Центральные процессоры

- Из простейших электронных элементов можно сконструировать АЛУ и память для хранения промежуточных результатов выполнения арифметических и логических операций

Центральные процессоры

- Из простейших электронных элементов можно сконструировать АЛУ и память для хранения промежуточных результатов выполнения арифметических и логических операций
- Микропроцессор — устройство, отвечающее за выполнение арифметических, логических операций и операций управления, записанных в машинном коде

Центральные процессоры

- Из простейших электронных элементов можно сконструировать АЛУ и память для хранения промежуточных результатов выполнения арифметических и логических операций
- Микропроцессор — устройство, отвечающее за выполнение арифметических, логических операций и операций управления, записанных в машинном коде
- Реализуется в виде одной или нескольких микросхем

Коды операций (инструкций). Регистры

- Простейшее АЛУ: операция для выполнения задается набором входных сигналов F0 и F1

Коды операций (инструкций). Регистры

- Простейшее АЛУ: операция для выполнения задается набором входных сигналов F0 и F1
- Каждой операции можно поставить в соответствие некоторое число: получаем набор операций (команд или инструкций) для данного АЛУ (процессора)

Коды операций (инструкций). Регистры

- Простейшее АЛУ: операция для выполнения задается набором входных сигналов F0 и F1
- Каждой операции можно поставить в соответствие некоторое число: получаем набор операций (команд или инструкций) для данного АЛУ (процессора)
- Для выполнения операций АЛУ должно получить доступ к операндам (иметь электрические контакты с устройством хранения). Такие устройства называются регистрами. Набор регистров называется файлом регистров.

Коды операций (инструкций). Регистры

- Простейшее АЛУ: операция для выполнения задается набором входных сигналов F0 и F1
- Каждой операции можно поставить в соответствие некоторое число: получаем набор операций (команд или инструкций) для данного АЛУ (процессора)
- Для выполнения операций АЛУ должно получить доступ к операндам (иметь электрические контакты с устройством хранения). Такие устройства называются регистрами. Набор регистров называется файлом регистров.
- Для удобства написания программы регистры часто именуются (0,1,2...; R1-R31; AX, BX...)

Коды операций (инструкций). Регистры

- Простейшее АЛУ: операция для выполнения задается набором входных сигналов F0 и F1
- Каждой операции можно поставить в соответствие некоторое число: получаем набор операций (команд или инструкций) для данного АЛУ (процессора)
- Для выполнения операций АЛУ должно получить доступ к операндам (иметь электрические контакты с устройством хранения). Такие устройства называются регистрами. Набор регистров называется файлом регистров.
- Для удобства написания программы регистры часто именуются (0,1,2...; R1-R31; AX, BX...)
- Названия регистров можно использовать в качестве операндов в операциях.

Машинный код, ассемблер

- Машинный код — последовательность команд в виде, готовом для исполнения на процессоре

Адрес в памяти	Содержимое памяти	Комментарий
00000	11100101	считываем слово в регистр AX...
00001	00000101	...из порта номер 5
00002	01000000	увеличиваем на 1 регистр AX
00003	11100111	записываем слово из регистра AX...
00004	00000010	...в порт номер 2
00005	11101011	повторяем действия с помощью перехода...
00006	11111001	...назад на 7 байт

- Ассемблер – язык, в котором машинные инструкции представляются символами (символическими именами)

```
CYCLE:
IN      AX, 5    ;считываем слова из порта номер 5 в регистр AX
INC     AX       ;увеличиваем на 1 регистр AX
OUT     2, AX    ;записываем слово из регистра AX в порт номер 2
JMP     CYCLE    ;повторяем действия
```

Инструкция	OpCode
IN	1110010-
INC	01000---
OUT	1110011-
JMP	11101011

Архитектура (Instruction Set Architecture) и Микроархитектура

- ISA
  - Состояние, видимое программисту (память, регистры, . . .)
  - Набор инструкций (синтаксис) и семантика их выполнения
  - Прерывания
  - Ввод/вывод

Архитектура (Instruction Set Architecture) и Микроархитектура

- ISA
  - Состояние, видимое программисту (память, регистры, ...)
  - Набор инструкций (синтаксис) и семантика их выполнения
  - Прерывания
  - Ввод/вывод
- Микроархитектура (реализация ISA в «железе»)
  - Количество и глубина конвейера
  - Микрокод
  - Кеш
  - ...

