$|(X - Mf(X))'| < 1.$$

На УАЯ алгоритм решения уравнения $X = \varphi(X)$ по методу итераций можно записать в следующем виде:

```
M0 : Начало;
     Ввести (X0, ε);
     U = X0;
M1 : X = φ(U);
     Если |X − U| < ε, то перейти к M2;
     U = X;
     Перейти к M1;
M2 : Вывести (X);
     Конец;
```

Переведем этот алгоритм на язык ПМК «Электроника Б3-34», распределив память следующим образом: $X_0 \to$ Р2, $\varepsilon \to$ Р3, $U \to$ Р4, $X \to$ Р5. Вычисление значений $\varphi(X)$ оформим в подпрограмму. Тогда получим:

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
00	ИП2	62	$X_0 \to$ РХ	$U = X_0$;
01	П4	44	$X_0 \to$ Р4	

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
02	ПП	53	Обращение к подпрограмме вычисления $\varphi(U)$	
03	21	21	Первый адрес подпрограммы	M1 : $X = \varphi(U)$;
04	П5	45	$\varphi(U) \to$ Р5	
05	ИП5	65	$X \to$ РХ	
06	ИП4	64	$U \to$ РХ, $X \to$ РУ	
07	—	11	$X - U \to$ РХ	
08	FX < 0	5[	Переход на 10 при $X - U < 0$	
09	11	11	Переход на 11 при $X - U \geqslant 0$	Если $\lvert X - U \rvert < \varepsilon$, то перейти к M2;
10	/—/	0L	Изменение знака $X-U$ при $X - U < 0$	
11	ИП3	63	$\varepsilon \to$ РХ, $\lvert X - U \rvert \to$ РУ	
12	—	11	$\lvert X - U \rvert - \varepsilon \to$ РХ	
13	FX ⩾ 0	59	Переход на 15 при $\lvert X - U \rvert \geqslant \varepsilon$	
14	19	19	Переход на 19 при $\lvert X - U \rvert < \varepsilon$	
15	ИП5	65	$X \to$ РХ	$U = X$;
16	П4	44	$X \to$ Р4	
17	БП	51	Безусловный переход	Перейти к M1;
18	02	02	Адрес перехода	
19	ИП5	65	$X \to$ РХ	M2: Вывести (X);
20	С/П	50	Стоп	Конец;

Пример. Найти приближенное решение уравнения

$$X^3 + 1{,}76439X^2 + 2{,}21584X - 3{,}31344 = 0$$

с точностью до $\varepsilon = 10^{-7}$, если известно, что корень уравнения находится на промежутке $[0, 1]$.

Представив это уравнение в виде

$$X = X - (X^3 + 1{,}76439X^2 + 2{,}21584X - 3{,}31344)/10,$$

можно убедиться в том, что

$$|\varphi'(X)| = |1 - (3X^2 + 3{,}52878X + 2{,}21584)/10|$$

лежит в пределах от 0 до 0,8 при X, находящемся в промежутке $[0, 1]$.

Для вычисления значений $\varphi(X)$ напишем подпрограмму, разместив коэффициенты $Q = 1{,}76439$, $R = 2{,}21584$ и $S = -3{,}31344$ в регистрах PB, PC и PD соответственно и размещая команды подпрограммы, начиная с адреса 21:

Адрес команды	Команда	Код команды	Примечание	Соответствующие операторы на УАЯ
21	П9	49	$U \to$ P9	
22	ИПВ	6L	$Q \to$ PX, $U \to$ PY	
23	+	10	$U + Q \to$ PX	
24	ИП9	69	$U \to$ PX, $U + Q \to$ PY	
25	×	12	$U(U + Q) \to$ PX	
26	ИПС	6[	$R \to$ PX, $U(U + Q) \to$ PY	
27	+	10	$U(U + Q) + R \to$ PX	
28	ИП9	69	$U \to$ PX, $U(U + Q) + R \to$ PY	
29	×	12	$U(U(U + Q) + R) \to$ PX	
30	ИПD	6Г	$S \to$ PX, $U(U(U + Q) + R) \to$ PY	
31	+	10	$U(U(U + Q) + R) + S \to$ PX	F: Подпрограмма; $X = (U - (U^3 + QU)^2 + RU + S)/10$; Конец F;
32	1	01	$10 \to$ PX, $U^3 + QU^2 + RU + S \to$ PY	
33	0	00		
34	÷	13	$(U^3 + QU^2 + RU + S)/10 \to$ PX	
35	ИП9	69	$U \to$ PX, $(U^3 + QU^2 + RU + S)/10 \to$ PY	
36	$\overrightarrow{\text{XY}}$ $\overleftarrow{\text{XY}}$	0—	$U \to$ PY, $(U^3 + QU^2 + RU + S)/10 \to$ PX	
37	—	11	$U - (U^3 + QU^2 + RU + S)/10 \to$ PX	
38	В/О	52	Выход из подпрограммы	

Пример. Найти приближенно площадь фигуры, ограниченной кривой $Y = f(X) = \sqrt{X}$ и прямыми $X = a = 0$, $X = b = 4$, $Y = 0$ (рис. 68).

Для решения задачи разобьем промежуток $[a, b]$ на некоторое число достаточно мелких промежутков длиной $h = (b - a)/n$, например $h = 0{,}25$ ($n = 16$). На каждом из этих промежутков заменим криволинейную трапецию, ограниченную линиями $X = X_i$, $X = X_{i+1}$, $Y = 0$ и $Y = f(X)$, прямолинейной с основаниями $f(X_i)$, $f(X_{i+1})$ и высотой h (рис. 69). Вычислив площади

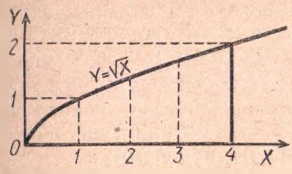


Рис. 68. Криволинейная трапеция, ограниченная линиями $Y = \sqrt{X}$, 0 и $X = 0, 4$

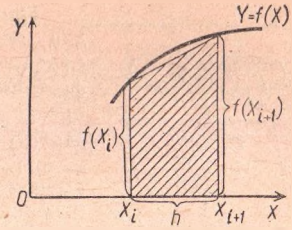


Рис. 69. Приближенная замена криволинейной трапеции прямолинейной

всех прямолинейных трапеций и сложив их, найдем приближенно искомую площадь.

Площадь прямолинейной трапеции с основаниями a, b и высотой h вычисляется по формуле

$$S = \frac{a+b}{2} \cdot h.$$

Поскольку в рассматриваемом примере высоты всех трапеций одинаковы и равны h, в сумме площадей

$$S = (f(X_0) + f(X_1)) \cdot h/2 + (f(X_1) + f(X_2)) \cdot h/2 + (f(X_2) + f(X_3)) \cdot h/2 + (f(X_3) + f(X_4)) \cdot h/2 + \ldots + (f(X_{n-1}) + f(X_n)) \cdot h/2 = ((f(X_0) + f(X_1)) + (f(X_1) + f(X_2)) + f(X_2) + f(X_3)) + ((f(X_3) + f(X_4)) + \ldots + (f(X_{n-1}) + f(X_n))) \cdot h/2$$

множитель $h/2$ можно вынести за скобки и прибавлять каждый раз лишь два соседних значения функции. При этом сумму двух первых значений $f(X_0) + f(X_1)$ нужно прибавлять к нулю, а каждых последующих двух значений — к ранее найденной сумме. Очередную пару аргументов следует выбирать так: первый раз взять пару аргументов $X_0 = a$ и $X_1 = X_0 + h$, затем [после вычисления значений $f(X_0)$ и $f(X_1)$ и прибавления их к сумме ранее накопленных результатов] оба аргумента X_0 и X_1 увеличить на h, полагая $X_0 = X_0 + h$, $X_1 = X_1 + h$. Если при этом X_1 еще не выходит за пределы промежутка $[a, b]$, то во вновь полученных точках X_0 и X_1 надо вычислить значения $f(X_0)$, $f(X_1)$, прибавить их к сумме ранее накопленных значений, после чего вновь нарастить аргументы X_0 и X_1 на h. Так нужно действовать до тех пор, пока X_1 не выйдет за пределы промежутка $[a, b]$. Как только X_1 выйдет за пределы промежутка $[a, b]$, это будет означать, что вновь полученная трапеция уже не входит в число прямолинейных трапеций, с помощью которых приближенно заменена криволинейная трапеция. В этом случае повторение вычислений необходимо прекратить, найденную сумму значений $\sum_{i=0}^{n-1} (f(X_i) + f(X_{i+1}))$ умножить на $h/2$ и полученный результат считать приближенным значением искомой площади криволинейной трапеции. При этом, чем меньше h, тем точнее можно вычислить площадь криволинейной трапеции [при условии, что значения $f(X_i)$ вычисляются точно].

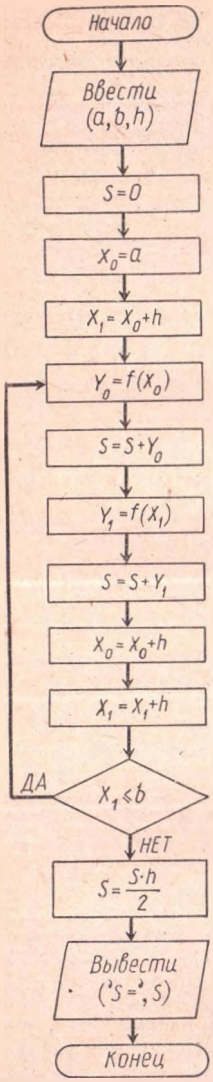


Рис. 70. Схема алгоритма приближенного вычисления площади криволинейной трапеции

Приведенные рассуждения действительны для любой функции $f(X)$, а не только для $f(X) = \sqrt{X}$. Поэтому вычисление значений функции $f(X)$ целесообразно оформить в подпрограмму с тем, чтобы при необходимости замены функции $f(X) = \sqrt{X}$ другой функцией можно было заменить только подпрограмму вычисления значений функции, не внося изменений в основную программу.

Описанный алгоритм приближенного вычисления площади криволинейной трапеции на УАЯ можно представить в виде:

A: Начало;
Ввести $(a,\ b,\ h)$;
$S = 0$;
$X_0 = a;\ X_1 = X_0 + h$;
M: $Z = X_0$;
Выполнить F;
$Y_0 = W$;
$S = S + Y_0$;
$Z = X_1$;
Выполнить F;
$Y_1 = W$;
$S = S + Y_1$;
$X_0 = X_0 + h;\ X_1 = X_1 + h$;
Если $X_1 \leqslant b$, то перейти к M;
$S = S \cdot h/2$;
Вывести ('S =', S);
Конец A;
F: Подпрограмма;
$W = \sqrt{Z}$;
Конец F;

Схема этого алгоритма изображена на рис. 70.

Переведем теперь описание алгоритма с УАЯ на язык МК, распределив память следующим образом: $S \to$ P0, $a \to$ P1, $b \to$ P2, $h \to$ P3, $X_0 \to$ P4, $X_1 \to$ P5, $Y_0 \to$ P6, $Y_1 \to$ P7. Перед обращением к подпрограмме аргумент для подпрограммы вычисления значения функции $f(X)$ будем записывать в регистр РХ, результат вычислений по подпрограмме также будем оставлять в регистре РХ. Управление операциями ввода — вывода, как и раньше, осуществляем вручную с клавиатуры МК.

Учитывая изложенное, программу решения рассмотренной задачи на МК можно записать в виде:

Адрес команды	Команда	Код команды	Примечание	Соответствующие пункты алгоритма на УАЯ
00 01	0 П0	00 40	$0 \to$ РХ $0 \to$ Р0	$S = 0$;
02 03	ИП1 П4	61 44	$a \to$ РХ $a \to$ Р4	$X_0 = a$;
04 05 06	ИП3 + П5	63 10 45	$h \to$ РХ, $a \to$ РУ $X_0 + h \to$ РХ $X_0 + h \to$ Р5	$X_1 = X_0 + h$;
07	ИП4	64	$X_0 \to$ РХ	М: $Z = X_0$;
08 09	ПП 43	53 43	Обращение к подпрограмме вычисления $f(X)$ Адрес первой команды подпрограммы	Выполнить F;
10	П6	46	$Y_0 \to$ Р6	$Y_0 = W$;
11 12 13	ИП0 + П0	60 10 40	$S \to$ РХ, $Y_0 \to$ РУ $S + Y_0 \to$ РХ $S + Y_0 \to$ Р0	$S = S + Y_0$;
14	ИП5	65	$X_1 \to$ РХ	$Z = X_1$;
15 16	ПП 43	53 43	Обращение к подпрограмме вычисления $f(X)$ Адрес первой команды подпрограммы	Выполнить F;
17	П7	47	$Y_1 \to$ Р7	$Y_1 = W$;
18 19 20	ИП0 + П0	60 10 40	$S \to$ РХ, $Y_1 \to$ РУ $S + Y_1 \to$ РХ $S + Y_1 \to$ Р0	$S = S + Y_1$;
21 22 23	ИП4 ИП3 +	64 63 10	$X_0 \to$ РХ $h \to$ РХ, $X_0 \to$ РУ $X_0 + h \to$ РХ	$X_0 = X_0 + h$;

Продолжение табл.

Адрес команды	Команда	Код команды	Примечание	Соответствующие пункты алгоритма на УАЯ
24	П4	44	$X_0 + h \to$ P4	
25	ИП5	65	$X_1 \to$ PX	
26	ИП3	63	$h \to$ PX, $X_1 \to$ PY	
27	+	10	$X_1 + h \to$ PX	$X_1 = X_1 + h$;
28	П5	45	$X_1 + h \to$ P5	
29	ИП2	62	$b \to$ PX, $X_1 \to$ PY	
30	$\vec{XY}$	0—	$b \to$ PY, $X_1 \to$ PX	
31	—	11	$b - X_1 \to$ PX	
32	FX $\geqslant$ 0	59	Если $b - X_1 \geqslant 0$, то перейти к 33; если $b - X_1 < 0$ — к 35	Если $X_1 \leqslant b$, то перейти к M;
33	36	35	Адрес перехода при невыполнении условия $b - X_1 \geqslant 0$ (или $X_1 \leqslant b$)	
34	БП	51	Безусловный переход	
35	07	07	Адрес перехода	
36	ИП0	60	$S \to$ PX	
37	ИП3	63	$h \to$ PX, $S \to$ PY	
38	×	12	$S \cdot h \to$ PX	
39	2	02	$2 \to$ PX, $S \cdot h \to$ PY	$S = S \cdot h/2$;
40	÷	13	$S \cdot h/2 \to$ PX	
41	П0	40	$S \cdot h/2 \to$ P0	
42	С/П	50	Стоп	Вывести ('S='S); Конец A;
43	F $\sqrt{-}$	21	$\sqrt{X} \to$ PX	F: Подпрограмма; $W = \sqrt{Z}$; Конец F;
44	В/О	52	Выход из подпрограммы	

§ 19. ЭЛЕМЕНТЫ СТРУКТУРНОГО ПРОГРАММИРОВАНИЯ

В настоящее время, когда сферы применения ЭВМ необычайно расширились, программы для ЭВМ интересуют многих. Поэтому создаются системы программ с учетом

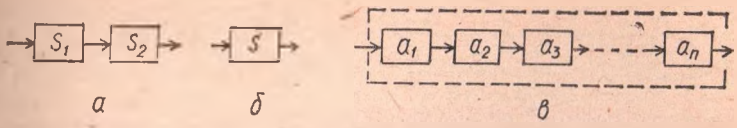


Рис. 71. К рассмотрению базовой структуры СЛЕДОВАНИЕ

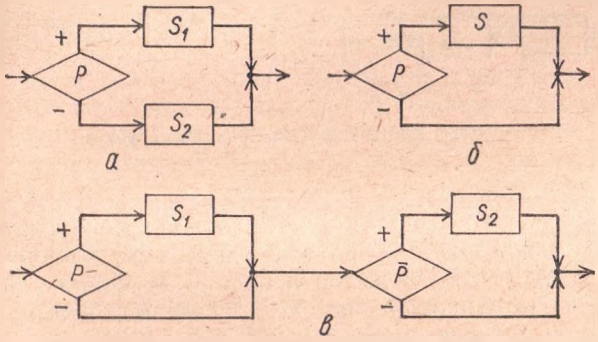


Рис. 72. К рассмотрению базовой структуры РАЗВИЛКА

многократного их использования различными пользователями, исходя из чего предъявляется ряд естественных требований к описанию алгоритмов. В частности, оно должно быть понятным (т. е. легко воспринимаемым), легко проверяемым и должно допускать возможность модификации без существенной перестройки структуры. С этой целью при разработке алгоритмов придерживаются особой методики, называемой *структурным подходом*, а сами алгоритмы конструируют из трех основных (базовых) структур: СЛЕДОВАНИЕ, РАЗВИЛКА и ЦИКЛ.

