Архитектура ИС

Лекция №3

ФУНКЦИОНИРОВАНИЕ ЭВМ

Организация функционирования ЭВМ с магистральной архитектурой

- Комплект интегральных схем, из которых состоит ЭВМ, называется микропроцессорным комплектом (МК)
- В состав МК входят: системный таймер, микропроцессор (МП), сопроцессоры, контроллер прерываний, контроллер прямого доступа к памяти (ПДП), контроллеры устройств ввода-вывода и др.

Организация функционирования ЭВМ с магистральной архитектурой

- Все устройства ЭВМ делятся на центральные и периферийные.
- Центральные устройства полностью электронные, периферийные могут быть либо электронными, либо электромеханическими с электронным правлением.

Организация функционирования ЭВМ с магистральной архитектурой

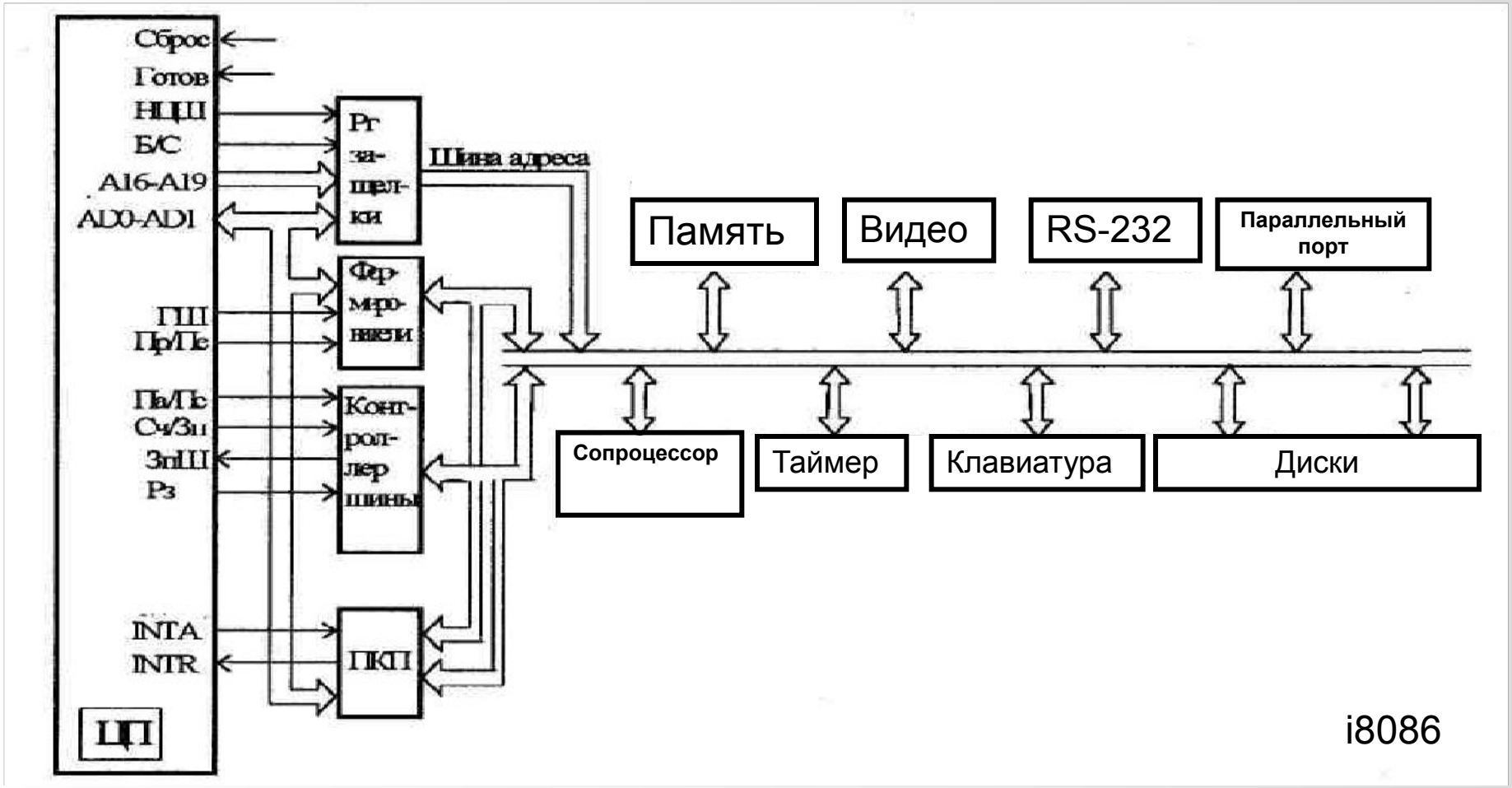
- В центральных устройствах основным узлом, связывающим микропроцессорный комплект в единое целое, является системная магистраль. Она состоит из трех узлов, называемых шинами:

1) шина данных (ШД)
2) шина адресов (ША)
3) шина управления (ШУ).

Организация функционирования ЭВМ с магистральной архитектурой

- Количество проводников в шине называется разрядностью (шириной) шины.
- Для ША ширина равна 20 - 64, что обеспечивает прямую адресацию 2^{20}-2^{64} устройств или ячеек памяти.
- ШД имеет ширину 8-64, обеспечивая параллельную передачу от одного до восьми байт.
- Ширина ШУ зависит от типа ЦП и определяется числом управляющих сигналов.

Структура ЭВМ минимальной конфигурации



Структура ЭВМ минимальной конфигурации

ШУ содержит четыре линии

Па/Пе	Память/Периферия	Различает обращение к памяти или периферии
Сч/Зп	Считывание/Запись	Операции считывания или записи информации
ЗпШ	Запрос Шины	Запрос на захват шины другим ведущим
РзШ	Разрешение Шины	Разрешение на захват шины другим ведущим

Структура ЭВМ минимальной конфигурации

Системные сигналы

Сброс	Внешний сигнал, осуществляющий начальный сброс системы, этот сигнал формируется при включении системы
Готов	Сигнал, получаемый от периферийных устройств о том, что в следующем такте шины данные будут восприняты или выданы устройством, служит признаком завершения шины в следующем такте

Структура ЭВМ минимальной конфигурации

Для управления ШД используются следующие сигналы

ГШ	Готовность Шины	Сигнал, показывающий готовность к выдаче или приему данных
Пр/Пе	Прием/Передача	Показывает периферийным устройствам должны они передавать или принимать данные

Структура ЭВМ минимальной конфигурации

Для управления ША используются следующие сигналы

НЦШ	Начальный Цикл Шины	Сигнал, показывающий наличие адреса на ША
Б/С	Байт/Слово	Признак размера передаваемых данных

Сигнал INTR - запрос на разрешения прерывания. Сигнал INTA - подтверждение запроса. После получения INTA ПКП выставляет на ШД вектор прерывания, устройства сделавшего запрос.