Архитектура (Instruction Set Architecture) и Микроархитектура

- ISA
  - Состояние, видимое программисту (память, регистры, . . .)
  - Набор инструкций (синтаксис) и семантика их выполнения
  - Прерывания
  - Ввод/вывод
- Микроархитектура (реализация ISA в «железе»)
  - Количество и глубина конвейера
  - Микрокод
  - Кеш
  - . . .
- Без абстракций сложно перейти от физических устройств к приложениям

Характеристики ISA

- Классы инструкций
  - Арифметико-логические
  - Контроль управления
  - Передача данных
  - Плавающая точка, мультимедиа, строки, ...

Характеристики ISA

- Классы инструкций
  - Арифметико-логические
  - Контроль управления
  - Передача данных
  - Плавающая точка, мультимедиа, строки, ...
- Адресация памяти (регистры, смещения, индексы, абсолютная, ...)

Характеристики ISA

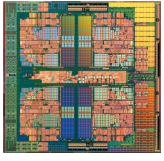
- Классы инструкций
  - Арифметико-логические
  - Контроль управления
  - Передача данных
  - Плавающая точка, мультимедиа, строки, ...
- Адресация памяти (регистры, смещения, индексы, абсолютная, ...)
- Кодирование инструкций
  - Фиксированный размер (RISC: MIPS, PowerPC, SPARC, ...)
  - Переменный размер (CISC: IBM360, x86, VAX, ...)
  - Mostly Fixed or Compressed
  - Very Long Instruction Word

Характеристики ISA

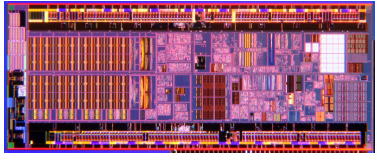
- Классы инструкций
  - Арифметико-логические
  - Контроль управления
  - Передача данных
  - Плавающая точка, мультимедиа, строки, ...
- Адресация памяти (регистры, смещения, индексы, абсолютная, ...)
- Кодирование инструкций
  - Фиксированный размер (RISC: MIPS, PowerPC, SPARC, ...)
  - Переменный размер (CISC: IBM360, x86, VAX, ...)
  - Mostly Fixed or Compressed
  - Very Long Instruction Word
- Почему отличаются ISA?
  - Влияние технологий (дорогие устройства хранения -> сжатие; Multicore/Manycore, ...)
  - Влияние приложений (инструкции для приложений: DSP; новые улучшенные компиляторы, ...)

Пример: единая ISA — различная микроархитектура

- X86 Instruction Set
- Quad Core
- 125W
- Decode 3 Instructions/Cycle/Core
- 64KB L1 I Cache, 64KB L1 D Cache
- 512KB L2 Cache
- Out-of-order
- 2.6GHz



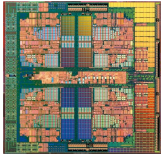
- X86 Instruction Set
- Single Core
- 2W
- Decode 2 Instructions/Cycle/Core
- 32KB L1 I Cache, 24KB L1 D Cache
- 512KB L2 Cache
- In-order
- 1.6GHz



Пример: различная ISA — различная микроархитектура

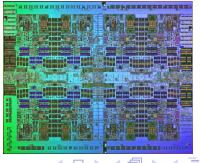
AMD Phenom X4

- X86 Instruction Set
- Quad Core
- 125W
- Decode 3 Instructions/Cycle/Core
- 64KB L1 I Cache, 64KB L1 D Cache
- 512KB L2 Cache
- Out-of-order
- 2.6GHz



IBM POWER7

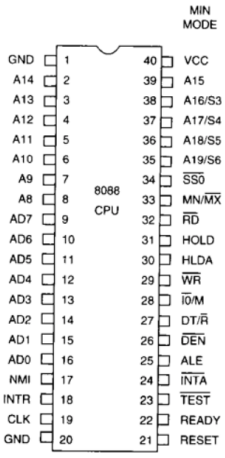
- Power Instruction Set
- Eight Core
- 200W
- Decode 6 Instructions/Cycle/Core
- 32KB L1 I Cache, 32KB L1 D Cache
- 256KB L2 Cache
- Out-of-order
- 4.25GHz



Процессор Intel 8086/8088

- 1978 г. Intel 8086 (первое поколение 16 разрядных процессоров Intel)
- 1979 г. Intel 8088 (восьмиразрядная шина данных для того, чтобы использовать уже существующие микросхемы «поддержки»: доступ к памяти, контроллер прерываний, I/O, . . .)
- Двадцатиразрядная шина адреса
- Мультиплексированная шина адреса/данных
- Частота 4-10 МГц, технология 3 мкм
- 29 000 транзисторов

