

Машинно-зависимые языки программирования, лекция 1

Каф. ИУ7 МГТУ им. Н. Э. Баумана, 2021 г.



Организация курса

- видео-, аудиозапись и фотосъёмка на занятиях запрещены
- 2 модуля + экзамен
- 8 лекций, ~12 лабораторных работ
- 38 часов самостоятельной подготовки (по учебному плану)

Литература

- Зубков С. В. "Assembler. Для DOS, Windows и Unix"



Цели и программа курса

- Изучение низкоуровневого устройства ЭВМ
- Понимание исполнения программ на аппаратном уровне. Работа процессора
- Умение составлять и читать программы на языках низкого уровня, включая:
 - написание программы на низкоуровневом языке “с нуля”;
 - взаимодействие программного кода с устройствами;
 - использование расширений процессоров;
 - отладку и реверс-инжиниринг исполняемых файлов.

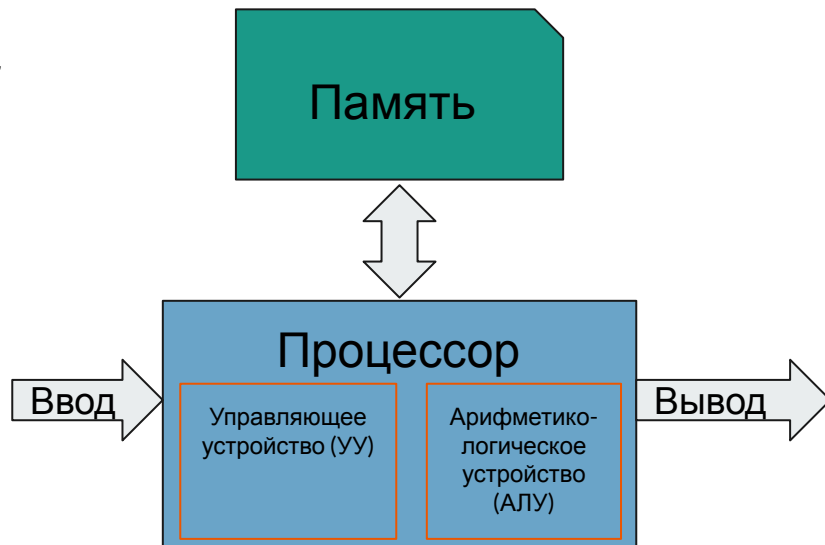
История создания ЭВМ. Появление вычислителей общего назначения.