Цикл работы и стандарты системной шины

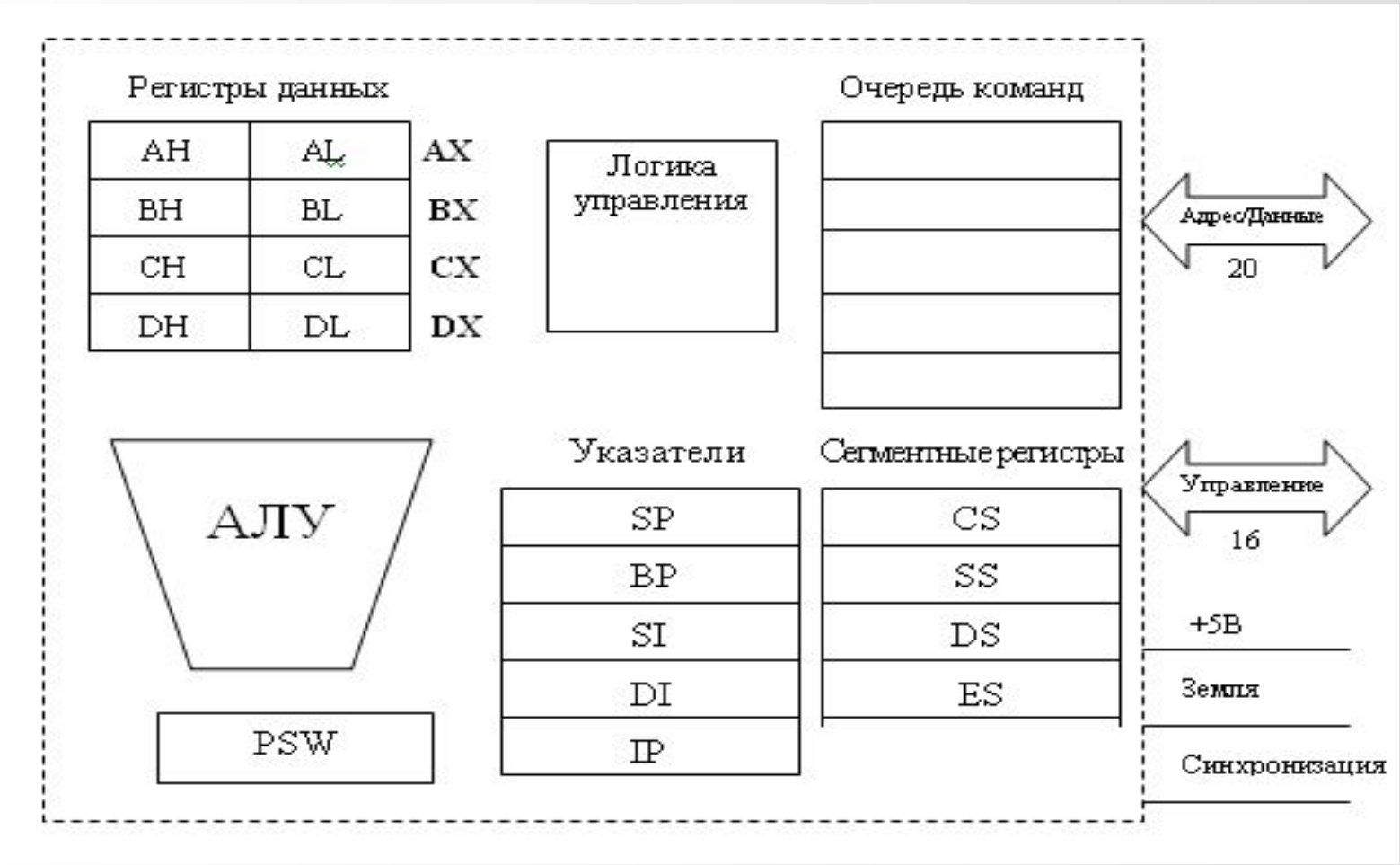
- СШ синхронизирована сигналами тактового генератора процессора. Цикл шины состоит из нескольких тактов: четырёх обязательных тактов (Т1 - Т4) и бесконечного числа тактов ожидания (Т0).
- Когда процессор готов инициировать цикл шины, он в такте Т1 выдаёт сигнал НЦШ и сигналы, определяющие вид информации (Б/С), адресат (Па/Пе), вид данных (Б/С), режимы (Сч/Зп) и (Пр/Пе) и выставляет на выводы адреса адрес порта периферийного устройства или ячейки памяти.

Цикл работы и стандарты системной шины

- В конце такта регистры защёлки фиксируют адрес и он снимается с контактов адреса ЦП.
- Во втором такте устанавливается сигнал ГШ, разрешающая работу формирователей.
- В третьем такте, если периферийные устройства или память могут принять/передать информацию, то данные помещаются на шину. Если к моменту Т3 ЦП не получил от устройства сигнал Готов, он вводит между тактами Т3 и Т4 такты ожидания Т0 до получения сигнала Готов.
- В начале четвёртого такта данные принимаются/передаются ЦП, снимаются сигналы ГШ и Пр/Пе и цикл СШ заканчивается.