Графическое изображение базовой структуры СЛЕДОВАНИЕ показано на рис. 71,*а*. Эта структура состоит из двух функциональных элементов S_1, S_2 и означает, что два таких элемента могут быть размещены один за другим.

Функциональному элементу, графически изображаемому в виде прямоугольника с одним входом и одним выходом (рис. 71,*б*), в языках программирования соответствуют операторы ввода и вывода или любой оператор (группа операторов) присваивания. В виде функционального элемента может быть изображена любая последовательность операторов, имеющих по одному входу и выходу (рис. 71,*в*).

Условная конструкция РАЗВИЛКА (рис. 72,*а*) включает проверку некоторого логического условия P, в зави-

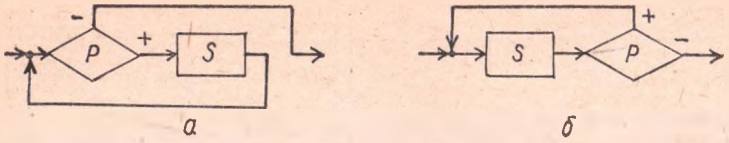


Рис. 73. К рассмотрению базовой структуры ЦИКЛ

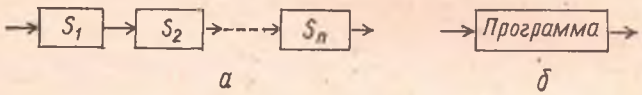


Рис. 74. К рассмотрению линейной цепочки базовых структур

симости от чего выполняется либо функциональный элемент S_1, либо функциональный элемент S_2. Частным случаем РАЗВИЛКИ является неполная условная конструкция, изображенная на рис. 72,*б*, когда при невыполнении условия P вообще не производится никакое действие.

Заметим, что полную условную конструкцию всегда можно реализовать с помощью двух неполных конструкций (рис. 72. в); поэтому в принципе любую программу можно составить с использованием только неполных условных конструкций, где $\overline{P}$ — условие, противоположное условию P.

Базовая структура ЦИКЛ может быть двух видов (рис. 73). В первом случае (рис. 73,*а*) функциональный элемент S размещен после проверки условия P и в принципе возможно, что функция данного элемента не выполнится ни разу. Такой вариант структуры ЦИКЛ называют ЦИКЛ — ПОКА (цикл с предусловием). Во втором случае (рис. 73,*б*) функция элемента S выполняется, по крайней мере, один раз. Этот вариант структуры ЦИКЛ называют ЦИКЛ — ДО (цикл с постусловием).

Важнейшей особенностью рассмотренных структур является то, что каждая из них имеет по одному входу и выходу. Элементы S_1 и S_2, входящие в состав этих структур, сами могут представлять собой одну из таких структур. При конструировании программ с использованием названных элементов последние соединяют в линейную цепочку так, чтобы выход одного из них подсоединялся к входу следующего, как показано на рис. 74,*а*.

Для построения более сложных алгоритмов из простейших (базовых) структур разрешается только подсоединять одну

структуру к другой с целью образования последовательности структур и заменять функциональные элементы S, S_1, S_2 любой из базовых структур. Придерживаясь этих правил, можно строить сложные алгоритмы для решения разных задач. В результате составленная программа будет иметь линейную структуру, причем порядок следования ее элементов соответствует порядку их выполнения. Такая структура облегчает чтение и понимание программы, упрощая доказательство ее правильности. Поскольку линейная цепочка элементов может быть объединена в один элемент, любая программа в конечном итоге может рассматриваться как единый функциональный элемент с одним входом и одним выходом (рис. 74, *б*).

При написании программы нужно выполнить обратное преобразование, т. е. разбить ее на последовательность сначала крупных частей, затем каждую часть — на последовательность более мелких частей. Этот процесс продолжают до тех пор, пока не будут получены «атомарные» элементы рассмотренных выше типов. Такой метод конструирования программы принято называть *нисходящим* (сверху вниз).

При нисходящем методе первоначально задачу рассматривают в целом. На каждом последующем этапе ее разбивают на более мелкие подзадачи, каждую подзадачу — на еще более мелкие подзадачи и так до тех пор, пока не будут получены такие подзадачи, которые легко программируются на выбранном языке программирования. При этом на каждом шаге уточняют все новые и новые детали (*пошаговая детализация*).

В процессе нисходящего разбиения задачи сохраняется строгая дисциплина программирования, так как разбиение на подзадачи осуществляется с использованием только рассмотренных типов структур, чем и обеспечивается хорошо структурированная программа, которую легко читать сверху донизу без перерыва.

Структурное программирование и преследует цель облегчить процесс проверки правильности программы, улучшить ее ясность и читабельность, повысить производительность труда программистов. Например, алгоритм отыскания наибольшего из трех чисел a, b, c графически можно изобразить так, как показано на рис. 75,*а*; алгоритм нахождения наибольшего из n чисел a_1, a_2, ..., a_n — так, как представлено на рис. 75,*б*.

Чтобы при записи алгоритма на алгоритмическом языке указать, где начинается и заканчивается группа операто-

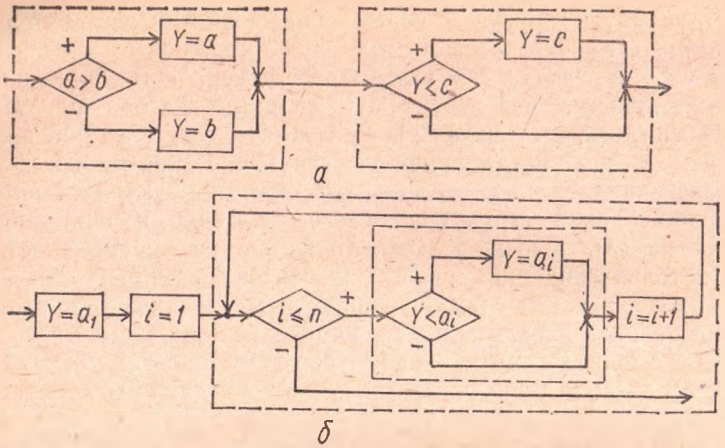


Рис. 75. Схемы алгоритмов отыскания наибольшего из трех чисел (*а*) и наибольшего из *n* чисел (*б*)

ров, такую группу заключают в так называемые *операторные скобки*. В качестве открывающей операторной скобки используют слова Начало группы (или просто Начало), а в качестве закрывающей — слова Конец группы (или просто Конец).

Полная условная конструкция РАЗВИЛКА на УАЯ описывается условным оператором вида:

Если P, то (первая группа операторов);

Иначе (вторая группа операторов);

По этой команде проверяется выполнение условия P. Если оно выполняется, то выполняется первая группа операторов, а вторая — пропускается. Если же условие P не выполняется, то первая группа операторов пропускается и выполняется вторая группа. Например, алгоритм решения квадратного уравнения $aX^2 + bX + c = 0$ можно записать так:

```
M0: Начало;
    Ввести (a, b, c);
    D = b^2 — 4ac;
    Если D ⩾ 0, то
    Начало;
    W = 2a;
    S = √D;
    X1 = (—b — S)/W;
    X2 = (—b + S)/W;
    Вывести ('X1 = ', X1,
                'X2 = ', X2);
    Конец;
    Иначе
    Начало;
    Вывести ('Действительных корней нет')
    Конец;
Конец M0;
```

Если в группе один оператор, то операторные скобки можно опустить.

Неполная условная конструкция РАЗВИЛКА на УАЯ описывается условным оператором вида:

Если P, то (группа операторов);

По этой команде проверяется выполнение условия P. Если оно выполняется, то выполняется группа операторов, записанная после слова ТО, а если нет, то данная группа операторов пропускается.

Группа операторов, записанная после слова ТО, может состоять из одного оператора. В частности, таким оператором может быть оператор безусловного перехода Перейти к М;.

Если в группе один оператор, то операторные скобки в начале и в конце группы можно опустить. Например, алгоритм выбора большего из двух чисел a и b можно представить в виде:

Если $a > b$, то $Y = a$; иначе $Y = b$;

или в виде:

$Y = b$;

Если $a > b$, то $Y = a$;

или в виде:

$Y = a$;

Если $b > a$, то $Y = b$;

Базовая структура ЦИКЛ — ПОКА на УАЯ описывается оператором вида:

Пока P выполнять (группа операторов);

При этом группа операторов цикла заключается в операторные скобки Начало цикла (иногда пишут только Начало) и Конец цикла (иногда пишут только Конец). Например, алгоритм произведения натуральных чисел $1 \cdot 2 \cdot \ldots \cdot n$ можно записать так:

```
М0: Начало;                    Начало цикла;
    Ввести (n);                N = N · k;
    k = 1;                     k = k + 1;
    Пока k ≤ n выполнять;      Конец цикла;
                               Конец М0;
```

Группа операторов цикла выполняется до тех пор, пока не нарушится условие P, записанное после слова ПОКА.

В алгоритмических языках используется еще один вид цикла — ЦИКЛ С ПАРАМЕТРОМ. Этот цикл записывается в виде:

Для $X = a$ до b с шагом h выполнять (группа операторов цикла);

Как и раньше, группа операторов цикла заключается в операторные скобки Начало цикла и Конец цикла (иногда пишут только Начало и Конец). Например, алгоритм вычисления всех значений функции

$$Y = \frac{2 \sin X + \cos X^2}{1 + X^2}$$

для значений X, изменяющихся с шагом h от $X = a$ до $X = b$, можно представить в виде:

```
M0: Начало;                          Y = (2 · sin(X) +
    Ввести (a, b, h);                + cos(X²))/(1 + X²);
    Для X = a до b с ша-             Вывести ('X = ', X,
        гом h выполнять;             ', Y = ', Y);
    Начало цикла;                    Конец цикла;
                                     Конец M0;
```

Ясно, что все рассмотренные здесь структуры могут быть реализованы и с помощью введенных ранее (см. § 14, 15) операторов. Для базовой структуры СЛЕДОВАНИЕ это очевидно: группы операторов S_1 и S_2 записываются одна за другой.

Полная условная конструкция РАЗВИЛКА может быть представлена так:

```
        Если P, то перейти к M1;
        (вторая группа операторов);
        Перейти к M2;
M1:     (первая группа операторов);
M2:
```

а неполная — так:

```
        M1: Если P, то перейти к M2;
            (группа операторов);
        M2:
```

ЦИКЛ — ПОКА можно представить в виде:

```
        M1: Если P, то перейти к M2;
            (группа операторов цикла);
        M2:
```

При этом последний оператор в группе операторов цикла должен передать управление на проверку условия P, т. е. должен быть оператором Перейти к M1;

ЦИКЛ — ДО можно представить в виде:

М:
(группа операторов цикла);
Если P, то перейти к М;

а ЦИКЛ С ПАРАМЕТРОМ — в виде:

$X = a$:
М1: Если $X > b$, то перейти к М2;
(группа операторов);
$X = X + h$;
Перейти к М1;
М2:

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как изображается базовая структура СЛЕДОВАНИЕ? 2. Как изображается функциональный элемент? 3. Как изображается полная условная конструкция РАЗВИЛКА? 4. Как изображается неполная условная конструкция РАЗВИЛКА? 5. Как изображается базовая структура ЦИКЛ? 6. Как обозначаются начало и конец группы операторов при ее записи на УАЯ? 7. Как на УАЯ описываются условные конструкции РАЗВИЛКА? 8. Как на УАЯ описываются структуры вида ЦИКЛ?

Глава 3

АВТОМАТИЗАЦИЯ ПРОГРАММИРОВАНИЯ

§ 20. ПОНЯТИЕ О МАШИННОМ ЯЗЫКЕ МИКРО-ЭВМ

Элементарные указания, которые может выполнять ЭВМ, называются *машинными командами*, а набор всех допустимых для данной ЭВМ указаний — *системой команд* ЭВМ. Каждая команда полностью определяет ту работу, которую должна выполнить ЭВМ в очередной промежуток времени.

В связи с этим в любой команде должна содержаться информация о том, какую операцию необходимо выполнить; должны быть указаны места в памяти ЭВМ, где хранятся числа, с которыми надо выполнить эту операцию: место для хранения полученного результата, а также адрес команды, которая должна выполняться следующей. Вся эта информация должна быть представлена в цифровой форме в виде двоичного числа, причем часть информации может не указываться в явном виде, а определяться по некоторым правилам, заложенным в конструкции машины.

Каждая операция имеет свое числовое обозначение — *код*, который и задает ее для выполнения. Информация о том, где хранятся числа, с которыми необходимо выполнить требуемую операцию, куда направить полученный результат, какой адрес следующей команды, содержится в адресной части команды или задается неявно.

Исходя из количества адресов в адресной части различают *безадресные*, *одно-*, *двух-* и *трехадресные* команды.

Трехадресная команда имеет вид:

КОП	AI	AII	AIII

где КОП — код операции, AI — первый адрес, AII — второй адрес, AIII — третий адрес. Например, для конкретной ЭВМ такая команда, записанная в виде

00000001	00010000	00010001	00010010

указывает на то, что необходимо взять числа из ячеек памяти с двоичными номерами 00010000 (первый адрес) и 00010001 (второй адрес), произвести над ними операцию, имеющую код 00000001, и полученный результат поместить в ячейку с адресом 00010010 (третий адрес). Указанный код может обозначать, например, операцию сложения.

Если адрес следующей команды явно не указан, то выполняется команда, адрес которой на единицу больше адреса предыдущей команды.

