**Система команд процессора и разновидности архитектур**

*Система команд* – это набор допустимых для данного процессора управляющих кодов и способов адресации данных. В общем случае система команд жестко связана с конкретным типом процессора, поскольку определяется аппаратной структурой блока дешифрации команд. Поэтому разные процессоры могу иметь как разные команды в своей системе команд, так и их разное количество.

Тем не менее все команды системы команд процессора разбиты на различные общепринятые группы команд:

* пересылки данных (наиболее многочисленная группа). В таких командах содержится информация о адресе источника и получателя операндов (адреса ячеек памяти, номера регистров процессора или информация о том, что операнды расположены в стеке), длина подлежащих пересылке данных, заданная явно или косвенно и способ адресации каждого из операндов, с помощью которого содержимое адресной части команды может быть пересчитано в физический адрес операнда;
* арифметической и логической обработки – команды, обеспечивающие арифметическую и логическую обработку информации в различных формах ее представления;
* ввода/вывода – это команды управления периферийным устройством, проверки состояния ввода/вывода, и непосредственно команды ввода и вывода;
* управления потоком команд – это команды способные изменить последовательность вычислений, т. е. безусловный переход, условный переход (ветвление), вызовы процедур и возвраты из процедур;
* битового управления – команды для реализации возможности независимого управления разрядами портов или регистров;
* преобразования и др.

*Классификация систем команд*

C точки зрения системы команд существуют следующие архитектуры:

1. CISC (Complicated Instruction Set Computer) – полная система команд;
2. RISC (Reduced Instruction Set Computer) – сокращенная система команд.

***CISC-архитектура***

* максимально полный набор команд, которые процессор может реализовывать. Такая система команд составляет сотни команд;
* команды имеют нефиксированный размер кратный байту (1-8 байт);
* система команд, как правило, неортогональна, то есть не все команды могут использовать любой из способов адресации применительно к любому из регистров процессора;
* выборка команды на исполнение осуществляется побайтно в течение нескольких циклов. Время выполнения команды может составлять от 1 до 12 циклов;
* обладает сложной кодировкой команд.

Достоинства CISC-архитектуры:

* компактность наборов инструкций уменьшает размер программ и уменьшает количество обращений к памяти;
* наборы инструкций включают поддержку конструкций высокоуровневого программирования.

Недостатки CISC-архитектуры:

* нерегулярность потока команд;
* высокая стоимость аппаратной части;
* сложности с распараллеливанием вычислений.

***RISC-архитектура***

* набор исполняемых команд сокращен до минимума. Такая система команд составляет десятки команд. Для реализации более сложных операций приходится комбинировать команды;
* все команды имеют формат фиксированной длины (не обязательно кратный байту);
* простая кодировка команд;
* выборка команды из памяти и ее исполнение осуществляется за один такт синхронизации;
* система команд предполагает возможность равноправного использования всех регистров процессора. Это обеспечивает дополнительную гибкость при выполнении ряда операций.

Достоинства RISC-архитектуры:

* высокая скорость выполнения команд;
* меньшая необходимая площадь кристалла самого процессора, а следовательно, и более низкая стоимость;
* снижение нерегулярности потока команд.

Недостатки RISC-архитектуры:

* для выполнения более сложных команд (отсутствующих в системе команд) требуется комбинирование нескольких простых команд;

*Следует отметить, что время выполнения всей программы зависит не только от количества команд в программе, но и от того, какие именно команды используются. Так абстрактно однозначно нельзя сказать какая из архитектур CISC или RISC будет более производительнее. Сравнить можно только на примере конкретной выполняемой программы.*

***Форматы команд и способы адресации***

 Формат команды определяет ее структуру, т. е. количество разрядов, отводимых под всю команду, а также количество, разрядность и расположение отдельных полей команды

Поле команды – это совокупность разрядов, кодирующих составную часть команды.

Длина команды влияет на организацию и емкость памяти, структуру шин и быстродействие процессора.