Архитектура фон Неймана

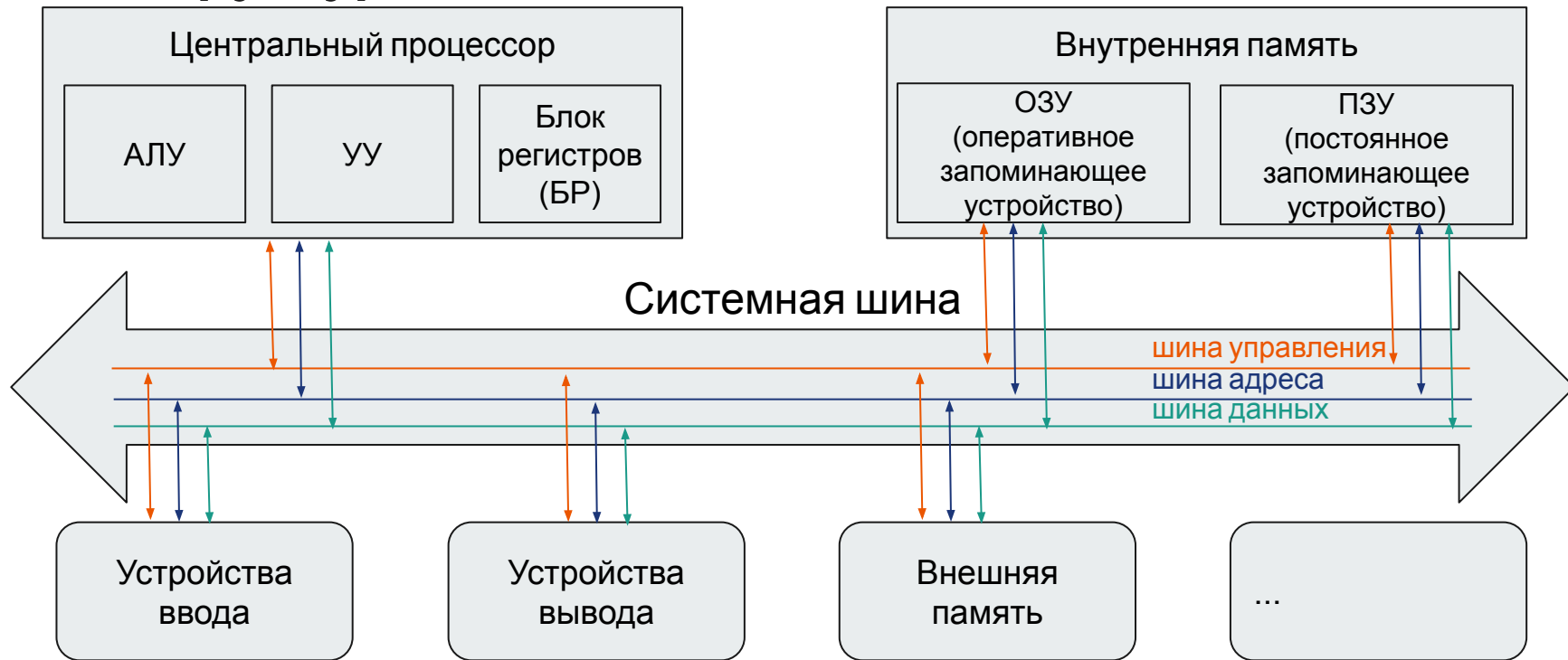
От решения частных вычислительных задач - к универсальным системам

Принципы фон Неймана:

1. Использование двоичной системы счисления в вычислительных машинах.
2. Программное управление ЭВМ.
3. Память компьютера используется не только для хранения данных, но и программ.
4. Ячейки памяти ЭВМ имеют адреса, которые последовательно пронумерованы.
5. Возможность условного перехода в процессе выполнения программы.



Структурная схема ЭВМ



Память. Единица адресации.

Минимальная адресуемая единица памяти - байт:

- 8 бит
- $2^8=256$ значений (0..255)
- $8 = 2^3$
- $256 = 2^8=10_{16}^2=100_{16}$

Машинное слово - машинно-зависимая величина, измеряемая в битах, равная разрядности регистров и шины данных

Параграф - 16 байт

ASCII (*аскú*) - *American standard code for information interchange*, США, 1963.

| ASCII Code Chart | | | | | | | | | | | | | | | | |
|------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|----|----|----|-----|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 0 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS | HT | LF | VT | FF | CR | SO | SI |
| 1 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US |
| 2 | | ! | " | # | \$ | % | & | ' | (|) | * | + | , | - | . | / |
| 3 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S | T | U | V | W | X | Y | Z | [| \ |] | ^ | _ |
| 6 | ` | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s | t | u | v | w | x | y | z | { | | } | ~ | DEL |

- 7-битная кодировка (в расширенном варианте - 8-битная)
- первые 32 символа - непечатные (служебные)
- старшие 128 символов 8-битной кодировки - национальные языки, псевдографика и т. п.



Системы счисления

Двоичная (binary)

- 0, 1, 10, 11, 100, 101...
- $2^8 = 256$
- $2^{10} = 1024$
- $2^{16} = 65536$
- Суффикс - b. Пример: 1101b

Шестнадцатеричная (hexadecimal)

- 0, 1, ..., 8, 9, A, B, C, D, E, F, 10, 11, 12, ..., 19, 1A, 1B, ...
- $2^4 = 10_{16}$
- $2^8 = 100_{16}$
- $2^{16} = 10000_{16}$
- Суффикс - h (10h - 16). Некоторые компиляторы требуют префикса 0x (0x10)

$$101101101111000_2 = B6F8_{16}$$



Представление отрицательных чисел

Знак - в старшем разряде (0 - "+", 1 - "-").