Двухадресная команда имеет вид:

Здесь третий адрес задан неявно. Как правило, третье число хранится в специальном регистре процессора — а к к у м у л я т о р е.

Одноадресная команда имеет вид:

Выполнение ее сводится чаще всего к выборке числа из ячейки с адресом, указанным в адресной части А1, и выполнению над ним и, возможно, числом, находящимся в аккумуляторе, операции, определяемой ее кодом. Результат операции остается в аккумуляторе.

Безадресная команда имеет вид:

Примером такой команды может быть команда для ПМК, где в ее коде содержится информация только о выполняемой операции. Однако конструкцией МК предусмотрено, что числа, с которыми будет выполняться требуемая операция, должны находиться в операционных регистрах РХ и РУ. Результат выполнения операции направляется в регистр РХ, хотя в коде команды об этом явно не указано.

Можно выделить следующие основные группы команд ЭВМ:

1. *Команды передачи данных*, обеспечивающие простую пересылку информации без выполнения каких-либо операций ее обработки. Команды этой группы делятся на команды, связанные с обращением к памяти, команды, связанные с обращением к специальным регистрам микропроцессора, и команды ввода — вывода.

2. *Команды управления* (делятся на команды безусловных и условных переходов).

3. *Команды обработки данных* (делятся на арифметические и логические).

4. *Специальные команды*, используемые для управления работой микропроцессора. По этим командам не передается и не обрабатывается информация. К ним относится, например, команда «Стоп» о прекращении выполнения программы.

Система команд микропроцессора зависит от его структуры. Для решения некоторой задачи с помощью микропроцессора необходимо описать алгоритм ее решения на языке микропроцессора. Иными словами, нужно записать алгоритм решения задачи через систему команд микропроцессора.

При подготовке задач к решению на микро-ЭВМ команды могут записываться в том же виде, в каком они будут храниться в ЗУ, т. е. в виде двоичных кодов. Вся совокупность таких последовательно записанных команд образует п р о г р а м м у на так называемом *машинном языке*.

Каждая ЭВМ имеет свой машинный язык, представляющий собой набор команд, записанных в специальном коде. Составление программ в кодах команд ЭВМ называется *программированием на машинном языке*. При составлении таких программ надо хорошо знать особенности структуры данной ЭВМ, организацию ее памяти и коды команд. Чтобы записывать программы на машинном языке, необходимо иметь таблицу кодов команд и четко представлять себе, как они выполняются.

На практике непосредственно в кодах микропроцессора программируют редко. Программа, составленная на машинном языке для одной ЭВМ, как правило, не годится для другой. Однако в отдельных случаях целесообразно составлять программы именно на машинном языке.

Трудности и недостатки программирования на машинном языке в значительной мере устраняются с помощью языка А с с е м б л е р а, который сохраняет все возможности машинного языка благодаря замене числового двоичного представления всех частей каждой машинной команды (кода операции, адреса и т. д.) символическими обозначениями — *мнемокодами*.

Язык Ассемблера, называемый иногда языком *символического кодирования* или *автокодом*, позволяет работать с командами, очень похожими на команды машинного языка. Отличительной особенностью языка Ассемблера

является то, что каждой его команде, как правило, соответствует точно одна команда машинного языка.

Для каждой ЭВМ создается свой язык Ассемблера. Программа, написанная на этом языке, переводится затем в машинную программу с использованием формальных правил. Этот перевод можно автоматизировать, т. е. производить его с помощью ЭВМ по специально составленной программе, которая называется транслятором. В частности, на клавиатуре ПМК указаны мнемокоды команд, так что программы для МК были записаны, по сути, на языке Ассемблера. При занесении программ в память ПМК команды с языка Ассемблера после нажатия соответствующих клавиш автоматически переводятся на машинный язык и в память МК записываются уже коды команд, т. е. команды, представленные на машинном языке.

Таким образом, для программирования задач и решения их на ЭВМ необходимо знать тот или иной язык программирования. Команды, записанные на языке программирования, должны восприниматься и выполняться данной ЭВМ непосредственно или после трансляции.

Языки программирования подразделяются на *машинно-ориентированные (машинно-зависимые)* и *машинно-независимые*. К машинно-ориентированным относятся, например, машинные языки и языки Ассемблера.

Машинно-независимые *(проблемно-ориентированные)* языки по структуре существенно ближе к обычному языку, чем к машинному, и обладают универсальностью. Программы, составленные на таких языках, почти не опираются на особенности конкретной ЭВМ и могут использоваться на различных машинах. Отметим, что УАЯ не ориентирован на какую-либо ЭВМ и относится к машинно-независимым алгоритмическим языкам. В то же время программы, написанные на языке ПМК «Электроника Б3-34», ориентированы только на такие МК. Поэтому язык МК относят к машинно-ориентированным алгоритмическим языкам.

Применение для программирования универсальных (не зависящих от конкретной машины) алгоритмических языков обеспечивает более высокий уровень автоматизации программирования. Алгоритмические языки допускают использование обычных математических символов и конструкций и некоторого небольшого числа словесных связок между этими конструкциями. Программа, написанная на алгоритмическом языке, напоминает обычный математический текст. Она состоит из операторов, представляющих собой мнемонические коды и определяющих действия,

которые машине надо выполнить (например, ввести данные, вычислить значение выражения, напечатать результат и т. д.). Каждый оператор программы может быть реализован несколькими машинными командами.

Программирование на алгоритмическом языке требует выполнения точных и однозначных правил, нарушение которых недопустимо и может привести либо к ошибке при исполнении алгоритма, либо к полной невозможности его исполнения. Чтобы программу, написанную на алгоритмическом языке, можно было выполнить на ЭВМ, имеющей свой индивидуальный язык, используют программу-транслятор. Последняя переводит программу, написанную на алгоритмическом языке, в программу, реализующую тот же алгоритм, но выраженную уже средствами языка машины.

В любом алгоритмическом языке можно выделить четыре элемента: символы, слова, выражения и операторы. В описание алгоритмического языка входят: 1) список символов; 2) правила образования слов; 3) описание выражений, имеющих смысл в данном языке; 4) разбор всех допустимых типов операторов.

Наиболее распространенными языками при работе на микро-ЭВМ являются ФОРТРАН, КОБОЛ, БЕЙСИК, ПЛ/1, ПАСКАЛЬ. Краткую характеристику и области применения каждого из них дает таблица:

Транскрипция		Краткая характеристика языка программирования
русская	английская	
АЛГОЛ	ALGOL (**ALGO**ritmik **L**anguage)	Для решения задач вычислительного характера
ФОРТРАН	FORTRAN (**FOR**mylas **TRAN**slation)	Для решения научных и инженерных задач
КОБОЛ	COBOL (**CO**mmon **B**usiness **O**riented **L**anguage)	Для задач обработки данных экономического (делового) характера
ПЛ/I	PL/I (Programming **L**anguage/**O**ne)	Многоцелевой универсальный для решения экономических задач, объединяет понятия языков АЛГОЛ-60, ФОРТРАН и КОБОЛ

Продолжение табл.

Транскрипция		Краткая характеристика языка программирования
русская	английская	
БЕЙСИК	BASIC (Beginner's All-purpose Symbolic Instruction Code)	Для решения различных задач вычислительного характера, возможен диалог человека с машиной

Каждый из этих языков требует специального изучения. Только после этого на нем можно составлять программы. Наиболее простым является алгоритмический язык БЕЙСИК, который используется при работе на многих микро-ЭВМ.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Что называется машинными командами? 2. Что называется системой команд ЭВМ? 3. Что называется кодом команды? 4. Что называется кодом операции? 5. Что указывают в адресной части команды? 6. Сколько адресов может содержать адресная часть команды? 7. Бывают ли безадресные команды? 8. Что такое язык Ассемблера? 9. Что называется трансляцией с языка Ассемблера в машинный язык? 10. Какой язык программирования называется машинным? 11. Назовите основные группы команд в системах команд микропроцессоров. 12. Какой язык программирования называется машинно-независимым? 13. Как на ЭВМ можно реализовать программу, написанную на машинно-независимом языке? 14. Почему УАЯ можно отнести к машинно-независимым языкам? 15. Можно ли отнести язык ПМК «Электроника Б3-34» к машинно-независимым алгоритмическим языкам?

§ 21. АЛФАВИТ АЛГОРИТМИЧЕСКОГО ЯЗЫКА БЕЙСИК

Рассмотрим правила описания алгоритмов на алгоритмическом языке БЕЙСИК (BASIC). Алфавит этого языка содержит:

1) 26 заглавных букв латинского алфавита (от A до Z);

2) 10 цифр (0, 1, 2, 3, 4, 5, 6, 7, 8, 9);

3) знаки арифметических операций: сложения (+), вычитания (—), умножения (*), деления (/), возведения в степень (↑);

4) знаки отношений: меньше (<), не больше (<=), равно (=), не равно (<>), больше (>), не меньше (>=);

5) разделительные знаки: открывающая скобка (( ), закрывающая скобка ( )), десятичная точка (.), запятая (,),

точка с запятой (;), кавычки ('), пробел (␣), возврат каретки (↓).

Числа в языке БЕЙСИК записывают в двух формах:

1. В обычной форме (с фиксированной запятой), например —3; 25.7; 0.73;—252.73 и т. д. Для отделения дробной части от целой в изображении числа используется десятичная точка.

2. В виде двух чисел — мантиссы и порядка, разделенных между собой буквой Е, например 5.31Е4; —0.2348Е—5; 561.027Е12 и т. д. Такая форма записи чисел называется *формой с плавающей запятой*. Например, число 30 можно записать в виде 30 или 3Е01, или 0.3Е02, или 300Е — 1; число — 0,0038 — в виде — 0.0038 или —3.8Е — 3; число 0,0023456 — в виде 0.23456Е — 2 или 23.456Е — 4.

Идентификаторы (обозначения) простых переменных в языке БЕЙСИК должны состоять либо из одной буквы, либо из буквы и следующей за ней цифры, например A; B2; C9; F4. Никакие другие наборы символов для обозначения переменных величин использовать не разрешается. Идентификаторы (обозначения) совокупностей чисел могут состоять только из одной буквы.

Изображения чисел и обозначения переменных и функций, объединенные с помощью знаков арифметических операций и скобок, образуют арифметические выражения, например X — Y ↑ 3*Z. В частности, арифметическое выражение может быть образовано из отдельного идентификатора или изображения отдельного числа.

Порядок выполнения операций в арифметических выражениях определяется общепринятым старшинством операций: 1) вычисление значений функций; 2) возведение в степень; 3) умножение и деление; 4) сложение и вычитание. Желаемый порядок выполнения операций может быть указан с помощью скобок. При этом разрешается использовать только круглые скобки.

При отсутствии скобок в выражении операции, имеющие одинаковый приоритет, выполняются слева направо, например выражение A + B * C ↑ D вычисляется так: C возводится в степень D, затем результат умножается на B и складывается с A.

Любые выражения, в том числе и арифметические, на языке БЕЙСИК записываются в одну строку. Надстрочные и подстрочные записи не допускаются. Не допускается использование «двухэтажных» записей вида a^2, $\frac{c}{d}$, X_1. В УАЯ эти ограничения были несущественны и поэтому не вводились. В реальных же алгоритмических языках линей-

ность записей обусловлена конструкцией устройств ввода информации в память ЭВМ. Например:

Общепринятая запись арифметического выражения	Запись арифметического выражения на языке БЕЙСИК
$\frac{2X-5}{3+X}$	(2 * X — 5)/(3 + X)
$a^3 - b^{-3}$	A ↑ 3 — B ↑ (—3)
$aX + b + 0{,}3$	A * X + B + 0.3
$\frac{aX^2 + bX + c}{d - 2{,}5}$	(A * X ↑ 2 + B * X + C)/(D — 2.5)
$(a^b)^c$	(A ↑ B) ↑ C
a^{bc}	A ↑ (B ↑ C)
$b + \sqrt{A}$	B + A ↑ 0.5

Программы вычисления значений наиболее часто встречающихся функций оформлены в виде стандартных подпрограмм. Обозначения этих функций могут быть включены в арифметические выражения. Для их обозначения используются трехбуквенные идентификаторы. Аргумент функции заключается в круглые скобки. В качестве аргумента может быть использовано любое арифметическое выражение. В тригонометрических функциях аргумент задается только в радианах.

В языке БЕЙСИК используются следующие функции:

Функция	Обозначение функции на языке БЕЙСИК	Примечание
$\sin X$	SIN (X)	Аргумент X в радианах
$\cos X$	COS (X)	То же
$\operatorname{tg} X$	TAN (X)	»
$\operatorname{arctg} X$	ATN (X)	Аргумент — произвольное число
e^X	EXP (X)	e^X меньше максимального числа, представимого в ЭВМ
$\ln X$	LOG (X)	$X > 0$
$\lvert X \rvert$	ABS (X)	Аргумент — произвольное число
$\sqrt{X}$	SQR (X)	$X \geqslant 0$

Упражнения. 1. Записать на языке БЕЙСИК выражения: а) $X^2 + 4{,}5X - 8{,}321$; б) $\sin X^2 - 5{,}782 \cos X + 3{,}281$;

в) $\operatorname{tg}^2 X - \sin 2X + 3\cos^3(2X^2 - 3{,}8)^2$;

г) $|X^2 - 3X - 12| - \sqrt{2{,}5X + 8{,}7}$;

д) $\sqrt{1 - \sin X^2} + \sqrt{1 - \cos X^2}$ е) $\frac{1 - X^2}{1 + X^2}$;

ж) $(a + bc - b)(2c - b)^2$.

2. Проверить, правильно ли записаны выражения на языке БЕЙСИК:

а) A * B + 2C — 3A; б) A * B — 2 * SIN X + 3 COS X;
в) 2A — 4.875 * 2 ↑ SIN (X); г) SIN (X)/(COS (X) — 2.752);
д) A2 + B2 — 3.44; е) A ↑ 2 + B ↑ 2 — C2 ↑ 4;
ж) SQR (1—X ↑ 2) + ABSX; з) TAN (X) — COS (X) +SIN ↑ 2 (X).

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Какие символы входят в алфавит языка БЕЙСИК? 2. В какой форме записываются числа на языке БЕЙСИК? 3. Из скольких и каких символов может быть образовано обозначение переменной величины на языке БЕЙСИК? 4. Может ли арифметическое выражение быть образовано из одного только идентификатора или изображения числа? 5. В каком порядке выполняются операции в арифметических выражениях, записанных на языке БЕЙСИК? 6. Можно ли в выражениях, записанных на языке БЕЙСИК, использовать фигурные или квадратные скобки? 7. Можно ли задавать на языке БЕЙСИК аргументы тригонометрических функций в градусах?

§ 22. ОСНОВНЫЕ ОПЕРАТОРЫ НА ЯЗЫКЕ БЕЙСИК

Для того чтобы составить любую программу на языке БЕЙСИК, необходимо, как и в УАЯ, знать правила записи операторов. Все операторы программы, написанной на языке БЕЙСИК, должны быть пронумерованы. Наименьший возможный номер оператора — 1, наибольший — 9999. Как правило, номера соседних операторов отличаются на несколько единиц. Это делается для того, чтобы при необходимости можно было между двумя операторами вставить третий, не меняя нумерацию всех ранее записанных операторов.

