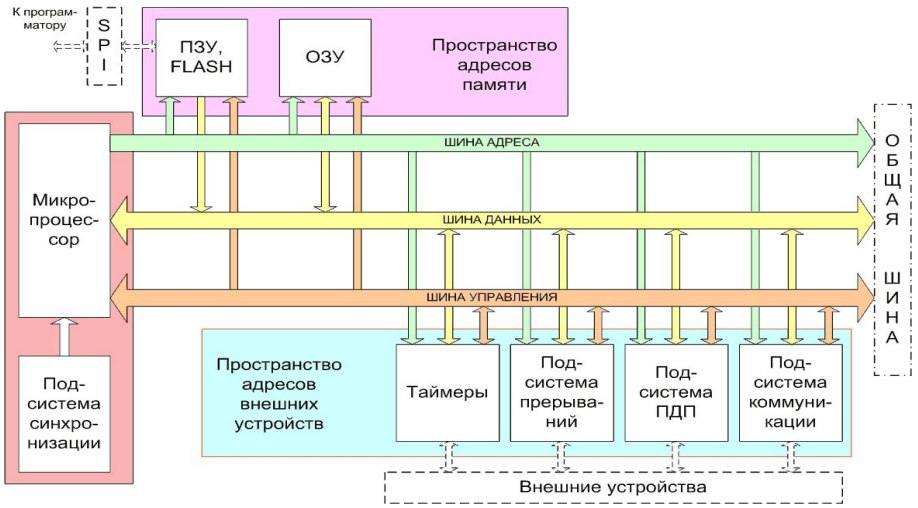
Внутренняя структура uP.

Классика принстонской архитектуры.

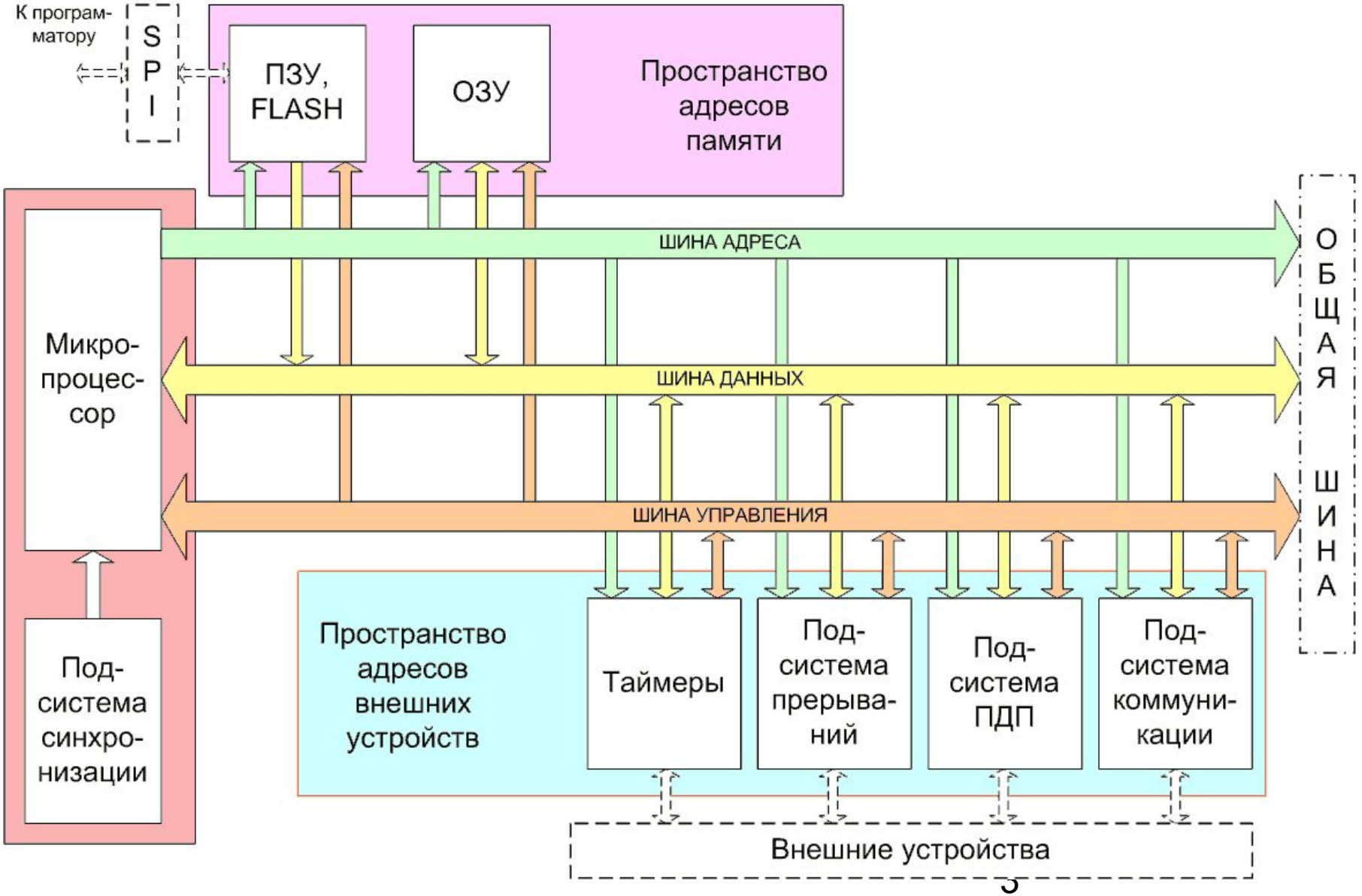
Структура микропроцессорного устройства



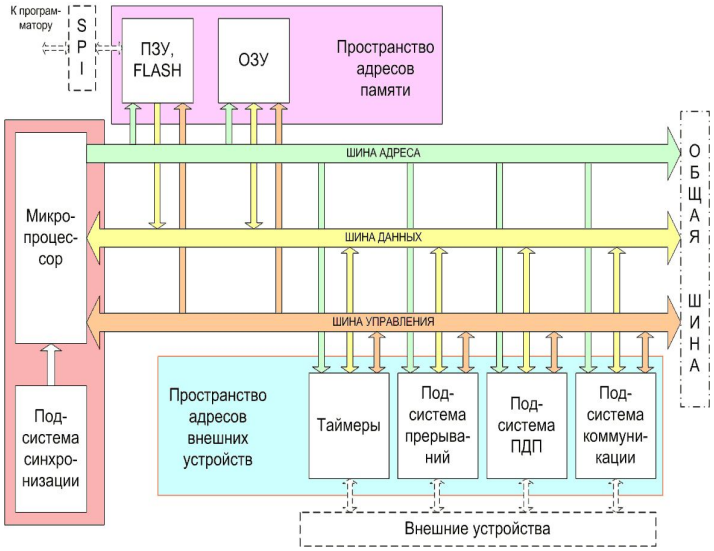
Наиболее широко применяется магистрально-модульный принцип построения микропроцессорных систем (МПС) при котором отдельные устройства (модули), входящие в состав системы, обмениваются информацией по общей системной шине - магистрали.

- ***Системная шина*** содержит большое количество проводников, которые в соответствии с их функциональным назначением подразделяются на отдельные *шины - адреса, данных и управления*.
- ***Шина адреса*** служит для передачи адреса, который формируется микропроцессором и позволяет выбрать необходимую ячейку памяти ОЗУ (ПЗУ) или требуемое внешнее устройство.
- ***Шина данных*** служит для выборки команд и данных из ПЗУ, обмена данными с ОЗУ или внешними устройствами.
- По ***шине управления*** передаются разнообразные управляющие сигналы, определяющие режимы работы памяти (запись или считывание), интерфейсных устройств (ввод или вывод информации) и микропроцессора (запуск, запросы внешних устройств на обслуживание, информация о текущем режиме работы и другие сигналы).

Трехшинная магистраль.



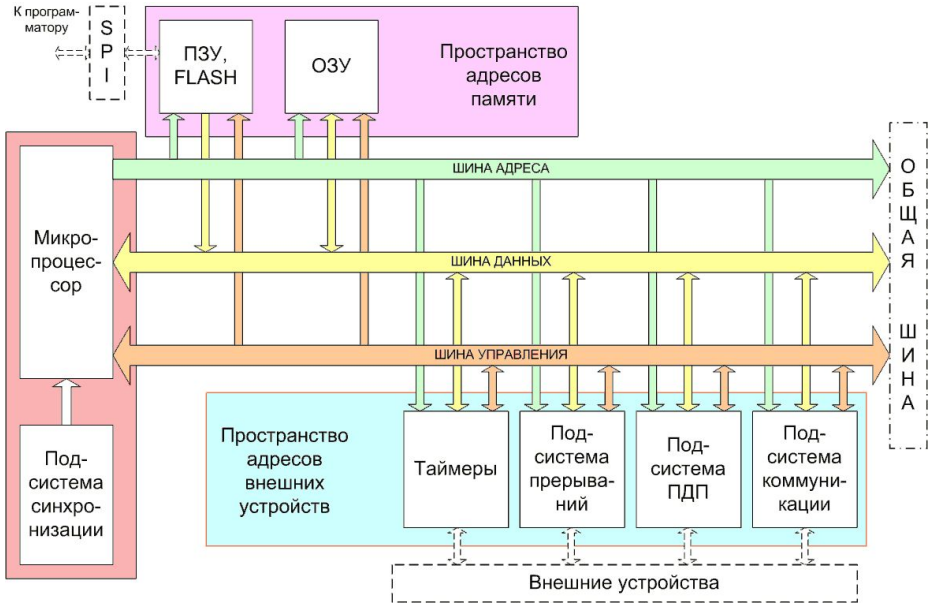
Архитектура Фон Неймана (Принстонская)



Основным преимуществом архитектуры Фон Неймана является простота. Содержимое ОЗУ (RAM - Random Access Memory) может быть использовано как для хранения данных, так и для хранения программ. В некоторых приложениях программе необходимо иметь доступ к содержимому стека. Это предоставляет большую гибкость для разработчика программного обеспечения.

- Упрощенная структура МПС *принстонской* архитектуры (фон-Неймана), использующая трехшинную магистраль. С целью сокращения количества физических соединений (проводов), возможно использование двухшинной магистрали. В случае двухшинной магистрали, шина адреса и данных объединена в единую, т.е. адреса и данные передаются по одним и тем же физическим проводникам с разделением по времени- *мультиплексирование линий адреса и данных*. Передача адреса при этом сопровождается дополнительным сигналом, указывающим на то, что в данный момент времени на шине выставлен адрес, а не данные. При этом падает скорость обмена между процессором и остальными узлами МПС.

Системная шина

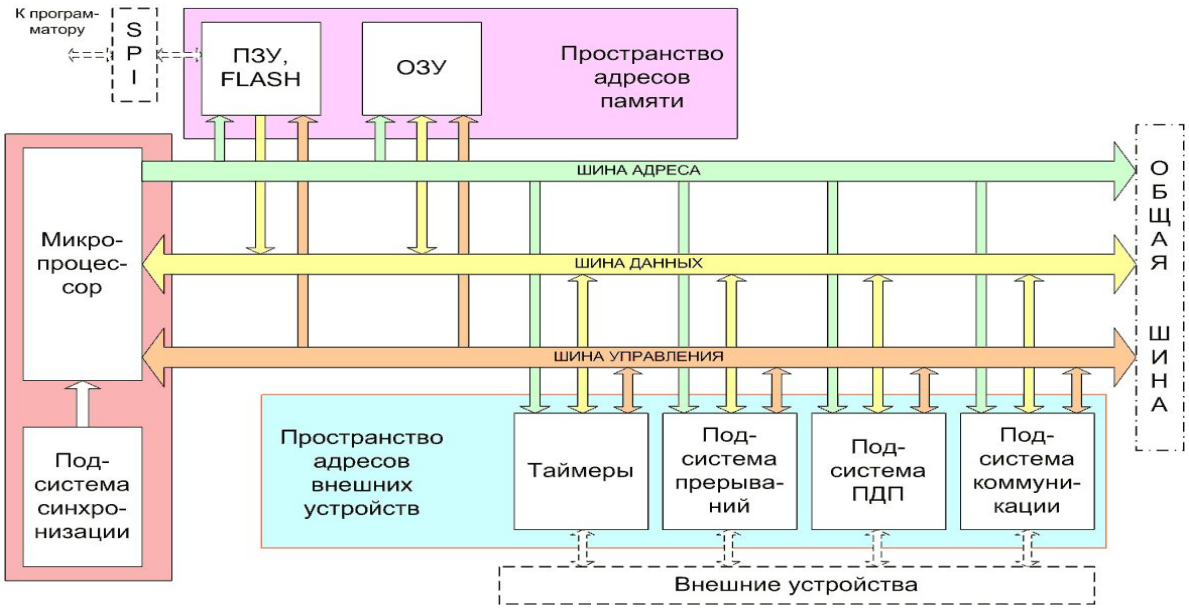


По системной шине происходит три вида передачи данных:

1. процессор - память;
2. процессор - интерфейс;
3. память - интерфейс.

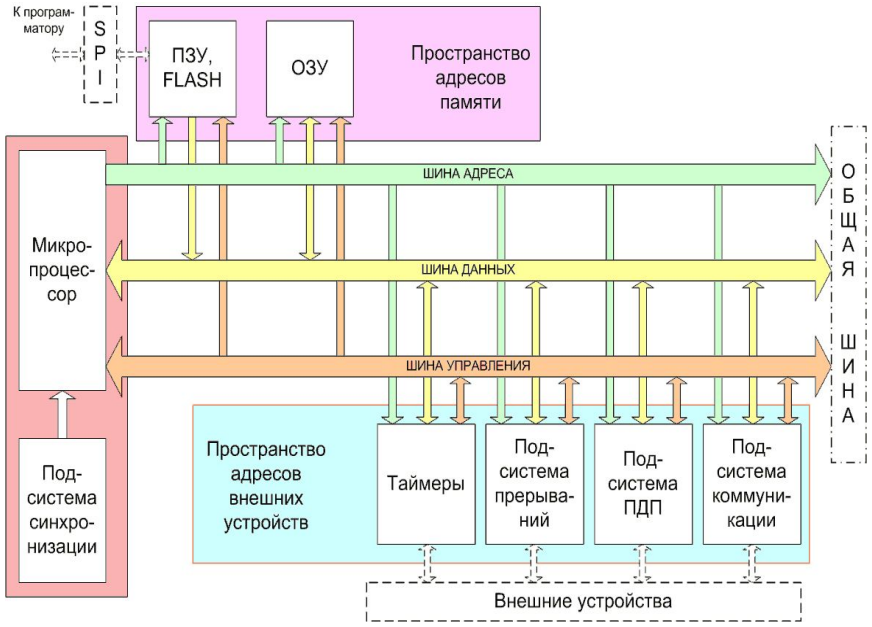
- В первых двух случаях передачей данных управляет процессор. Память или интерфейс по управляющему сигналу от процессора осуществляют передачу данных.
- Направление передачи данных, как правило, определяется процессором.
- Пересылка данных с шины данных внутрь процессора называется считыванием, обратный процесс - записью.
- Третий случай осуществляется механизмом прямого доступа к памяти ПДП и будет рассмотрен позднее.

ЦПУ, Синхронизация



- **Центральным процессорным устройством** (ЦПУ) в системе является микропроцессор (МП), выполняющий арифметические и логические операции над данными, осуществляющий управление выборкой команд и данных из памяти и организующий взаимодействие всех устройств, входящих в систему.
- Работа МП происходит под воздействием тактовых сигналов, вырабатываемых *схемой синхронизации*, часто выполняемой в виде отдельной микросхемы (генератора тактовых импульсов).

Разрядность шины. Шина данных.

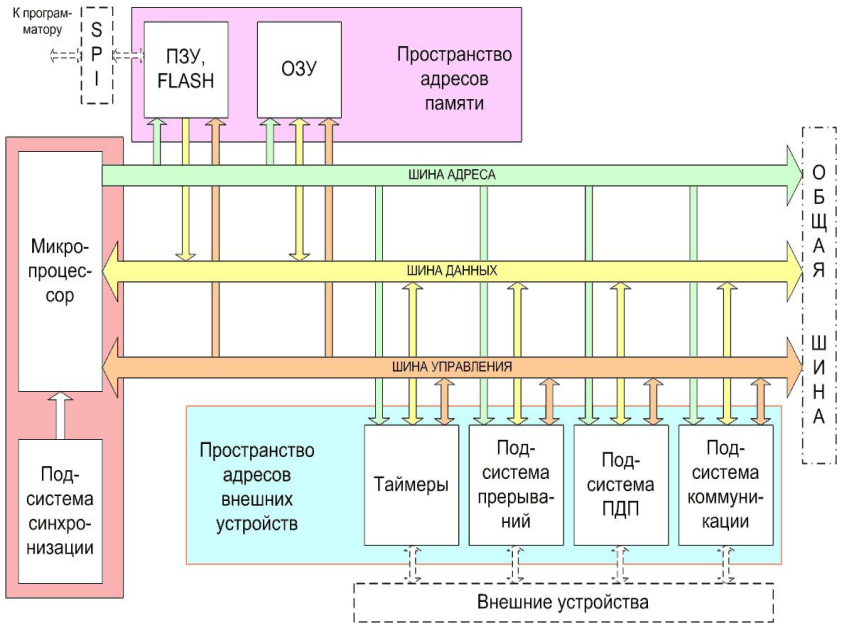


Если в качестве шины использовать восемь сигнальных проводов, можно параллельно (одновременно) передавать 8 бит информации, соответственно по 16-жильному проводу можно передавать 16 бит информации.

Количество сигнальных линий определяет разрядность шины. Обычно разрядность шины данных и длину слов, обрабатываемых процессором, выбирают одинаковыми.