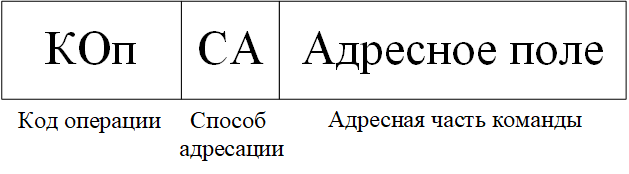
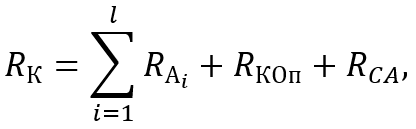
Наиболее классическое представление формата команды проиллюстрировано на рисунке 3.9.  


Рисунок 3.9 – Представление формата команды

Длина команды может быть определена как



где *l* – количество адресов в адресном поле команды (адресность);  
*R*А – количество разрядов для записи *i*-гo адреса;   
      *R*КОп – разрядность поля кода операции;   
      *R*CA – разрядность поля способа адресации.

Количество двоичных разрядов, отводимых под код операции, выбирается таким образом, чтобы каждая команда в системе команд имела уникальный номер. Если система команд предполагает *N* различных команд, то минимальная разрядность поля кода операции *R*КОп*min* определяется как



*Адресность* – это количество адресов в адресном поле команды.

В фон Неймановских машинах максимальная адресность команды может быть равна трем, а минимальная – единице (или при неявной адресации – нулю).

Трехадресный формат представлен на рисунке 3.10 (а).

Если результат будет записан на место одного из операндов, то можно получить двухадресный формат команды (рисунок 3.10 (б)). В этом случае соответствующий замещаемый операнд после выполнения операции теряется.

Если один из операндов берется из регистра или стека, или выполняется одноместная операция (операция над одним операндом), то можно сократить формат команды до одноадресного (рисунок 3.10 (в)).

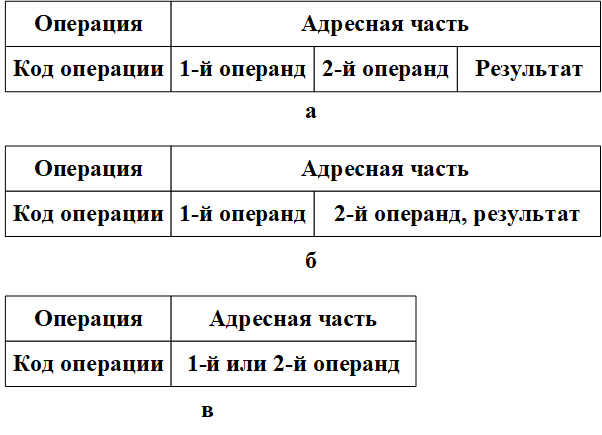


Рисунок 3.10 – Адресность формата команды

Возможен также вариант, когда в адресном поле отсутствует адресная информация, при этом говорят о неявной адресации (например, адресного поля либо просто нет, либо отсутствующий адрес подразумевается кодом операции).

***Способы адресации***

*Исполнительный адрес операнда* (Аисп) – это двоичный код номера ячейки памяти, служащий источником или приемником операнда. По этому коду происходит фактическое обращение к указанной ячейке.

*Адресный код команды* (Ак) – это двоичный код в адресном поле команды, из которого формируется исполнительный адрес операнда.

*Способ адресации* – это способ формирования исполнительного адреса операнда Аисп по адресному коду команды Ак.

***Непосредственная адресация (НА)***

При таком способе адресации в адресном поле команды вместо адреса содержится непосредственно само значение операнда (рисунок 3.11 (а)).

Этот способ может применяться при выполнении арифметических операций, операций сравнения, а  также для загрузки констант в регистры. Когда операндом является число, оно обычно представляется в дополнительном коде. При записи в регистр, имеющий разрядность, превышающую длину непосредственного операнда, операнд размещается в младшей части регистра, а оставшиеся свободными позиции заполняются значением знакового бита операнда.

