**ВОПРОС 16**

Внутренняя структура процессора i8086.

**Особенности i8086**

• CISC архитектура

• 20 адресных линий

• может адресовать 2 Мб оперативной памяти

• *поддерживает прямой доступ к памяти*

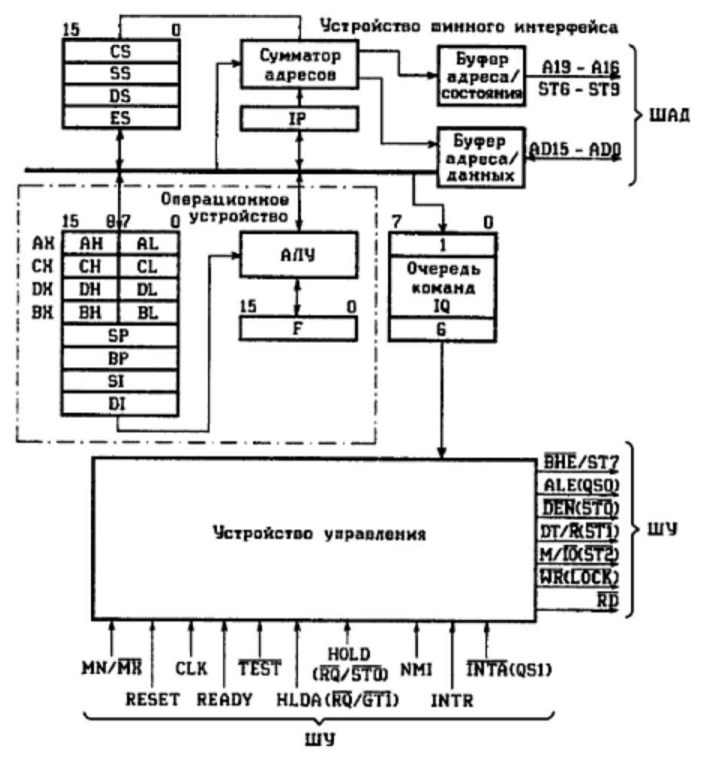
*• обмен по запросу на прерывание*

• внутренние регистры 16-ти разрядные

• *поддерживает конвейер команд* (который скорее очередь команд)

• тактируется от внешнего генератора

Разработала команда интел. Началась разработка в 1976 году и был положен в основу IBM PC.

**Внутренняя структура**

Условно можно разделить на 3 блока:

Операционное устройства, куда входят АЛУ, регистры общего назначения, регистр флагов, который содержит признаки выполнения предыдущей операции, либо управляющие инструкции для процессора, как выполнять следующие операции.

Конвейер команд здесь представлен очередью команд. Очередь может иметь до 6 команд.

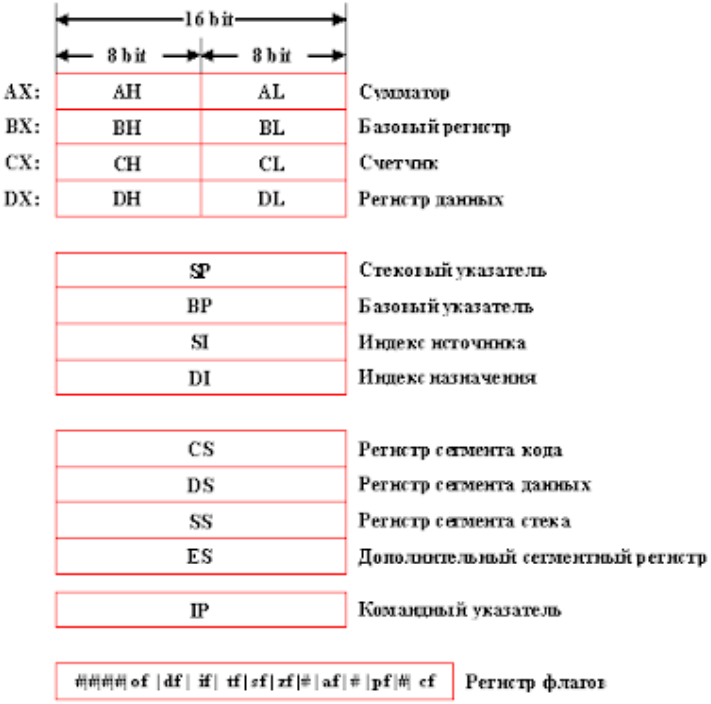
Сверху над операционным устройством находится устройство шинного интерфейса. Обеспечивает сопряжение процессора с внешней средой. В частности шинный интерфейс занимается формированием адресов (сумматор адресов, которому подается информация с регистра IP - счетчик команд в которой хранится номер текущей команды). Слева шестнадцатиразрядные сегментные регистры, которые определяют расположение в памяти соответствующего сегмента. CS - код, SS - стек, **DS** - данные, ES - дополнительный сегмент. Также имеются буферные регистры на выходе. (Совмещенная шина адреса и данных).

Устройство управление находится снизу. Содержит множество входов и выходов, которые совмещаются в шину управление.

Получается шина адреса и данных совмещены. А шина управления отдельно.

Это был первый 16-битный процессор с 16-битным ALU, 16-битными регистрами, внутренней шиной данных и 16-битной внешней шиной данных, что привело к более быстрой обработке.

**Программная модель процессора**



Под программной моделью подразумевается его внутренние регистры. У первых четырех регистров есть особенность. Они могут использоваться как один шестнадцатиразрядный, либо как два восьмиразрядных.

**AX** - сумматор, он же аккумулятор у нас в курсаче. Он выступает в качестве приемника результата и часто в качестве второго операнда.

