1. Принципы организации ЭВМ

В основе организации большинства современных ЭВМ лежат принципы Дж. фон Неймана:

1. Двоичное кодирование информации, разделение ее на слова фиксированной разрядности.

2. Размещение слов в памяти с линейно-адресной организацией *(N* ячеек, *n* разрядов). Ячейки пронумерованы по порядку (О, 1, .., *N*—1). Номер ячейки называется *адресом.* Адрес является идентификатором переменной, хранящейся в соответствующей ячейке.

3. Представление алгоритма программой, состоящей из команд. ***Команда*** - это предписание, определяющее шаг процесса выполнения программы. Она содержит указание операции, адрес операндов и другие служебные признаки.

4. Хранение команд и данных в одной памяти с линейно-адресной организацией. Различие заключается только в способе использования считанного слова.

5. Вычислительный процесс организуется как последовательное выполнение команд в порядке, определяемом программой.

6. Максимальная простота процессора и машинного языка. **Процессом** называют выполнение программы в ЭВМ. **Процессор** - это устройство ЭВМ, которое вызывает процесс и управляет им, т. е. обеспечивает выборку очередной команды и ее исполнение. Другими словами, процессор переводит процесс из одного состояния в другое. Команды ЭВМ, которые выполняет процессор, образуют язык, называемый **машинным**.

7. Жесткость архитектуры. Это означает неизменяемость в процессе работы структуры ЭВМ, списка команд, методов кодирования данных.

На рис. 1.1 представлена структурная схема ЭВМ фон Неймана, состоящая из устройства ввода-вывода (УВВ), оперативного запоминающего устройства (ОЗУ), внешнего запоминающего устройства (ВЗУ), центрального устройства управления (ЦУУ) и арифметико-логического устройства (АЛУ). Все устройства совместно функционируют на основе принципа программного управления. Программа вместе с исходными данными вводится в ОЗУ ЭВМ с помощью УВВ. Она представляет собой последовательность команд, реализующих алгоритм решаемой задачи.

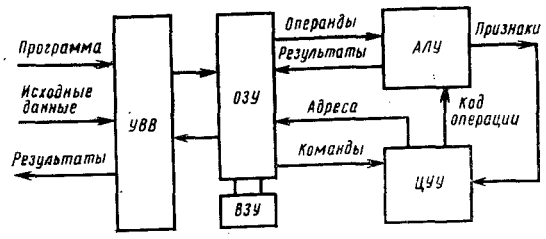

Рис. 1.1. Структура ЭВМ фон Неймана

Формат команды обычно содержит два поля: **код операции и адресную часть** (рис. 1.2).

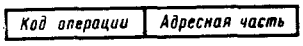

Рис. 1.2. Формат команды

Код операции указывает тип операции, выполняемой в данной команде, адресная часть определяет местонахождение данных, участвующих в операции.

Решение задачи начинается с выборки первой команды программы в ЦУУ и ее обработки. Далее ЭВМ работает автоматически по программе, последовательно исполняя все команды программы до последней, на которой и заканчивается решение задачи.

Принцип программного управления предполагает, что для новой задачи достаточно составить и ввести в ОЗУ новую программу.

Обработка i-й команды программы предполагает последовательность действий, которая реализуется в ЦУУ: выборку очередной i-й команды, ее исполнение, подготовку следующей команды. В состав ЦУУ входят **счетчик команд**, хранящий номер очередной команды, и **регистр команд**. По содержимому счетчика команд (адресу команды) извлекается из ОЗУ очередная i-я команда и записывается в регистр команд. Исполнение i-й команды начинается с анализа двух полей команды. Код операции настраивает АЛУ на выполнение заданной операции. Адреса поступают в ОЗУ, где по ним считываются операнды, участвующие в операции, и записывается результат. Заканчивается цикл, как правило, увеличением содержимого счетчика команд на единицу, подготавливая адрес следующей, (i+1)-й, команды. Такая естественная последовательность адресов команд при необходимости может быть изменена командами управления — безусловного и условного переходов. Команда безусловного перехода определяет адрес следующей команды; команда условного перехода указывает на два адреса следующей команды в зависимости от признака результата ◆, получаемого в АЛУ при исполнении данной или предыдущей команды программы. Если ◆==0, то выполняется (i+1)-я команда программы, если ◆=1, то адресом следующей команды является код, записанный в адресном поле данной команды условного перехода.