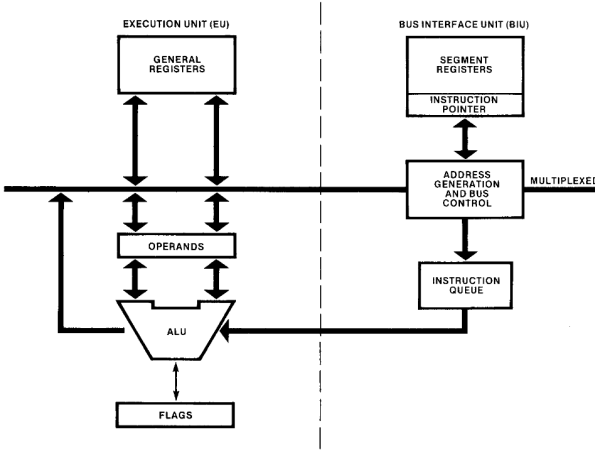
Intel 8088 - описание контактов



Контакты процессора Intel 8088

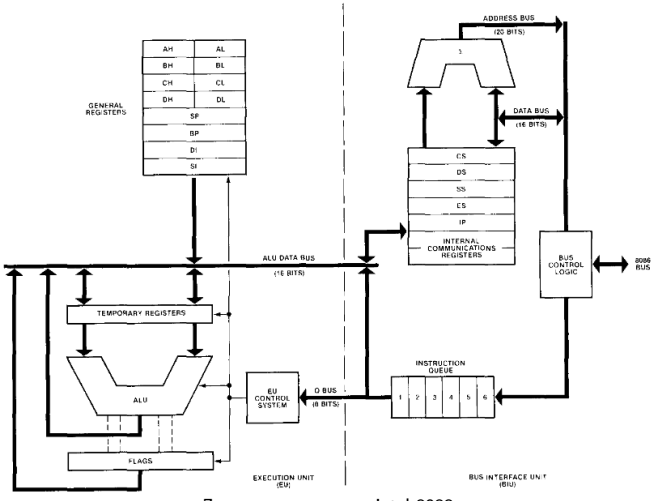
Intel 8086/8088: архитектура

- BIU (Bus Interface Unit) — **Блок интерфейса шины**: извлекает инструкции, считывает операнды и записывает результат
- EU (Execution Unit) — **Блок исполнения**: выполняет инструкции
- Организована очередь инструкций для ускорения доступа



Общая архитектура Intel 8088

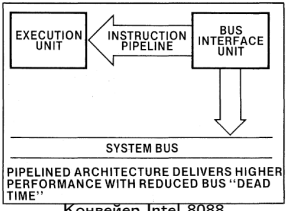
Intel 8086/8088: детали архитектуры



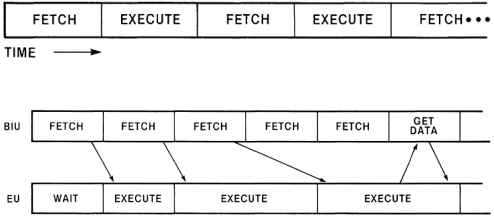
Детали архитектуры Intel 8088

Intel 8086/8088: конвейер

- Для ускорения выполнения последовательности инструкций
- Блоки EU и BIU работают параллельно (одновременно)

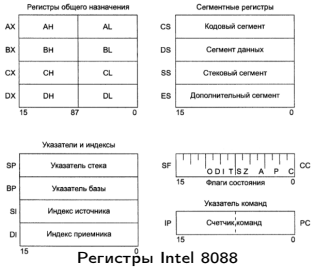


Конвейер Intel 8088

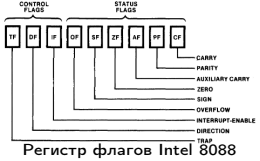


Intel 8086/8088: регистры

- DATA — AX, BX, CX, DX
  регистры общего назначения, для арифметических и логических операций; некоторые специальные
- POINTER и INDEX — BP, SP
  указывают на стек, SI (Source), DI (Destination) — для работ со строками, имеют автоувеличение и автоуменьшение
- CONTROL — IP (указатель инструкций), flags (флаги) — результат операции