**BX** - базовый регистр, используется для базовой адресации

**CX** - счетчик, используется в циклах

**DX** - регистр данных

**SP** - стековый указатель, хранит адрес вершины стека, соответственно с SS (регистром сегмента) позволяет формировать адреса в стеке

**BP** - базовый указатель, в базово-индексной адресации хранит второй из трех слагаемых. (индекс)

**SI** - индекс источника при различных способах адресации

**DI** - индекс приемника назначения

Основная функция сегментных регистров CS, DS, SS, ES формирование различных адресов

**CS** - хранит начальный адрес сегмента кода в памяти

**DS** - хранит начальный адрес сегмента данных в памяти

**SS** - хранит начальный адрес сегмента стека

**ES** - хранит начальный адрес дополнительного сегмента данных

**IP** - командный указатель. Указывает на текущую выполняемую команду.

Формирование адреса происходит следующим образом: содержимое сегментного регистра сдвигается на 4 разряда и к этому прибавляется смещение. При формировании различных адресов юзаются содержимое разных сегментных регистров и разные регистры используются для смещения.

Например для получение адреса следующий команды необходимо содержимое регистра сегмента кода сдвинуть на 4 разряда влево (тоже самое, что умножить на 16) и прибавить значение командного указателя.

Для обращение к адресу вершины стека в берется SS сдвигается на 4 разряда и прибавляется SP

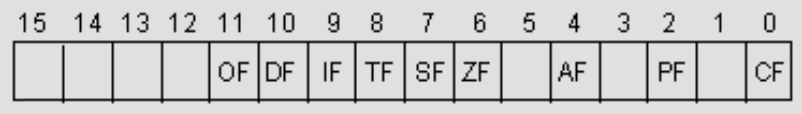
Для обращения к данным DS или ES, сдвигается на 4 разряда, в качестве смещение используется DI или SI.

[Там в 10 лекции есть более подробная табличка]

**Регистр флагов**

Каждый бит регистра флагов отвечает за определённый признак предыдущей операции или управляющая инструкция, определяющая поведение процессора.

Состоит из 16 бит, из них используется только 9, остальные зарезервированные.



*Признаки предыдущей операции*

**CF** — содержит перенос(заем) из старшего бита после арифметических операций и сдвигах

**PF** — показывает четность младших восьми битовых данных (1 – четное, 0 – нечетное) (по сути, дублирует младший бит и инвертирует его)

**AF** — признак меж тетрадного переноса (из 3 бита) Если осуществляется перенос или заем в третьем бите тетрады (от слова тетрада - 4 бита), то выставляется в 1. Используется в двоично-десятичных операциях.

**ZF** — флаг нулевого результата. (если вышло 0, то флаг 1 и далее можно строить работу по нему. В курсаче для условных переходов, опираясь на этот флаг.)

**SF** — флаг знака (совпадает со старшим битом результата, 0 – плюс, 1 - минус).

*Флаги, которые влияют на режим работы процессора*

**TF** — флаг пошагового режима (используется при отладке)

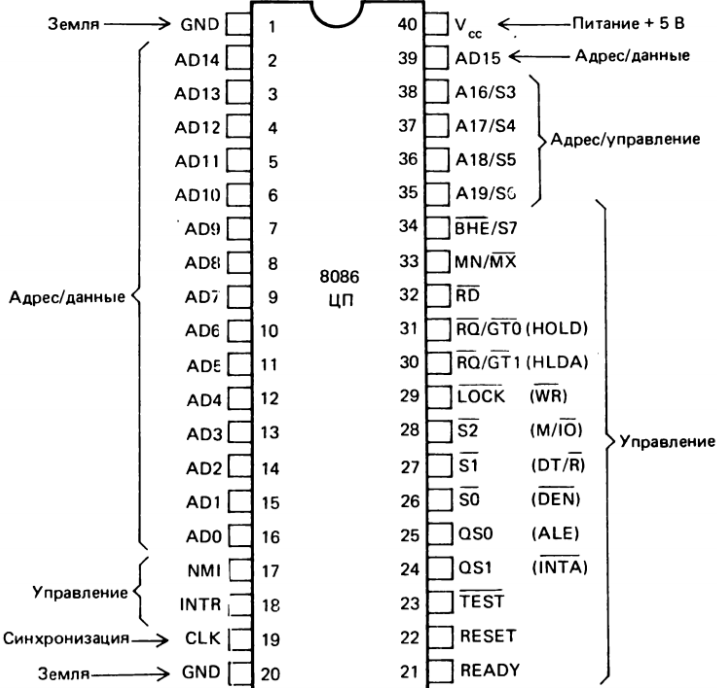
**IF** — флаг разрешения аппаратных прерываний. Разрешает(1) или запрещает(0) маскируемые прерывания (немаскируемые прерывания нельзя программно запретить. Юзаются в

критических ситуаций, процессор всегда их обработает. А маскируемые только при наличии разрешения на обработку. Как раз этот флаг разрешает или нет)

**DF** — флаг направления при строковых операциях. Обозначает левое или правое направление

**OF** — указывает на переполнение старшего бита (есть в курсаче команда условного перехода, но у некоторых вариантов).

**Входы и выходы процессора (микросхема)**



Поддерживает 2 режима минимальный и максимальный, результат (всм текущий режим) в 33 MN/MX (1 минимальный, 0 максимальный)

Например в максимальной используются S3, а в минимальном A16.

Мы рассматриваем в минимальном режиме.

Макс режим юзается в спец устройствах. Например если юзаем несколько процессоров таких в объеденинении.

Также уточню, что если сверху черта, то это означает, что 0 означает логическую единицу. (то бишь сигнал на выходе инвертирован)