Наличие команд управления позволяет использовать ЭВМ как систему с обратной связью, способную учитывать результаты выполненных преобразований.

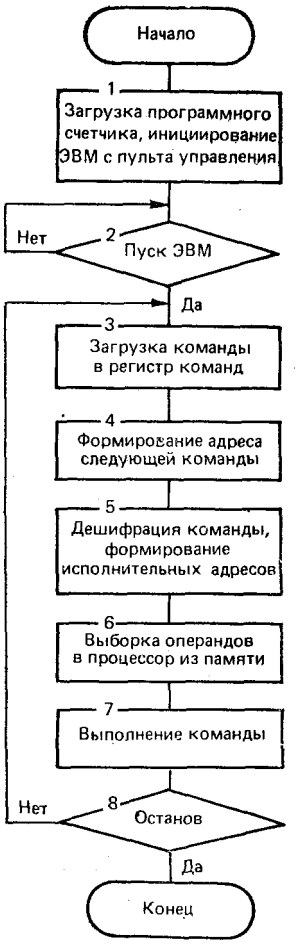

Рис.1.2а. Обобщенный алгоритм функционирования ЭВМ

Поясним необходимость памяти двух уровней: ОЗУ и ВЗУ. Оперативное запоминающее устройство является быстродействующей памятью, так как обменивается данными с АЛУ и ЦУУ, определяя время обработки программы. Однако информационная емкость ОЗУ ограничена. Внешнее запоминающее устройство обладает большой емкостью, но меньшим быстродействием, позволяя хранить программы большого объема. Во время работы ЭВМ необходимая информация из ВЗУ переносится в ОЗУ, а временно ненужная - из ОЗУ в ВЗУ.

Изменения элементно-технологической базы привели к изменению сложившихся принципов проектирования вычислительной техники. На смену им приходят идеи проблемной ориентации систем, параллельной и конвейерной обработки информации, использования табличных методов обработки данных и принятия решений, развиваются принципы регулярности и однородности структур на различных уровнях организации, становится реальной возможностью идея создания адаптивно перестраиваемых вычислительных систем, конфигурация которых изменяется в процессе решения задачи с целью наиболее эффективной организации вычислительного процесса и обеспечения живучести системы.

По мере расширения сферы использования ЭВМ, в частности в связи с их широким применением для управления техническими системами, увеличивается потребность в их проблемной ориентации. Это осуществляется как за счет изменения программ, так и комплексированием аппаратных средств. В связи с этим в организации ЭВМ используют принципы **модульности и магистральности**.

Модульность означает, что ЭВМ как система строится из набора модулей, выполненных в одинаковом конструктиве и одинаковым образом включаемых в состав системы.

Магистральность означает наличие в системе магистрали (системного канала), по которой осуществляется обмен информацией между модулями. Магистральный принцип обеспечивает обмен информацией между функциональными и конструктивными модулями различного уровня с помощью магистралей, объединяющих входные и выходные шины. Обычно выделяются следующие магистрали: шины данных, шины адресные, шины управления. Адресные шины и шины данных обычно одно- двух- или четырехбайтные.

Магистральный обмен информацией является одним из аспектов обеспечения регулярности структуры ЭВМ как на уровне БИС, так и на уровне связей между конструктивными модулями ЭВМ.

Применение магистрального обмена позволяет минимизировать число связей между блоками, обеспечить регулярность операционного устройства управления путем сосредоточения большинства комбинационных схем в АЛУ и коммутаторах, обеспечить стандартизацию интерфейсов, сократить число выводов БИС.

На рис. 1.3 приведена структурная схема ЭВМ, построенной с использованием принципов модульности и магистральности. Возможности таких ЭВМ можно изменять, меняя число и состав модулей. Такие ЭВМ называют магистрально-модульными вычислительными системами.