Структура микропроцессора i8086/8088

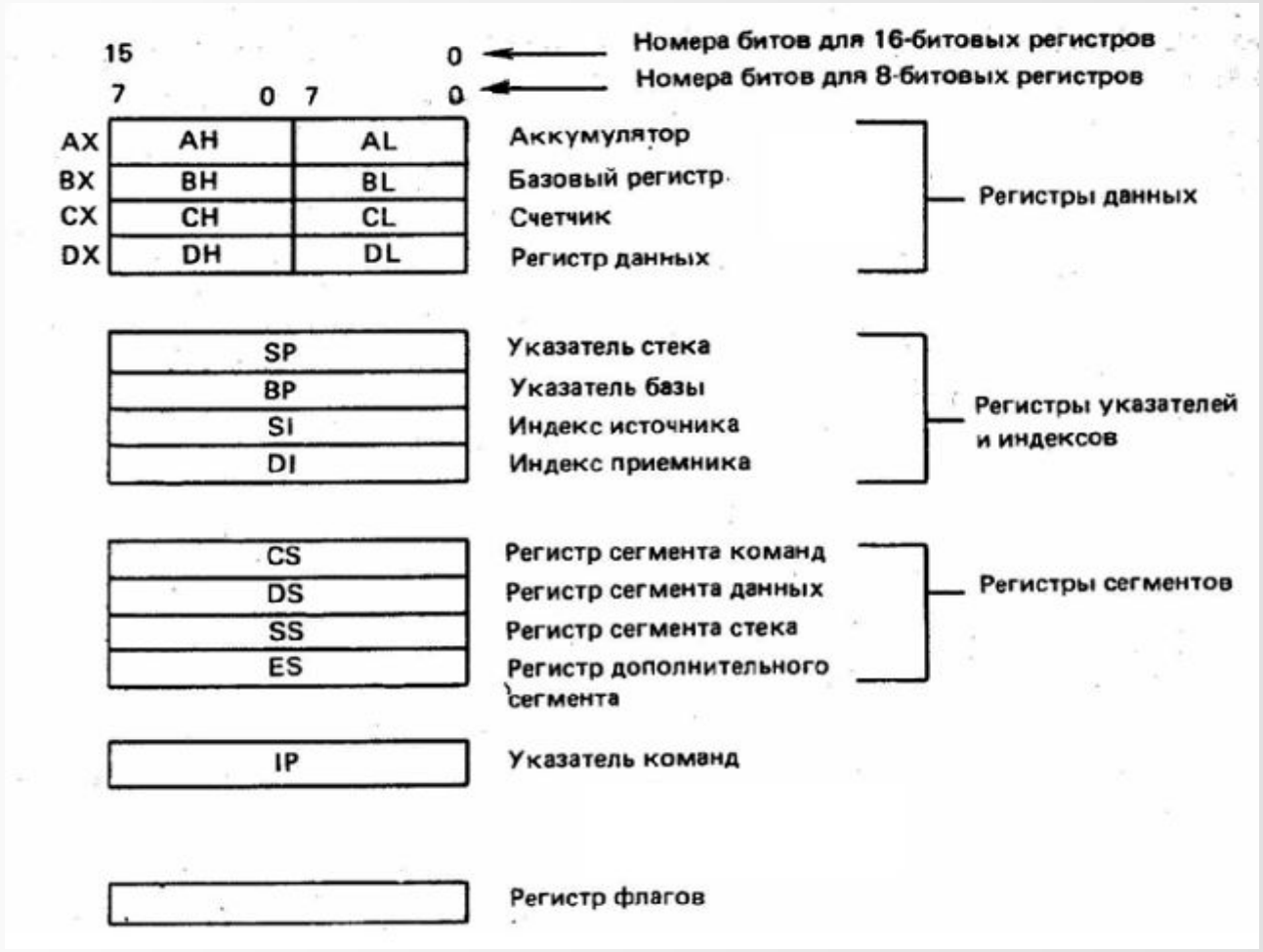
АРХИТЕКТУРА ЦЕНТРАЛЬНОГО ПРОЦЕССОРА



АРХИТЕКТУРА ЦЕНТРАЛЬНОГО ПРОЦЕССОРА

- Внутри микропроцессора информация содержится в группе 16-битовых элементов, называемых регистрами.
- Всего он имеет 14 регистров:
- 12 регистров данных и адресов и в дополнение к ним указатель команд (регистр адреса команд) и регистр состояния (регистр флагов).
- Можно подразделить 12 регистров данных и адресов на три группы по четыре регистра, а именно на регистры данных, регистры указателей и индексов и регистры сегментов.

АРХИТЕКТУРА ЦЕНТРАЛЬНОГО ПРОЦЕССОРА



Регистры данных

- В зависимости от того, чем Вы оперируете: 16-битовыми словами или 8-битовыми байтами, регистры данных можно рассматривать как четыре 16-битовых или восемь 8-битовых регистров. В первом случае регистры имеют имена AX, BX, CX, DX. Эти регистры образованы из 8-битовых регистров AL, AH, BL, BH, CL, CH, DL и DH

Регистры данных

- Регистр AX, аккумулятор (accumulator), используется при умножении и делении слов, в операциях ввода-вывода и в некоторых операциях над строками.
- Регистр AL используется при выполнении аналогичных операций над байтами, а также при преобразовании десятичных чисел и выполнении над ними арифметических операций.
- Регистр AH используется при умножении и делении байтов.