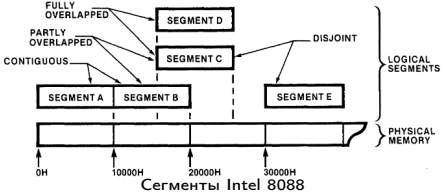
Регистры Intel 8088



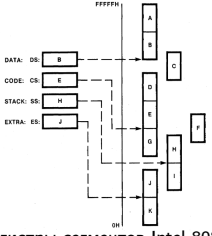
Регистр флагов Intel 8088

Intel 8086/8088: организация памяти

- 20-разрядная шина адреса, до 1Мб памяти
- С точки зрения хранения: память — линейная последовательность байт
- С точки зрения программы: память разбивается на сегменты по 64Кб (логическое деление)
  - Каждый сегмент имеет начальный (базовый) адрес, выровненный по границе 16 байт (4 младших бита == 0)
  - Сегменты могут пересекаться
  - Каждое приложение определяет и использует сегменты самостоятельно
  - Возможно динамическое перераспределение программы в памяти



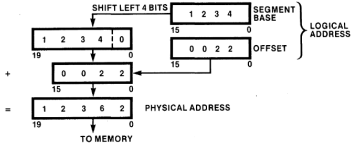
Сегменты Intel 8088



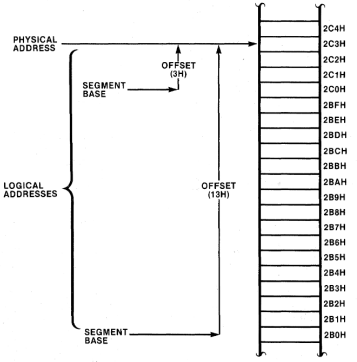
Регистры сегментов Intel 8088

Intel 8086/8088: формирование физического адреса

- Физический адрес = Сегмент * 16 + Смещение
- Осуществляется блоком BIU
- Смещение (в случае эффективного адреса) вычисляется блоком EU

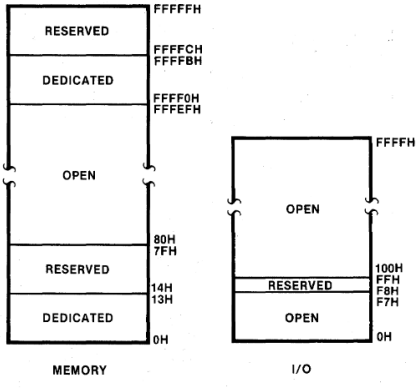


Тип ссылки	Базовый регистр по умолчанию	Альтернативный базовый регистр	Смещение
Извлечение инструкции	CS	None	IP
Операции со стеком	SS	None	SP
Данные (переменные)	DS	CS, ES, SS	Эфф. адрес
Источник строки	DS	CS, ES, SS	SI
Приемник строки	ES	None	DI
BP как базовый регистр	SS	CS, DS, ES	Эфф. адрес



Intel 8086/8088: зарезервированная память

- Intel зарезервировала часть оперативной памяти для специальных функций процессора
- Используется для прерываний и обработки системного сброса (system reset)



зарезервированная память

Intel 8086/8088: ввод/вывод

- Ввод/вывод осуществляется через нумерованные ячейки — так называемые порты ввода/вывода
- Пространство портов ввода/вывода, отдельное от памяти
  - «+»: быстрая работа, не расходуется память
  - «-»: отдельные инструкции, сложнее программировать
- Также может использоваться память для реализации портов ввода/вывода
- Пространство портов не сегментируется

Intel 8086/8088: прерывания

- Прерывание — изменение последовательности выполнения команд
  - внешние, аппаратные (при возникновении сигнала на специально выделенных для этих целей входных контактах процессора)
  - внутренние процессора (деление на ноль, трассировка программы)
  - программные (специальная инструкция, чаще всего для операций ввода/вывода или вызова функций BIOS)
- У каждого прерывания есть номер, по которому процессор определяет подпрограмму для обработки этого прерывания (обработчик прерывания)
  - Поддерживается 256 прерываний
  - Существует таблица прерываний, содержащая адреса обработчиков. Порядковый номер элемента в этой таблице соответствует номеру прерывания. Эта таблица хранится в памяти
  - Содержимое элемента таблицы прерываний (вектор прерывания) – двойное слово – адрес вызываемой процедуры (обработчика)
- Маскируемые и немаскируемые прерывания: можно наложить маску (фильтр), которая запретит поступление определенных прерываний в процессор