- **Шина данных** используется как для передачи данных в направлении процессор - память и процессор - интерфейс, так и для их передачи в обратном направлении. Шина данных является *двунаправленной*.
- Передача по шине данных может производиться в обоих направлениях, однако в каждый заданный момент времени она осуществляется лишь в одном направлении. Это означает, что для передачи данных в систему и их приема из системы микропроцессор меняет направление работы шины данных. По всем разрядам шины в текущий момент времени данные передаются лишь в одном направлении, т. е. в любой момент по всем линиям шины они могут либо только, передаваться либо только приниматься процессором.).

Шина адреса.



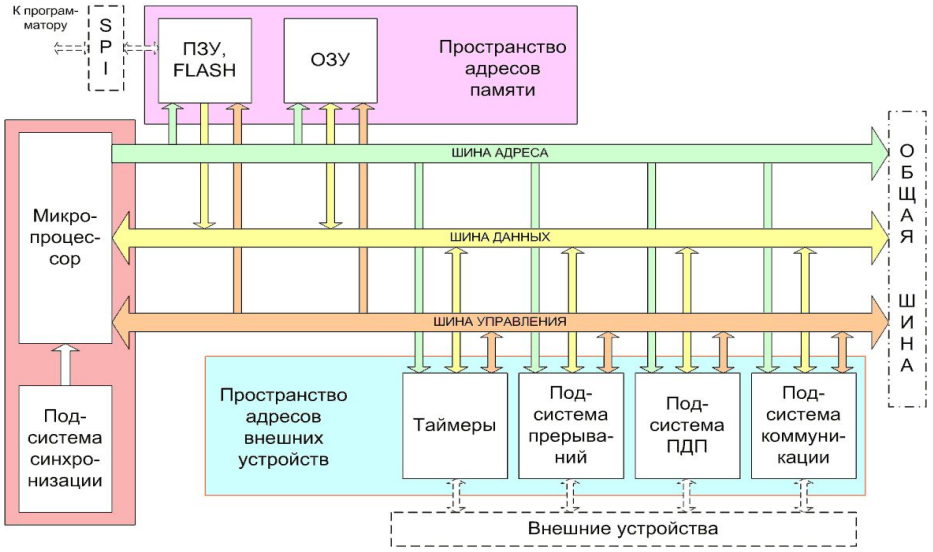
Передача адресов происходит по шине адресов. Адрес, по которому осуществляет доступ, определяет процессор. ***Эта шина является выходной однонаправленной по отношению к процессору.***

Если обозначить разрядность адресной шины n, то наибольшее число адресов, к которым можно обращаться по этой шине, равно 2^n. Так, для 16-разрядной шины это число равно 64К, для 24-разрядной - 16М, для 32-разрядной - 4G (где 1К=2^{10}. 1М=2^{24}, 1G = 2^{32}).

Распределяемая область адресов называется **адресным пространством**.

- Все **адресное пространство** МПС разделено на *пространство адресов памяти* и пространство адресов *внешних устройств* - иначе устройств *ввода-вывода.* Это означает, что по одному и тому же адресу, выставляемому процессором на шину адреса, возможно обращение (обмен данными) к ячейке памяти или контроллеру внешнего устройства. Куда конкретно произойдет обращение, в этом случае, определяют сигналы шины управления. *Адреса ОЗУ и ПЗУ находятся в пространстве адресов памяти.*
- Остальные устройства находятся, *как правило*, в пространстве адресов ввода-вывода и называются внешними устройствами. Внешние устройства подключаются посредством специальных программируемых устройств сопряжения – интерфейсов.

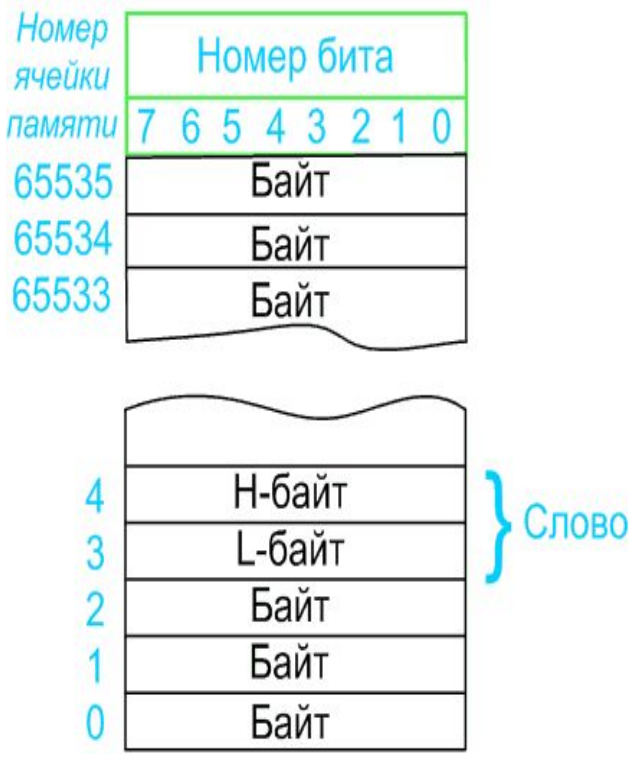
Шина управления.



Шина управления служит для передачи различных управляющих сигналов, часть которых является *выходными сигналами*, а другая часть - *входными сигналами* по отношению к процессору. Конкретный состав этих сигналов зависит от типа процессора.

- Основными сигналами, передаваемыми по этой шине, являются сигналы управления записью-чтением из памяти и сигналы управления вводом-выводом данных из соответствующего порта. Если применяется способ изолированного ввода-вывода, передаются четыре управляющих сигнала:
  - MEMRD (MRD) - (“memory read” -считывание данных из памяти);
  - MEMWR(MWR) - (“memory write” -запись данных в память)
  - IORD - (“in-out read” -ввод данных из порта ввода)
  - IOWR - (“in-out write” - вывод данных в порт вывода).
- В случае ввода-вывода с отображением адресов УВВ на адреса памяти, порты ввода-вывода и память не различаются, обращение производится одними и теми же командами, генерирование специальных сигналов управления не происходит, поэтому можно использовать только сигналы RD-чтение и WR-запись.

Подсистема памяти. Линейная адресация.



Линейная организация памяти

- Линейная и сегментная адресация.
- Память представляет собой единую совокупность ячеек для хранения информации.
- Каждой ячейке присваивается адрес в следующем порядке – нулевой адрес, первый адрес и т.д. В ряде случаев память МПС (с точки зрения программиста) представляет собой линейно упорядоченный набор n-разрядных ячеек с произвольным доступом— *линейная память.*
- Каждой ячейке набора соответствует число, называемое ее *адресом.*
- Все адреса занимают целочисленный диапазон от 0 до 2^{m-1}, который образует *адресное пространство* памяти *разрядности* ***m***.
- В тех случаях, когда наименьшая адресуемая единица - байт (n=8), память имеет байтовую организацию.

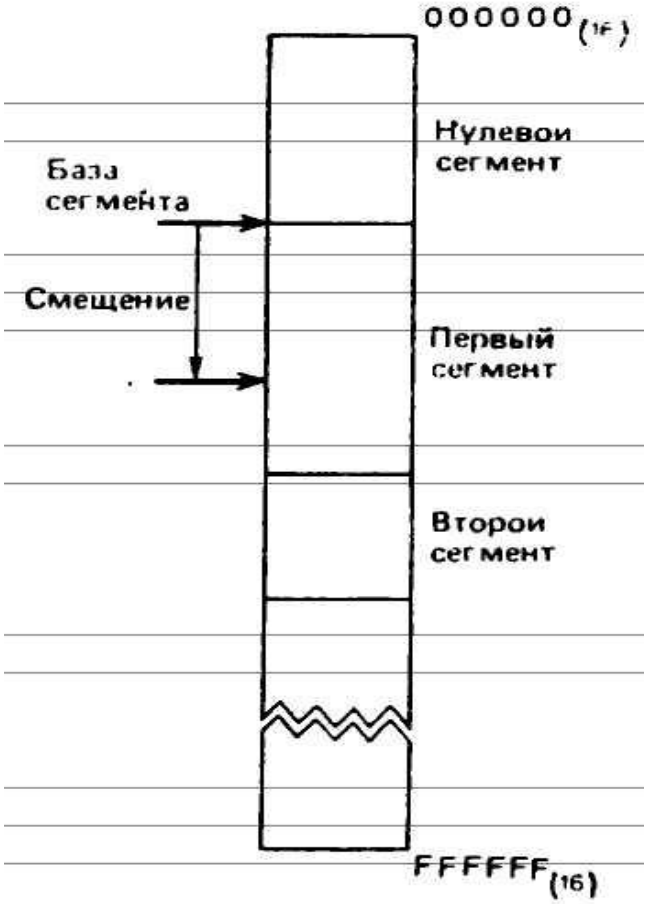
Подсистема памяти. Подсистема памяти. Сегментная адресация.

- При *сегментной адресации* все пространство адресов делится на множество сегментов.
- Пространство, разбитое на такие сегменты, называется сегментированным пространством адресов.
- Начальный адрес сегмента называется *базовым*. За каждым сегментом закреплен соответствующий номер.
- Порядок разбиения пространства может быть произвольным, и, после того как он установлен, адрес можно представить с помощью номера сегмента и некоторого *смещения*.
- При сегментной адресации такое разбиение позволяет представить адрес в виде двух целочисленных величин - *номера сегмента и смещения*.
- Вычисление адреса в принципе сводится только к вычислению смещения, т. е. для вычисления адреса можно воспользоваться тем же 16-разрядным АЛУ, не прибегая к помощи 32-разрядных регистров и сумматоров, что значительно упрощает структуру процессора.

Сегментация, логические и физические адреса.

- Программы «видят» пространство памяти как группу сегментов, определяемых самой программой.
- *Сегмент* представляет собой логическую единицу памяти определенного размера, например, 64Кбайт. Он состоит из смежных ячеек памяти и является независимой и отдельно адресуемой единицей памяти.
- Каждому сегменту программой назначается *базовый* (начальный) *адрес*, являющийся адресом его первого байта в адресном пространстве памяти.
- Все сегменты начинаются на 16-байтных границах памяти, называемых *границами параграфов.* Других ограничений на размещение сегментов в памяти нет - сегменты могут «быть соседними (смежными), не перекрывающимися (непересекающимися), частично или полностью перекрывающимися.
- В каждом применении сегменты определяются и используются по-разному. Если начального разделения рабочего пространства достаточно, то можно инициализировать сегментные регистры, а затем забыть о них. Если приходится модифицировать содержимое сегментных регистров, то необходимо тщательно следить за их определением и использованием.
- Сегментная организация памяти рассчитана на *модульные программы,* так как во многих ситуациях сегментация обеспечивает определенные преимущества.

Сегментные регистры процессора

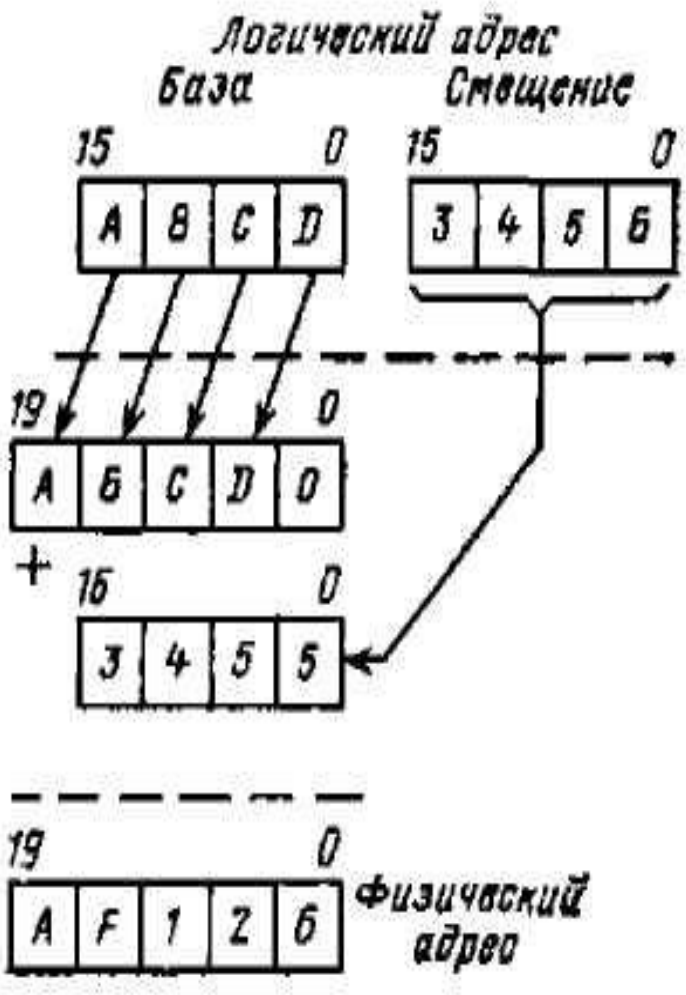


- Физическая ячейка памяти может принадлежать одному или нескольким сегментам.
-
- Сегментные регистры процессора:
- **CS – code segment** – сегмент команд. Определяет базовый адрес сегмента памяти, в котором хранится выполняемая программа,
- **DS – data segment** – сегмент данных. Определяет базовый адрес сегмента памяти, в котором хранятся обрабатываемые данные,
- **SS – stack segment** – сегмент стека. Определяет базовый адрес сегмента памяти, доступ к которому возможен с помощью стековой адресации,
- **ES – extra segment** – дополнительный сегмент. Определяет базовый адрес сегмента памяти, в котором наряду с DS могут храниться обрабатываемые данные,

Физические и логические адреса. Сегмент, смещение.

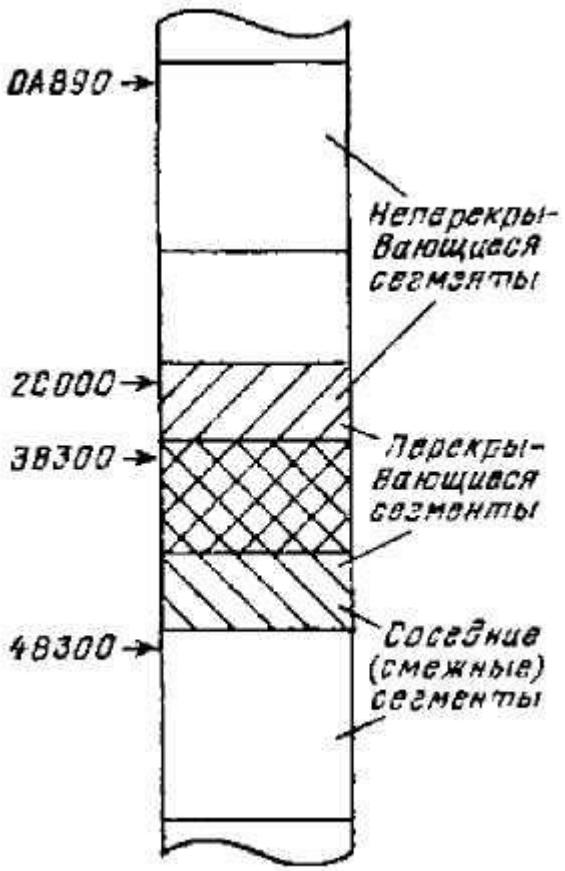
- Каждая ячейка памяти имеет два адреса: физический и логический.
- ***Физический адрес*** представляет собой, например, 20-битное значение в диапазоне от 00000h до FFFFFh, которое однозначно идентифицирует положение каждого байта в пространстве памяти 1М байт. Именно физический адрес выдается на шину адреса в начале каждого цикла шины, связанного с обращением к памяти.
- Программы оперируют не физическими, а ***логическими адресами***, что позволяет разрабатывать их без априорного знания того, как конкретная программа размещается в памяти. Кроме того, упрощается динамическое управление памятью, что имеет большое значение в мультипрограммной среде.
- Логический адрес в этом случае состоит из двух 16-битных беззнаковых значений: ***базового*** (начального) адреса сегмента, который называется также просто базой или сегментом, и внутрисегментного адреса или ***смещения***.
- Для любой ячейки памяти база идентифицирует адрес первого байта, сегмента, которому принадлежит эта ячейка, т.е. начало сегмента, в котором расположена ячейка, а смещение определяет расстояние в байтах от начала сегмента до этой ячейки.
- Нулевое смещение имеет байт с наименьшим внутрисегментным адресом, а максимальное смещение равно FFFFh.

Физические и логические адреса. Сегмент, смещение.