Регистры данных

- Регистр BX, базовый регистр (base register), часто используется при адресации данных в памяти.
- Регистр CX, счетчик (count register), используется как счетчик числа повторений цикла и в качестве номера позиции элемента данных при операциях над строками.
- Регистр CL используется как счетчик при операциях сдвига и циклического сдвига на несколько битов.
- Регистр DX, регистр данных (data register), используется при умножении и делении слов. Кроме того, в операциях ввода-вывода он используется как номер порта.

Регистры сегментов

- Регистр сегмента команд CS (code segment) указывает на сегмент, содержащий текущую исполняемую программу. Для вычисления адреса следующей исполняемой команды микропроцессор добавляет к содержимому регистра CS содержимое указателя команд IP.
- Регистр сегмента стека SS (stack segment) указывает на текущий сегмент стека. Стек представляет собой область памяти, используемую для временного хранения данных и адресов. Микропроцессор 8088 использует стек для хранения адреса возврата из текущей подпрограммы, но стек можно использовать также для восстановления содержимого регистров, изменяемых при работе программы.

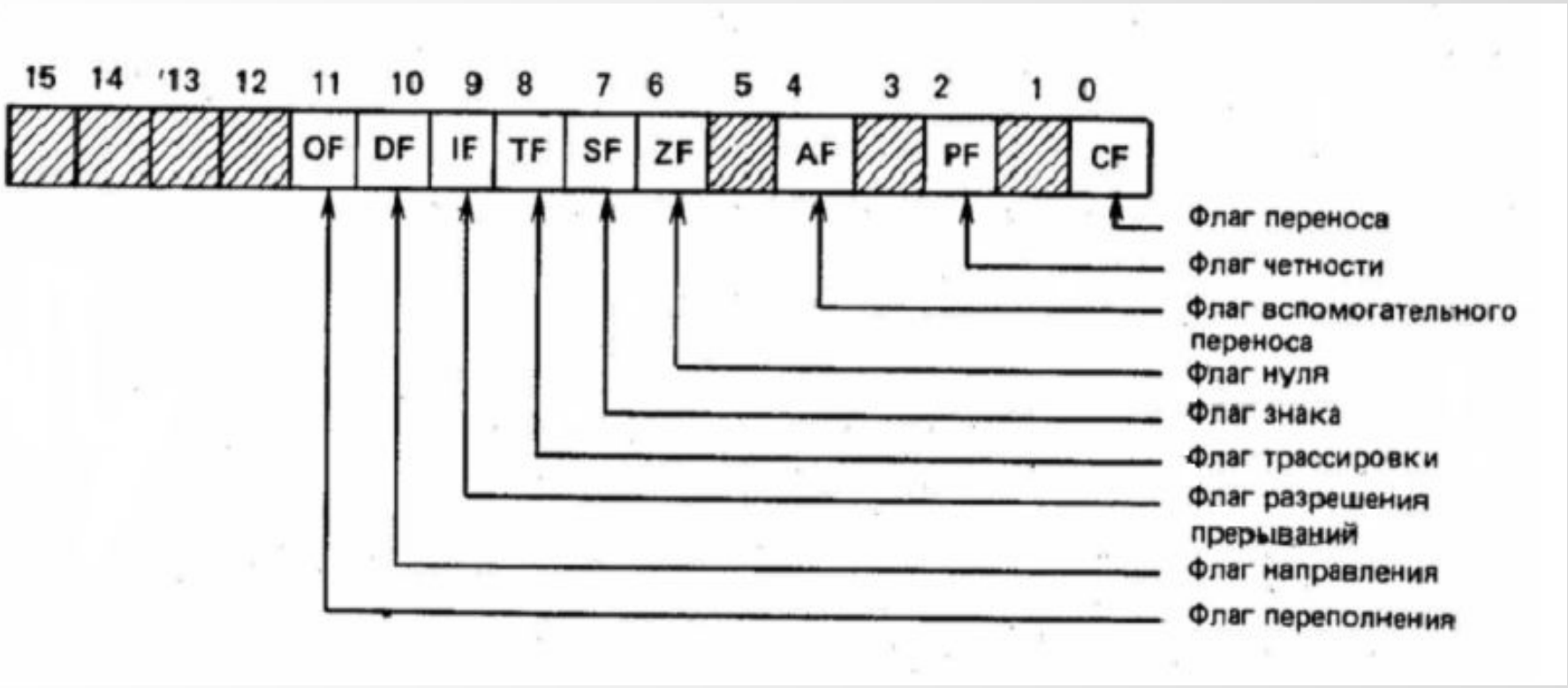
Регистры сегментов

- Регистр сегмента данных DS (data segment) указывает на текущий сегмент данных, обычно содержащий используемые в программе переменные.
- Регистр дополнительного сегмента ES (extra segment) указывает на текущий дополнительный сегмент, который используется при выполнении операций над строками.

Регистры указателей и индексов

- Для вычисления адреса команды в сегменте команд микропроцессор извлекает номер блока памяти из регистра CS, а смещение - из регистра IP.
- Для доступа к сегменту данных микропроцессор извлекает номер блока из регистра DS, а смещение - из регистра BX или индексного регистра (SI или DI).
- Для доступа к сегменту стека микропроцессор извлекает номер блока из регистра SS, а смещение - из регистра указателя (SP или BP).
- Выбирая номер блока из регистра ES, микропроцессор может также получить доступ к дополнительному сегменту.

флаги


15 14 '13 12 11 10 9 8 7 6 5 4 3 2 1 0
OF DF IF TF SF ZF AF PF CF
Флаг переноса
Флаг четности
Флаг вспомогательного переноса
Флаг нуля
Флаг знака
Флаг трассировки
Флаг разрешения прерываний
Флаг направления
Флаг переполнения

флаги