Intel 8086/8088: таблица прерываний, обработка аппаратных прерываний

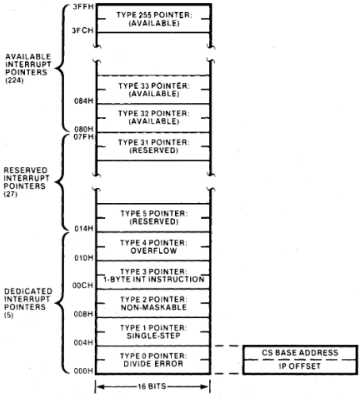
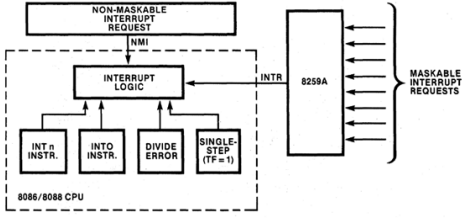


Таблица прерываний

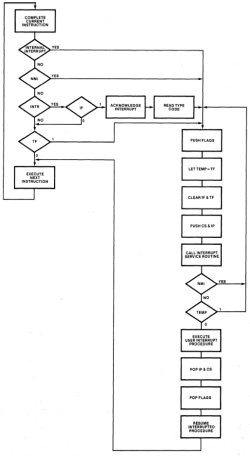
- Маскируемые прерывания поступают через контроллер прерываний
- Немаскируемые – через специальный контакт процессора



Поступление аппаратных прерываний

Intel 8086/8088: обработка прерываний

- Содержимое регистра флагов помещается на стек
- Флаги IF и TF очищаются для запрещения прерываний (очищается сигнал INTR) и возможной передачи управления отладчику
- Содержимое регистра CS помещается на стек
- Содержимое регистра IP помещается на стек
- Извлекается вектор прерывания и его содержимое помещается в регистры IP и CS соответственно. Следующей командой будет выполняться первая команда обработчика прерывания
- Возврат из обработчика прерывания должен быть осуществлен специальной инструкцией IRET, которая восстанавливает флаги и возобновляет работу с инструкции, на которую до появления прерывания указывал IP



Intel 8086/8088: аппаратные прерывания

Линия прерывания	Типичное использование
IRQ3	Последовательный порт 2
IRQ4	Последовательный порт 1
IRQ5	Параллельный порт 2
IRQ6	Драйвер НГМД
IRQ7	Параллельный порт 1
IRQ9, IRQ10, IRQ11, IRQ15	None
IRQ12	Интерфейс мыши
IRQ14	Драйвер НЖМД

Линии прерываний на шине ISA

Линия прерывания	Типичное использование
IRQ0	Системный таймер
IRQ1	Интерфейс клавиатуры
IRQ2	Прерывание от дополнительного контроллера PIC
IRQ8	Часы реального времени
IRQ13	Сопроцессор

Линии прерываний, подключенные к контроллерам прерываний

Intel 8086/8088: команды

- Арифметические и логические инструкции
- Инструкции пересылки данных
- Инструкции работы со строками (до 64K)
- Инструкции передачи управления
- Инструкции управления процессором

Intel 8086/8088: команды

Арифметические и логические инструкции

Логические	
NOT	„NOT“байта или слова
AND	„AND“байт или слов
OR	„OR“байт или слов
XOR	„Исключающее OR“байт или слов
TEST	„AND“байт или слов без записи результата
Сдвиг	
SHL/SAL	Логический и арифметический сдвиг влево
SHR	Логический сдвиг вправо
SAR	Арифметический сдвиг вправо
Циклический сдвиг	
ROL	Циклический сдвиг влево
ROR	Циклический сдвиг вправо
RCL	Циклический сдвиг влево с использованием CF
RCR	Циклический сдвиг вправо с использованием CF

Сложение	
ADD	Сложение байта или слова
ADC	Сложение с переносом (+ CF)
INC	Увеличение на 1
AAA	ACSII корректировка после сложения BCD
DAA	Десятичная корректировка после сложения BCD
Вычитание	
SUB	Вычитание
SBB	Вычитание с заёмом (- CF)
DEC	Уменьшение на 1
NEG	Изменение знака на противоположный
CMP	Сравнение
AAS	ACSII корректировка после вычитания BCD
DAS	Десятичная корректировка после вычитания BCD
Умножение	
MUL	Беззнаковое умножение
IMUL	Целочисленное умножение
AAM	ACSII корректировка после умножения BCD
Деление	
DIV	Беззнаковое деление
IDIV	Целочисленное деление
AAB	ACSII корректировка после деления BCD
CBW	Преобразование байта в слово
CWD	Преобразование слова в двойное слово