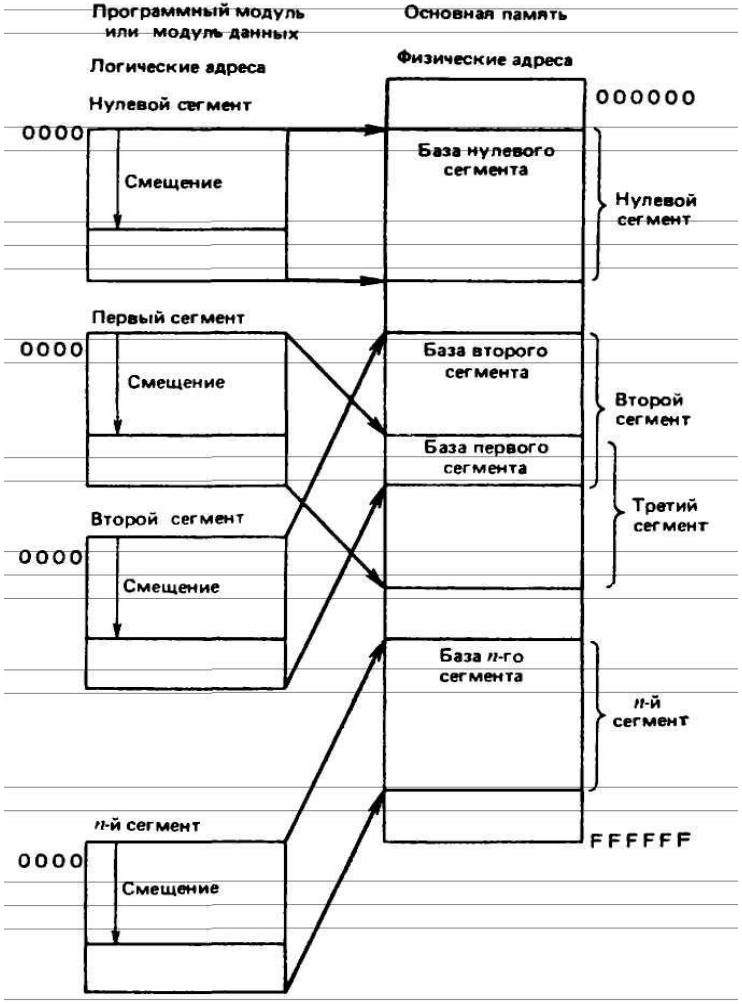
- Перед обращением к памяти для выборки команды или считывания/записи данных, необходимо преобразовать логический адрес (сегмента и смещения) в физический адрес.
- Для этого база сегмента сдвигается влево на 4 бита и суммируется со смещением.
- Принцип формирования физического адреса предполагает, что абсолютные адреса сегментов оканчиваются четырьмя двоичными нулями, например FFFF0h.
- Такие адреса называются границами сегментов, а четыре старшие FFFF0h 16-ричные цифры, (т.е. содержимое сегментного регистра) - номерами параграфов.
- Термины «база», «сегмент», «номер параграфа» являются по существу синонимами.
- Возникающий при суммировании перенос из старшего бита игнорируется. Это приводит к так называемой кольцевой организации памяти — за ячейкой с максимальным адресом FFFFFh следует ячейка с нулевым адресом.

Сегментная адресация.



- Команды всегда выбираются из текущего сегмента кода - базовый адрес сегмента находится в регистре CS, а смещение - в регистре PC.
- Стековые команды всегда обращаются к текущему сегменту стека: базовый адрес находится в регистре SS, а смещение - в регистре SP.
- Считается, что большинство переменных (операндов) находится в текущем сегменте данных, адресуемом регистром DS, но программист может заставить МП обращаться к переменной, находящейся в другом сегменте (под «другим» здесь понимается сегмент, базовый адрес которого находится не в регистре DS, а в каком-то другом сегментном регистре).
- Смещение переменной вычисляет операционное устройство в соответствии с определенным в команде режимом адресации. Результат этого вычисления называется *эффективным* или *исполнительным адресом* (EA) операнда в памяти.

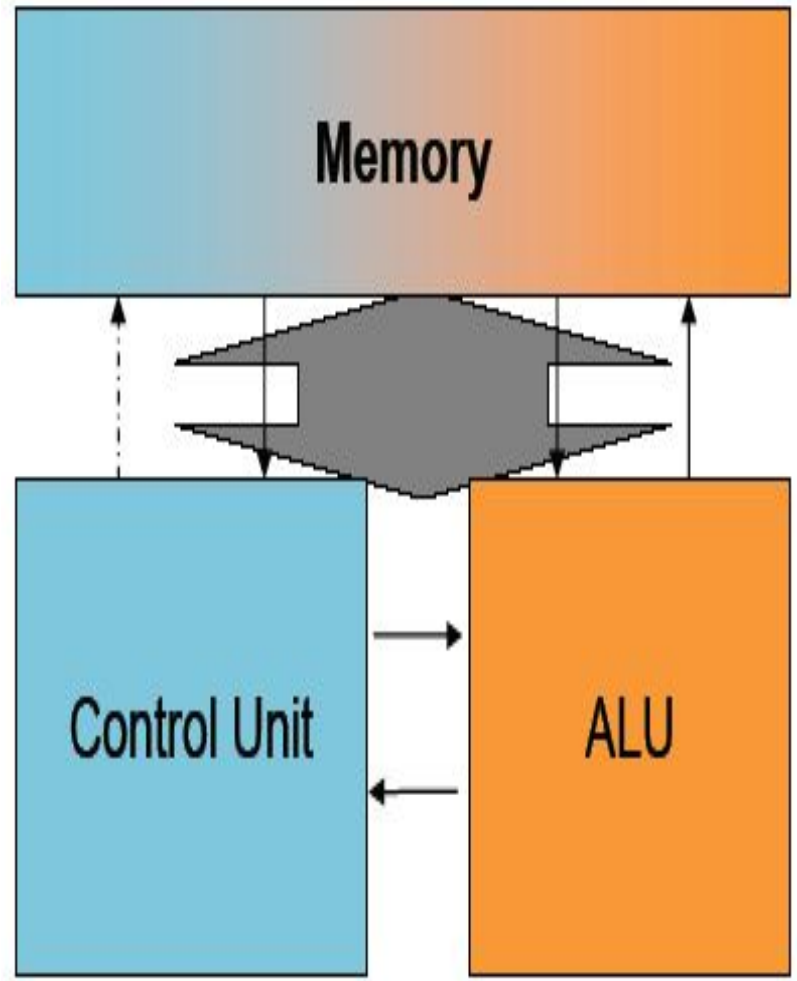
Защита памяти

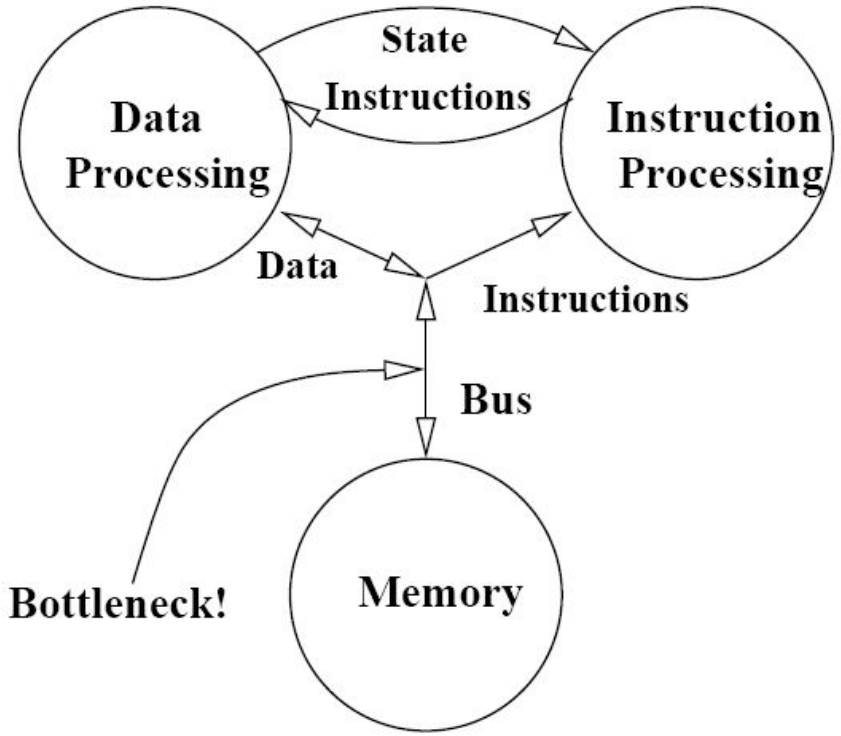


Взаимосвязь физических и логических адресов памяти

- Ошибочный переход к области данных может привести к их уничтожению. Кроме того, опасна запись данных в область программы. Для предотвращения таких ситуаций необходима защита памяти.
- Для ее осуществления кроме базового адреса сегмента указываются размер и *атрибут сегмента*.
- В этом случае ситуация, когда величина смещения превышает размер сегмента, считается нарушением из-за возможности проникновении в область соседнего сегмента.
- Атрибуты сегментов могут быть самыми разными; из них широко используются следующие: разделение на системную область и область пользователя; разделение на область программ и область данных; в случае работы с областью программы разделение на области, допускающие только ее выполнение или также и считывание; в случае работы с областью данных разделение на области, допускающие только считывание данных или также запись.

«Бутылочное горлышко» архитектуры Фон-Неймана

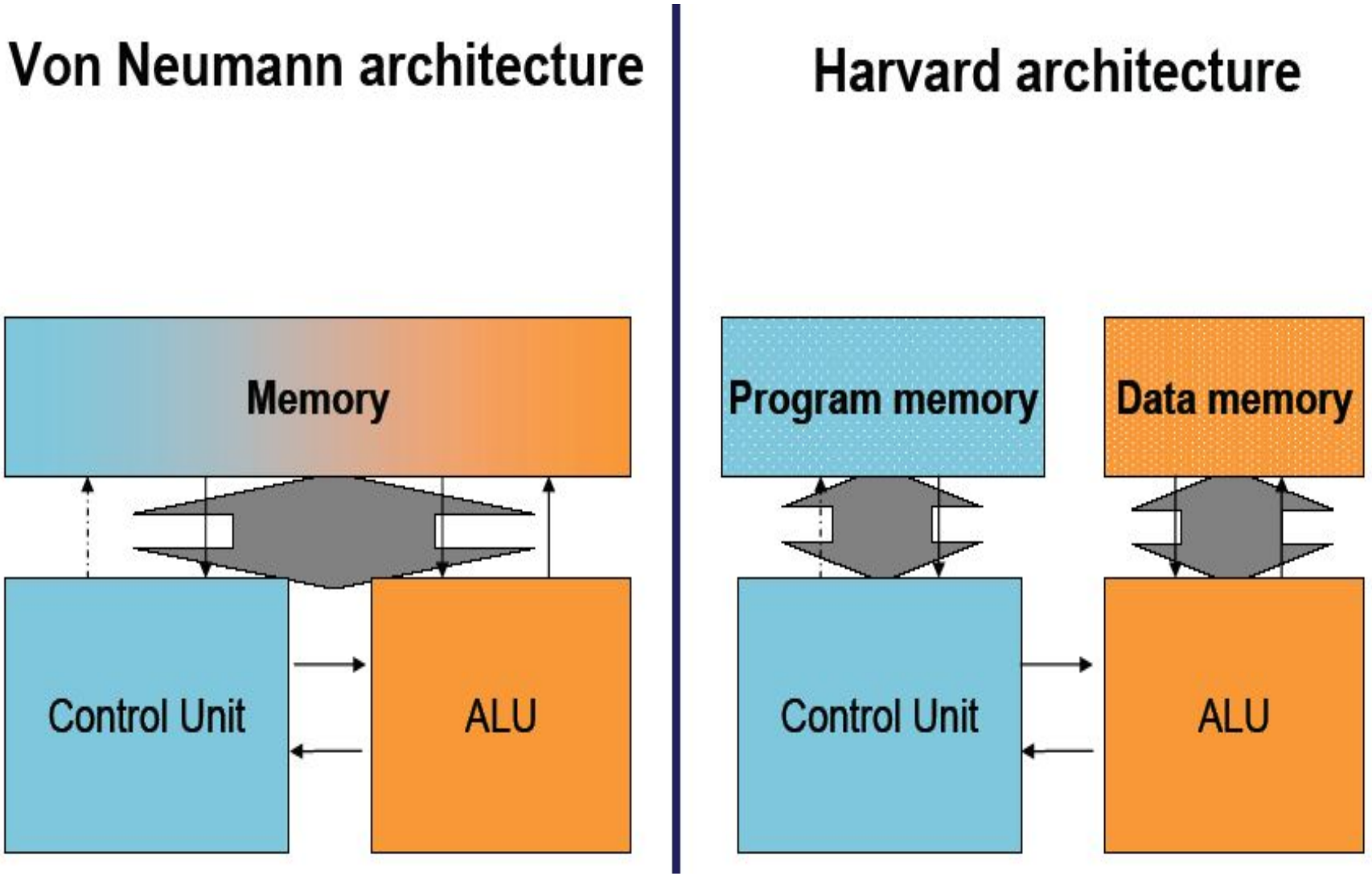




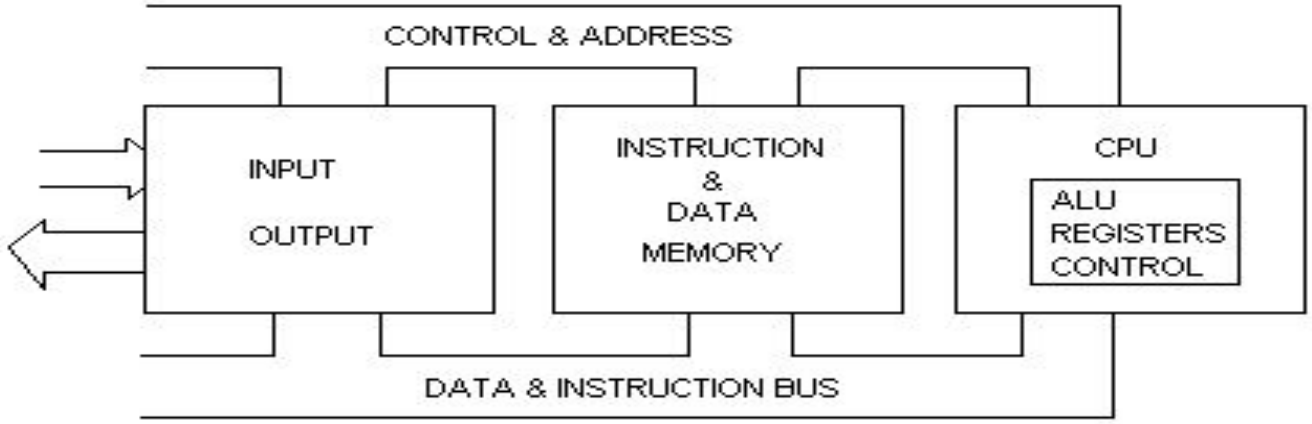
- A common bus is used for data as well as instructions.
- The system can become ‘bus bound’.

Гарвардская архитектура почти не использовалась до конца 70-х.

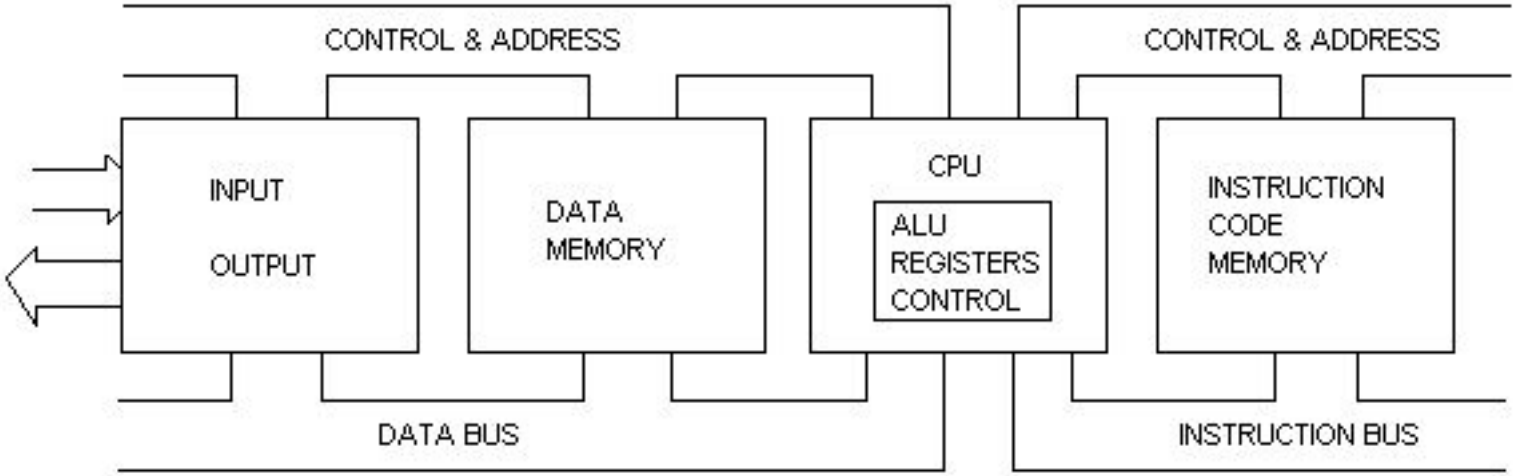
Гарвард vs Фон Нейман.



Гарвард vs Фон Нейман.