- В 16-битовом регистре флагов фиксируется информация о текущем состоянии дел, которая может помочь программе принять решение. Шесть битов регистра служат для хранения состояний, а три других могут быть использованы для программного управлений режимом работы микропроцессора

флаги

1. Бит 0, флаг переноса CF (carry flag), равен 1, если произошел перенос единицы при сложении или заем единицы при вычитании. В противном случае он равен нулю. Кроме того, CF содержит значение бита, который при сдвиге или циклическом сдвиге регистра или ячейки памяти вышел за их границы, и отражает результат операции сравнения. Наконец, CF служит индикатором результата умножения.

флаги

2. Бит 2, флаг четности PF (parity flag), равен 1, если в результате операции получено число с четным числом единиц в его битах. В противном случае он равен нулю. Флаг PF в основном используется в операциях обмена данными.

флаги

3. Бит 4, вспомогательный флаг переноса AF (auxiliary carry flag), аналогичен флагу CF, только контролирует перенос или заем для третьего бита данных. Полезен при выполнении операций над упакованными десятичными числами.
4. Бит 6 флаг нуля ZF (zero flag), равен 1, если в результате операции получен нуль; ненулевой результат сбрасывает ZF в нуль.

флаги

5. Бит 7, флаг знака SF (sign flag), имеет значение только при операциях над числами со знаком. Флаг SF равен 1, если в результате арифметической или логической операции, сдвига или циклического сдвига получено отрицательное число. В противном случае он равен нулю.

флаги

6. Бит 8, флаг трассировки TF (trap flag), разрешает микропроцессору исполнять программу "по шагам" и используется при отладке программ.
7. Бит 9, флаг прерывания IF (interrupt enable flag), разрешает микропроцессору реагировать на прерывания от внешних устройств. Сбрасывание IF в нуль заставляет микропроцессор игнорировать прерывания до тех пор, пока IF не станет равным 1.

флаги

8. Бит 10, флаг направления DF (direction flag), заставляет микропроцессор уменьшать на единицу (DF = 1) или увеличивать на единицу (DF = 0) регистр(ы) индекса после выполнения команды для работы со строками. Если DF = 0, то микропроцессор будет обрабатывать строку "слева направо" (от младших адресов к старшим). Если DF = 1, то обработка пойдет в обратном направлении (от старших адресов к младшим или справа налево).