Номера операторов играют роль меток. Рассмотрим правила записи основных операторов на языке БЕЙСИК. При этом будем сравнивать их с соответствующими записями на УАЯ.

Оператор

Начало;

на языке БЕЙСИК не имеет записи.

Оператор окончания вычислений, записываемый на УАЯ в виде

Конец;

на языке БЕЙСИК записывается так:

N0 END

где N0 — номер (метка) оператора, причем этот номер должен быть больше, чем номер любого другого оператора в программе. Слово END в переводе означает «конец».

Оператор присваивания, записываемый на УАЯ в виде

Y = A;

на языке БЕЙСИК записывается так:

N0 LET Y = A

Здесь A — некоторое арифметическое выражение, задающее правило вычисления одного числового значения; Y — переменная, которой присваивается полученное числовое значение; слово LET в переводе означает «пусть»; N0 — номер оператора. Например:

5 LET X = 1 15 LET Y = X ↑ 2 + Y ↑ 2
10 LET Y = 2.5 20 LET Z = SIN (2 * Y + 7.5)

Выполнение оператора присваивания сводится к двум последовательным шагам: 1) вычислению значения выражения, записанного справа от знака равенства; 2) занесению найденного значения в ячейку оперативной памяти, выделенную для хранения значений переменной, указанной слева от знака равенства.

Оператор безусловного перехода, записываемый на УАЯ в виде

Перейти к M;

на языке БЕЙСИК записывается так:

N0 GO TO N1

Слова GO TO в переводе означают «перейти к »; N1 — номер оператора, который будет выполняться вслед за оператором с номером N0.

Оператор безусловного перехода используется в тех случаях, когда необходимо нарушить естественный порядок выполнения операторов программы и осуществить переход к заданному оператору независимо от выполнения каких-либо условий. Например, во фрагменте программы

```
10 LET A = 3.17
20 LET B = 1.24E2
30 LET A = A + 2 * B
40 LET C = SIN (A) + B ↑ 2/COS (A * B)
```

```
50 GO TO 70
60 LET B = LOG (C)
70 END
```

в строке под номером 50 записан оператор GO TO, который предписывает перейти к оператору под номером 70. Оператор с номером 60 при этом не выполняется.

Оператор условного перехода, записываемый на УАЯ в виде

Если A Θ B, то перейти к M;

на языке БЕЙСИК записывается так:

N0 IF A Θ B THEN N1

Слово IF означает «если», слово THEN — «то»; A и B — некоторые арифметические выражения; Θ — один из знаков отношения $<$, $<=$, $=$, $<>$, $>$, $>=$. Если условие A Θ B выполняется, то естественный порядок выполнения операторов нарушается и вслед за оператором с номером N0 выполняется оператор с номером N1. Если же условие A Θ B не выполняется, то естественный порядок выполнения операторов не нарушается.

Пусть, например, требуется выбрать большее из двух чисел M1 и M2 и полученное значение присвоить переменной Y. Алгоритм решения задачи на языке БЕЙСИК можно записать так:

```
10 IF M1 < M2 THEN 40    40 LET Y = M2
20 LET Y = M1            50 END
30 GO TO 50
```

Здесь, если условие $M1 < M2$ выполняется, то вслед за оператором с меткой 10 сразу же выполняется оператор с меткой 40, согласно которому переменной Y присваивается значение M2. Если условие $M1 < M2$ не выполняется, то вслед за оператором с меткой 10 выполняется оператор с меткой 20, согласно которому переменной Y присваивается значение M1. Оператор с меткой 30 передает управление оператору с меткой 50, что позволяет обойти оператор с меткой 40 в случае невыполнения условия $M1 < M2$. Если оператор с меткой 30 опустить, то, очевидно, переменной Y будет присвоено значение M2 независимо от того, выполняется условие $M1 < M2$ или нет.

Операции присваивания исходных значений переменным величинам на языке БЕЙСИК могут быть описаны разными способами:

1. С использованием оператора присваивания. Пусть, например, требуется выполнить некоторые вычисления при значениях переменных X, Y, Z, равных соответственно —0,25; 73,81; 25,7. Тогда указанные значения переменным X, Y, Z можно присвоить с помощью команд

```
10 LET X = —0.25
20 LET Y = 73.81
30 LET Z = 25.7
```

2. С помощью оператора READ (читать). Этот оператор всегда используется в паре с оператором DATA (данные). Например, выполняя указания

```
10 DATA — 0.25, 73.81, 25.7
20 READ X,Y,Z
```

исполнитель должен присвоить переменным X,Y,Z значения —0,25; 73,81 и 25,7 соответственно.

3. С помощью оператора INPUT (ввести). Если в программе встречается оператор INPUT, то машина печатает номер и текст этого оператора на пишущей машине и приостанавливает автоматическую работу по программе. После того, как на клавиатуре пишущей машины будут набраны все необходимые значения, отделенные одно от другого запятыми, и нажата клавиша ↓ (возврат каретки), автоматическая работа по программе будет продолжена. При этом переменным, указанным в списке элементов ввода после слова INPUT, будут присвоены значения, набранные на клавиатуре пишущей машины. Например, выполняя указание

```
10 INPUT X,Y,Z
```

ЭВМ приостанавливает автоматическую работу, печатая на пишущей машине текст:

```
10 INPUT X,Y,Z
```

Если теперь на клавиатуре пишущей машины набрать символы

```
—0.25, 73.81, 25.7,
```

то переменным X,Y,Z будут присвоены значения —0,25; 73,81 и 25,7 соответственно. Затем автоматическая работа по программе будет продолжена, начиная с оператора, следующего за оператором с меткой 10.

Все эти три способа присваивания исходных значений переменным X,Y,Z на УАЯ описываются оператором

```
Ввести (X, Y, Z);
```

Операции вывода значений переменных и пояснительных текстов на языке БЕЙСИК описываются оператором PRINT (печатать). После слова PRINT следует список элементов вывода, отделенных один от другого точкой с запятой либо запятой. В список элементов вывода могут входить арифметические выражения либо взятые в кавычки произвольные наборы символов, включая русские буквы. Числовые константы и взятые в кавычки наборы символов по команде PRINT печатаются на пишущей машине без изменений. Если же в списке элементов вывода будет идентификатор переменной величины или арифметическое выражение, то печатается предварительно найденное значение указанной переменной величины или арифметического выражения.

Если элементы списка вывода разделены между собой точкой с запятой, то после печати каждого числового значения делается один пробел, а если запятой, то для печати каждого числового значения на бумаге отводится 15 позиций. По каждой команде PRINT печатание начинается с новой строки. Например, после выполнения команды

```
15 PRINT 'ЗНАЧЕНИЕ КОРНЯ'
```

на пишущей машине будет напечатан текст:

```
ЗНАЧЕНИЕ КОРНЯ
```

а после выполнения команды

```
15 PRINT 0,1,2,3,4,5,6,7,8,9
```

указанные числа будут напечатаны так:

```
0  1  2  3
4  5  6  7
8  9
```

После выполнения команды

```
15 PRINT 0; 1; 3; 4; 5; 6; 7; 8; 9
```

те же числа будут напечатаны иначе:

```
0 1 2 3 4 5 6 7 8 9
```

В результате выполнения команд

```
5 PRINT 'ВЫЧИСЛЕНИЕ СРЕДНЕГО
        ГЕОМЕТРИЧЕСКОГО'
15 DATA 0.8,5.0
20 READ X,Y
30 LET M = (X * Y) ↑ 0.5
```

35 PRINT X,Y, 'СРЕДНЕЕ = ';M

будет напечатано:

ВЫЧИСЛЕНИЕ СРЕДНЕГО ГЕОМЕТРИЧЕСКОГО
0.8 5 СРЕДНЕЕ = 2

Оператор PRINT соответствует оператору УАЯ
Вывести (...);
и почти полностью идентичен ему.

Упражнения. 1. Записать на языке БЕЙСИК операторы присваивания:

а) переменной Z значения арифметического выражения $aX^2 + bX + c$;

б) переменной Y значения $\sin^2\alpha + \cos^2\beta + \cos^2\gamma$;

в) переменной α значения $\operatorname{arctg}(Y/X)$;

г) переменной Y значения $|\sin X|$.

2. Задана точка M с координатами X, Y. На языке БЕЙСИК записать оператор условного перехода на оператор с меткой 950 в случае, если точка M не находится внутри круга единичного радиуса.

3. Записать на языке БЕЙСИК оператор ввода, присваивающий переменным a, b, c, d значения — 37,84; —15,29; 1,38 и 17,59 соответственно.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как записывается на языке БЕЙСИК оператор Конец;? 2. Как записывается на языке БЕЙСИК оператор присваивания? 3. Как записывается на языке БЕЙСИК оператор безусловного перехода? 4. Как записывается на языке БЕЙСИК оператор условного перехода? 5. Нарушается ли естественный порядок выполнения операторов в соответствии с оператором условного перехода в случае, если проверяемое условие не выполняется? 6. Как могут быть записаны на языке БЕЙСИК операции присваивания исходных значений переменным величинам? 7. Как записывается на языке БЕЙСИК оператор вывода? 8. Какие элементы могут входить в список элементов ввода? 9. Какие элементы могут входить в список элементов вывода?

§ 23. ОСНОВНЫЕ ТИПЫ ПРОГРАММ НА ЯЗЫКЕ БЕЙСИК. ПОДПРОГРАММЫ

Рассмотрим примеры описания на языке БЕЙСИК линейных, разветвляющихся и циклических вычислительных процессов.

1. Пусть требуется описать алгоритм вычисления значения переменной Y, определяемой выражением

$$Y = \cos^2(c + a/d) + \ln b + 3,$$

если переменные a, b, c, d принимают значения 5; 52; 2,14

и 3,57 соответственно. Результат вычислений должен быть выведен на печать.

Схема алгоритма решения задачи показана на рис. 76. Описание алгоритма на языке БЕЙСИК имеет вид:

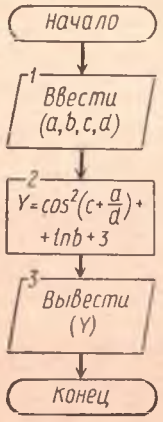
```
10 DATA 5, 52, 2.14, 3.57
20 READ A, B, C, D
30 LET Y = (COS(C + A/D)) ↑ 2 + LOG(B) + 3
40 PRINT Y
50 END
```

Символу *1* схемы алгоритма в его описании на языке БЕЙСИК не ставится в соответствие никакой оператор. Символ *2* реализуется двумя операторами: оператором чтения 20 и оператором определения данных 10, символ *3* — оператором присваивания 30, символ *4* — оператором 40 вывода текущего значения переменной Y на печать, символ *5* — оператором конца программы: 50 END. Так как все операторы алгоритма выполняются строго в том порядке, как они записаны, и только по одному разу, то рассмотренный алгоритм (программа) линейный.

2. Пусть требуется описать алгоритм решения системы двух линейных алгебраических уравнений с двумя неизвестными:

$$\begin{cases} a_1X + b_1Y = c_1 \\ a_2X + b_2Y = c_2 \end{cases}$$

для значений переменных величин a_1, b_1, c_1, a_2, b_2, c_2, равных 0,57; 3,29; —5,81; 2,77; —3,98 и —7,89 соответственно. Учитывая, что алгоритм должен быть массовым, необходимо предусмотреть возможность решения задачи при любых исходных данных, а не только указанных.



Домножив первое уравнение на b_2, а второе на b_1 и вычтя из первого второе, получим

$$(b_2a_1 - b_1a_2)\,X = b_2c_1 - b_1c_2,$$

откуда $X = (c_1b_2 - c_2b_1) / (a_1b_2 - a_2b_1)$.

Величину $c_1b_2 - c_2b_1$ обозначим через ΔX.

Рис. 76. Схема алгоритма вычисления значения переменной $Y = \cos^3(c + a/d) + \ln b + 3$

Аналогично, умножив первое уравнение на a_2, а второе на a_1 и вычтя из второго первое, найдем

$$Y = (a_1c_2 - a_2c_1)/(a_1b_2 - a_2b_1).$$

Величину $a_1c_2 - a_2c_1$ обозначим через ΔY. Величина $\Delta = a_1b_2 - a_2b_1$ называется *определителем системы уравнений*.

Если $\Delta = a_1b_2 - a_2b_1 = 0$, т. е. $a_1/b_1 = a_2/b_2$, то прямые, соответствующие первому и второму уравнениям, параллельны, поскольку равны угловые коэффициенты в уравнениях этих прямых:

$$Y = -\frac{a_1}{b_1}X + \frac{c_1}{b_1};$$

$$Y = -\frac{a_2}{b_2}X + \frac{c_2}{b_2}.$$

При этом система имеет бесчисленное множество решений, если кроме условия $a_1/b_1 = a_2/b_2$ выполняется также условие $c_1/b_1 = c_2/b_2$, или, что то же, $c_1b_2 - c_2b_1 = 0$, так как в этом случае прямые, изображаемые первым и вторым уравнениями, совпадают. Если же $a_1/b_1 = a_2/b_2$, но $c_1/b_1 \neq c_2/b_2$, то у параллельных прямых общих точек нет и система уравнений не имеет решений.

Если $\Delta = a_1b_2 - a_2b_1 \neq 0$, то прямые не параллельны (имеют единственную точку пересечения) и система уравнений имеет единственное решение.

Схема одного из возможных вариантов решения рассматриваемой задачи показана на рис. 77. Поскольку в алфавит языка БЕЙСИК греческие буквы не входят, Δ переобозначим через D, ΔX — через D1, ΔY — через D2. Тогда описание алгоритма на языке БЕЙСИК примет вид:

```
10 DATA 0.57, 3.29, —5.81, 2.77, —3.98, —7.89
20 READ A1, B1, C1, A2, B2, C2
30 LET D = A1 * B2 — A2 * B1
40 IF D = 0 THEN 110
50 LET D1 = C1 * B2 — C2 * B1
60 LET D2 = A1 * C2 — A2 * C1
70 LET X = D1/D
80 LET Y = D2/D
90 PRINT 'X = '; X; ',Y = '; Y
100 GO TO 200
110 LET D1 = C1 * B2 — C2 * B1
```

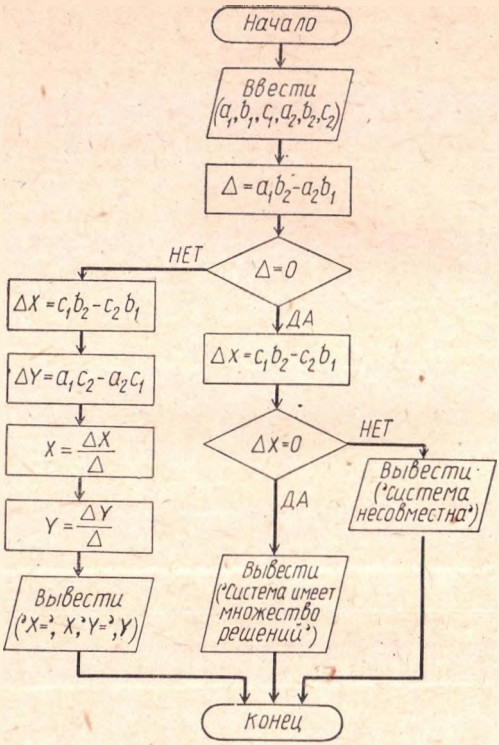


Рис. 77. Схема алгоритма решения системы двух линейных алгебраических уравнений с двумя неизвестными

```
120 IF D1 = 0 THEN 150
130 PRINT 'СИСТЕМА НЕСОВМЕСТНА'
140 GO TO 200
150 PRINT 'СИСТЕМА ИМЕЕТ
          МНОЖЕСТВО РЕШЕНИЙ'
200 END
```

В описанном алгоритме вычисления разветвляются по трем возможным направлениям, но в зависимости от того, выполняются или нет условия, проверяемые операторами 40 и 120, при конкретных исходных данных вычисления будут идти только по одному из этих путей.