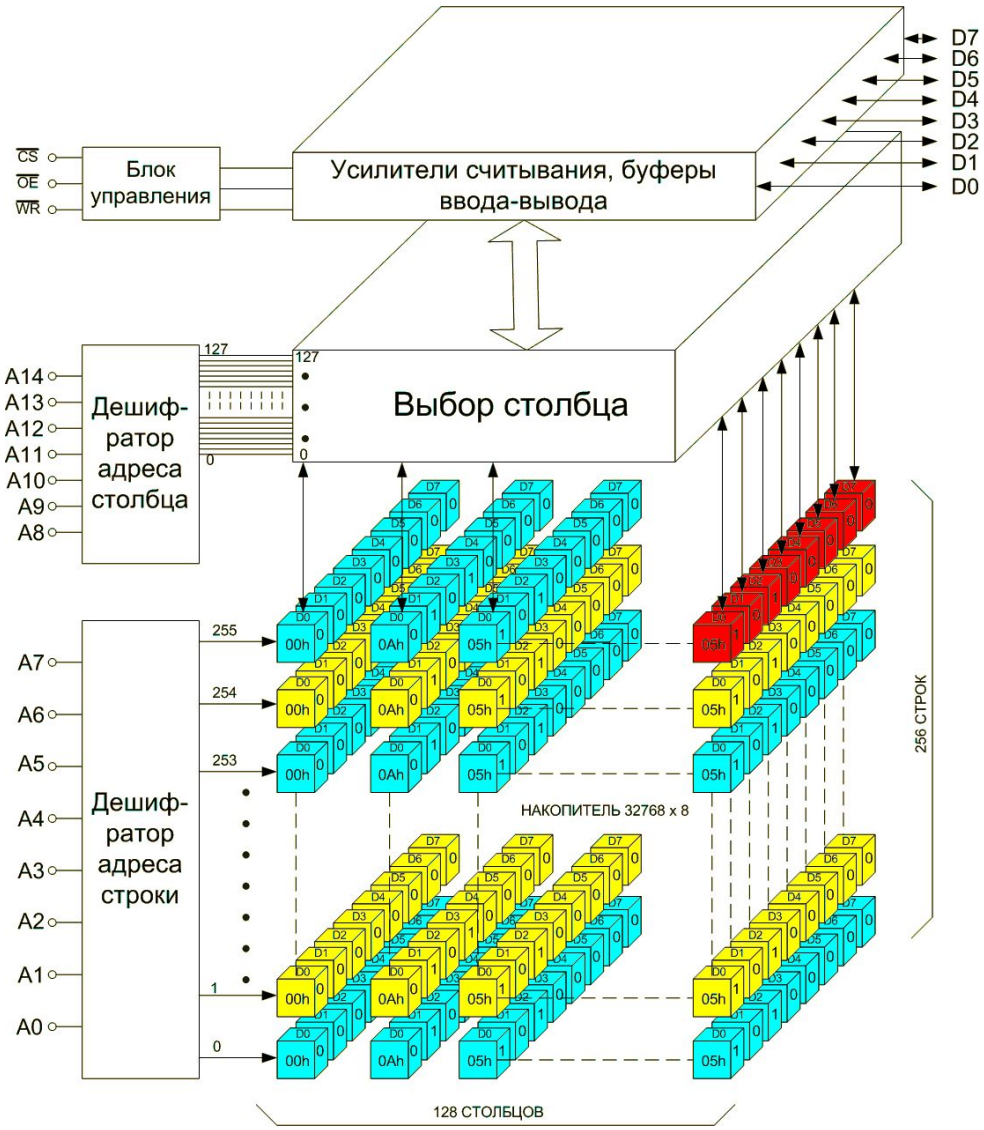


VON NEUMANN ARCHITECTURE MICROPROCESSOR



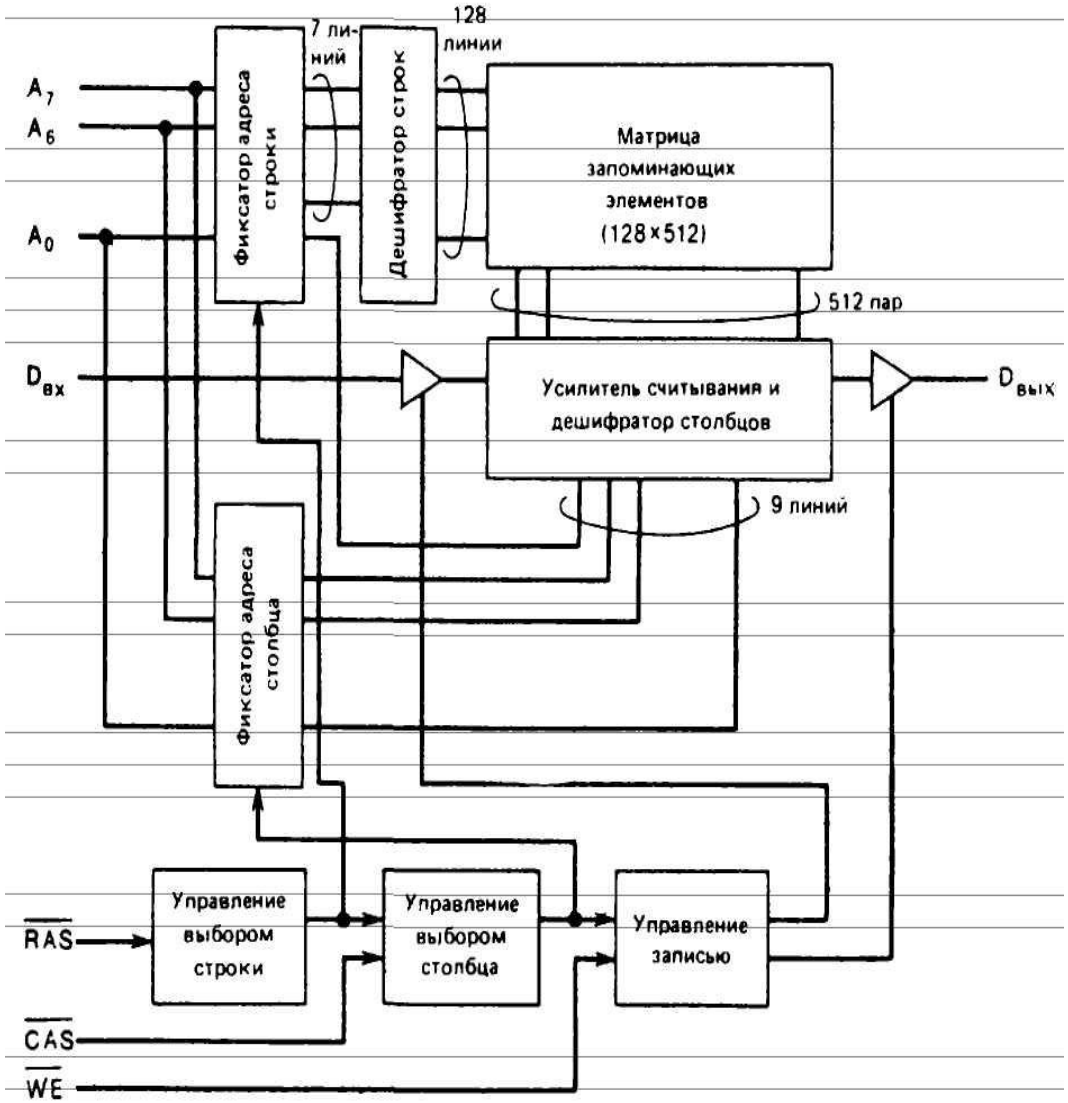
HARVARD ARCHITECTURE MICROPROCESSOR

Статическое ОЗУ 32к*8

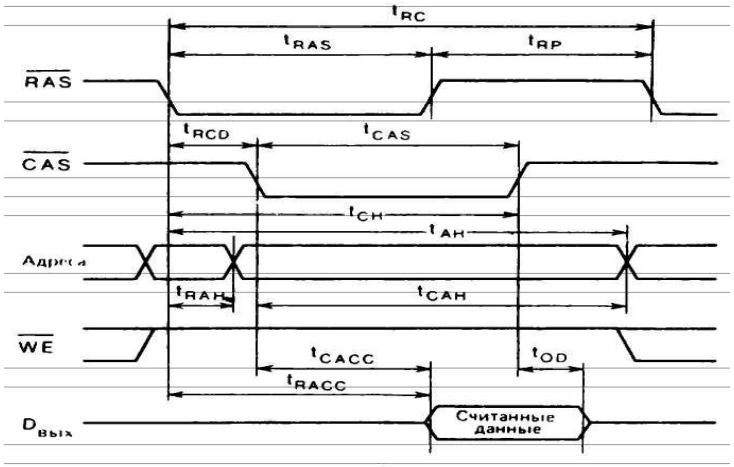


- Структура статического запоминающего устройства объемом 32кбайт, организацией 32к * 8бит.

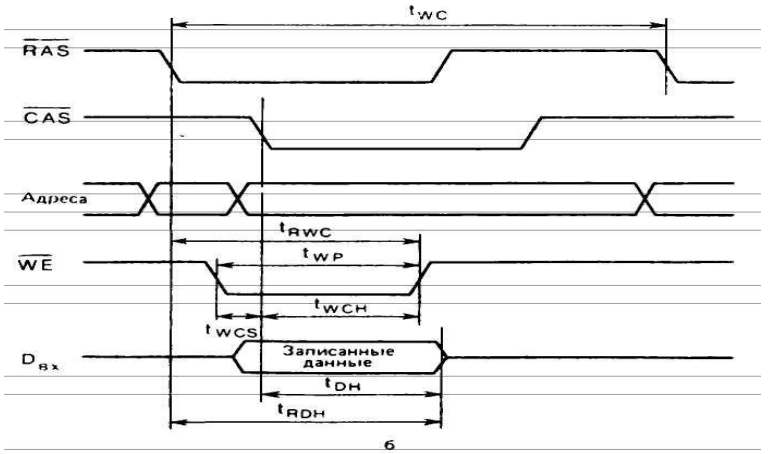
Динамическое ОЗУ.



Временные диаграммы

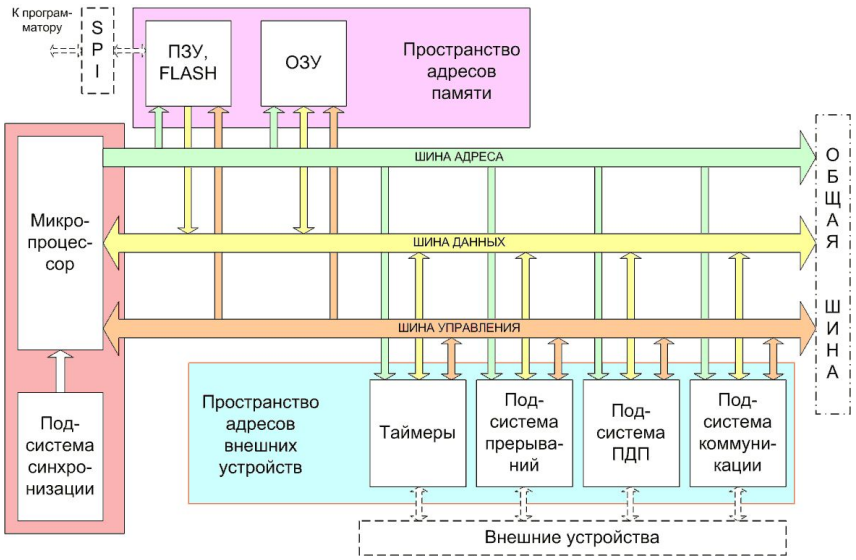


считывания



записи

Порты ввода-вывода.



При обращении к периферийным устройства имеются две области адресов, предназначенные для операции ввода и вывода данных.

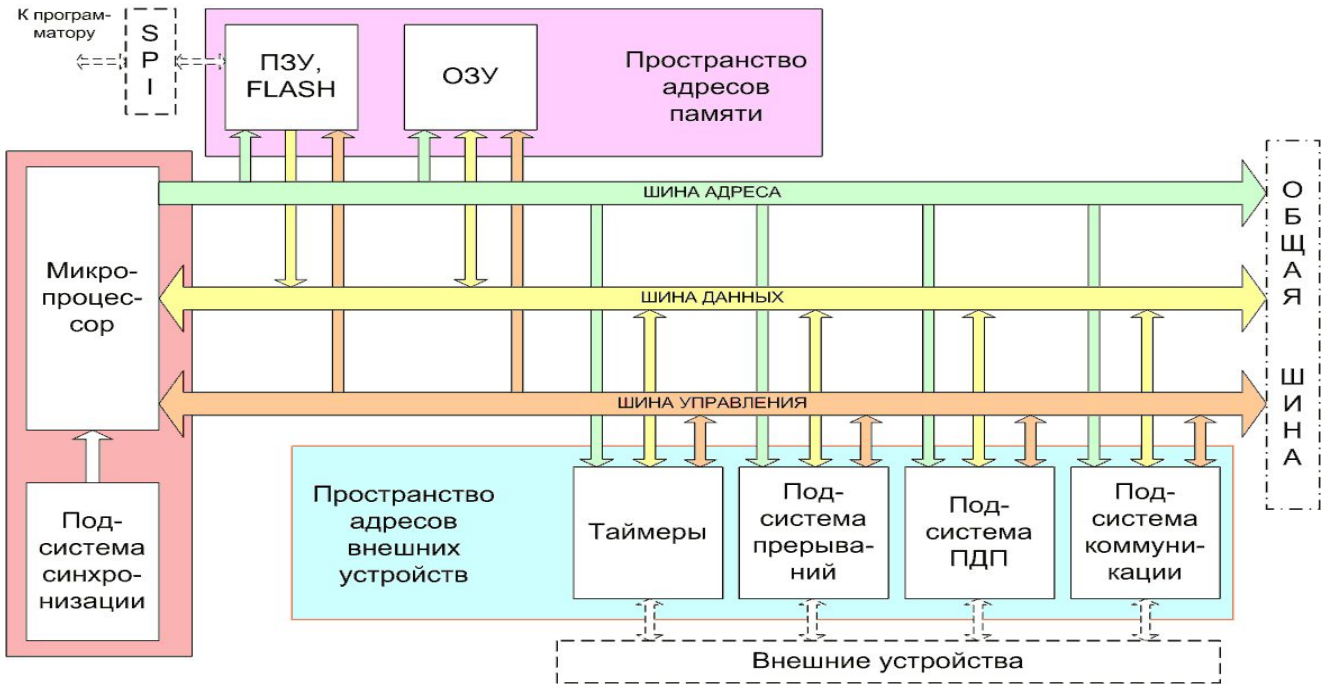
- Первая из них называется адресами *портов ввода*,
- вторая — адресами *портов вывода*.

- Считывание данных из порта ввода называется *вводом данных*, а запись в порт вывода - *выводом данных*.
- Для обращения к портам ввода-вывода существуют следующие два способа:
  - 1) изолированный ввод-вывод;
  - 2) ввод-вывод с отображением на память.
- При первом способе память и порты ввода-вывода используются как различные устройства. При втором – часть адресного пространства памяти отводится для адресации внешних устройств.

Порты ввода-вывода.

- Как правило, на практике используется первый способ.
- Для осуществления ввода-вывода при этом способе применяются *специальные команды ввода-вывода*. Они отличны от команд обеспечивающих работу с памятью. Порту ввода и порту вывода присваиваются номера (адреса), не зависящие от адресов памяти. Для передачи адреса порта также используется шина адреса.
- Использование специальных команд обращения к устройствам ввода-вывода (УВВ) *автоматически* активирует специальные сигналы, передаваемые по шине управления, которые указывают, что происходит обращение к УВВ, а не к памяти.
- Микропроцессорные устройства и МПС содержат различные средства ввода-вывода информации. Периферийные устройства подсоединяются к шинам через программируемые периферийные адаптеры, осуществляющие передачу информации в параллельном или последовательном кодах.
- Наличие программно-настраиваемых адаптеров делает весьма гибкой и функционально богатой систему ввода-вывода информации в МПС.
- Порты ввода-вывода в самом общем случае можно представить регистры с определенным алгоритмом доступа.

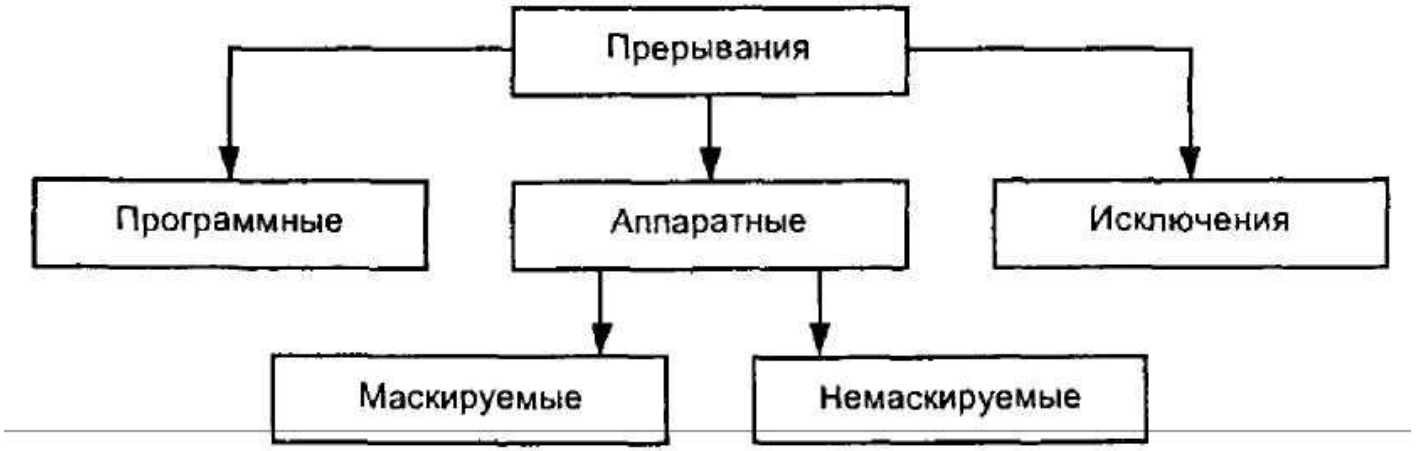
Таймеры



- **Таймеры** предназначены для
  - формирования временных интервалов,
  - подсчета входных сигналов,
  - деления частоты.
- Применяются, например, при настройке скорости приема-передачи последовательного канала связи.
- Размещаются, как правило, в пространстве адресов ввода-вывода.