***Прямая адресация (ПА)***

При прямой адресации адресный код прямо указывает номер ячейки памяти, к которой производится обращение, т. е. адресный код команды Ак = исполнительному адресу операнда Аисп (рисунок 3.11 (б)).

Недостаток данного способа адресации – это ограниченный размер адресного поля команды, т. к. для адресации к памяти большой емкости нужно большое количество разрядов адресного поля.

***Косвенная адресация (КА)***

При косвенной адресации с помощью ограниченного адресного поля команды указывается адрес ячейки небольшого адресуемого пространства памяти, в свою очередь, содержащей полноразрядный адрес операнда (рисунок 3.11 (в)).

При косвенной адресации содержимое адресного поля команды остается неизменным, в то время как косвенный адрес в процессе выполнения программы можно изменять. Это позволяет проводить вычисления, когда адреса операндов заранее неизвестны и появляются лишь в процессе решения задачи. Также такой прием упрощает обработку массивов и списков, а также передачу параметров подпрограммам.

***Регистровая адресация (РА)***

При регистровой адресации адресное поле команды содержит не адрес ячейки памяти, а адрес регистра процессора, по которому находится операнд (рисунок 3.11 (г)).

Обычно размер адресного поля в данном случае составляет четыре-шесть бит, что позволяет указать на один из 16-64 регистров общего назначения (РОН).

Основными преимуществами регистровой адресации являются короткое адресное поле в команде и исключение обращений к памяти. Возможности использования регистровой адресации ограничены числом РОН в составе процессора.

***Относительная адресация (ОА)***

При относительной адресации, для получения исполнительного адреса операнда, необходимо адресный код команды сложить со значением содержимого счетчика команд (рисунок 3.11 (д)).

Таким образом, адресный код команды представляет собой смещение относительно адреса текущей команды. Однако, в момент вычисления исполнительного адреса операнда в счетчике команд может уже быть сформирован адрес следующей команды, что нужно учитывать при выборе величины смещения.

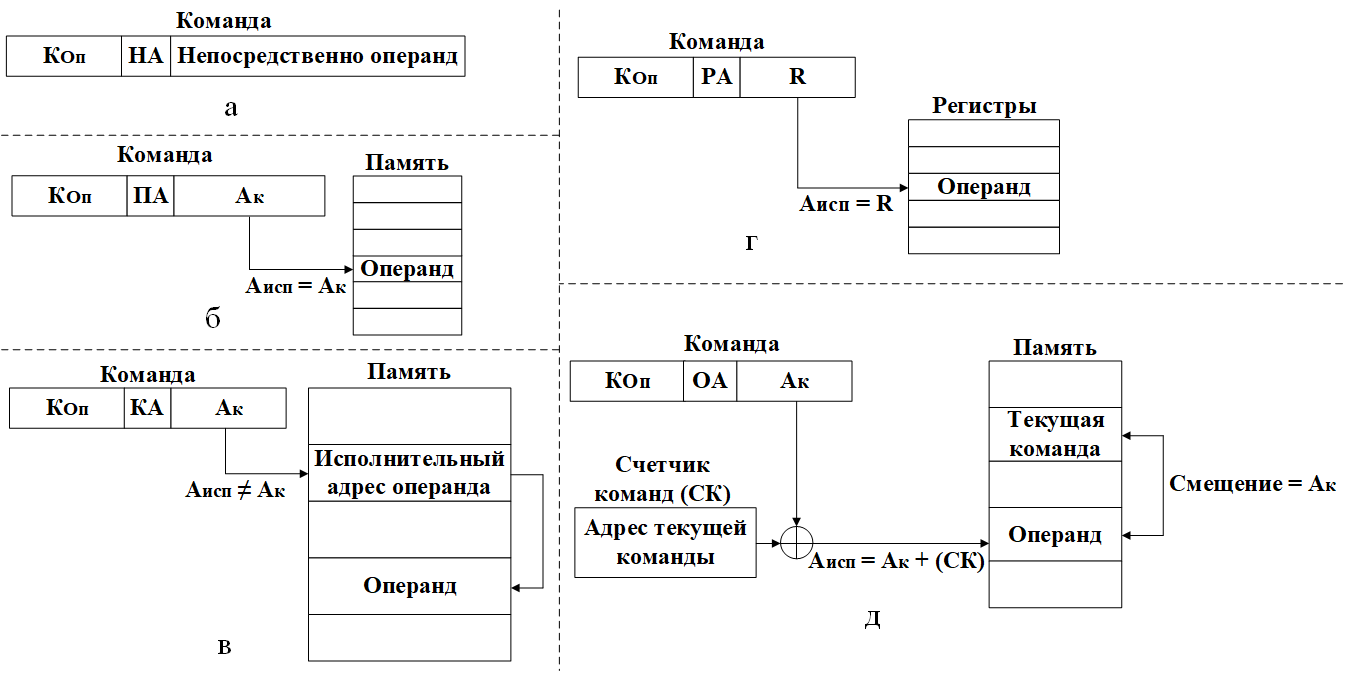
Поскольку смещение будет складываться со значением счетчика команд, а операция сложения не выполняется в прямом коде, то чаще всего адресный код команды уже записан в дополнительном коде.  