Для обеспечения возможностей замены модулей и наращивания ресурсов системы подключением новых модулей требуется унификация средств обмена информацией. Периферийные устройства (ПУ) подключаются к магистрали через контроллеры периферийных устройств (КПУ). Модули объединяются в единую ЭВМ на основе стандартной системы связей - стандартного интерфейса (сопряжения). Интерфейс объединения модулей в ЭВМ называется ***внутримашинным интерфейсом*** и представляет собой совокупность унифицированных линий связи, сигналов, правил кодирования и алгоритмов передачи информации.

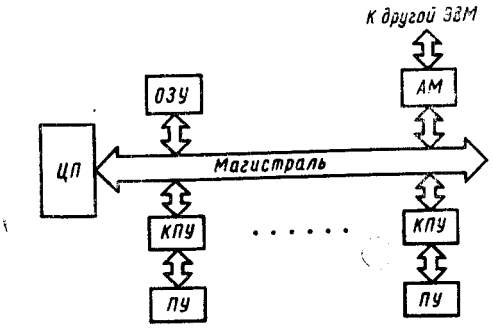

Рис. 1.3. Структура магистрально-модульной системы

Обмен по магистрали осуществляется под управлением ЦП. Однако часть функций обмена при этом реализуется не в ЦП, а в модулях с помощью специальных электронных схем, обеспечивающих подключение модуля к магистрали. В зависимости от сложности реализуемых функций и их состава эти электронные схемы называют **контроллерами** того или иного устройства (ОЗУ, периферийного устройства), **интерфейсными блоками, адаптерами интерфейсов**. Связь с другой ЭВМ осуществляется подключением к ее магистрали через адаптер магистрали АМ.

1.2. Многоуровневая организация вычислительных процессов.

Структурная и функциональная сложность ЭВМ привела к необходимости иерархического подхода с выделением нескольких уровней организации. На рис. 1.4 представлены пять уровней организации вычислительных процессов в ЭВМ: концептуальный, языков высокого уровня, машинных команд, регистровых передач и логических вентилей.

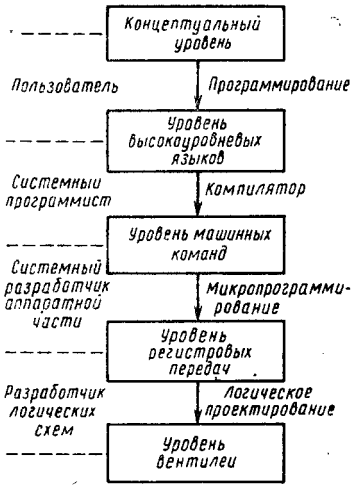

Рис. 1.4. Многоуровневая организация ЭВМ

На **концептуальном уровне** пользователь ЭВМ анализирует задачу, разрабатывает алгоритм ее решения, определяет содержимое обработки информации.

На **уровне языков программирования** высокого уровня изучается

алгоритм решения задачи, составляется детальный проект решения (определяются структуры данных, содержание отдельных программных модулей, связи между ними), пишется программа на одном из языков высокого уровня.

На **уровне машинных команд** обеспечивается связь программных и аппаратных средств. На этом уровне составляется список команд, определяются способы кодирования кодов операций и адресов, число адресных полей и другие параметры, заложенные в структуру ЭВМ. При переходе на данный уровень программа пользователя, написанная на языке высокого уровня, преобразуется в программу, составленную из машинных команд с помощью компилятора соответствующего языка высокого уровня. Компилятор — это специальная системная программа, входящая в состав программного обеспечения ЭВМ. Чем ближе состав операторов языка программирования к списку команд ЭВМ, тем проще компилятор и меньше затраты времени на компиляцию. В частности, минимальных затрат требует ассемблер - компилятор для перевода с языка Ассемблера в машинные коды.

На **уровне регистровых передач** осуществляются элементарные операции, выполняемые аппаратурой ЭВМ. Это операции преобразования информации, образующие функционально полный набор в структурах арифметики и булевой алгебры, а также операции чтения и записи в запоминающее устройство и регистры, операции коммутации, обеспечивающие передачу слов между отдельными модулями и блоками. Операции этого уровня представляют собой пересылки сигналов между регистрами через логические схемы, в частности через комбинационную схему АЛУ. Поэтому этот уровень называют уровнем регистровых передач, а выполняемые элементарные операции - **микрооперациями (МО)**. Для настройки логических схем на выполнение требуемых МО необходимо сформировать соответствующий набор управляющих сигналов (УС). Код набора УС называют **микрокомандой**. Последовательность микрокоманд образует **микропрограмму**. Каждой машинной команде соответствует своя микропрограмма. На уровне регистровых передач осуществляется интерпретация машинных команд программы с использованием микропрограмм и аппаратуры. Поэтому этот уровень организации ЭВМ называют также **микропрограммным.**