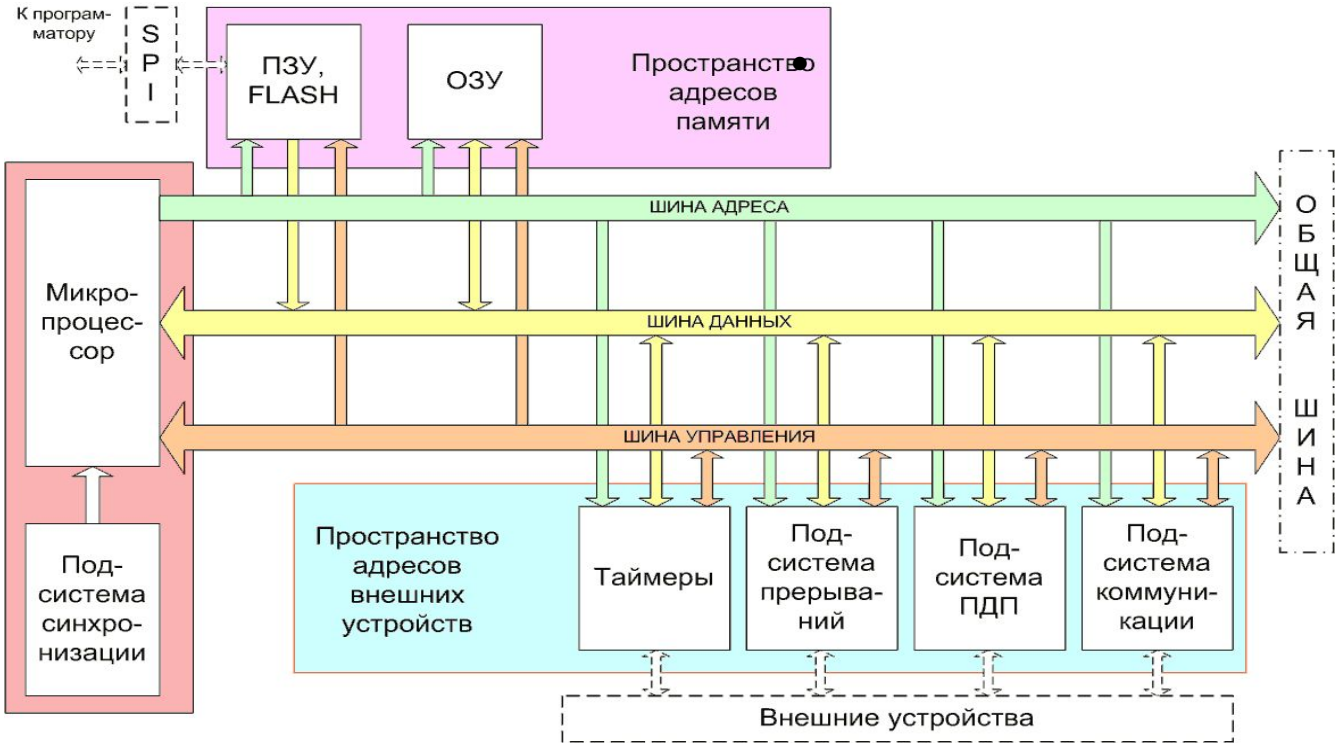
Прерывания

- *Прерываниями* (interruption) называются ситуации, вызывающие нарушение естественной последовательности выполнения программы.
- Они возникают при поступлении соответствующих команд (***программные прерывания***) или сигналов - запросов от внешних устройств (***аппаратные прерывания***).
- *Исключениями* (exception) являются нештатные ситуации (ошибки), возникающие при работе процессора. При выявлении таких ошибок соответствующие блоки, контролирующие работу процессора, вырабатывают внутренние сигналы запроса, обеспечивающие вызов необходимой подпрограммы обслуживания.



Классификация прерываний и исключений

Подсистема прерываний



- **Контроллер прерываний** осуществляет прием сигналов - запросов аппаратных прерываний от различных внешних устройств, генерирование сигнала запроса прерывания процессору, генерирование векторов аппаратных прерываний, осуществляет механизм приоритета прерываний.

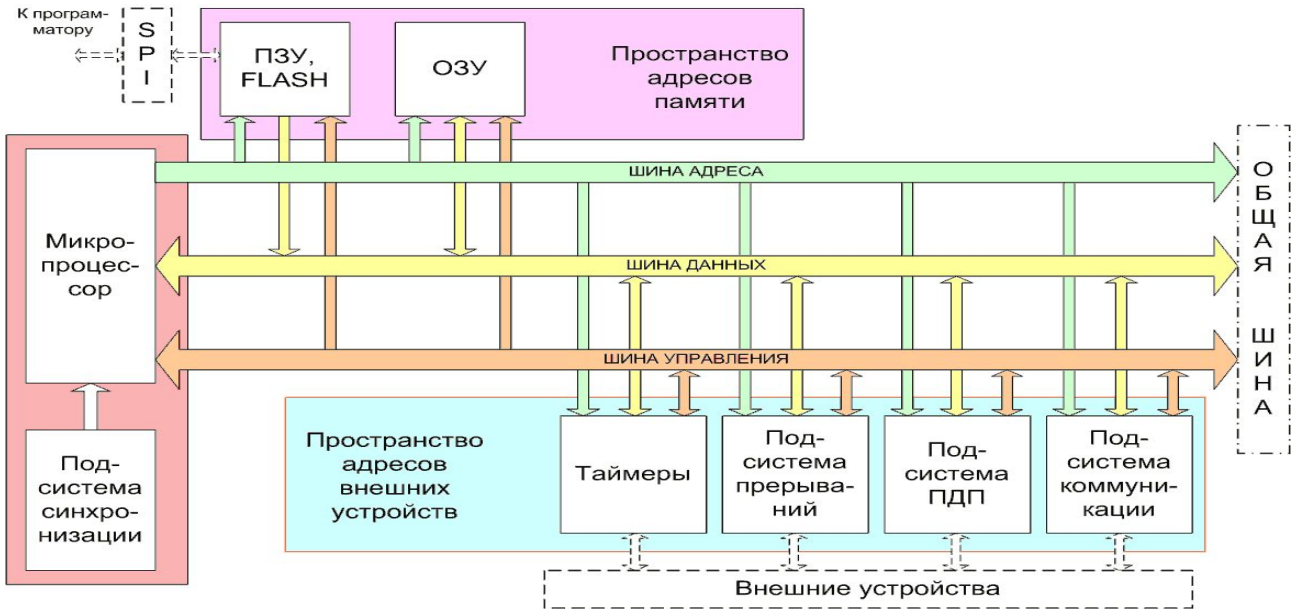
Подсистема прерываний

- Во всех этих ситуациях микропроцессор завершает выполнение очередной команды и заносит в стек текущее содержимое программного счетчика (PC – program counter), которое является адресом возврата к прерванной программе после реализации подпрограммы обслуживания прерывания, и содержимое регистра состояний (SR – status register).
- Если запрос прерывания поступает от внешнего устройства, то процессор формирует сигнал подтверждения прерывания, который информирует устройство, инициировавшее прерывание, о том, что начато обслуживание данного запроса.
- Затем в счетчик PC загружается вектор прерывания - начальный адрес соответствующей подпрограммы обслуживания. Векторы являются адресами входов в подпрограммы обслуживания и хранятся в таблице векторов прерываний, которая обычно записывается в ОЗУ или поступают по шине данных от контроллера прерываний. Адреса могут быть также фиксированы на аппаратном уровне, т.е. их адреса поменять невозможно.
- Завершается подпрограмма обслуживания прерывания специальной командой возврата из прерывания, по которой происходит выборка из стека содержимого PC и SR и загрузка его обратно в эти регистры, обеспечивая возвращение к выполнению прерванной программы.

Прерывания

- Аппаратные прерывания могут быть ***маскируемые*** или ***немаскируемые****.*
- Запросы **маскируемых прерываний** обслуживаются только в том случае, если они были разрешены с помощью специальных команд. Можно разрешить или запретить обслуживание поступивших аппаратных прерываний при выполнении определенных программ или их фрагментов.
- При одновременном поступлении нескольких запросов обслуживание запросов реализуется в соответствии с их ***приоритетом***. В ряде микропроцессорных систем для обеспечения приоритетного обслуживания запросов от многих внешних устройств включаются специальные микросхемы - *контроллеры прерываний.* Некоторые типы микропроцессоров имеют внутренние контроллеры для организации приоритетных прерываний.
- **Немаскируемые запросы** на прерывания обслуживаются в первоочередном порядке и не могут быть маскированы. Обычно микропроцессор имеет один вход для подачи немаскируемых запросов, которые формируются при возникновении каких-либо аварийных ситуаций, например, сбоя по питанию.
- Причинами ***исключений*** могут быть различные ошибки и нештатные ситуации, возникающие при работе системы. Различные типы микропроцессоров контролируют разные варианты такого рода ситуаций. Типичными причинами исключений являются, например, использование нулевого делителя при выполнении команды деления (деление на 0); выборка неправильного кода команды; выход за границы разрешенного сегмента памяти, поступление команд, выполнение которых запрещено при данном режиме функционирования микропроцессора и ряд других. Соответствующие причины возникновения ситуации исключений зависят от конкретных типов микропроцессоров.
- В простейших микропроцессорах нет механизма обслуживания исключений

Прямой доступ к памяти

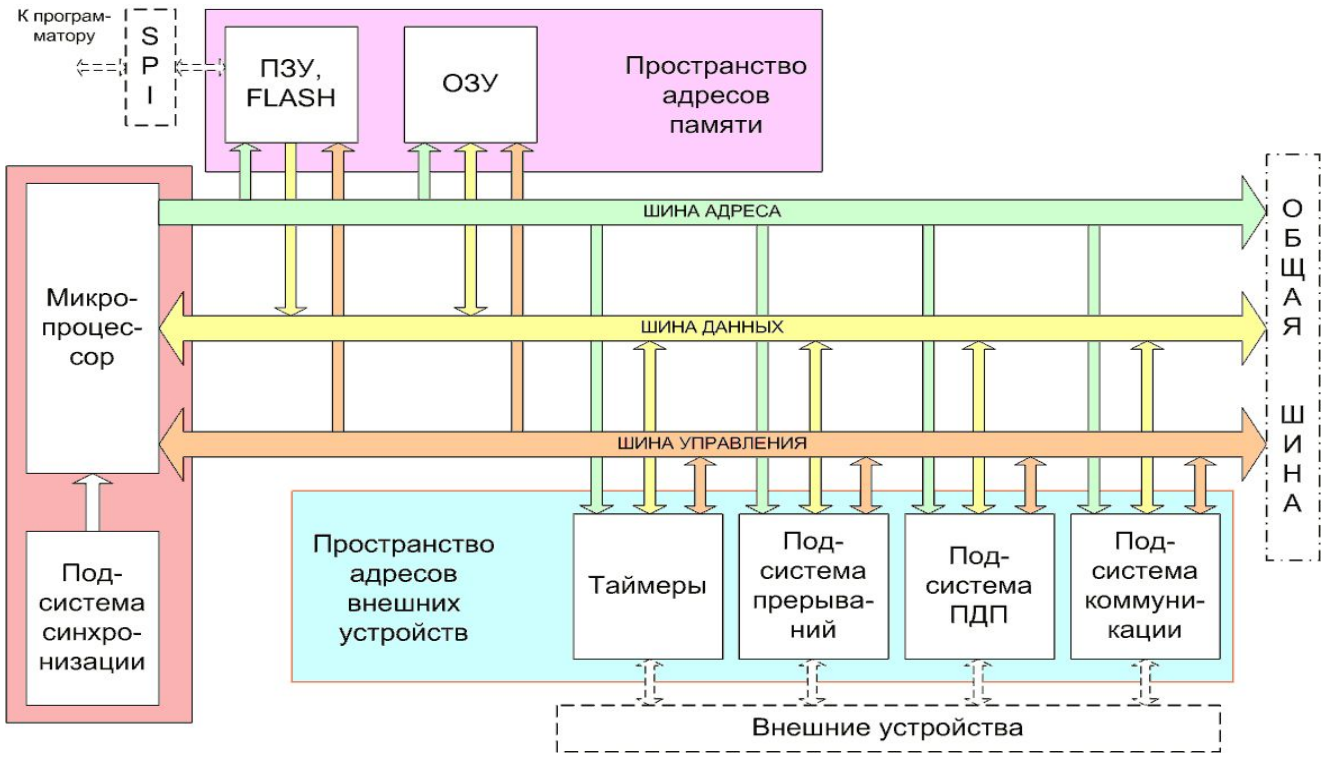


- **Контроллер прямого доступа к памяти** осуществляет механизм прямого доступа к памяти **(DMA, ПДП).** Режим прямого доступа к памяти DMA (Direct Memory Access) используется, если необходимо произвести пересылку значительного массива информации между ОЗУ и каким-либо внешним устройством, которое подает в систему соответствующий запрос. Реализация такой пересылки с помощью соответствующей программы обмена требует выполнения отдельной команды пересылки для передачи каждого байта или слова. При этом необходим определенный объем памяти для хранения программы и требуется значительное время для ее выполнения.

Механизм ПДП

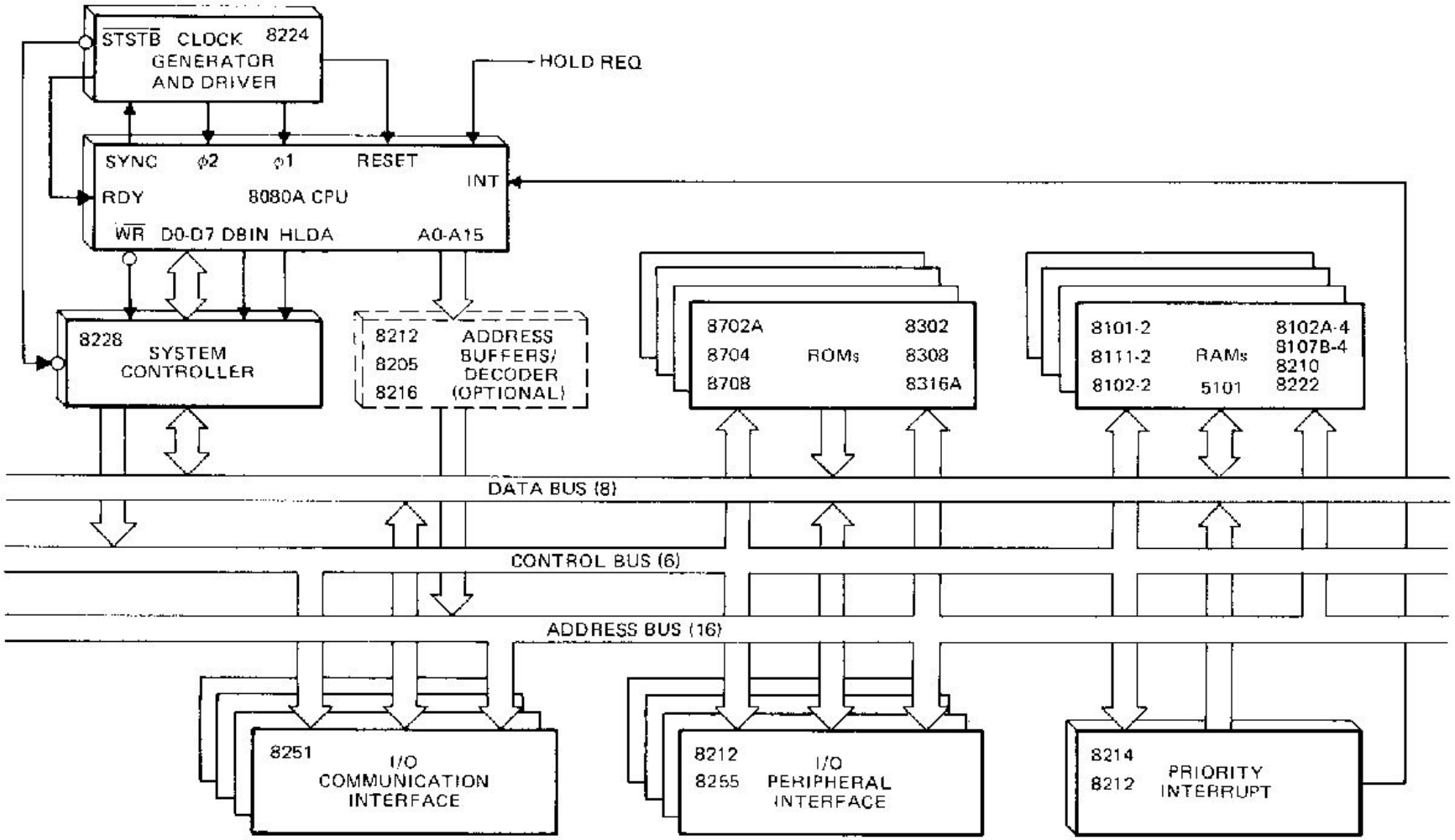
- В большинстве современных микропроцессорных систем пересылка массивов информации обеспечивается с помощью специальных устройств - контроллеров DMA, которые реализуют режим прямого доступа к памяти.
- При поступлении запроса от внешнего устройства контроллер выдает соответствующий сигнал микропроцессору. Микропроцессор завершает очередной цикл обмена по системной шине и отключается от нее, то есть переводит свои выводы, подключенные к шинам в отключенное (высокоимпедансное) состояние. При этом микропроцессор выдает контроллеру DMA сигнал разрешения на осуществление прямого доступа.
- Получив сигнал разрешения, контроллер принимает на себя управление системой. Он выдает на шину адреса ячеек ОЗУ, с которыми выполняется текущий цикл обмена, формирует необходимые сигналы, определяющие режим работы ОЗУ (запись или считывание) и интерфейсного устройства, через которое производится пересылка информации (ввод или вывод).
- Взаимодействие между микропроцессором и контроллером DMA производится по соответствующим линиям шины управления.
- Предварительно контроллер DMA должен быть запрограммирован для выполнения указанных функций. В соответствующие регистры контроллера DMA записываются начальные адреса массивов памяти, с которых начинается процесс обмена, и размеры массивов, подлежащих пересылке.
- Обычно контроллер DMA обслуживает запросы от нескольких внешних устройств, поэтому он программируется на реализацию определенного приоритета обслуживания в случае одновременного поступления нескольких запросов. Программирование контроллера производится путем записи необходимых управляющих слов в управляющие регистры контроллера DMA.