Возможные способы:

- прямой код
- обратный код (инверсия)
- **дополнительный код (инверсия и прибавление единицы)**

Примеры доп. кода на 8-разрядной сетке

-1:

1. 00000001
2. 11111110
3. 11111111

Смысл: $-1 + 1 = 0$ (хоть и с переполнением):

$$11111111 + 1 = (1)\underline{00000000}$$

-101101:

1. 00101101
2. 11010010
3. 11010011



Виды современных архитектур ЭВМ

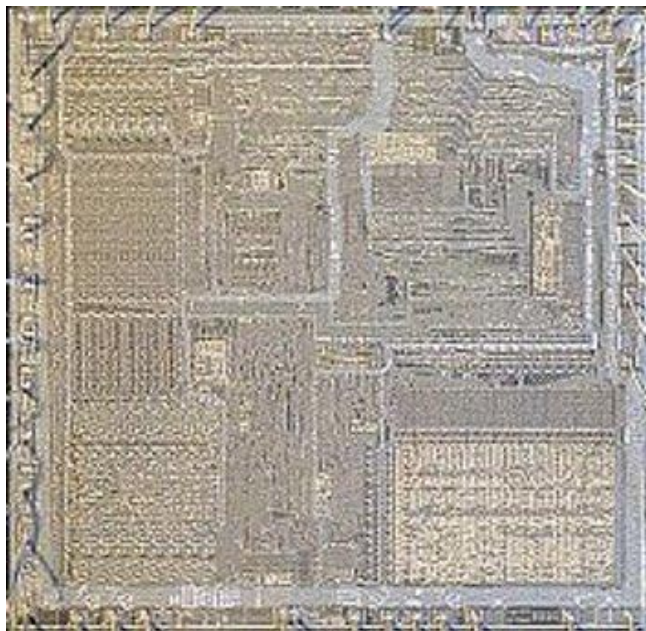
- x86-64 8086 (16-разр.) ➤ x86 (32-разр.) ➤ x86-64 (64-разр.)
- ARM
- IA64
- MIPS (включая Байкал)
- Эльбрус

Семейство процессоров x86 и x86-64

- Микропроцессор 8086: 16-разрядный, 1978 г., 5-10 МГц, 3000 нм
- Предшественники: 4004 - 4-битный, 1971 г.; 8008 - 8-битный, 1972 г.; 8080 - 1974 г.
- Требует микросхем поддержки
- 80186 - 1982 г., добавлено несколько команд, интегрированы микросхемы поддержки
- 80286 - 1982 г., 16-разрядный, добавлен защищённый режим
- 80386, 80486, Pentium, Celeron, AMD, ... - 32-разрядные, повышение быстродействия и расширение системы команд
- x86-64 (x64) - семейство с 64-разрядной архитектурой
- Отечественный аналог - K1810BM86, 1985 г.