Рисунок 3.11 – Способы адресации

**Основные устройства процессора**

 Современные процессоры в своем составе имеют такие устройства, как: устройство управления (УУ), арифметико-логическое устройство (АЛУ), наборы различных регистров (РОН, регистры специальных функций, аккумуляторы, блок управляющих регистров и т. д.), контроллеры и кэш-память.

Простейшая структура процессора представлена на рисунке 3.12.

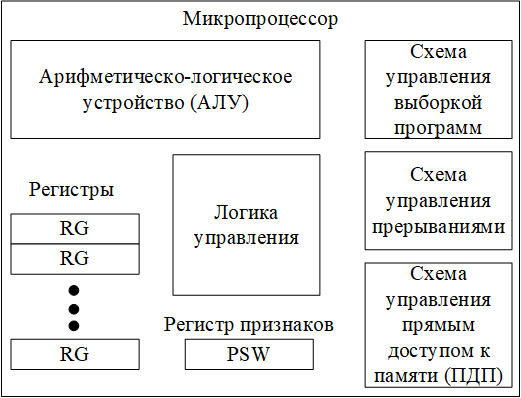


Рисунок 3.12 – Простейшая структура процессора

УУ организует автоматическое выполнение программ Пересылка информации между любыми элементами ЭВМ инициируется своим сигналом управления (СУ), т. е. управление вычислительным процессом сводится к выдаче нужного набора СУ в нужной временной последовательности. Основной функцией УУ является формирование управляющих сигналов, отвечающих за извлечение команд из памяти в порядке, определяемом программой, и последующее исполнение этих команд. Кроме того, УУ формирует СУ для синхронизации взаимодействия внутренних и внешних устройств ЭВМ.

АЛУ обеспечивает арифметическую и логическую обработку двух входных операндов, в результате чего формируется значение выходной переменной. Функции АЛУ обычно сводятся к простым арифметическим и логическим операциям, а также операциям сдвига.

Помимо вычисления результата арифметические и логические операции сопровождаются формированием в АЛУ признаков (флагов), характеризующих этот результат. Наиболее часто фиксируются такие признаки, как: Z (Zero) – нулевой результат; N (Negative) – отрицательный результат; V (Overflow) – переполнение разрядной сетки; С (Carry) – наличие переноса.

В процессоре может быть одно универсальное АЛУ для выполнения всех основных арифметических и логических преобразований или несколько специализированных АЛУ или операционных блоков для отдельных видов операций, например АЛУ для чисел с фиксированной точкой и АЛУ для числе с плавающей точкой.

Блок управляющих регистров предназначен для временного хранения управляющей информации. Он содержит регистры и счетчики, участвующие в управлении вычислительным процессом: регистры, хранящие информацию о состоянии процессора, регистр-счетчик адреса команды – счетчик команд, счетчики тактов, регистр запросов прерывания и др.