Подсистема коммуникации

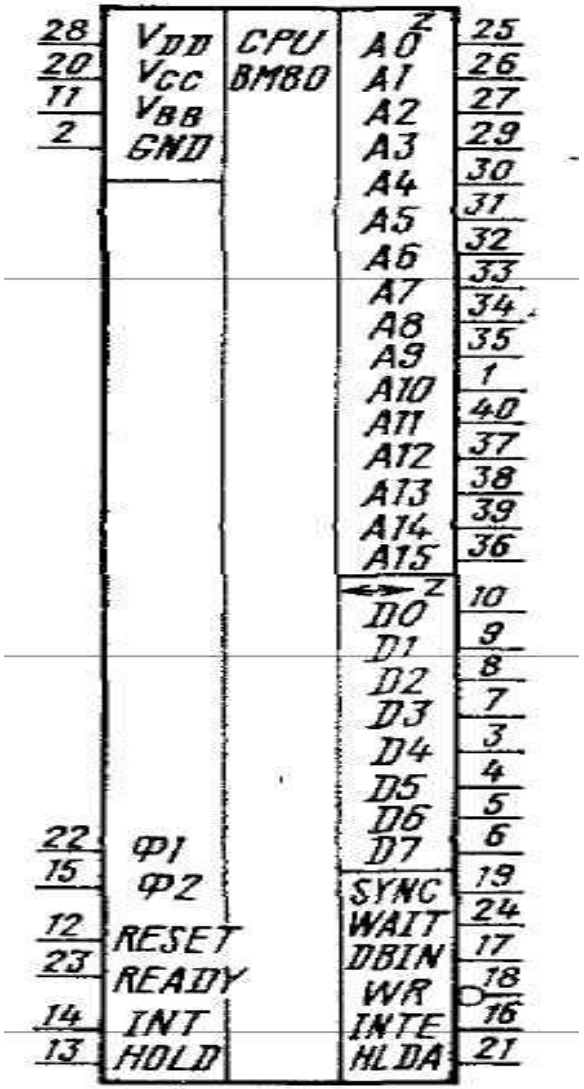


- **Подсистема коммуникации** осуществляет связь МП системы с удаленными объектами.
- В нее могут входить параллельные периферийные адаптеры, последовательные адаптеры, сетевые адаптеры и т.п.

Архитектура Фон Неймана (Принстонская)

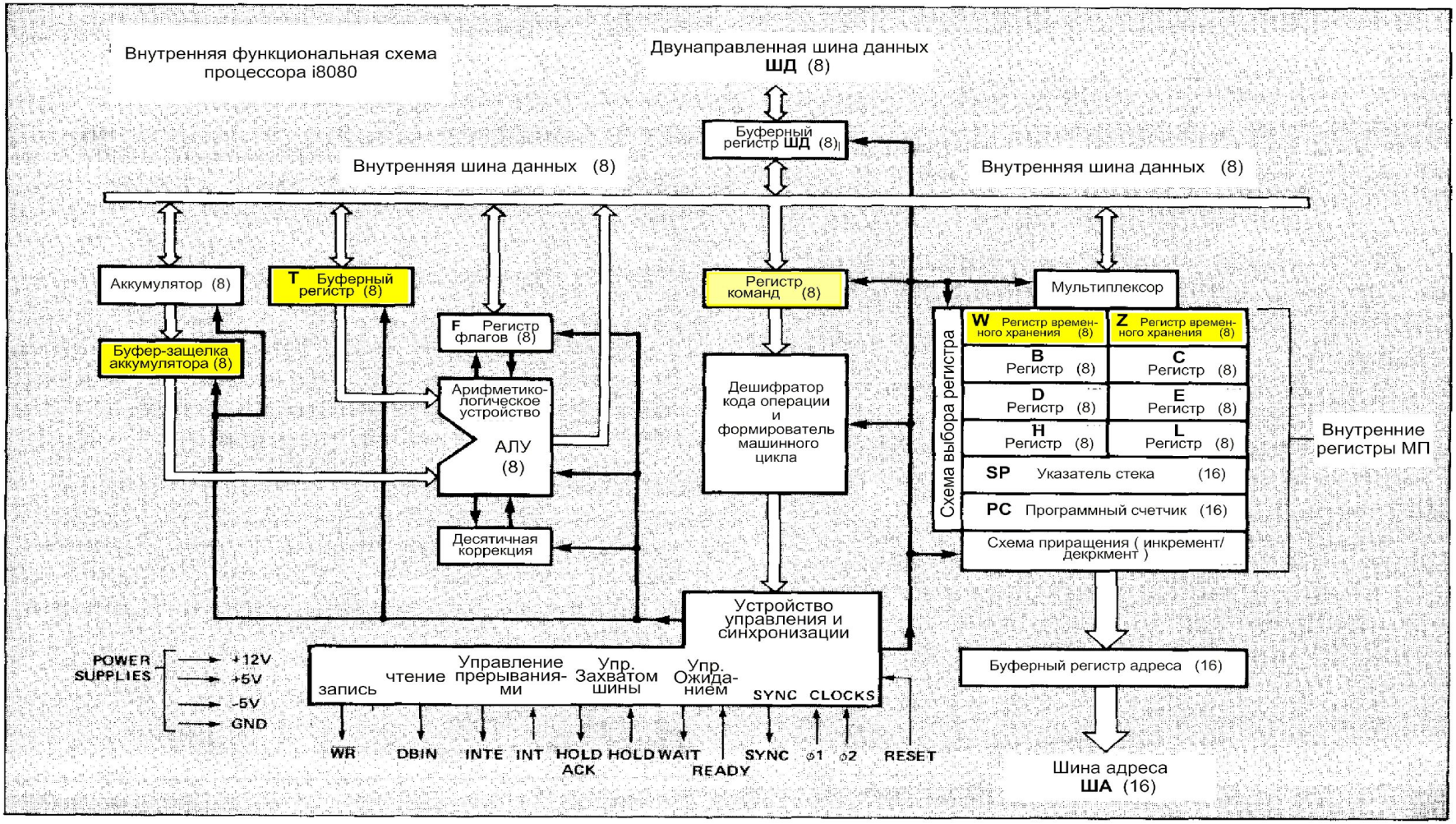


Внешние сигналы микропроцессора на примере i8080



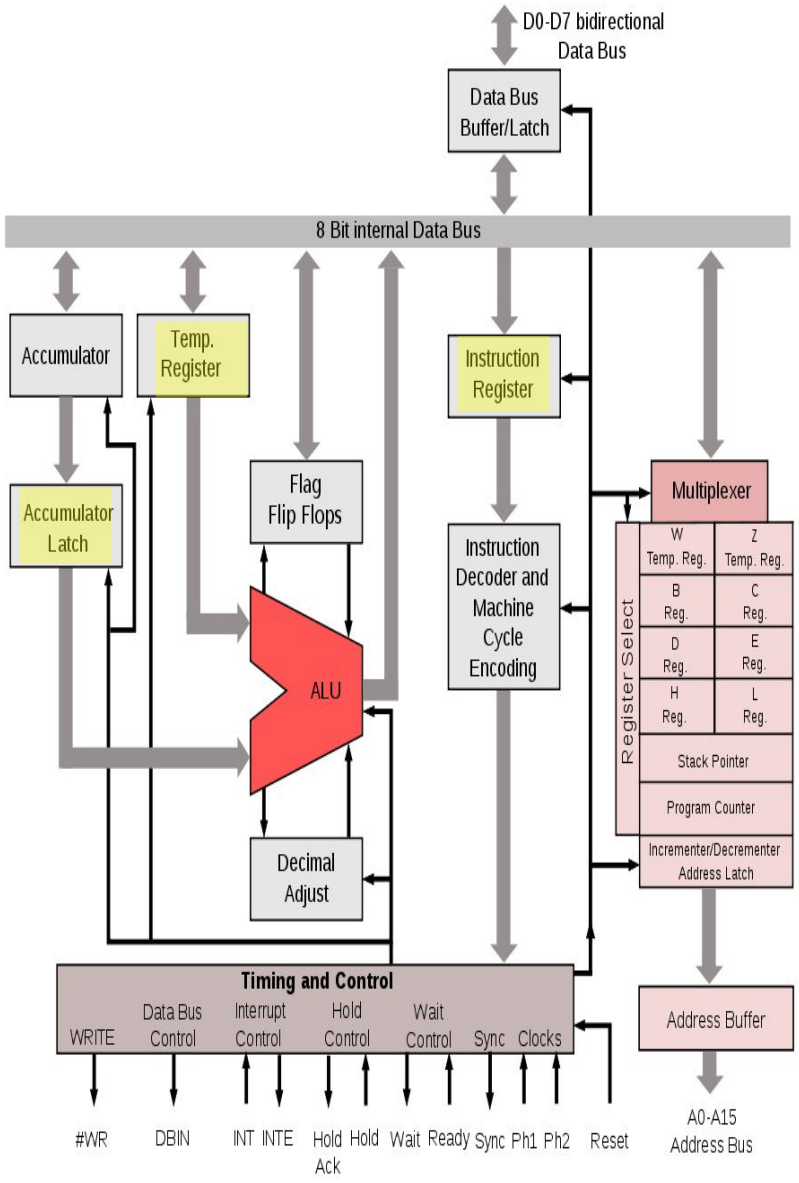
- A0…A15 - шина адреса. Однонаправленная с третьим состоянием. Выход
- D0…D7 - двунаправленная шина данных с третьим состоянием.
- WR - запись - выход строба выдачи данных.
- DBIN - чтение - выход строба приема данных.
- INT - вход - запрос на прерывание от ВУ.
- INTE - выход - разрешение прерывания
- SYNC - выход - признак начала нового машинного цикла.
- WAIT - выход - признак режима ожидания.
- HOLD - вход - запрос захвата шины.
- HLDA - выход - сигнал подтверждения захвата шины.
- READY - вход - готовность.
- RESET - вход - сброс.
- Ф1, Ф2 - вход – тактовый сигнал синхронизации.
- Vdd, Vbb,Vcc – входы напряжений питания.
- GND - обший провод - земля.

Структурная схема uP



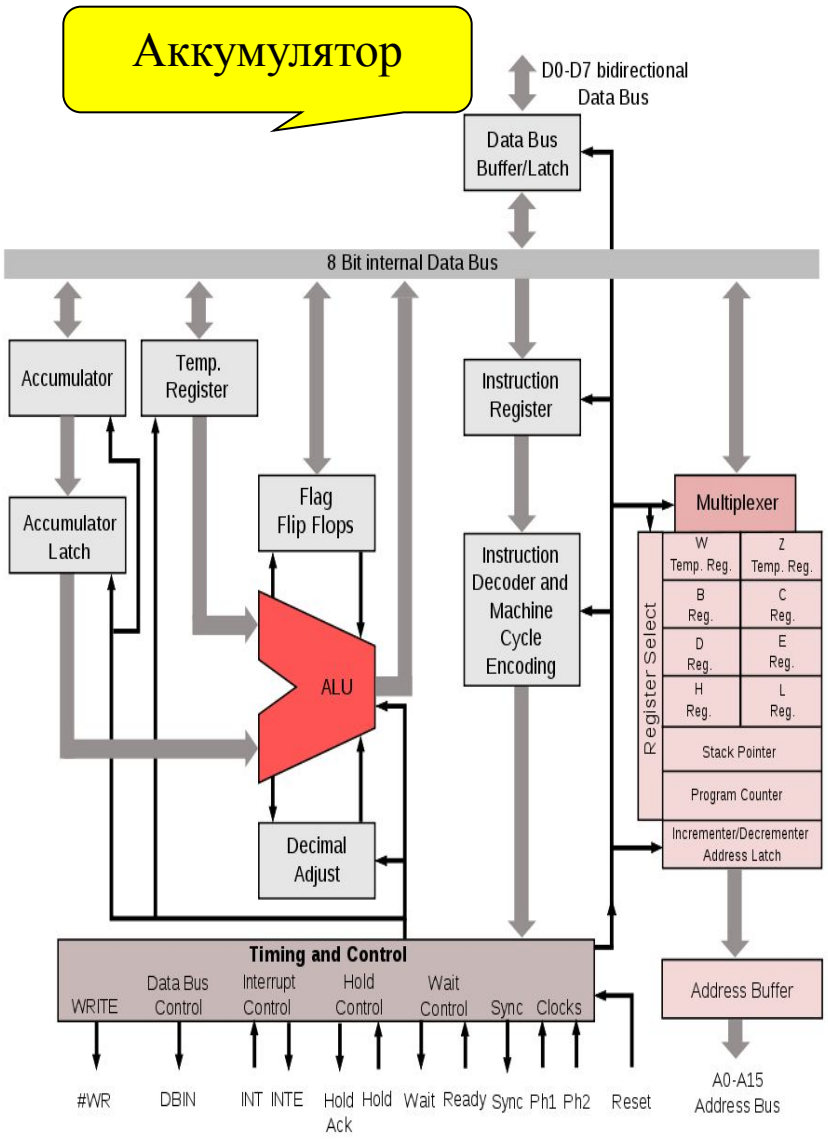
- Арифметико-логическое устройство (АЛУ),
- блок регистров, буферные схемы,
- управляющее устройство.

Регистры uP



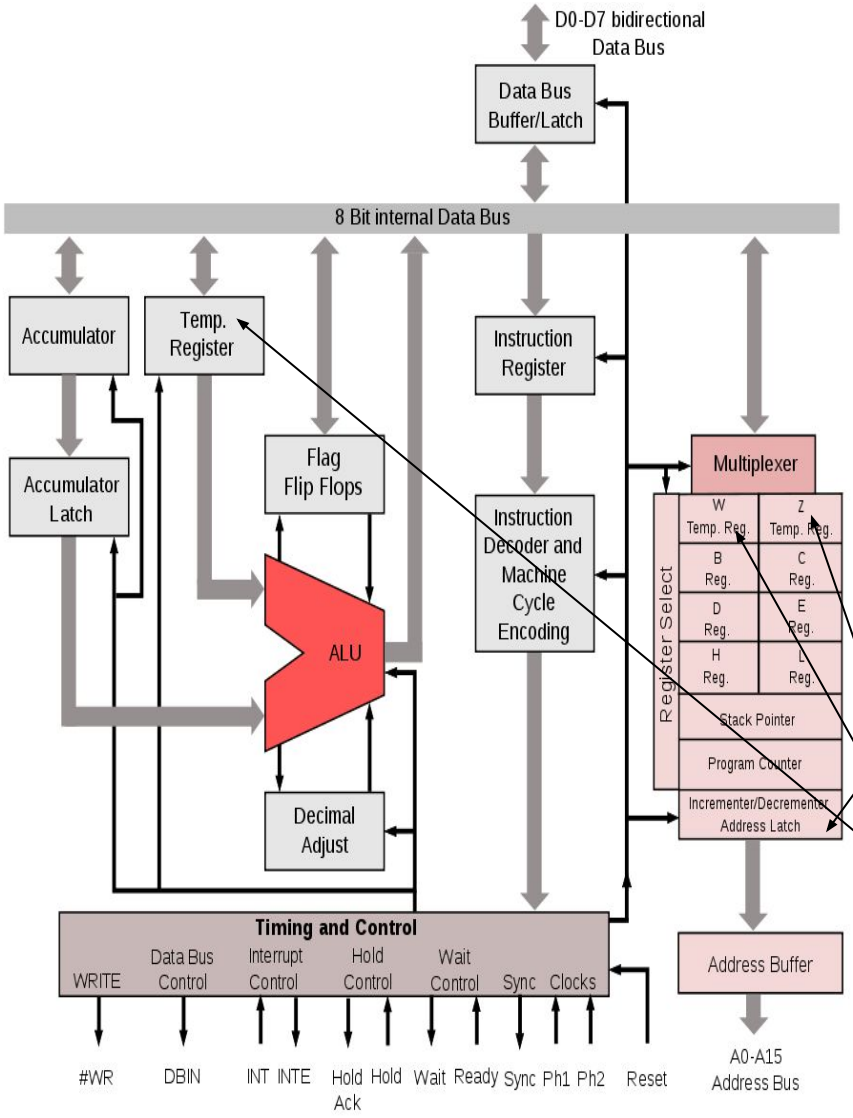
- Блок регистров содержит 8-разрядные программно-доступные регистры: A (регистр-аккумулятор); B, C, D, E, H, L (регистры общего назначения - РОН); F (регистр признаков-флагов);
- 16-разрядные специализированные регистры; программный счетчик (PC); регистр-указатель стека (SP); сдвоенный регистр косвенного адреса HL (H-регистр старшего полуадреса, L-регистр младшего полуадреса).
- Имеются непосредственно недоступные программе (отмечены желтым цветом) регистры: 8-разрядный регистр команд; 8-разрядные регистры временного хранения T, W, Z; 16-разрядный регистр адреса PA.
- Содержимое пар регистров B и C, D и E, H и L можно использовать в качестве слов двойной длины, т.е. 16-разрядных слов.