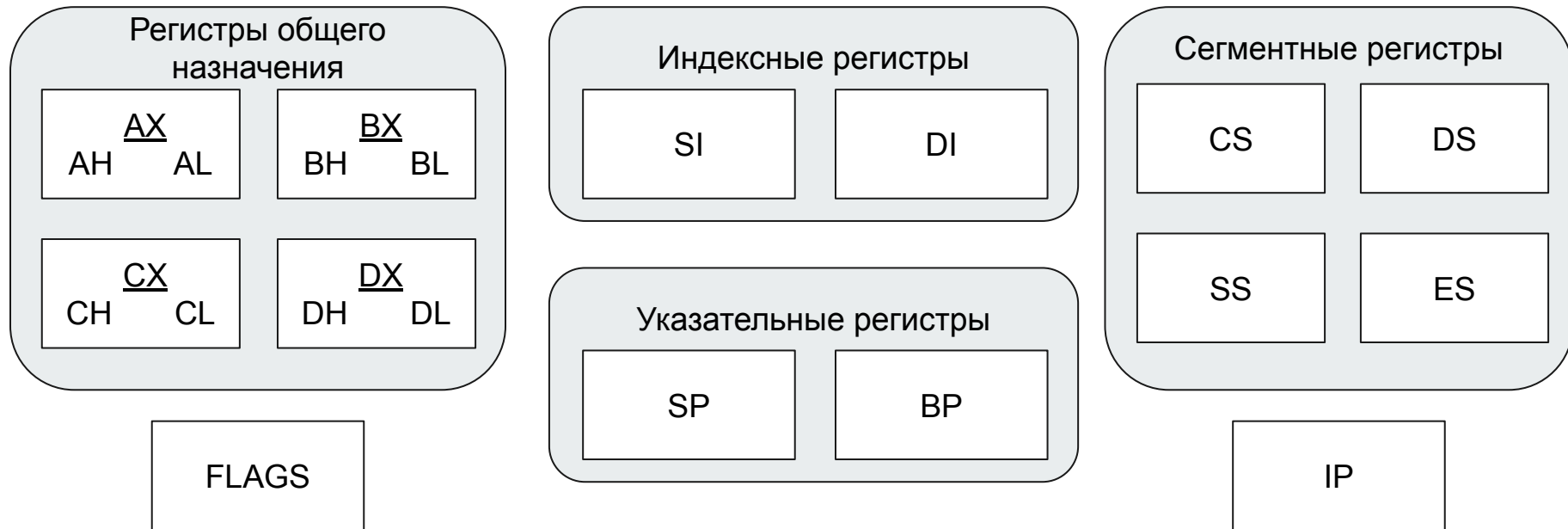


Устройство 8086



| | | | MAX MODE | (MIN MODE) |
|------|----|----|-----------------|---------------|
| GND | 1 | 40 | U _{CC} | |
| AD14 | 2 | 39 | AD15 | |
| AD13 | 3 | 38 | A16/S3 | |
| AD12 | 4 | 37 | A17/S4 | |
| AD11 | 5 | 36 | A18/S5 | |
| AD10 | 6 | 35 | A19/S6 | |
| AD9 | 7 | 34 | BHE/S7 | |
| AD8 | 8 | 33 | MN/MX | |
| AD7 | 9 | 32 | RD | |
| AD6 | 10 | 31 | RQ/GT0 | (HOLD) |
| AD5 | 11 | 30 | RQ/GT1 | (HLDA) |
| AD4 | 12 | 29 | LOCK | (WR) |
| AD3 | 13 | 28 | S2 | (M/IO) |
| AD2 | 14 | 27 | S1 | (DT/R) |
| AD1 | 15 | 26 | S0 | (DEN) |
| AD0 | 16 | 25 | QS0 | (ALE) |
| NMI | 17 | 24 | QS1 | (INTA) |
| INTR | 18 | 23 | TEST | |
| CLK | 19 | 22 | READY | |
| GND | 20 | 21 | RESET | |

Архитектура 8086 с точки зрения программиста (структура блока регистров)





Язык ассемблера

Машинная команда - инструкция (в двоичном коде) из аппаратно определённого набора, которую способен выполнять процессор.


Машинный код - система команд конкретной вычислительной машины, которая интерпретируется непосредственно процессором.

Язык ассемблера - машинно-зависимый язык программирования низкого уровня, команды которого прямо соответствуют машинным командам.