На уровне вентилей разрабатываются логические схемы при логическом проектировании аппаратуры ЭВМ. Если на уровне регистровых передач рассматриваются элементарные операции с «n-разрядными» словами, то на уровне вентилей - операции с отдельными двоичными переменными, в частности с отдельными разрядами слов.

1.3. Аппаратные средства и программное обеспечение ЭВМ

ЭВМ связывает две категории специалистов: **разработчиков ЭВМ и пользователей**. Разработчики заинтересованы в простоте аппаратуры и, как следствие, в ограничении числа и простоте машинных команд, пользователи - в удобстве программирования и снижении трудоемкости разработки и отладки программ. Чем более ограничен список команд, тем длиннее программа, тем сложнее ее разработка. Выход из этого противоречия дает **многоуровневая организация вычислительного процесса**, приводящая к понятию виртуальных машин.

Обозначим: Я1—машинный язык, выбранный разработчиком, а Я2—алгоритмический язык, выбранный пользователем. Для выполнения программы, написанной на языке Я2, возможны два способа.

Способ компиляции состоит в замене команд программы на языке Я2 эквивалентными по функциям последовательностями команд Я1 и получении новой программы на языке Я1. Программа, реализующая этот способ, называется компилятором.

Способ интерпретации заключается в использовании специальной программы - интерпретатора на языке Я1. Для нее команды программы на Я2 являются как бы входными данными. В процессе выполнения каждая выбранная из ЗУ команда Я2 анализируется и выполняется эквивалентной последовательностью команд Я1.

ЭВМ с k уровнями функциональной организации можно рассматривать как k виртуальных (кажущихся, гипотетических) машин, каждая из которых имеет свой машинный язык. Для выполнения программ на верхних уровнях требуется использование аппаратуры нижнего уровня ЭВМ. Связь между уровнями осуществляется методами компиляции и интерпретации.

Многоуровневая организация процессов в ЭВМ упрощает ее использование, но снижает эффективность работы за счет возрастающих «накладных расходов» на управление. В процессе работы происходит «челночное» взаимодействие уровней, что требует дополнительного расхода ресурсов ЭВМ (памяти, процессорного времени).

Аппаратные средства ЭВМ – это электронные схемы процессора, основная часть которых реализуется в виде БИС, устройства памяти и ввода-вывода. Таким образом, аппаратные средства состоят из материальных объектов: ***микросхем, соединительных кабелей и шин, читающих и печатающих устройств, терминалов, источников питания***.

Программное обеспечение ЭВМ — **это представление алгоритмов в виде совокупности программ, обеспечивающих заданные функции системы, ее эффективное функционирование и предоставляющих определенный сервис пользователю**.

Аппаратные средства и программное обеспечение выполняют функции преобразования информации по определенным алгоритмам. Любая функция может быть реализована как аппаратно, так и программно. Способ реализации выбирается с учетом стоимости, быстродействия, надежности, частоты возможных изменений (гибкости). Аппаратные средства позволяют наиболее быстро реализовать требуемую функцию.

При разработке многоуровневой машины необходимо решать, какими средствами реализовать ее функции. В настоящее время в большинстве ЭВМ уровни вентилей, регистровых передач и машинных команд реализуются аппаратно, а более высокие уровни — программно.

2. ПРОЦЕССОРЫ

2.1. Структура процессора. Система команд

Процессором называется устройство ЭВМ, непосредственно осуществляющее процесс переработки цифровой информации и управление им в соответствии с заданным алгоритмом, который, как правило, представлен программой.

В некоторых ЭВМ переработку информации выполняют несколько процессоров.

Процессор, управляющий вычислительным процессом, называют центральным.

Большинство ЭВМ содержит только ЦП. Далее ЦП будем называть процессором. Там, где рассматривается дополнительный процессор (например, арифметический или ввода-вывода), это будет поясняться.