Аккумулятор (А)



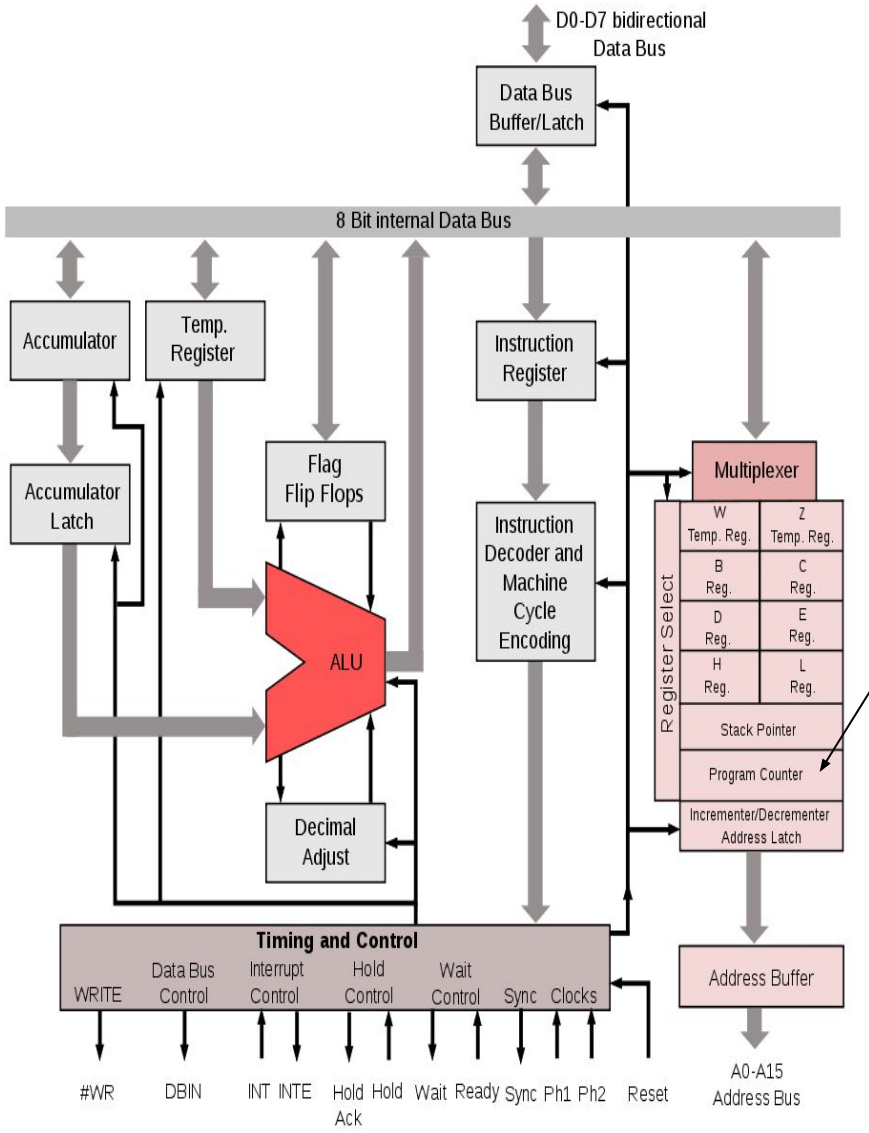
- **Аккумулятор (А)** используется в качестве источника одного из операндов и приемника результата выполнения операции, в нем хранятся результаты текущих арифметических и логических операций, выполняемых в АЛУ.
- По результату выполнения операции, помещенному в аккумулятор, часто принимается решение о дальнейшем ходе вычислительного процесса в микро-ЭВМ, т.е. осуществляется ветвление.
- *Разрядность аккумулятора определяет разрядность информационного слова микро-ЭВМ.*

Регистры общего назначения (РОН)



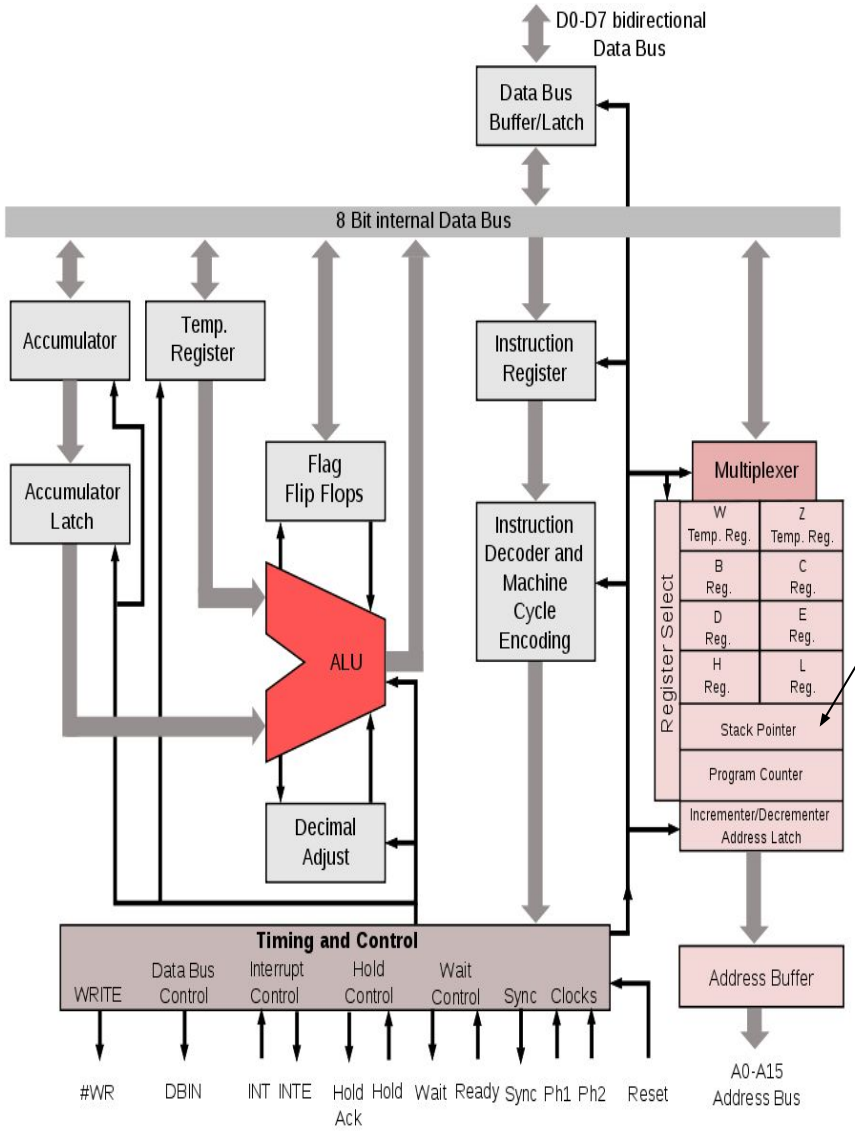
- для хранения операндов, промежуточных и конечных результатов, а также адресов при косвенной адресации данных.
- Позволяет при выполнении команд уменьшить количество обращений к памяти и повысить скорость выполнения программы.
- *Схема приращений* (инкремента/ декремента), выполняет операцию прибавления/ вычитания единицы над содержимым регистров. Схема инкремента/декремента позволяет реализовать процедуры автоматического задания приращений при операциях с адресами как в указателе стека (регистре SP), так и в программном счетчике (регистре PC).
- Регистры T, W, Z служат для временного хранения данных. Они являются служебными регистрами процессора и программно недоступны.

Программный счетчик (PC)



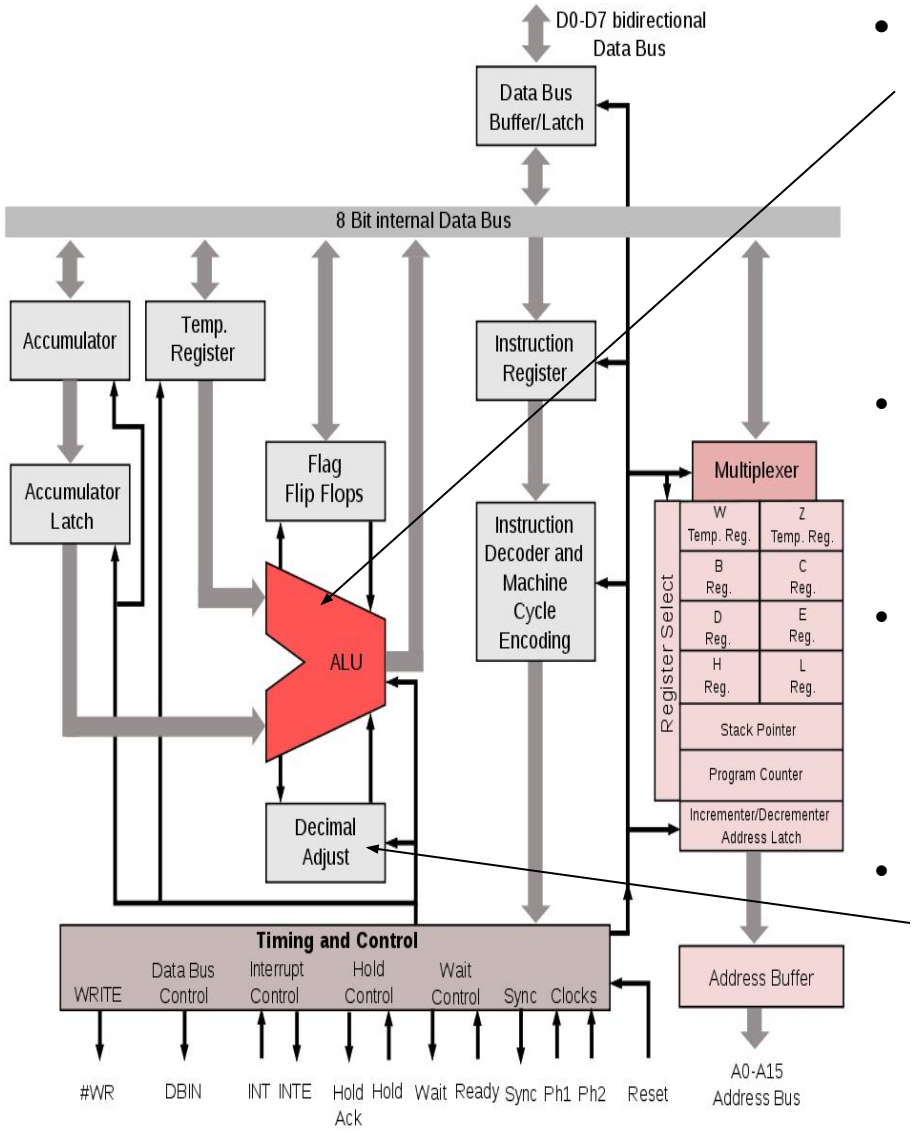
- **Программный счетчик (PC) -** указывает адрес где находится в памяти очередной байт команды. Так как команды, как правило, хранятся в ячейках памяти с последовательно возрастающими адресами, то после выполнения очередной команды содержимое PC увеличивается на единицу, в результате чего автоматически указывается адрес следующей команды. Иногда содержимое PC изменяется под действием самой программы. В результате этого осуществляется принудительная передача управления той области памяти, в которой помещается следующий фрагмент программы.

Указатель стека (SP)



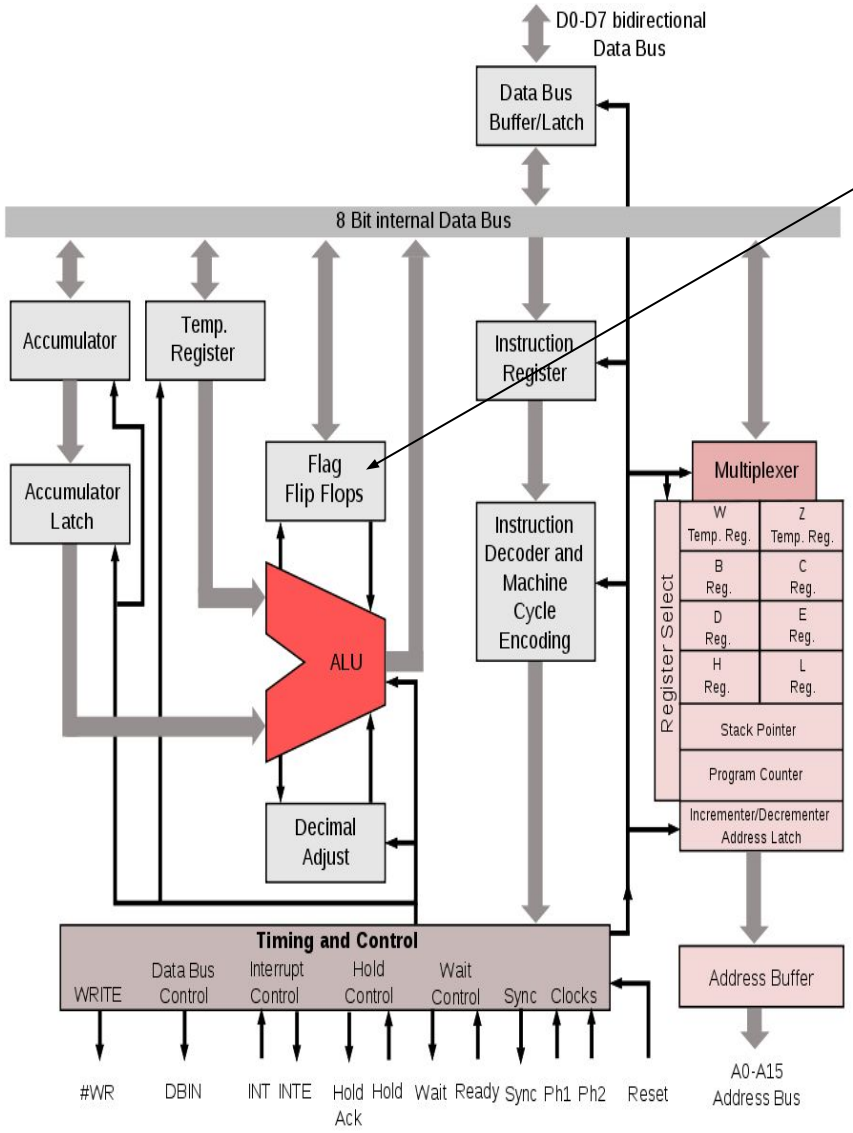
- **Указатель стека (SP -** Stack Pointer**)** регистр с автомодификацией служит для адресации особого вида памяти, называемого стеком в котором хранятся адреса возврата к прерванным подпрограммам. Стек может также использоваться для временного хранения локальных переменных и передачи входных или выходных параметров при вызова подпрограмм.
- *Стек*—это область памяти с доступом типа «последний пришел — первый вышел» или LIFO (Last Input — First Output).
- Стек обычно заполняется в сторону уменьшения адресов, при этом указатель стека показывает на последнюю заполненную ячейку стека — *вершину стека* TOS (Top of Stack).

Арифметико-логическое устройство (АЛУ)



- **Арифметико-логическое устройство (АЛУ)** выполняет арифметические и логические операции над двоичными числами и представляет собой комбинационно-логическую схему. К одному из входов схемы АЛУ подключен аккумулятор, к другому через регистр Т может быть подключен любой из регистров общего назначения.
- АЛУ выполняет арифметические операции сложения, вычитания; логические: конъюнкции, дизъюнкции, сложения по модулю 2; операции сравнения, сдвига.
- Выполнение более сложных операций (умножение, деление, вычисление элементарных функций) для может быть реализовано как аппаратно, так и потребовать программной реализации (в зависимости от конкретного процессора).
- В состав АЛУ входит комбинационная схема десятичной коррекции ДК, которая при выполнении специальной команды преобразует результат из двоичной формы в двоично-десятичную.

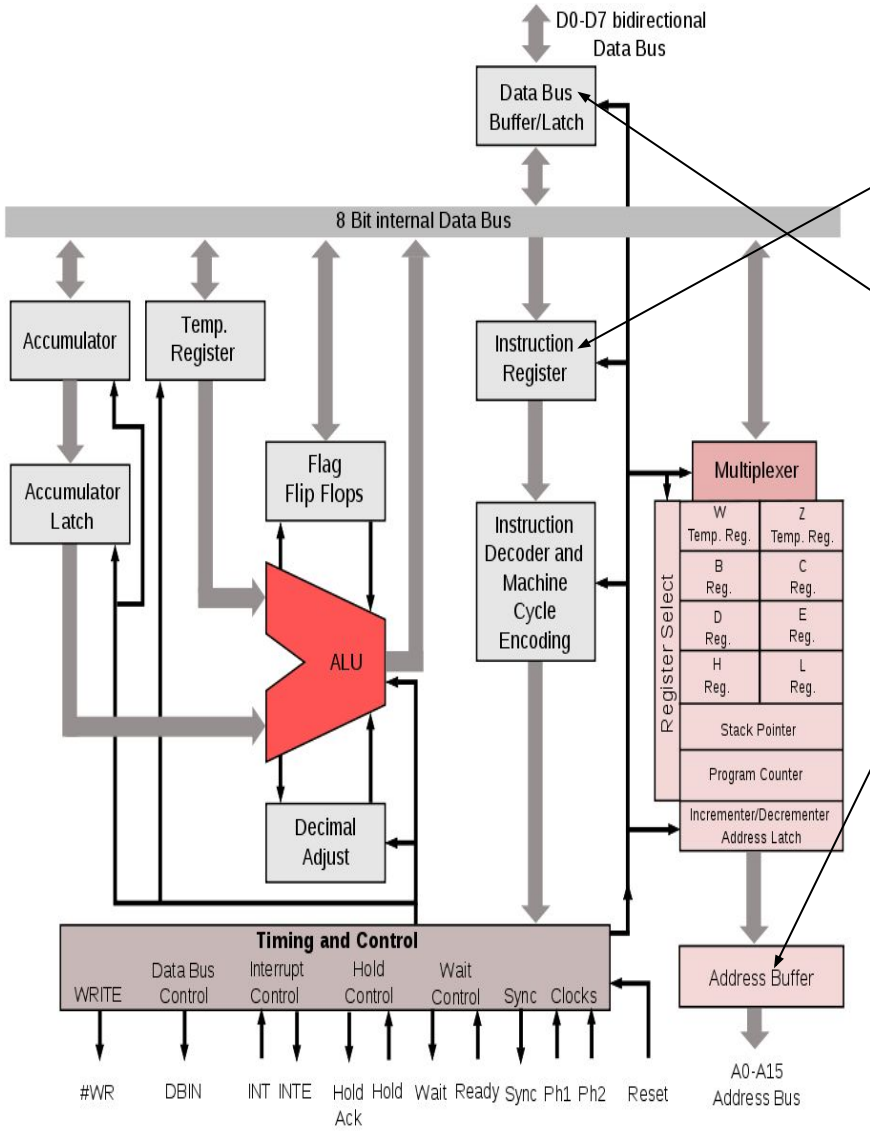
Регистр флагов.