флаги

9. Бит 11, флаг переполнения OF (overflow flag), в первую очередь служит индикатором ошибки при выполнении операций над числами со знаком. Флаг OF равен 1, если результат сложения двух чисел с одинаковым знаком или результат вычитания двух чисел с противоположными знаками выйдет за пределы допустимого диапазона значений операндов. В противном случае он равен 0. Кроме того, OF = 1, если старший, (знаковый) бит операнда изменился в результате операции арифметического сдвига. В противном случае он равен 0. В сочетании с флагом CF флаг OF указывает длину результата умножения. Если старшая половина произведения отлична от нуля, то OF и CF равны 1; в противном случае оба эти флага равны 0. Наконец, OF = 0, если частное от деления двух чисел переполняет результирующий регистр.

Безусловный переход

jmp [< тип > **ptr**] операнд.

<тип> - тип перехода **short** (короткий) – смещение 127 байтов вперёд или 128 байтов назад, **near** (близкий) – смещение в пределах сегмента (64 Кбайта), **far** (дальний) – в любой сегмент с любым смещением.

- **ptr** – приставка, которую можно перевести как *указанный в*.
- Если тип не задан, по умолчанию принимается **near**.

Безусловный переход

Название	Мнемоника	Описание
внутрисегментный прямой короткий	jmp short <операнд>	IP ← (IP) + 8-битное смещение, определяемое операндом
внутрисегментный прямой близкий переход	jmp near ptr <операнд>	IP ← (IP)+16-битное смещение, определяемое операндом
внутрисегментный косвенный переход	jmp <адрес операнда>	IP ← 16-битный адрес перехода
Межсегментный прямой далекий переход	jmp far ptr <операнд>	IP ← смещение операнда в сегменте CS ← адрес сегмента, содержащего операнд
Межсегментный косвенный далёкий переход	jmp far ptr <адрес операнда>	IP ← операнд CS ← адрес операнда +2

Безусловный переход

	Флаги	Смысл
ja/jnbe	CF or ZF=0	выше /не ниже и не равно
jae/jnb	CF=0	выше или равно/не ниже
jb/jnae	CF=1	ниже/не выше и не равно
jbe/jna	CF or ZF=1	ниже или равно/не выше
je/jz	ZF=1	равно/нуль
jne/jnz	ZF=0	не равно/не нуль
jg/jnle	(SF xor OF) or ZF=0	больше/не меньше и не равно
jge/jnl	SF xor OF=0	больше или равно/не меньше
jl/jnge	(SF xor OF)=1	меньше/не больше и не равно
jle/jng	((SF xor OF) or ZF)=1	меньше или равно/не больше
jp/jpe	PF=1	есть паритет/паритет четный
jnp/jpo	PF=0	нет паритета/паритет нечетный
jc	CF=1	перенос
jnc	CF=0	нет переноса
jo	OF=1	переполнение
jno	OF=0	нет переполнения
jns	SF=0	знак +
js	SF=1	знак -

Циклы

- **loop**[<условие повторения цикла>] <метка короткого перехода>
- Инструкция **loop** использует содержимое регистра CX как счетчик повторений цикла. Команда **loop** уменьшает содержимое регистра CX на 1 и передает управление по адресу, определяемому меткой перехода, если содержимое CX ≠ 0, в противном случае выполняется следующая за LOOP инструкция.
- Добавление к инструкции **loop** <условие повторения цикла> позволяет ввести дополнительные логические условия на повторение цикла:
- **loope/loopz** – повторять, пока ноль;
- **loopne/loopnz** – повторять, пока не ноль.

Пример

- Дан массив из десяти слов, содержащих целые числа. Требуется найти максимальное значение в массиве.

Пример

```
data segment
max dw ?
mass dw 10,24,76,479,-347,281,-24,70,124,97
data ends
code segment
assume cs: code, ds: data
start:     mov ax, data
mov ds, ax          ; Загрузить сегментный адрес данных
           lea bx, mass      ; Загрузить адрес смещения массива
           mov cx, 10  ; Установить счетчик повторений цикла
           mov ax, [bx]      ; Первый элемент массива в Аккумулятор
```

Пример

```
beg:        cmp [bx], ax      ; Сравнить текущий элемент
            jl no             ; он меньше
            mov ax, [bx]; он больше или равен
 no:        inc bx            ; Следующий элемент
            inc bx            ; массива
            loop beg
            mov max, ax
 quit:      mov ax,4C00h   ; Код завершения 0
            int 21h        ; Выход в DOS
code ends
end start
```