Понятие «микропроцессор» в функциональном отношении совпадает с понятием «процессор» и отражает лишь особенности, связанные с использованием технологии СБИС при его реализации.

Процессор - основной блок структуры ЭВМ (рис. 4.1). В нем можно выделить две основные части: **управляющую и операционную**.

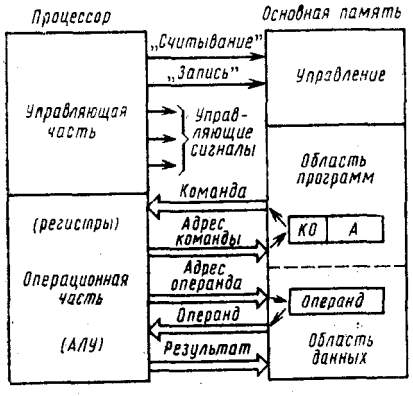

Рис. 4.1. Обобщенная структура процессора

Операционная часть содержит средства, необходимые для выполнения некоторого функционально полного набора элементарных операций. **Управляющая часть** формирует управляющие сигналы (УС) для настройки и коммутации элементов и узлов операционной части в соответствии с выполняемой операцией, а также УС, необходимые для организации обмена информацией с основной памятью и внешними устройствами.

Процессор реализует программное управление вычислительным процессом. Необходимую для этого управляющую информацию он получает в виде команд, хранимых в ОП. Адрес команды, подлежащей выполнению, хранит счетчик команд (СК), называемый также программным счетчиком (ПС). Счетчик команд является одним из операционных элементов процессора. Считанная из ОП команда во время ее исполнения хранится в регистре команд. Команда имеет операционную и адресную части. Адресная часть (А) указывает адреса ячеек ОП, в которых хранятся данные (операнды). Операционная часть содержит код операции (КО), характеризующий соответствующий тип операции, которую должен выполнить процессор. Результат операции в зависимости от типа команды может храниться в одном из регистров процессора либо передается в ОП.

Операционная часть процессора, в которой происходит преобразование информации (выполняются элементарные операции из функционального набора), содержит арифметико-логическое устройство (АЛУ) и различные регистры. Операции двоичной арифметики и булевой алгебры реализуются в АЛУ на уровне регистровых передач между регистрами - источниками операндов - и регистром результата. Эти операции осуществляются за счет пересылки через комбинационную схему, настраиваемую на выполнение определенной элементарной операции.

Процесс решения любой задачи в ЭВМ определяется алгоритмом. ***Алгоритм***—это однозначное предписание последовательности операций, обеспечивающей решение задачи.

Для процессора это предписание задается в виде программы.

Программа - это последовательность команд, представляющая полное описание алгоритма решения задачи.

Команда - это управляющее слово в закодированном виде (например, в двоичном коде), обозначающее одну из операции (сложение, вычитание, сдвиг и т. п.).

Процесс решения задачи заключается в исполнении в процессоре последовательности команд в соответствии с программой, которая предварительно помещается в ОП. Автоматическое управление процессом решения задачи на основе программы, последовательно считываемой из ОП команда за командой, с последующим исполнением закодированных в них операций, является основой ***принципа программного управления***.

Каждая ЭВМ имеет определенную ***систему команд***, характеризуемую списком команд и их структурой (форматом).

Список команд. Он отражает набор операций, выполнение которых предусмотрено на данной ЭВМ. Список команд является одной из важных архитектурных характеристик машины. В современных ЭВМ он включает до 70—100 наименований и более. В любом списке команд можно выделить следующие группы операций: 1) арифметические; 2) логические; 3) пересылки; 4) обращения к периферийным устройствам; 5) передачи управления.

Арифметические операции - сложения, вычитания, умножения, деления, причем в большинстве простых микропроцессоров и микроЭВМ используются только простые (короткие) операции типа сложения, вычитания, добавления или вычитания единицы. «Длинные» арифметические операции типа умножения и деления имеются в списке команд только сложных микропроцессоров. В простейших микропроцессорах таких команд нет и эти операции выполняются программным путем, что требует существенно больших затрат времени. В некоторых процессорах система команд ориентирована только на обработку двоичных чисел в форме с фиксированной запятой. Операции с другими формами представления двоичных чисел и числами в двоично-десятичной системе счисления выполняются программно.