3. Рассмотрим задачу приближенного определения площади криволинейной трапеции, ограниченной линиями $X = a$, $X = b$, $Y = 0$ и $Y = f(X)$ (рис. 78). Приближенно

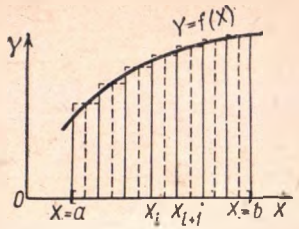


Рис. 78. Криволинейная трапеция, ограниченная линиями $Y = f(X)$, 0 и $X = a$, b

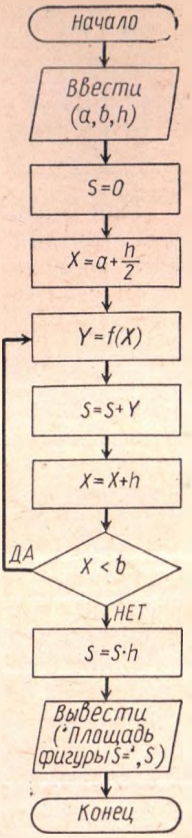


Рис. 79. Схема алгоритма приближенного определения площади криволинейной трапеции по формуле средних прямоугольников

ее площадь можно определить следующим образом. Разобьем промежуток $[a, b]$ на n частей длиной $h = (b - a)/n$ точками деления $X_0 = a, X_1, \ldots, X_n = b$. Площадь фигуры, ограниченной линиями $X = X_i$, $X = X_{i+1}$ и $Y = 0$, $Y = f(X)$, приблизительно можно считать равной площади прямоугольника с длиной основания $h = X_{i+1} - X_i$ и высотой $f(X_i + h/2)$. Складывая площади отдельных прямоугольников, находим площадь всей фигуры:

$$S \approx f\left(X_0 + \frac{h}{2}\right) \cdot h + f\left(X_1 + \frac{h}{2}\right) \times$$

$$\times h + f\left(X_2 + \frac{h}{2}\right) \cdot h + \cdots$$

$$\cdots + f\left(X_{n-1} + \frac{h}{2}\right) \cdot h = h\left(\sum_{i=0}^{n-1} f\left(X_i + \frac{h}{2}\right)\right).$$

Эту формулу называют *формулой средних прямоугольников* приближенного определения площади криволинейной трапеции.

Схема алгоритма решения рассматриваемой задачи показана на рис. 79.

Прежде чем записать алгоритм приближенного определения площади криволинейной трапеции по формуле средних прямоугольников, необходимо задать конкретную функцию $f(X)$. Пусть $f(X) = \sqrt{X}$; тогда на языке БЕЙСИК программа решения задачи примет вид, представленный ниже.

```
10 INPUT A, B, H
20 LET S = 0
30 LET X = A + H/2
40 LET Y = SQR(X)
50 LET S = S + Y
60 LET X = X + H
70 IF X < B THEN 40
80 LET S = S * H
90 PRINT' ПЛОЩАДЬ
   ФИГУРЫ S = '; S
9999 END
```

При реализации этой программы нужно задать конкретные значения переменных величин A, B, H в ответ на запрос по оператору 10 INPUT A, B, H.

Рассмотренный алгоритм является циклическим с пошаговым изменением аргумента X. Операторы цикла с метками 30, 40, 50, 60 и 70 выполняются до тех пор, пока условие, проверяемое по оператору 70, выполняется. Оператор 70 управляет количеством повторений операторов цикла.

4. При составлении программы зачастую удобно ее отдельные части оформить в виде подпрограммы. Пусть, например, рассмотренную задачу о вычислении площади криволинейной трапеции надо решить для нескольких функций $Y = f(X)$. Чтобы не менять каждый раз в программе операторы, вычисляющие значение $Y = f(X)$, вычисление этого значения можно оформить как подпрограмму.

Написание подпрограммы на языке БЕЙСИК ничем не отличается от написания программы. Единственное отличие то, что последним оператором подпрограммы должен быть оператор N0 RETURN (слово RETURN означает «возврат»). Обращение к подпрограмме осуществляется с помощью оператора

N0 GOSUB N1,

где N1 — номер первого оператора подпрограммы. По команде N0 GOSUB N1 управление вычислениями передается оператору с номером N1. Выполнение вычислений по подпрограмме продолжается до тех пор, пока не встретится оператор RETURN, который передает управление оператору, следующему за оператором N0 GOSUB N1.

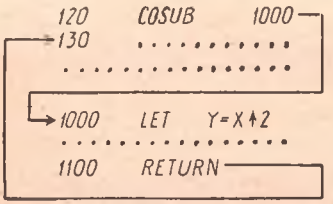


Рис. 80. Обращение к подпрограмме на языке БЕЙСИК

Таким образом, команды подпрограммы как бы вставляются между оператором N0 GOSUB N1 и следующим за ним оператором (рис. 80). Номер оператора END конца

программы должен быть больше, чем номера всех операторов подпрограмм.

С учетом изложенного рассмотренную программу решения задачи можно записать так:

```
10 INPUT A, B, H          80 LET S = S * H
20 LET S = 0              90 PRINT' ПЛОЩАДЬ
30 LET X = A + H/2            ФИГУРЫ S = '; S
40 GOSUB 110              100 GO TO 9999
50 LET S = S + Y          110 LET Y = SQR(X)
60 LET X = X + H          120 RETURN
70 IF X < B THEN 40       9999 END
```

В данной программе вычисление значения функции выделено в подпрограмму только для иллюстрации.

Использование подпрограмм особенно удобно и эффективно, если часть алгоритма, записанная в подпрограмме, довольно сложна и содержит большое число операторов.

Упражнения. 1. Записать на языке БЕЙСИК программы для вычисления:

а) начальной скорости движения электрона по формуле

$$v = \sqrt{\frac{2leU}{md \sin a}},$$

если $l = 25$, $e = 1{,}6 \cdot 10^{-19}$, $U = 2{,}1 \cdot 10^3$, $m = 9{,}1 \cdot 10^{-31}$, $d = 56$, $a = 2$;

б) площади треугольника ABC по формуле

$$S = \frac{a^2 \sin B \sin C}{2 \sin (B + C)},$$

если $a = 48{,}5$; $B = 1$ рад, $C = 2{,}3$ рад;

в) координат центра тяжести трех материальных точек с массами m_1, m_2, m_3 и координатами (X_1, Y_1), (X_2, Y_2), (X_3, Y_3) по формулам

$$X_c = (X_1 m_1 + X_2 m_2 + X_3 m_3) / (m_1 + m_2 + m_3);$$
$$Y_c = (Y_1 m_1 + Y_2 m_2 + Y_3 m_3) / (m_1 + m_2 + m_3),$$

если $X_1 = 7{,}5$, $X_2 = 8{,}9$, $X_3 = 7{,}7$, $Y_1 = 3{,}4$, $Y_2 = 9{,}5$, $Y_3 = -5{,}6$, $m_1 = 4{,}5$, $m_2 = 8{,}2$, $m_3 = 9{,}4$;

г) длины медианы треугольника со сторонами длиной a, b, c по формуле

$$m_a = 0{,}5 \sqrt{2b^2 + 2c^2 - a^2},$$

если $a = 3{,}28$, $b = 5{,}64$, $c = 4{,}57$;

д) количества теплоты Q по формуле

$$Q = cm\,(t_2 - t_1),$$

если $c = 4{,}21 \cdot 10^3$ Дж/(кг · °С), $m = 2{,}3$ кг, $t_1 = 20$ °С, $t_2 = 100$ °С;

е) значения функции (с выводом на печать)

$$f(X) = \begin{cases} \sin\dfrac{X}{3} & \text{при } X < 1; \\ (X+3)^2 & \text{» } 1 \leqslant X \leqslant 3; \\ \ln\dfrac{X}{2} & \text{» } X > 3, \end{cases}$$

если значение X будет задано по запросу оператора INPUT;

ж) корней квадратного уравнения

$$aX^2 + bX + c = 0,$$

если значения коэффициентов a, b, c будут заданы по запросу оператора INPUT. Если действительных корней уравнение не имеет, то на печать должен быть выведен текст «Действительных корней не существует».

2. Составить программу для определения наибольшего среди трех чисел a, b, c. Результат вывести на печать. Числа a, b, c будут заданы по запросу оператора INPUT.

3. Записать на языке БЕЙСИК программу вычисления $n!$ Число n будет задано по запросу оператора INPUT.

4. Составить схему алгоритма вычисления произведения

$$\left(\frac{2}{1}\cdot\frac{2}{3}\right)\left(\frac{4}{3}\cdot\frac{4}{5}\right)\left(\frac{6}{5}\cdot\frac{6}{7}\right)\left(\frac{8}{7}\cdot\frac{8}{9}\right)\left(\frac{10}{9}\cdot\frac{10}{11}\right)$$

описать его на языке БЕЙСИК.

5. Составить схему алгоритма вычисления суммы

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15}$$

и описать его на языке БЕЙСИК.

6. Составить схемы алгоритмов вычислений и описать их на языке БЕЙСИК:

а) $S = 1^2 + 2^2 + 3^2 + \cdots + 100^2$;

б) $S = \sum\limits_{k=1}^{20} \dfrac{k}{k+1} = \dfrac{1}{2} + \dfrac{2}{3} + \dfrac{3}{4} + \cdots + \dfrac{20}{21}$;

в) $S = \sum\limits_{k=1}^{30} \dfrac{1}{k^3} = \dfrac{1}{3^3} + \dfrac{1}{2^3} + \dfrac{1}{3^3} + \cdots + \dfrac{1}{30^3}$;

г) $S = \sum_{k=1}^{10} \frac{1}{k!} = \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \frac{1}{1 \cdot 2 \cdot 3 \cdot 4} +$

$+ \frac{1}{1 \cdot 2 \cdot 3 \cdot \ldots \cdot 10}$;

д) $S = \sum_{k=0}^{n} \frac{1}{(2k+1)^2} = \frac{1}{1^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \cdots$

$\cdots + \frac{1}{(2n+1)^2}$.

§ 24. ПОДГОТОВКА ПРОГРАММ К РЕАЛИЗАЦИИ НА ЭВМ

Программа вводится в память ЭВМ с терминала (пишущей машины, телетайпа) пооператорно. При этом сначала набирается номер оператора, затем текст, описывающий оператор, после чего набирается символ ↓ (возврат каретки к новой строке). Если текст оператора набран неверно, достаточно набрать его заново вместе с номером. Например, в результате набора команд

20 DATA —1, 2.1↓
20 LET X = —1↓

в память ЭВМ под номером 20 будет занесен оператор LET X = —1 вместо ранее занесенного оператора DATA —1, 2.1.

Если в память ЭВМ вводится совершенно новая программа, то перед ее вводом следует набрать команду (без номера)

NEW↓

Слово NEW здесь означает «новый». По этой команде вся ранее загруженная информация (операторы, данные) аннулируется (обнуляется), т. е. происходит очистка памяти ЭВМ.

После того как программа введена в память ЭВМ, для начала ее автоматической трансляции на машинный язык с языка БЕЙСИК дается команда

RUN↓

(RUN означает «работа», «пуск», «прогон»).

Если ошибок в программе нет, то сразу после трансляции начинается счет по программе, результаты которого

выводятся через терминал (пишущую машину). Если же в программе содержатся ошибки, то, обнаружив по программе-транслятору ошибочный оператор, ЭВМ печатает на терминале неправильный оператор и приостанавливает работу. Для продолжения работы ЭВМ достаточно набрать заново оператор, в котором обнаружена ошибка, и снова дать команду RUN↓

Пусть, например, в ЭВМ введена программа:

```
NEW
 5 LET X = 0.5
10 LET Y = SIH(X)
20 PRINT X, Y
30 END
```

Так как оператор с меткой 10 набран неверно, то через некоторое время ЭВМ выведет на пишущую машину текст:

```
10 LET Y = SIN (X)
```

и приостановит работу. После этого достаточно набрать оператор

```
10 LET Y = SIN (X)
RUN↓
```

Теперь ошибок в наборе операторов не содержится и после выполнения трансляции, а затем вычислений на печать будут выведены результаты:

```
 0.5            0.47942
```

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как вводится в память ЭВМ программа, составленная на языке БЕЙСИК? 2. В каком порядке вводятся операторы в память ЭВМ? 3. Как исправить неправильно набранный текст оператора? 4. Какую команду следует набрать перед вводом в память ЭВМ новой программы? 5. Что происходит в памяти ЭВМ после набора команды NEW↓? 6. Какую команду необходимо дать для начала автоматической трансляции программы с языка БЕЙСИК на машинный язык? 7. В каком случае ЭВМ переходит на автоматическое выполнение вычислений по программе после ее трансляции? 8. С помощью каких устройств выводятся из памяти ЭВМ результаты вычислений? 9. Как ЭВМ сообщает о том, что при трансляции в программе обнаружен неправильно записанный оператор? 10. Как продолжить работу, если ЭВМ сообщила, что в одном из операторов имеется ошибка?

§ 25. ПОНЯТИЕ О ПРОГРАММНОМ ОБЕСПЕЧЕНИИ ЭВМ. ОСНОВНЫЕ ЭТАПЫ РЕШЕНИЯ ЗАДАЧ НА ЭВМ

Устройства и схемы, входящие в состав ЭВМ, называются *аппаратурой* ЭВМ. Набор программ, составленных для ЭВМ и предназначенных для решений некоторого круга задач с помощью данной ЭВМ, составляет ее *программное обеспечение*.

Вычислительная машина вместе с программным обеспечением образует *вычислительную систему* (рис. 81). Состав программного обеспечения ЭВМ зависит от круга решаемых на ней задач. Например, если микро-ЭВМ управляет станком, обрабатывающим только три вида деталей, то ее программное обеспечение может состоять из трех различных программ, каждая из которых предназначена для управления станком при обработке определенного вида деталей. Для переналадки станка на обработку новой детали необходимо обеспечить работу управляющей ЭВМ по соответствующей программе, составленной заранее и хранящейся в специальном ЗУ.