Исполняемые файлы. Компиляция. Линковка

- Исполняемый файл - файл, содержащий программу в виде, в котором она может быть исполнена компьютером (то есть в машинном коде).
- Получение исполняемых файлов обычно включает в себя 2 шага: компиляцию и линковку.
- Компилятор - программа для преобразования исходного текста другой программы на определённом языке в объектный модуль.
- компоновщик (линковщик, линкер) - программа для связывания нескольких объектных файлов в исполняемый.



Исполняемые файлы. Запуск программы.

Отладчик

- В DOS и Windows - расширения .EXE и .COM
- Последовательность запуска программы операционной системой:
 1. Определение формата файла.
 2. Чтение и разбор заголовка.
 3. Считывание разделов исполняемого модуля (файла) в ОЗУ по необходимым адресам.
 4. Подготовка к запуску, если требуется (загрузка библиотек).
 5. Передача управления на точку входа.
- Отладчик - программа для автоматизации процесса отладки. Может выполнять трассировку, отслеживать, устанавливать или изменять значения переменных в процессе выполнения кода, устанавливать или удалять контрольные точки или условия остановки.



“Простейший” формат исполняемого файла

.COM (command) - простейший формат исполняемых файлов DOS и ранних версий Windows:

- не имеет заголовка;
- состоит из одной секции, не превышающей 64 Кб;
- загружается в ОЗУ без изменений;
- начинает выполняться с 1-го байта (точка входа всегда в начале).

Последовательность запуска COM-программы:

1. Система выделяет свободный *сегмент* памяти нужного размера и заносит его адрес во все сегментные регистры (CS, DS, ES, FS, GS, SS).
2. В первые 256 (100h) байт этого сегмента записывается служебная структура DOS, описывающая программу - PSP.
3. Непосредственно за ним загружается содержимое COM-файла без изменений.
4. Указатель стека (регистр SP) устанавливается на конец сегмента.
5. В стек записывается 0000h (начало PSP - адрес возврата для возможности завершения командой ret).
6. Управление передаётся по адресу CS:0100h.



Классификация команд процессора 8086

- Команды пересылки данных
- Арифметические и логические команды
- Команды переходов
- Команды работы с подпрограммами
- Команды управления процессором



Команда пересылки данных MOV

MOV <приёмник>, <источник>

Источник: непосредственный операнд (константа, включённая в машинный код), РОН, сегментный регистра, переменная (ячейка памяти).

Приёмник: РОН, сегментный регистр, переменная (ячейка памяти).

- MOV AX, 5
- MOV BX, DX
- MOV [1234h], CH
- MOV DS, AX

- 
- MOV [0123h], [2345h]
 - MOV DS, 1000h



Целочисленная арифметика (основные команды)

- ADD <приёмник>, <источник> - выполняет арифметическое сложение приёмника и источника. Сумма помещается в приёмник, источник не изменяется.
- SUB <приёмник>, <источник> - арифметическое вычитание источника из приёмника.
- MUL <источник> - беззнаковое умножение. Умножаются источник и AL/AX, в зависимости от размера источника. Результат помещается в AX либо DX:AX.
- DIV <источник> - целочисленное беззнаковое деление. Делится AL/AX на источник. Результат помещается в AL/AX, остаток - в AH/DX.
- INC <приёмник> - инкремент на 1
- DEC <приёмник> - декремент на 1



Побитовая арифметика (основные команды)

- AND <приёмник>, <источник> - побитовое “И”. AND al, 00001111b
- OR <приёмник>, <источник> - побитовое “ИЛИ”. OR al, 00001111b
- XOR <приёмник>, <источник> - побитовое исключающее “ИЛИ”. XOR AX, AX
- NOT <приёмник> - инверсия



Команда безусловной передачи управления JMP

JMP <операнд>

- Передаёт управление в другую точку программы (на другой адрес памяти), не сохраняя какой-либо информации для возврата.
- Операнд - непосредственный адрес (вставленный в машинный код), адрес в регистре или адрес в переменной.



Команда NOP (no operation)

- Ничего не делает
- Занимает место и время
- Размер - 1 байт, код - 90h
- Назначение - задержка выполнения либо заполнение памяти, например, для *выравнивания*

Пример

...