- АЛУ непосредственно связано **регистром флагов (признаков) (F)**, в соответствующих разрядах которого фиксируются особенности выполнения каждой операции **например**:
- Флаг **Z** - нулевой результат в аккумуляторе.
- Флаг **C** - перенос из старшего/ заем в старший разряд.
- Флаг **S** - знак результата.
- Флаг **P** - четность (паритет).
- Флаг **AC** - вспомогательный перенос из младшего полубайта.

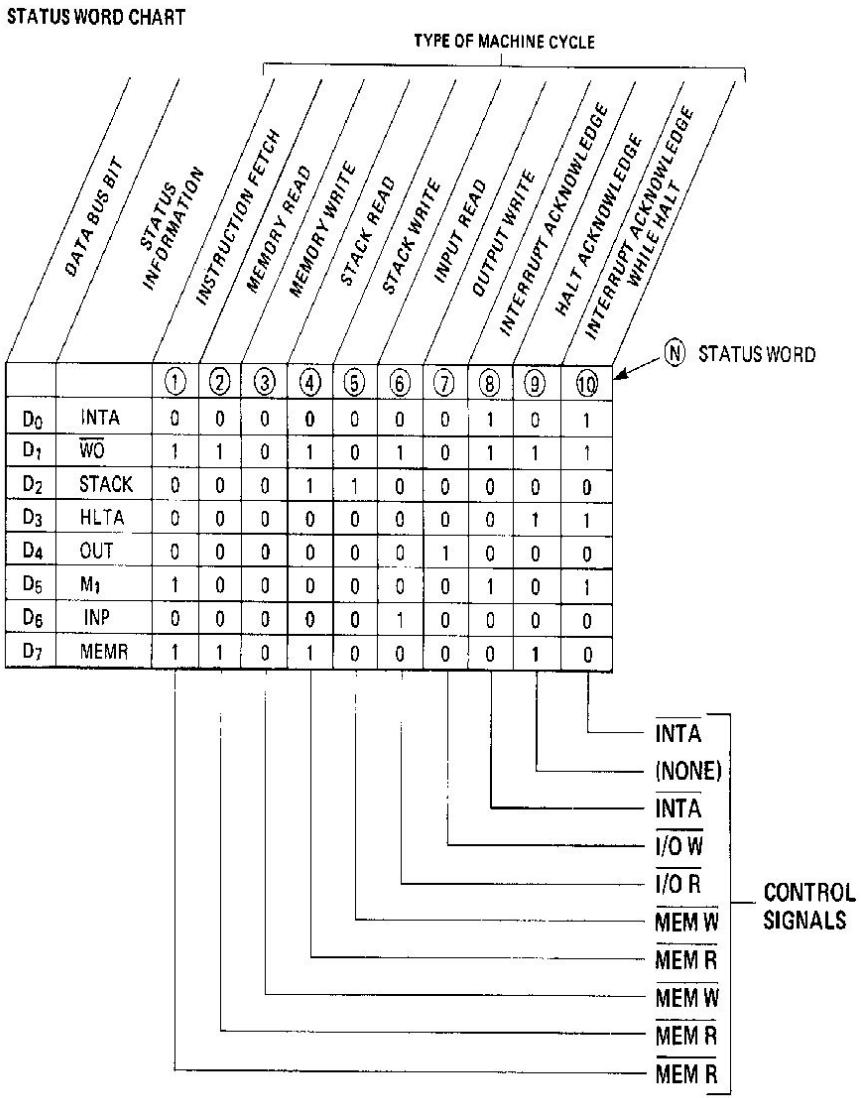
- Регистр флагов и аккумулятор образуют регистр слова состояния программы (Program Status Word) – **PSW**

Регистр команд, буфер шины.



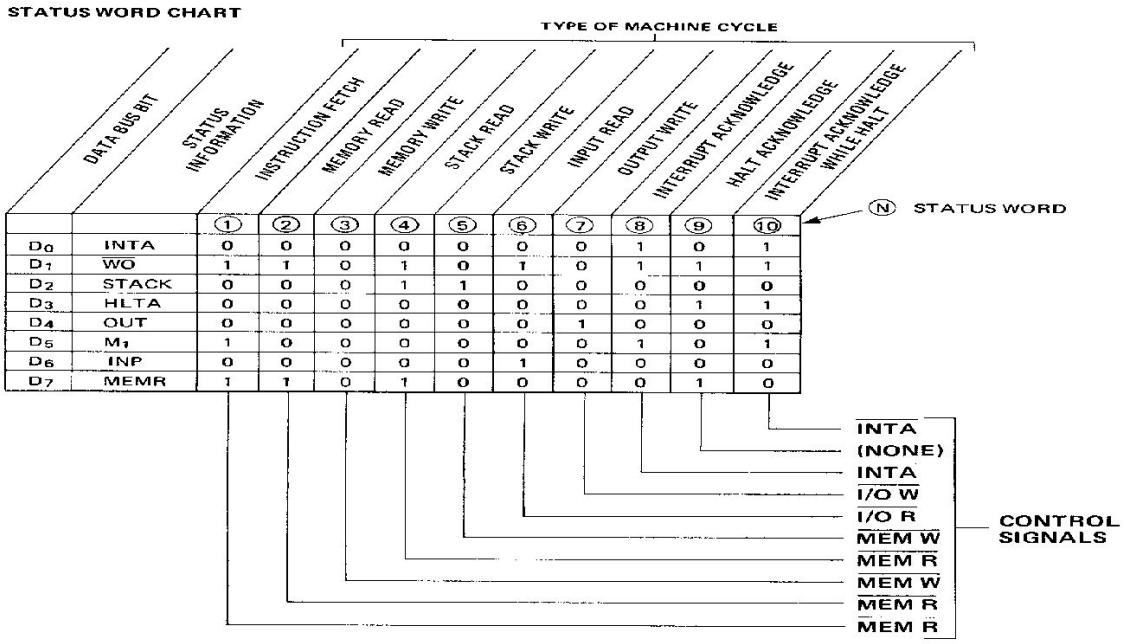
- **Регистр команд.** В него поступает первый байт команды содержащий код операции. Программно недоступен.
- **Буфер шины данных и буфер шины адреса** обеспечивают связь центрального процессора с внешними шинами данных и адреса. Использование буферов с тремя состояниями позволяет процессору отключаться от внешних шин, предоставляя их в распоряжение внешних устройств, а также позволяет использовать одну и ту же шину как для приема данных так и для передачи.

Слово состояния микропроцессора (SW).



SW (Status Word)

Состояние (машинный цикл МП)	Разряды регистра состояния МП							
	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
Выборка команды	1	0	1	0	0	0	1	0
Чтение байта из памяти	1	0	0	0	0	0	1	0
Запись байта в память	0	0	0	0	0	0	0	0
Чтение байта из стека	1	0	0	0	0	1	1	0
Запись байта в стек	0	0	0	0	0	1	0	0
Ввод байта с внешнего устройства	0	1	0	0	0	0	1	0
Вывод байта на внешнее устройство	0	0	0	1	0	0	0	0
Прерывание	0	0	1	0	0	0	1	1
Останов	1	0	0	0	1	0	0	0
Прерывание в останове	0	0	1	0	1	0	1	1



Сигнал	Разряд SW	Пояснения
INTA	D0	Сигнал подтверждения прерывания. Используется для ввода на ШД команды RST.
WO	D1	Указывает, что в текущем машинном цикле выполняется запись в память или операция вывода.
STACK	D2	Указывает наличие на ША содержимого указателя стека.
HLTA	D3	Сигнал подтверждения выполнения МП команды останов (HLT).
OUT	D4	Указывает, что на ША находится адрес устройства вывода, на которое будет выводиться байт данных по сигналу WR.
M1	D5	Указывает, что МП находится в цикле выборки первого байта команды.
INP	D6	Указывает, что на ША находится адрес устройства ввода, с которого будет введен байт данных по сигналу DBIN.
MEMR	D7	Указывает, что в текущем машинном цикле будет производиться чтение данных из памяти.

Типы команд, машинный цикл, такт.

Мнемоника		Код	Число циклов ВМ80	Число тактов		Флажки	Содержание
				ВМ80	ВМ85А	C Y Z M P A C	
ADD	src	10000SSS	1	4	4	+ + + + +	A ← A + A + src
ADC	src	10001SSS	1	4	4	+ + + + +	A ← A + src + CY
SUB	src	10010SSS	1	4	4	+ + + + +	A ← A − src
SBB	src	10011SSS	1	4	4	+ + + + +	A ← A − src − CY
ADD	M	86	2	7	7	+ + + + +	A ← A + (HL)
ADC	M	8E	2	7	7	+ + + + +	A ← A + (HL) + CY

Машинный цикл - промежуток между двумя обращениями процессора к (системной магистрали) внешнему по отношению к нему устройству (например, памяти).

Состоит из одного или нескольких **тактов**.

Команды (примеры):

- пересылки,
- логической обработки,
- арифметической обработки,
- передачи управления,
- управления процессором.

Машинные циклы (пример)

- Чтение команды из памяти;
- Чтение из памяти байта данных;
- Чтение байта из устройства ввода-вывода;
- Чтение из стека байта данных;
- Чтение вектора прерывания;
- Запись в память байта данных;
- Запись в стек байта данных;
- Передача в УВВ байта данных.

Примеры команд разной длины.

Примеры однобайтных команд:

MOV r_d, r_s , где r_d и r_s - имя регистра-приемника и имя регистра-источника операнда соответственно
Пример конкретной однобайтной команды (переслать содержимое регистра A в регистр C):
MOV C, A , (A)→C
sss=111 - код регистра A,
ddd=001 - код регистра C,
01=КОП (код операции)
Машинный код команды MOV C,A: 01 001 111.

Пример двухбайтных команд

MVI B, 75h , 75→B
OUT, 87h , (A)→87h

Пример трехбайтных команд

LXI B, 7C, 24h , 7Ch→B, 24h→C
JNZ, b_3b_2 , если результат не нулевой, т е Z=0,
то $b_3\,b_2$→PC, т.е. переход

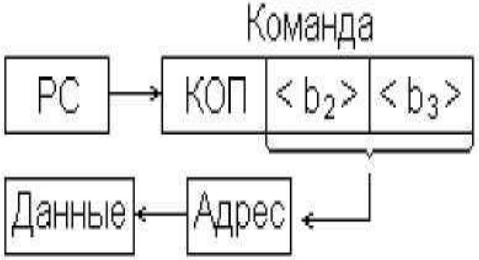
Способы адресации в МП системах (примеры).

- **Неявная адресация -** при выполнении команды места нахождения операндов и результата фиксировано. Пример – переслать в SP содержимое регистровой пары или инвертировать разряды содержимого аккумулятора.

код операции

- **Прямая (абсолютная) адресация.** Это наиболее простая, но наименее экономичная адресация. В поле самой команды содержится полный 16-битный адрес операнда в памяти. С помощью прямой адресации можно обращаться к любой ячейке в адресном пространстве.

код операции
младший байт операнда
старший байт операнда



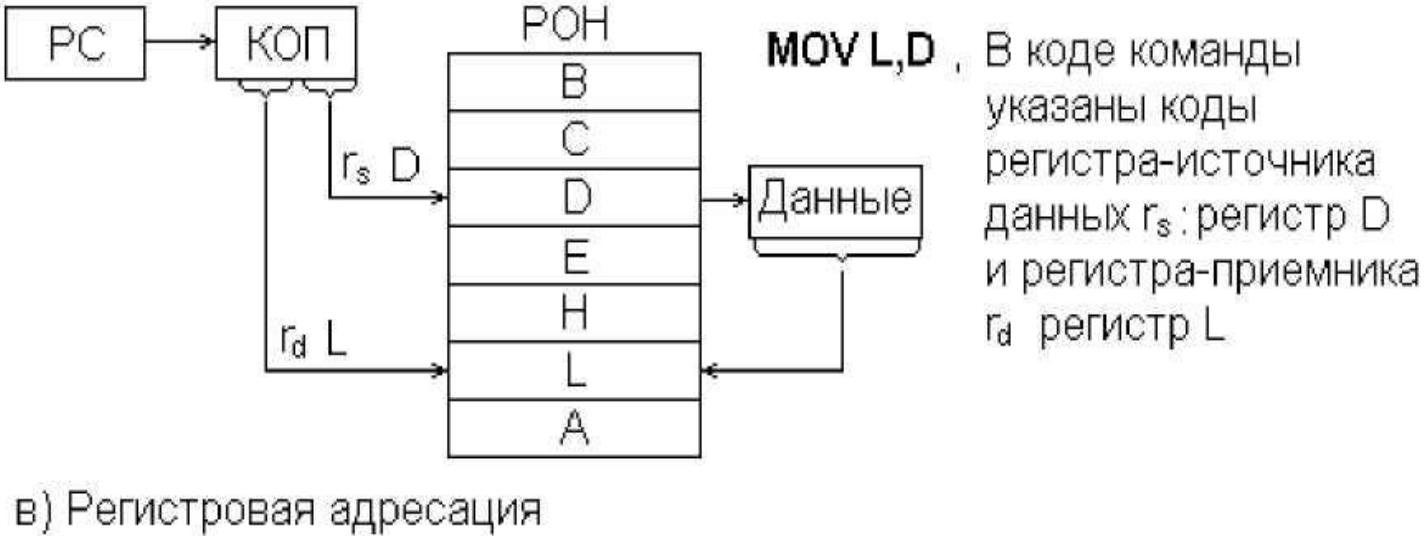
а) Прямая адресация

STA $b_3 b_2$, (A)→$b_3 b_2$ Содержимое регистра А запоминается в ячейке по адресу $b_3 b_2$

LDA $b_3 b_2$, ($b_3 b_2$)→A Содержимое ячейки $b_3 b_2$ заносится в регистр А

Способы адресации в МП системах (примеры).

- **Прямая регистровая адресация.** Вместо адреса в коде команды присутствует имя регистра или регистровой пары, в которых хранится операнд. Пример SUB B – прибавить содержимое регистра B к содержимому аккумулятора. Результат сохранить в аккумуляторе.

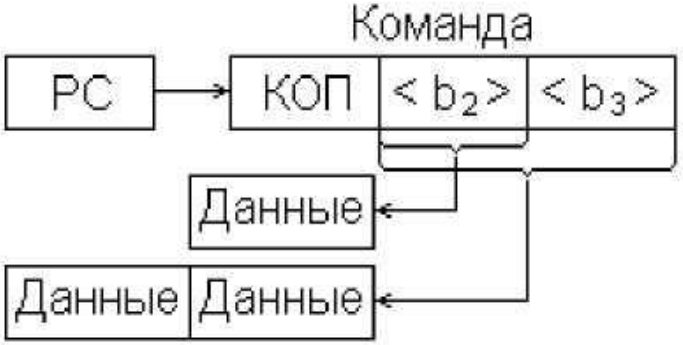


в) Регистровая адресация

Способы адресации в МП системах (примеры).

- **Непосредственная адресация.** При этом методе адресации операнд находится непосредственно за кодом операции.

код операции	
операнд	(8)
операнд	(16)



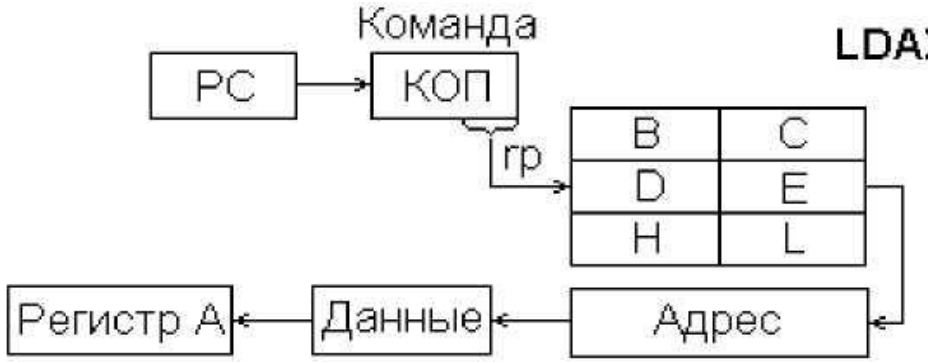
б) Непосредственная адресация

MVI C, 28h , Число 28h заносится в регистр C

ADI, 67h , (A)+67h→A Содержимое Аккумулятора суммируется с числом 67h и результат в Аккумуляторе

Способы адресации в МП системах (примеры).

- **Косвенная адресация**. При этом методе операнд содержится в ячейке, адрес которой помещен в одну из регистровых пар.



г) Косвенная регистровая адресация

LDAX D , В команде в КОП указан код регистровой пары DE, содержащей адрес данных, которые пересылаются в регистр А

Способы адресации в МП системах (примеры).

- **Стековая адресация.** Это косвенная регистровая адресация с автоинкрементом или автодекрементом при которой регистр с указателем адреса операнда задается неявно. Ячейку памяти, на которую указывает содержимое неявно заданного регистра (указателя стека) называют вершиной стека. Пример: PUSH, POP, RET.
- **Относительная адресация.** Адрес операнда вычисляется суммированием содержимого счетчика команд PC и, содержащегося в коде команды смещения (положительного или отрицательного). Позволяет создавать перемещаемые (по адресам памяти) программы, поскольку выполнение программы не зависит от ее абсолютного расположения в памяти.
- **Индексная адресация**. Аналогична относительной, но базой выступает не счетчик команд PC, а специальные индексные регистры. Удобна при обработке массивов. Адрес операнда в этом случае вычисляется суммирование адреса первого элемента массива с целым положительным смещением – содержимым индексного регистра.
- **Базовая адресация.** При этом способе адрес операнда вычисляется суммированием содержимого базового регистра (смещения) и адреса, расположенного в коде команды.
- **Битовая адресация.** Позволяет работать непосредственно с отдельно взятыми битами. Особенно актуально для микроконтроллеров.
- **Смешанная адресация.** Операнды в коде команды адресованы по-разному.

Схема включения микропроцессора (пример).