Если ЭВМ используется как обучающее и контролирующее устройство, то в состав ее программного обеспечения должны входить программы, обеспечивающие автоматизацию обучения и контроля, диалог обучаемого с ЭВМ, причем при обучении иностранному языку необходим один набор

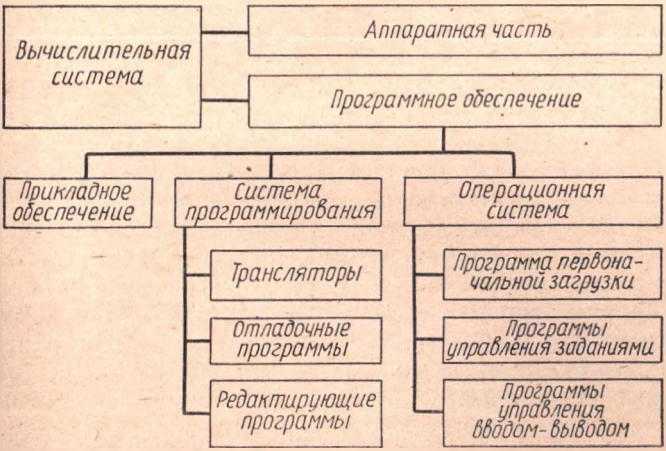


Рис. 81. Структурная схема вычислительной системы

обучающих программ, при обучении физике — другой, при обучении кулинарному искусству — третий набор программ. В состав программного обеспечения ПМК «Электроника Б3-34» входят программы выполнения арифметических операций, вычисления тригонометрических и обратных тригонометрических функций, логарифмических функций и др. Эти программы хранятся в ПЗУ МК.

Программное обеспечение современных универсальных ЭВМ содержит программы, предназначенные для решения очень широкого круга задач. Сюда относятся: программы, обеспечивающие перевод программ с машинно-независимых алгоритмических языков, например БЕЙСИК, на язык данной ЭВМ; программы, обеспечивающие автоматический ввод и вывод информации через разнообразные устройства ввода — вывода (дисплеи, пишущие машины, магнитофоны); программы, контролирующие работу устройств ЭВМ, а также управляющие работой аппаратуры и самого программного обеспечения. В программном обеспечении можно выделить *специальное (прикладное)* и *системное* обеспечение.

Специальное (прикладное) программное обеспечение состоит из совокупности программ решения конкретных задач. Подбор таких программ зависит от характера решаемых задач. Так, для решения коммерческих задач необходимы программы обработки главным образом буквенно-цифровой информации, а в научных исследованиях — программы обработки числовых данных. Знакомство с прикладным обеспечением нужно для того, чтобы знать возможности вычислительной системы.

Системное обеспечение состоит из системы программирования и операционных систем. Этот набор программ обеспечивает работу вычислительной системы, а также разработку, отладку и выполнение программ. Система программирования представляет собой набор программ, обеспечивающих автоматизацию программирования, и включает в себя трансляторы с различных языков программирования, отладочные и другие программы, позволяющие автоматизировать конструирование и отладку программ. Операционная система управляет всеми устройствами ЭВМ и самим программным обеспечением. Основные задачи операционной системы — управление выполнением программ, обеспечение взаимодействия человека с машиной, оптимальное использование оборудования. Операционная система — это набор управляющих программ, которые обеспечивают работу вычислительной системы, а именно:

1) управляют вводом — выводом, обменом данными с внешней памятью и обеспечивают доступ к ним;

2) организуют непрерывную работу машины от программы к программе без вмешательства человека;

3) планируют использование возможностей вычислительной системы и осуществляют решение задач.

Операционная система обеспечивает режим диалога человека с ЭВМ. В этом режиме осуществляется поочередный обмен информацией между человеком и машиной. При этом машина может вести диалог одновременно со многими абонентами, успевая решать поставленные ими задачи за счет своего высокого быстродействия. Каждому абоненту отводятся небольшие промежутки времени, достаточные для решения небольших задач. Такой режим работы ЭВМ называется *режимом разделения времени.*

Для того чтобы общаться с ЭВМ, решать задачи с ее помощью, недостаточно знать структурную схему ЭВМ, характеристику и назначение ее основных устройств, алгоритмические языки. Необходимо уметь осуществлять основные этапы решения задачи с помощью ЭВМ:

1) постановка задачи;

2) математическое описание задачи;

3) выбор численного метода решения задачи;

4) запись алгоритма для решения задачи;

5) составление программы на каком-то алгоритмическом языке;

6) перенесение программы, записанной на специальных бланках, на перфоленты или перфокарты;

7) отладка программы;

8) решение задачи на машине;

9) анализ полученных результатов.

На первом этапе выбирается общий подход к решению, устанавливается, каким целям должно служить решение задачи и при каких условиях оно будет существовать. Здесь производится разбиение задачи на более мелкие, определяется последовательность их решения и выполняется ряд других действий, т. е. осуществляется общая постановка задачи.

После этого этапа необходимо дать математическое описание задачи. Когда составлена полная система уравнений, определяющих исследуемое явление, задача становится уже чисто математической — задача математически поставлена. Теперь нужно разработать метод решения.

За математической постановкой и разработкой метода решения следует реализация выбранного метода на ЭВМ.

Этот этап содержит в себе исследование математической модели явления и программирование, которое состоит из двух частей: составления схемы программы (графического изображения последовательности действий) и написания программы на языке, который ЭВМ понимает непосредственно или с помощью транслятора.

Контроль за выполнением программы машиной включает в себя отладку программы и получение результатов.

Последний этап решения задачи — анализ полученных результатов.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Что называется программным обеспечением ЭВМ? 2. От чего зависит состав программного обеспечения ЭВМ? 3. Что называется системой программирования? 4. Для чего предназначена операционная система? 5. Какие основные этапы решения задач с использованием ЭВМ?

Глава 4

МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ

§ 26. ПОНЯТИЕ О МИКРОПРОЦЕССОРНОЙ СИСТЕМЕ

Микропроцессор — это устройство, которое по заданной программе может обрабатывать информацию, а также управлять другими устройствами. В настоящее время существует около 100 типов различных микропроцессоров, которые применяются главным образом как строительный элемент для создания ЭВМ различных типов и назначений. Сам по себе микропроцессор не может решить ту или иную конкретную задачу. Чтобы решить задачу, его нужно запрограммировать и соединить с другими устройствами. Микропроцессор отличается от микро-ЭВМ тем, что в его состав не входят оперативная память и устройства ввода — вывода данных. В микропроцессоре предусмотрены внешние входы, к которым следует подсоединить ЗУ и устройства ввода — вывода, чтобы получить микро-ЭВМ.

Некоторая совокупность соединенных друг с другом устройств, включающая в себя микропроцессор, память и устройства ввода — вывода данных, нацеленная на выполнение некоторой четко определенной функции, называется *микрокомпьютером* или *микропроцессорной системой*.

Микропроцессор, ЗУ и устройства, обеспечивающие ввод — вывод информации, представляют собой большие интегральные микросхемы (БИС). Поэтому микропроцессорная система является совокупностью БИС.

Состав микропроцессорной системы зависит от ее назначения, т. е. от тех функций, для реализации которых она предназначена. Примером микропроцессорной системы может быть микро-ЭВМ — вычислительная или управляющая система, выполненная на основе микропроцессора, в состав которой могут также входить: программная память (обычно — ПЗУ), память данных (обычно — ОЗУ), устройства ввода — вывода данных, генератор тактовых сигналов, а также другие устройства, выполненные с использованием БИС или элементов с меньшей степенью интеграции.

К микропроцессорным системам относятся и МК. В состав ПМК включены: микропроцессор, ПЗУ, ППЗУ, ОЗУ, устройства ввода — вывода данных. В непрограммируемых МК ППЗУ отсутствует.

Микропроцессорные системы предназначены для создания автоматизированных систем, использования в составе различного рода оборудования, агрегатов, приборов, станков, средств передачи данных, в качестве органов автоматического управления. Каждый день открываются новые области применений микропроцессоров. В настоящее время они используются в контрольно-измерительных приборах, в кассовых аппаратах магазинов. Они оказываются жизненно важным элементом в управлении химическими процессами, периферийными устройствами больших компьютеров, уличным движением, сложными бытовыми электроприборами и автомобилями.

При создании автоматизированных систем различного назначения широко применяются два типа цифровой техники: 1) устройства с жесткой структурой; 2) ЭВМ. Первые обычно содержат большое число интегральных микросхем малой и средней степеней интеграции. Эти схемы устанавливаются на платах, а их выводы соединяются в соответствии с тем, какие функции должно выполнять устройство. Любое изменение выполняемых устройством функций требует изменения схемы, т. е. перепайки соединений или замены интегральных микросхем. Поэтому внесение изменений в такое управляющее устройство связано с большими трудностями.

Системы на основе микро-ЭВМ могут легко перенастраиваться на выполнение новой функции. Для перенастройки достаточно составить и занести в память управляющей микро-ЭВМ новую программу. Таким образом, если при разработке автоматизированных систем на основе устройств с жесткой структурой применяются только аппаратные средства, то при построении систем на основе микропроцессорной техники можно использовать как аппаратные, так и программные средства. Настройка системы на выполнение определенной функции осуществляется программой, управляющей микропроцессором.

Замена устройств с жесткой структурой микропроцессорной техникой дает целый ряд преимуществ:

1. Микропроцессорные системы обладают значительно большей гибкостью. Функции, выполняемые такими устройствами, определяются программами, хранящимися в ПЗУ, ППЗУ или ОЗУ. В результате становится возмож-

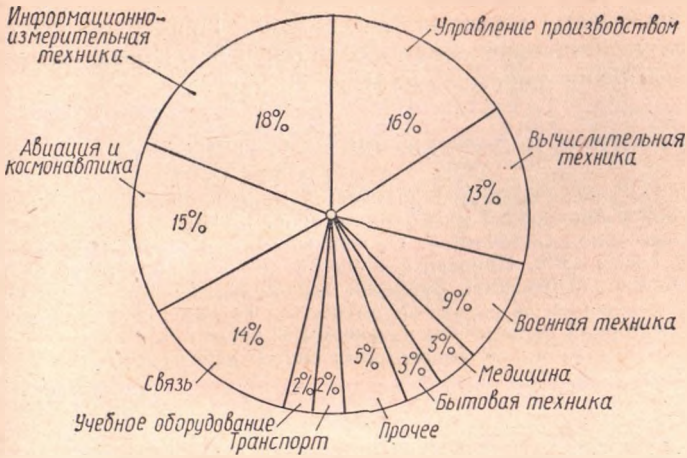


Рис. 82. Примерное распределение микропроцессорных систем в различных областях применения

ным изменять характеристики системы только за счет замены программы в памяти без внесения каких-либо изменений в монтаж или печатные платы. Это позволяет легко вносить изменения в уже изготовленную систему в процессе ее отладки, легко перенастраивать систему для изменения ее характеристик, создавать множество систем различного назначения на основе одного и того же микропроцессорного комплекта БИС.

2. Системы на основе микропроцессоров гораздо дешевле, чем системы с жесткой структурой. БИС микропроцессора обычно заменяет 75—200 корпусов интегральных микросхем малой и средней степеней интеграции. В микро-ЭВМ меньше соединений, печатных плат. Настройка и переделка микро-ЭВМ значительно проще, стоимость проектирования микро-ЭВМ ниже.

3. Затраты времени на разработку систем на основе микропроцессоров существенно ниже.

4. Надежность микро-ЭВМ намного выше надежности устройств с жесткой структурой благодаря резкому сокращению числа межсоединений.

Микропроцессорная система может заменить в целом ряде областей большую ЭВМ. Микропроцессорные системы все шире применяются в периферийном оборудовании больших ЭВМ, системах управления технологическими процессами, электронных кассовых аппаратах, цифровых измерительных приборах, радиоэлектронике и связи, меди-

цинском оборудовании, бытовой технике, обучающих устройствах, торговых терминалах. Примерное распределение микропроцессорных систем в различных областях применения показано на рис. 82.

ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОКОНТРОЛЯ

1. Какое устройство называется микропроцессором? 2. Какое основное применение микропроцессоров? 3. Что называется микропроцессорной системой? 4. От чего зависит состав микропроцессорной системы? 5. Приведите примеры микропроцессорных систем. 6. Можно ли изменить функции, выполняемые жесткой системой, без изменения ее схемы? 7. Можно ли изменить функции микропроцессорной системы без изменения ее схемы? 8. Какие основные преимущества микропроцессорных систем перед системами с жесткой структурой?

§ 27. ОБЛАСТИ ПРИМЕНЕНИЯ МИКРОПРОЦЕССОРНЫХ СИСТЕМ

Микропроцессоры и микропроцессорные системы находят широкое применение в различных автоматических и автоматизированных системах.

Автоматической системой называется совокупность устройств, выполняющих некоторую работу без участия человека (рис. 83). Такие системы могут использоваться для изготовления промышленной продукции, управления движущимися объектами (кораблями, самолетами), автоматизации различных услуг населению и выполнения ряда других работ. Автоматические системы могут управлять различными удаленными объектами и приспосабливаться к реальной обстановке, т. е. самонастраиваться в зависимости от условий окружающей среды. Они позволяют повысить производительность человеческой деятельности не только в сфере материального производства, но и в разнообразных непроизводственных областях.

Любая автоматическая система состоит из объекта управления и управляющей системы. Объектом управления могут быть станок, летательный аппарат, совокупность машин, цех. Управляющая система вырабатывает управляющие сигналы по наперед заданному алгоритму. Управляющие сигналы могут быть заданы в виде электрических напряжения и тока, давления жидкости или газа, механических сил. Управляющая система может реагировать на различные виды влияния внешней среды (температура, радиация, давление).

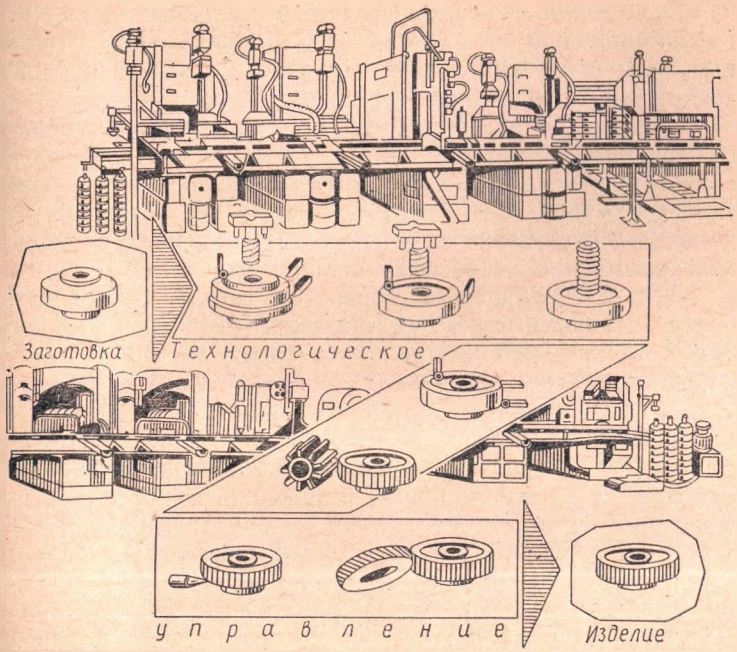


Рис. 83. Автоматическая система

Автоматическая система, в которой объектом управления является сверлильный, токарный, фрезерный или другой станок для обработки металла, пластмассы, дерева, называется *станком с числовым программным управлением*. В таких станках установлены независимые приводы по каждой из координат (вперед — назад, влево — вправо, вверх — вниз). Основная особенность привода станка с числовым программным управлением заключается в его дискретном принципе работы, т. е. любое действие здесь осуществляется как некоторая конечная совокупность достаточно мелких шагов. Важнейшая составная часть станка с числовым программным управлением — устройство управления. В зависимости от типа и назначения станков устройства управления различаются алгоритмами управления.