К логическим относятся операции И, ИЛИ, НЕ, а также сравнения и различные логические сдвиги. Логические сдвиги отличаются от арифметических тем, что в них участвуют все разряды чисел, включая и знаковые.

Операции пересылок и обращения к периферийным устройствам обеспечивают перемещение операнда из одной части ячеек ОП в другую, а также между регистром процессора и ОП. В некоторых ЭВМ регистры данных (порты) периферийных устройств имеют адреса, как и ячейки ОП в общем адресном пространстве. В этих случаях для ввода и вывода информации также используются операции пересылок. Понятия «ввод» и «вывод» определены по отношению к процессору. Ввод—это перемещение операнда из порта в процессор, вывод—наоборот.

Во многих ЭВМ адреса портов периферийных устройств выделены в отдельное адресное пространство. Для обращения к ним используются специальные ***команды ввода-вывода***. Они образуют группу команд обращения к периферийным устройствам.

Операции передачи управления предназначены для организации ветвлений в программе и обращений к подпрограммам и выхода из них.

Структура команд. Каждая команда ЭВМ в закодированном виде содержит информацию, необходимую для ответов на следующие вопросы:

Какая операция должна выполняться на данном шаге алгоритма (программы)?

Над какими данными (операндами) выполняется операция?

Куда необходимо поместить результат операции?

Куда (к какой команде) перейти после выполнения данной операции?

В команде выделяется несколько групп разрядов (полей), которые

делят на две части: операционную и адресную. Соглашение о распределении разрядов между полями и о способе кодирования информации определяет структуру команды или ее формат (рис. 4.2).

В операционной части команды, состоящей из n - k двоичных разрядов, содержится код операции (КО), обеспечивающий кодирование 2^{n-k} операций и определяющий, какие при этом будут задействованы устройства в процессоре или вне его.

В большинстве случаев ссылка на операнды в команде осуществляется с помощью адресов ячеек ОП, где они хранятся. В общем случае коды этих адресов, а также адрес результата и (перехода к следующей команде приводятся в адресной части команды, которая может содержать до четырех адресных полей (А1,..., Ак).

Не для всех операций необходим такой полный набор адресов. Поэтому в ЭВМ могут применяться четырех-, трех-, двух- и одноадресные команды, а также и безадресные команды. Адреса A_1, А2 во всех случаях, как правило, означают адрес ячеек ОП, где хранятся первый и второй операнды. Там, где присутствует адрес Аз,

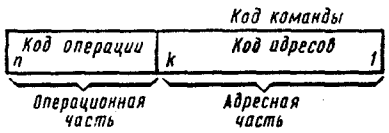

Рис. 4.2. Обобщенная структура команды

а это адрес ячейки, куда помещается результат; А4 — это адрес ячейки, где размещается следующая команда, называемый ***адресом перехода***. ЭВМ с такой структурой команд называют ЭВМ с ***принудительным (произвольным) порядком следования команд***.

Из-за ограниченности числа разрядов в машинном слове, обычно равной разрядности процессора и ОП, актуальны способы повышения экономичности представления информации в команде. В большинстве ЭВМ отказываются от адреса А4, это обусловлено тем, что большинство команд относятся к линейным участкам алгоритмов и могут быть размещены в ячейках ОП с последовательно возрастающими адресами. В этом случае для получения адреса следующей команды к адресу текущей достаточно прибавить, например, единицу, что и выполняет счетчик команд. Такой способ адресации команд называют ***естественным***. При нарушении естественного порядка следования команд (разветвления, объединения, циклы) используют специальные команды управления переходами, в которых имеется адрес перехода, но зато в них не требуются адреса операндов.

Можно отказаться и от адреса результата А3. Это обосновано тем, что при арифметических и логических операциях над двумя операндами в ряде случаев нет необходимости пересылать результат из процессора в ОП, так как он является еще промежуточным. При этом достигается не только сокращение разрядности команды, но и повышается производительность процессора (ЭВМ), так как исключаются лишние пересылки. В тех случаях,

когда необходимо результат переслать в ОП, адресом может быть А1 или А2, т. е. результат записывается на место операнда. Число адресов в адресной части команды можно уменьшить до одного, если принять соглашение, что один ив операндов и полученный результат помещаются в определенный регистр процессора, называемый ***аккумулятором***. Этому случаю соответствуют одноадресные команды. В общем случае для выполнения операции над двумя операндами потребуется три таких команды: загрузка операнда из ОП в аккумулятор; операция над операндом из аккумулятора и другим операндом из ОП; пересылки результата из аккумулятора в ОП. Если в аккумуляторе остается промежуточный результат, используемый в следующем шаге, то число пересылок уменьшается.