Intel 8086/8088: команды

Инструкции передачи управления

Безусловный переход	
CALL	Вызов процедуры
RET	Возврат из процедуры
JMP	Переход по адресу
Управление циклами	
LOOP	Переход по метке
LOOPE/LOOPZ	Переход по метке, если равно/ноль
LOOPE/LOOPZ	Переход по метке, если не равно/не ноль
JCXZ	Переход, если регистр CX=0
Прерывания	
INT	Вызов прерывания
INTO	Вызов прерывания, если переполнение
IRET	Возврат из прерывания

Условный переход	
JA/JNBE	Переход, если выше/не ниже не равно (беззнаковый)
JAE/JNB	Переход, если не ниже/выше или равно (беззнаковый)
JB/JNAE	Переход, если ниже/не выше или не равно (беззнаковый)
JBE/JNA	Переход, если ниже или равно/не выше (беззнаковый)
JBC	Переход, если возник перенос
JE/JZ	Переход, если равно/ноль
JG/JNLE	Переход, если больше/не меньше и не равно
JGE/JNL	Переход, если больше или равно/не меньше
JL/JNGE	Переход, если меньше/не больше и не равно
JLE/JNG	Переход, если меньше или равно/не больше
JNC	Переход, если нет переноса
JNE/JNZ	Переход, если не равно/не ноль
JNO	Переход, если нет переполнения
JNP/JPO	Переход, если нет четности/нечетная четность
JNS	Переход, если нет знака
JO	Переход, если есть переполнение
JP/JPE	Переход, если есть четность/четность четная
JS	Переход, если есть знак

Intel 8086/8088: команды

Инструкции пересылки данных

GENERAL PURPOSE	
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
XCHG	Exchange byte or word
XLAT	Translate byte
INPUT/OUTPUT	
IN	Input byte or word
OUT	Output byte or word
ADDRESS OBJECT	
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
FLAG TRANSFER	
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack

Intel 8086/8088: команды

Инструкции работы со строками

REP	Repeat
REPE/REPZ	Repeat while equal/zero
REPNE/REPNZ	Repeat while not equal/not zero
MOVS	Move byte or word string
MOVSB/MOVSW	Move byte or word string
CMPS	Compare byte or word string
SCAS	Scan byte or word string
LODS	Load byte or word string
STOS	Store byte or word string

SI	Index (offset) for source string
DI	Index (offset) for destination string
CX	Repetition counter
AL/AX	Scan value Destination for LODS Source for STOS
DF	0 = auto-increment SI, DI 1 = auto-decrement SI, DI
ZF	Scan/compare terminator

Intel 8086/8088: команды

Инструкции управления процессором

FLAG OPERATIONS	
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag
EXTERNAL SYNCHRONIZATION	
HLT	Halt until interrupt or reset
WAIT	Wait for $\overline{\text{TEST}}$ pin active
ESC	Escape to external processor
LOCK	Lock bus during next instruction
NO OPERATION	
NOP	No operation

Сопроцессор Intel 8087

- Для увеличения быстродействия при выполнении вычислений с плавающей точкой
- Собственный набор инструкций
- Вычисление экспоненты, логарифмов, тригонометрических функций
- Расширенный 80-битный формат чисел
- CPU — общее управление процессом вычислений, NPX (Numeric Processor Extension) — только свои команды

Data Type	Bits	Significant Digits (Decimal)	Approximate Range (decimal)
Word Integer	16	4	$-32{,}768 \leqslant X \leqslant +32{,}767$
Short Integer	32	9	$-2\times10^{9} \leqslant X \leqslant +2\times10^{9}$
Long Integer	64	18	$-9\times10^{18} \leqslant X \leqslant +9\times10^{18}$
Packed Decimal	80	18	$-99\ldots99 \leqslant X \leqslant +99\ldots99$ (18 digits)
Short Real*	32	6-7	$8.43\times10^{-37} \leqslant \lvert X\rvert \leqslant 3.37\times10^{38}$
Long Real*	64	15-16	$4.19\times10^{-307} \leqslant \lvert X\rvert \leqslant 1.67\times10^{308}$
Temporary Real	80	19	$3.4\times10^{-4932} \leqslant \lvert X\rvert \leqslant 1.2\times10^{4932}$