Подготовка станка к работе заключается в составлении программы и занесении ее в память ЭВМ. Программа составляется на основе рабочего чертежа детали. В программе указываются количество импульсов, которые надо подать на обмотки дискретных двигателей, чтобы переместить обрабатываемую деталь на требуемое расстояние в одном

из направлений либо повернуть ее на заданный угол. Многофункциональные станки позволяют производить различные виды обработки детали (фрезеровка, сверление, расточка) без ее перестановки.

В станках с программным управлением широко применяются микро-ЭВМ. В памяти такой машины хранятся в закодированном виде программы выполнения отдельных типовых циклов. Составлять программу обработки детали, используя программы отдельных типовых циклов, можно непосредственно на рабочем месте, у станка.

Необходимые режимы обработки деталей микро-ЭВМ определяет по сигналам, поступающим от специальных датчиков. Микро-ЭВМ контролирует работу станка, что исключает поломку инструмента и брак детали. При этом сокращается количество рабочих-станочников и вместо фрезеровщика, токаря, шлифовщика у станка работает оператор — обычно это рабочий со средним образованием, знакомый с основами программирования. Так, на станке с числовым программным управлением модели 16К20Т1 можно обрабатывать наружные и внутренние поверхности деталей со ступенчатым, конусным и радиусным профилями, а также нарезать резьбу. Обработка различных деталей осуществляется автоматически по наперед составленным программам. Программное обеспечение избавляет от необходимости обращаться к услугам специалистов вычислительных центров для составления программ. Программа вводится с пульта управления, смонтированного на станке. Отработанная программа хранится в оперативной памяти микро-ЭВМ или переписывается для длительного хранения на внешнее ЗУ (кассетный магнитофон). Труд токаря на таком станке становится более квалифицированным при существенном его облегчении.

Рассмотрим, как работает станок, управляющий прессом, который пробивает отверстие в плоском листе металла. Пресс состоит из стола, перемещающего лист вдоль двух взаимно перпендикулярных осей в горизонтальной плоскости, и пуансона над столом, который перемещается вертикально и пробивает отверстие в листе с помощью находящейся внизу матрицы. Перемещения стола вдоль каждой из горизонтальных осей осуществляются посредством винтовых подач, винты которых вращаются дискретными двигателями. Каждый двигатель управляется схемой, заставляющей его совершить некоторый единичный поворот по часовой стрелке в ответ на каждый импульс на одной входной линии и против часовой стрелки в ответ на каждый

импульс на другой входной линии. Схема управления двигателем обеспечивает полный цикл штамповки отверстия на каждый поступающий на схему импульс. На базе микропроцессора спроектирован контроллер для такого пресса. Контроллер последовательно перемещает лист металла в соответствии с координатами, записанными в памяти, и в каждой позиции пробивает отверстие. Для перемещения стола контроллер подает нужное число импульсов на схемы управления двигателями стола, определяемое разностями координат новой и старой позиций.

Следующий шаг в развитии программного управления станками — это создание автоматических участков и цехов. На таких участках автоматизированы погрузочно-разгрузочные работы, склады инструментов, заготовок и готовых деталей. Эту работу выполняют роботы.

Операторы работают в первую смену, подготавливая участки к работе (комплектуют склады, проверяют оборудование, вводят программы обработки и контроля в управляющие системы). Во вторую и третью смены цеха работают автоматически, без вмешательства человека.

Р о б о т — автоматическое устройство, способное производить по заданной программе механическую работу посредством специальных рабочих органов. Роботы предназначены для выполнения многочисленных производственных операций, в особенности тяжелой однообразной и монотонной работы, как, например, подача заготовок к станкам и прессам, съем обработанных изделий и их упаковка, складирование. Особенно важна роль роботов там, где требуется тяжелый физический труд, в условиях, когда приходится иметь дело с токсическими, взрывчатыми веществами, в условиях повышенной влажности, вибрации, шума, необычных температурных режимов, в недоступных для человека местах (в печах, на других планетах, в морских глубинах).

Роботы незаменимы в условиях агрессивной среды (наличие отравляющих веществ, кислот и др.), а также при высоких уровнях радиации и температуры. Одной из целей применения роботов является повышение производительности труда и качества продукции.

Робот — сложная машина. Он состоит из двух основных частей: механического агрегата и системы управления. Рабочим органом робота считается рука с кистью-схватом, которая может совершать движения по вертикальной и горизонтальной осям и повороты вокруг них. Кроме того, обеспечивается независимое движение схвата. Эти функции

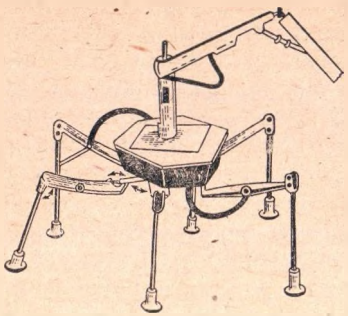


Рис. 84. Шестиногий робот с гидравлическим приводом

робота реализуются устройствами, называемыми *силовыми приводами*. Широко применяются гидравлические (рис. 84), пневматические, электрические и смешанные приводы. Источником энергии является давление жидкости в гидромагистрали или воздуха в пневмосистеме, в электроприводе используется электроэнергия.

Система управления роботом содержит микро-ЭВМ. Для подготовки робота к работе предварительно составляется карта технологического процесса с указанием действий, например: 1) взять заготовку со стеллажа; 2) повернуться к прессу; 3) положить заготовку в ручей штампа; 4) включить штамп; 5) взять готовую деталь; 6) повернуться на 70°; 7) положить деталь в ящик; 8) повернуться еще на 30°. Последовательность действий робота с карты операций заносится в программную память микро-ЭВМ после соответствующего кодирования.

Занесение программы в память сопровождается пробными включениями робота — его о б у ч е н и е м. При пробных включениях уточняются линейные и угловые движения рабочих органов робота и вносятся поправки в программу. Тем самым производится отладка программы, по окончании которой робот запускается в работу.

Чтобы по возможности освободить человека от работы во вредной для него среде, насыщенной испарениями красок и лаков, а также при песко- и дробеструйной обработках деталей, разработан промышленный робот. Он позволяет автоматизировать нанесение на детали термоизоляционных, порошкообразных и других покрытий. Обучение (программирование) робота производит оператор, который вручную ведет его по нужной траектории. Эта траектория запоминается в памяти управляющего устройства, а затем в процессе работы автоматически воспроизводится. В память управляющего устройства можно занести до 65 различных программ.

Разработан также универсальный сборочный робот, который сравним с человеческой рукой и может выполнять работу значительно быстрее ее. Робот легко программируется благодаря применению специального, относительно

простого, *роботного языка*. В память управляющей микро-ЭВМ можно ввести пять разных программ. Переход на работу по новой программе происходит практически мгновенно. Обучение робота производится с пульта, оснащенного телевизионным монитором.

Создан робот, предназначенный для обслуживания станков заготовками массой до 60 кг. Направляясь к станку, он в одном захвате несет заготовку, а вторым (свободным) захватом снимает уже обработанную деталь. Программирование робота проводит оператор с ручного пульта. Последовательность характерных точек траектории движения руки и их координаты вводятся в ЗУ системы управления, после чего робот может работать автоматически по заданной программе.

Разработан робот для дуговой электросварки. Манипулятор горелки может перемещаться вдоль трех координатных осей и, кроме того, вращать горелку и поворачивать ее под разными углами. Система управления, основой которой является микро-ЭВМ, позволяет настраивать робот на необходимый режим работы, планировать и изменять траекторию перемещения горелки относительно изделия, хранить библиотеку программ. Микропроцессоры управляют приводами манипуляторов горелки и изделия в реальном масштабе времени и сварочной аппаратурой в соответствии с заданной циклограммой технологического процесса сварки. Кроме того, в случае выявления аварийной ситуации они блокируют соответствующие устройства.

Чтобы настроить робот на автоматическую работу, вначале нужно его запрограммировать, т. е. научить выполнять в определенной последовательности ряд технологических операций, которые и обеспечат сварку данного вида изделий. Для этого на манипулятор устанавливают образец изделия и, не включая сварочного оборудования, нажатием соответствующих клавиш на пульте управления заставляют манипулятор горелки провести ее по будущей траектории сварки.

Координаты опорных точек этой траектории записываются в память микро-ЭВМ. Одновременно с того же пульта в память микро-ЭВМ вводят значения технологических параметров режима сварки. После такого обучения робот может вести сварку сам в автоматическом режиме.

Робот можно запрограммировать на сварку разных изделий и хранить такие программы во внешней памяти. Набор этих программ образует библиотеку, откуда любая программа может быть вызвана при необходимости.

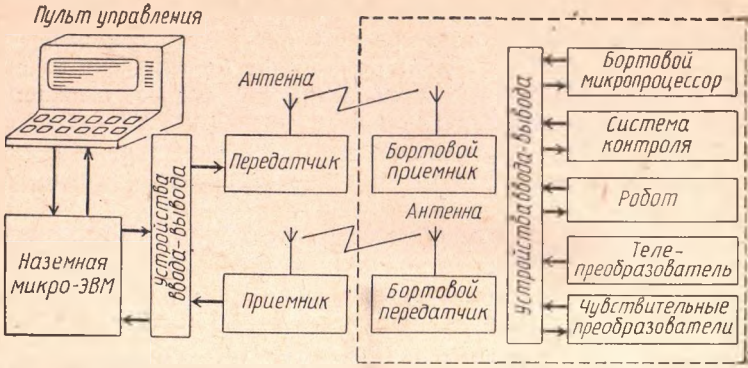


Рис. 85. Структурная схема управления роботом-луноходом

Одним из примеров робота является луноход, управляемый как наземной ЭВМ, так и бортовыми микро-ЭВМ. Структурная схема управления роботом-луноходом показана на рис. 85.

Существует множество других разновидностей роботов, используемых в разных сферах автоматизации человеческой деятельности.

Робот, управляемый микропроцессором, может заменить рабочего при серийной сборке различных узлов на автомобильном и других конвейерах, осуществлять сварку, литье под давлением, транспортировку деталей и многие другие операции. В целом, согласно оценке специалистов, роботы могут выполнять до 7 млн. видов рабочих операций, причем если робототехника пока еще нашла свое место в основном в машиностроительных и обрабатывающих отраслях промышленности, то теперь сфера ее применения значительно расширяется. Например, уже созданы сельскохозяйственные роботы, которым приходится взаимодействовать не с металлом и прочими неодушевленными предметами, а с живыми организмами, живой природой. На рис. 86 показана модель такого робота типа МАР-1.

У него пара рук, корпус вращается в любую сторону вокруг вертикальной оси. Гидравлические «мускулы» каждой руки поднимают до 75 кг груза. Измерительные преобразователи позволяют пальцам регистрировать температуру от 0,4 до 180 °С и влажность от 3 до 90%.

Во время работы автоматическое устройство контролирует состояние окружающей среды — оно «чувствует» влажность, загазованность, может определять и температуру

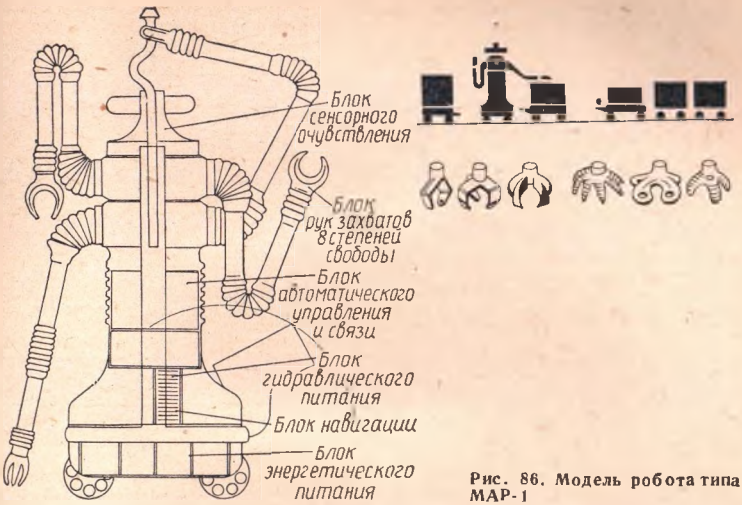


Рис. 86. Модель робота типа МАР-1

животных. Прикоснувшись к свинье ультразвуковым датчиком, робот определяет толщину прослойки сала. Прикатив на свое рабочее место (он движется на колесиках), робот сам подсоединяется к электросети, линии связи, пульту управления или ЭВМ. Обо всем, что «увидел» или «почувствовал», робот может информировать центральный пункт управления.

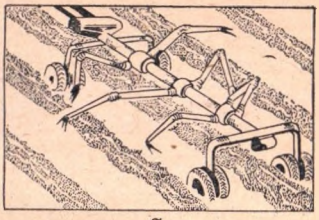
Продолжается работа и над другими моделями. Интерес представляет робот-дояр. У него четыре — по количеству сосков у вымени коровы — руки. Ведутся работы по проектированию мобильных автоматических устройств для открытых полей и садов (рис. 87).

Широкое применение находят микропроцессорные системы в области автоматизации управления производственными процессами (рис. 88). Микропроцессоры используются в составе технических средств практически на всех уровнях управления — от средств сбора и первичной обработки данных до ЭВМ.

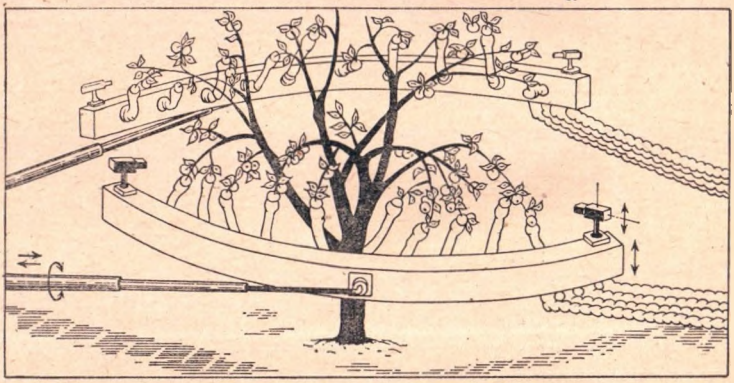
Появилась возможность значительного усовершенствования автоматизированных систем управления на основе программируемых управляющих и контролирующих устройств, которые обеспечивают изменение программы работы системы, не требуя отключения технологического оборудования и перепроектирования управления.