К безадресным командам относятся операции пуска, останова и др. Безадресные команды не могут образовать законченную систему команд и применяются только вместе с адресными.

Адресность команд (число полей Ai) влияет на время решения задач, затраты памяти, сложность процессора. Значение этих характеристик зависит также от класса решаемых задач. В частности, для научно-технических расчетов, в которых большой объем занимают многошаговые вычисления, более эффективными оказываются одноадресные команды. Для задач управления, где большую долю составляют пересылки, логические операции и операции управления, эффективнее двухадресные команды. В большинстве ЭВМ используются одновременно двух-, одно- и безадресные команды.

2.2. Архитектуры CISC и RISC.

Двумя основными архитектурами набора команд, используемыми компьютерной промышленностью на современном этапе развития вычислительной техники являются архитектуры CISC и RISC. Основоположником CISC-архитектуры можно считать компанию IBM с ее базовой архитектурой IBM360, ядро которой используется с1964 года и дошло до наших дней, например, в таких современных мейнфреймах как IBM ES/9000.

Лидером в разработке микропроцессоров с полным набором команд (CISC -Complete Instruction Set Computer) считается компания Intel со своей серией x86 и Pentium. Эта архитектура является практическим стандартом для рынка микрокомпьютеров. Для CISC-процессоров характерно: сравнительно небольшое число регистров общего назначения; большое количество машинных команд, некоторые из которых нагружены семантически аналогично операторам высокоуровневых языков программирования и выполняются за много тактов; большое количество методов адресации; большое количество форматов команд различной разрядности; преобладание двухадресного формата команд; наличие команд обработки типа регистр-память.

Основой архитектуры современных рабочих станций и серверов является архитектура компьютера с сокращенным набором команд (RISC -

Reduced Instruction Set Computer). Корни этой архитектуры уходят к компьютерам CDC6600, которые одни из первых начали оснащаться упрощенным набором команд для увеличения быстродействия. RISC в современном его понимании сформировалось на базе трех исследовательских проектов компьютеров: процессора 801 компании IBM, процессора RISC университета Беркли и процессора MIPS Стенфордского университета.

Эти три машины имели много общего. Все они придерживались архитектуры, отделяющей команды обработки от команд работы с памятью, и делали упор на эффективную конвейерную обработку. Система команд разрабатывалась таким образом, чтобы выполнение любой команды занимало небольшое количество машинных тактов (предпочтительно один машинный такт). Сама логика выполнения команд с целью повышения производительности ориентировалась на аппаратную, а не на микропрограммную реализацию. Чтобы упростить логику декодирования команд использовались команды фиксированной длины и фиксированного формата.

Среди других особенностей RISC-архитектур следует отметить наличие достаточно большого регистрового файла (в типовых RISC-процессорах реализуются 32 или большее число регистров по сравнению с 8 - 16 регистрами в CISC-архитектурах), что позволяет большему объему данных храниться в регистрах на процессорном кристалле большее время и упрощает работу компилятора по распределению регистров под переменные. Для обработки, как правило, используются трехадресные команды, что помимо упрощения дешифрации дает возможность сохранять большее число переменных в регистрах без их последующей перезагрузки.

С 1986 года началась активная промышленная реализация архитектуры RISC. К настоящему времени эта архитектура прочно занимает лидирующие позиции на мировом компьютерном рынке рабочих станций и серверов.

Следует отметить, что в последних разработках компании Intel, а также ее последователей-конкурентов (AMD, Cyrix, NexGen и др.) широко используются идеи, реализованные в RISC-микропроцессорах, так что многие различия между CISC и RISC стираются. Однако сложность архитектуры и системы команд x86 остается и является главным фактором, ограничивающим производительность процессоров на ее основе.