XOR AX, AX

MOV BX, 5

label1:

INC AX

ADD BX, AX

JMP label 1



| | | | | | | | |
|-------|--------|----|------|-----|------|----------|------|
| AX | 0000 | SI | 0000 | CS | 19F5 | IP | 0100 |
| BX | 0000 | DI | 0000 | DS | 19F5 | | |
| CX | 0024 | BP | 0000 | ES | 19F5 | HS | 19F5 |
| DX | 0000 | SP | FFFE | SS | 19F5 | FS | 19F5 |
| CMD > | | | | | | | |
| 0100 | 33C0 | | | XOR | | AX, AX | |
| 0102 | BB0500 | | | MOV | | BX, 0005 | |
| 0105 | 40 | | | INC | | AX | |
| 0106 | 03D8 | | | ADD | | BX, AX | |
| 0108 | EBFB | | | JMP | | 0105 | |
| 010A | BA1401 | | | MOV | | DX, 0114 | |
| 010D | CD21 | | | INT | | 21 | |
| 010F | B44C | | | MOV | | AH, 4C | |

Взаимодействие программы с внешней средой (ОС, пользователь, ...)

Прерывания - аппаратный механизм для приостановки выполнения текущей программы и передачи управления специальной программе - обработчику прерывания.

Основные виды:

- аппаратные
- программные

`int <номер>` - вызов (генерация прерывания)

21h - прерывание DOS, предоставляет прикладным программам около 70 различных функций (ввод, вывод, работа с файлами, завершение программы и т.д.)

Номер функции прерыванию 21h передаётся через регистр АН. Параметры для каждой функции передаются собственным способом, он описан в документации. Там же описан способ возврата результата из функции в программу.



Память в реальном режиме работы процессора

Реальный режим работы - режим совместимости современных процессоров с 8086.

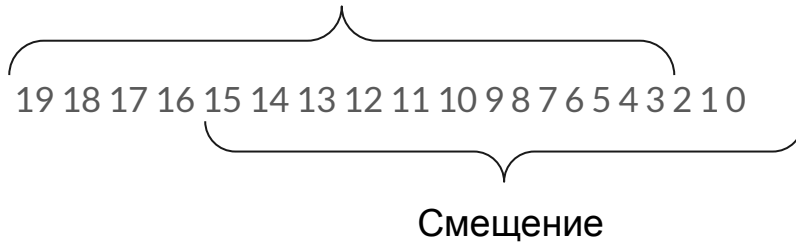
Доступен 1 Мб памяти (2^{20} байт), то есть разрядность шины адреса - 20 разрядов.

Физический адрес получается **сложением** адреса **начала сегмента** (на основе сегментного регистра) и **смещения**.

Сегментный регистр хранит в себе **старшие 16 разрядов** (из 20) адреса начала сегмента. 4 младших разряда в адресе начала сегмента всегда нулевые. Говорят, что сегментный регистр содержит в себе **номер параграфа начала сегмента**.

Память в реальном режиме работы процессора - пример

Номер параграфа начала сегмента



[SEG]:[OFFSET] => физический адрес:

1. SEG необходимо побитово сдвинуть на 4 разряда влево (или умножить на 16, что тождественно)
2. К результату прибавить OFFSET

5678h:1234h =>

$$\begin{array}{r} 56780 \\ + 1234 \\ \hline 579B4 \end{array}$$

Вычисление физического адреса выполняется процессором аппаратно, без участия программиста.

Распространённые пары регистров: CS:IP, DS:BX, SS:SP



Структура памяти программы. Виды сегментов. Назначение отдельных сегментных регистров

- Сегмент кода - регистр CS. Командой MOV изменить невозможно, меняется автоматически по мере выполнения команд.
- Сегмент данных. Основной регистр - DS, при необходимости дополнительных сегментов данных задействуются ES, FS, GS.
- Сегмент стека - регистр SS