Использование встроенных микропроцессоров в системах телеуправления — важное преимущество таких си-

Рис. 87. Внешний вид робота для полей (*а*) и садов (*б*)



а



б

стем. Системы телемеханики на основе микропроцессорных комплектов БИС осуществляют первичную обработку информации, что уменьшает нагрузку на каналы связи и центральную ЭВМ системы управления, обеспечивает повышение оперативности обмена за счет автоматического сжатия данных и исключения сообщений, не несущих информации.

Примером управляющих микропроцессорных систем может быть и управление светофором, установленным, например, на пересечении главной и второстепенной улиц. В этой системе управления есть микропроцессор, ЗУ и контроллер, к которому подключены сигнализаторы наличия автомобилей, кнопки для пешеходов и переключатели светофора. Сигнал прерывания от сигнализатора или кнопки переключает микропроцессор на обслуживающую программу, которая вызывает переключение светофора на заданные промежутки времени. Он открывает движение на второстепенной улице, если установленные на ней извещатели сообщают о наличии автомобилей или если пешеход нажимает кнопку управления сфетофором на переходе. Движение по главной улице согласно данной про-

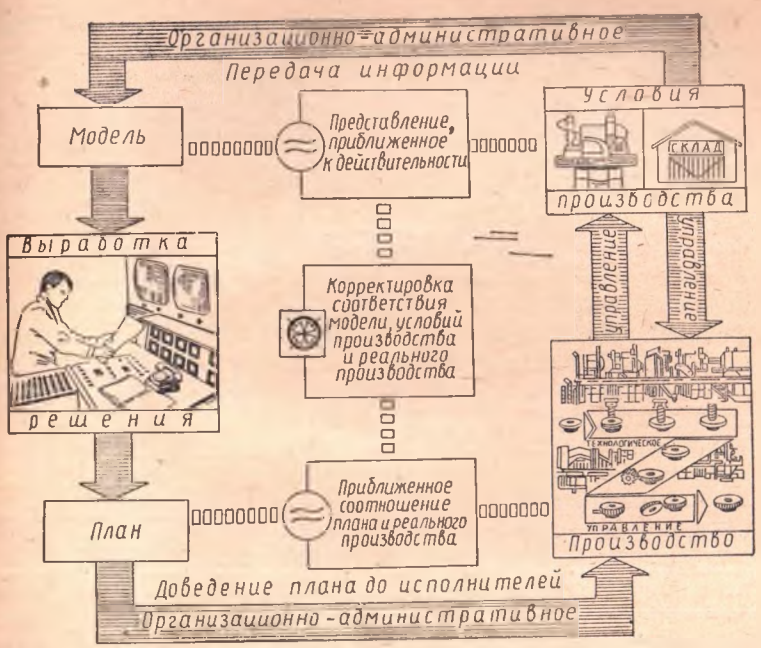


Рис. 88. Автоматизированная система

грамме останавливается только в случае, если с момента последней остановки прошло не менее 2 мин. Движение по второстепенной улице разрешается на 45 с.

Создана микропроцессорная система управления работой пассажирского транспорта «АСУ — рейс». Эта система состоит из датчиков, установленных в контрольных пунктах вдоль маршрута движений, микропроцессорных устройств с дисплеями в кабинах водителей автобусов и управляющей ЭВМ типа СМ-2М в центральной диспетчерской.

В случае отклонения от графика движения в кабине на экране дисплея появляются цифры: это время отклонения. Электронный диспетчер по радиоканалу дальнего действия (20—30 км) подсказывает водителю, что делать (на экране дисплея появляются команды ЭВМ -диспетчера, режим движения машины). При работе системы регулярность движения резко возрастает, отклонения автобусов от графика не превышают одной минуты. Если же один или несколько автобусов сошли с линии, система выравнивает интервалы между ними.

В памяти ЭВМ записаны «десятки ситуаций», позволяющие водителю принять верное решение, вызвать техническую помощь в случае аварии, пожарную машину при пожаре или скорую помощь для заболевшего пассажира. В Москве по этой программе ЭВМ управляет движением десятков автобусов.

Подобная микропроцессорная система управляет движением троллейбусов в г. Воронеже. В качестве ЭВМ-диспетчера служит «Электроника-60».

Сегодня уже широко применяются информационные системы со встроенными микропроцессорами продажи авиационных или железнодорожных билетов. ЭВМ такой системы обслуживает значительное число удаленных пользователей, связанных с ней телефонными или телеграфными линиями. Во внешней памяти ЭВМ хранятся сведения о наличии мест на все рейсы самолетов или поездов на ближайшие две недели. К ЭВМ подключено оборудование касс, состоящее из пишущей машины и печатающих устройств для заполнения бланка билета. Кассир вводит с пишущей машины запрос, который поступает в оперативную память ЭВМ и сравнивается со сведениями о нужном рейсе. Если запрос не удовлетворен, система может выдать на пишущую машину возможные варианты его удовлетворения: сведения о наличии мест на другие рейсы, сведения о другом типе мест в поезде и т. п.

Микропроцессор применяется и в электронных весах, которые можно встретить в продовольственном магазине или в любом другом месте, где продаются товары по массе. Задача состоит в индикации массы товара и его цены в цифровой форме. Входная информация поступает в двух формах: аналоговая величина, соответствующая массе, и цифровая величина, соответствующая цене за килограмм. Напряжение, пропорциональное массе товара, снимается с датчика давления. Это напряжение преобразуется в микропроцессоре в двоично-кодированный десятичный вес, который умножается на двоично-кодированную десятичную цену за килограмм, вводимую в микропроцессор с десятипозиционных наборных переключателей. Для выполнения этих двух операций (аналого-цифрового преобразования и умножения), по существу, и служит микропроцессорная система.

Электронные весы работают так: датчик давления воспринимает массу положенного на весы товара и формирует напряжение, пропорциональное массе. Напряжение усиливается и поступает на микропроцессор. Преобразование

аналогового сигнала массы в цифровую форму осуществляется с участием микропроцессора.

Все шире применяется микропроцессорная техника в медицине. Одним из примеров здесь может быть автоматизированная система наблюдения за состоянием кардиологических больных, построенная на базе микро-ЭВМ. В эту систему входят прикроватные кардиомониторы и центральный пост.

Прикроватный монитор, который может работать не только в системе, но и автономно, рассчитан на обслуживание одного пациента. Микро-ЭВМ в его составе выполняет первичную обработку данных, анализирует кардиосигнал, устанавливает диагноз, формирует сигналы тревоги, запоминает кардиосигналы патологических участков, передает данные на центральный пост.

Центральный пост контролирует и отображает текущее состояние и историю развития болезни восьми пациентов, прогнозирует состояние больных и подготавливает отчетные документы. При этом микро-ЭВМ дополнительно обрабатывает информацию, полученную от кардиомонитора с учетом данных клинического обследования, более точно устанавливает диагноз, формирует оценку тенденций в изменении состояния каждого пациента.

Полианализаторы, оснащенные микропроцессорами, применяются при массовых обследованиях больных. Ими можно быстро и точно измерить характеристики сердца и сосудов, дыхания и кровообращения. Первый в мире портативный эхокардиограф «Аргумент» успешно прошел проверку в космосе на орбитальной станции «Салют-7». Теперь он работает на Земле — в клиниках и машинах скорой помощи.

Одна из областей применения микро-ЭВМ — автоматизированные системы обучения. Так, система программ «Школьница», созданная для микро-ЭВМ «Агат», специально предназначена для обеспечения учебного процесса (см. форзац в конце этой книги).

Каждый учащийся имеет на своем столе цветной телевизор (дисплей) с клавиатурой. Преподаватель со своего пульта может управлять всеми дисплеями учащихся: на их экранах появляются вопросы, условия задач. Учащиеся отвечают на эти вопросы, набирая ответы на клавиатуре. Машина показывает на экране образцы выполнения упражнений, затем предлагает примеры для самостоятельного выполнения. При наличии ошибок машина укажет на них, попросит повторить упражнение.

ЭВМ может обучать самым различным предметам — математике, физике, иностранному языку, программированию и многим другим, для изучения которых разработано программное обеспечение ЭВМ.

Встраивание микропроцессоров в контрольно-измерительную и регистрирующую аппаратуру позволяет в несколько раз повысить точность, скорость и надежность измерений, снизить стоимость приборов, осуществить автоматизацию измерений и обработку результатов. Микропроцессоры применяются в цифровых вольтметрах, самописцах, генераторах сигналов, осциллографах, графопостроителях, автоматических тестерах, медицинских приборах и т. д.

Насчитываются уже десятки тысяч различных по своим назначениям применений микропроцессоров, которые осуществляются по трем основным направлениям.

Первое, самое большое, охватывает приборы и установки, в которые вмонтируются микропроцессоры с целью придания таким установкам качественно новых функций. Это так называемые *микропроцессорные контроллеры* (от англ. to control — «управлять») с программируемыми функциями. Они заменяют разнообразную совокупность схем с жесткой логикой, т. е. всю автоматическую аппаратуру. Например, телефон, в который встроен микропроцессор, кроме выполнения своей обычной функции, еще умеет запоминать номера абонентов, которые звонили во время отсутствия владельца, а также информацию, которую они хотели сообщить; может автоматически повторять вызов при получении сигнала «занято» до тех пор, пока канал связи не освободится, передавать при необходимости отдельные сообщения (предварительно записанные в память устройства) и служить приставкой к телевизору.

Второе направление в применении микропроцессоров предусматривает создание основного функционального блока универсальных микро-ЭВМ. Представителями класса микро-ЭВМ являются персональные компьютеры. Имеется в виду массовое проникновение в нашу повседневную жизнь вычислительной техники благодаря компактности и простоте использования новых машин.

Третье направление в применении микропроцессоров — это создание многопроцессорных систем.

Микро-ЭВМ с успехом используются и в других областях автоматизации управления процессами как производственного, так и непроизводственного характера и будут находить все новые и новые применения.

ОГЛАВЛЕНИЕ

Мирослав Иванович Жалдак
Наталия Викторовна Морзе

ОСНОВЫ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

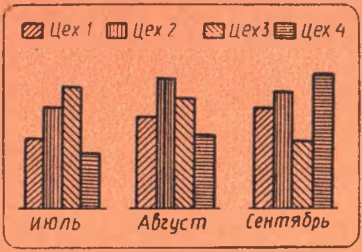
Научный редактор *В. Ф. Хмель*
Оформление художника
В. А. Гурлева
Художественный редактор
С. П. Духленко
Технический редактор
С. Л. Светлова
Корректор
Е. В. Ткачук

Информ. бланк № 11295

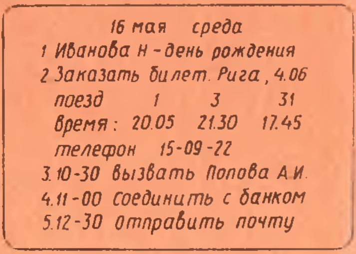
Сдано в набор 31.05.85. Подп. в печать 12.11.85. БФ 02723 Формат 84×108¹/₃₂. Бумага типогр. № 3. Лит. гарн. Выс. печать. У. л. печ. л. 10,5+0,21 форз. Усл. кр.-отт. 10,97. Уч.-изд. л. 11,13+0,18 форз. Тираж 10000 экз. Изд. № 7359. Зак. 5-1287. Цена 75 к.

Головное издательство издательского объединения «Вища школа», 252054, Киев-54, ул. Гоголевская, 7

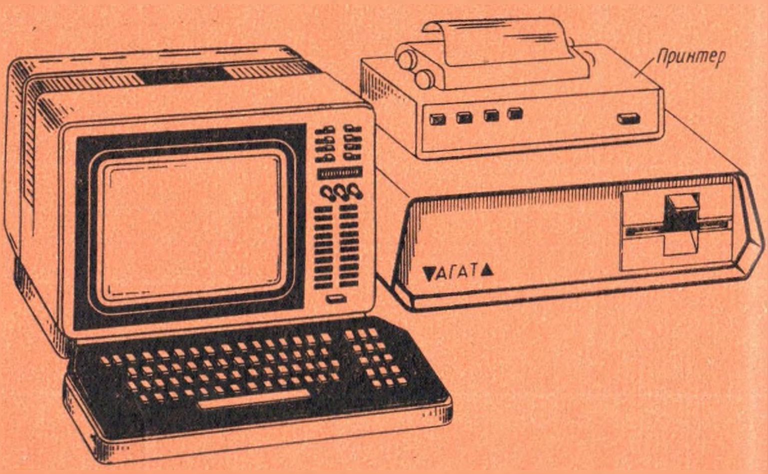
Книжная фабрика им. М. В. Фрунзе, 310057, Донец-Захаржевского, 6/8

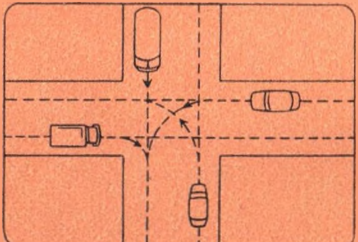


Помощник руководителя

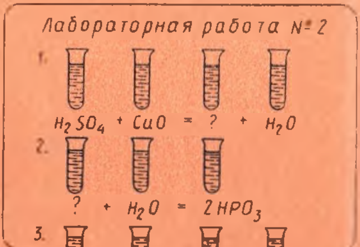


Личный секретарь



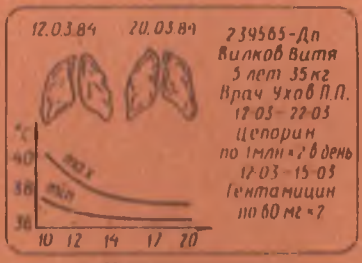


Профессиональная ориентация
и подготовка, профессиональный
отбор, трудоустройство

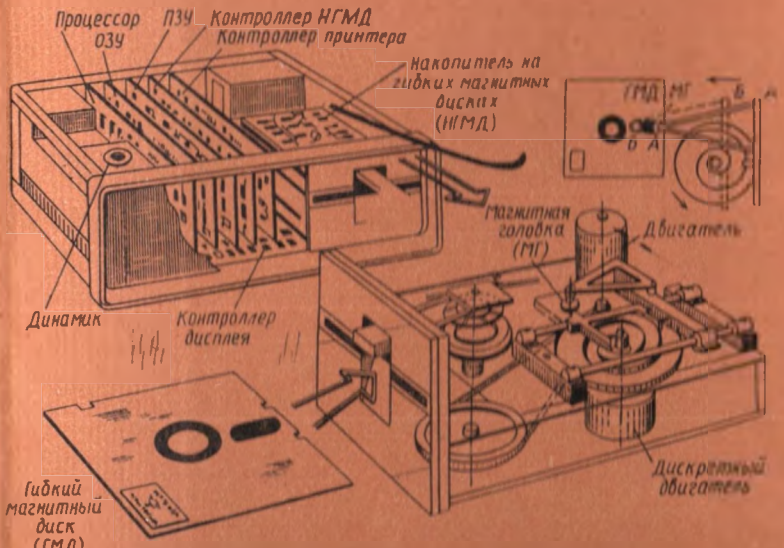


Образование и обучение

Помощник исследователя, инженера



Помощник врача, медицинского работника



Игры

Домашнее хозяйство