© 2023 МГУ/ВМК/СП

Лекция 2

11 февраля

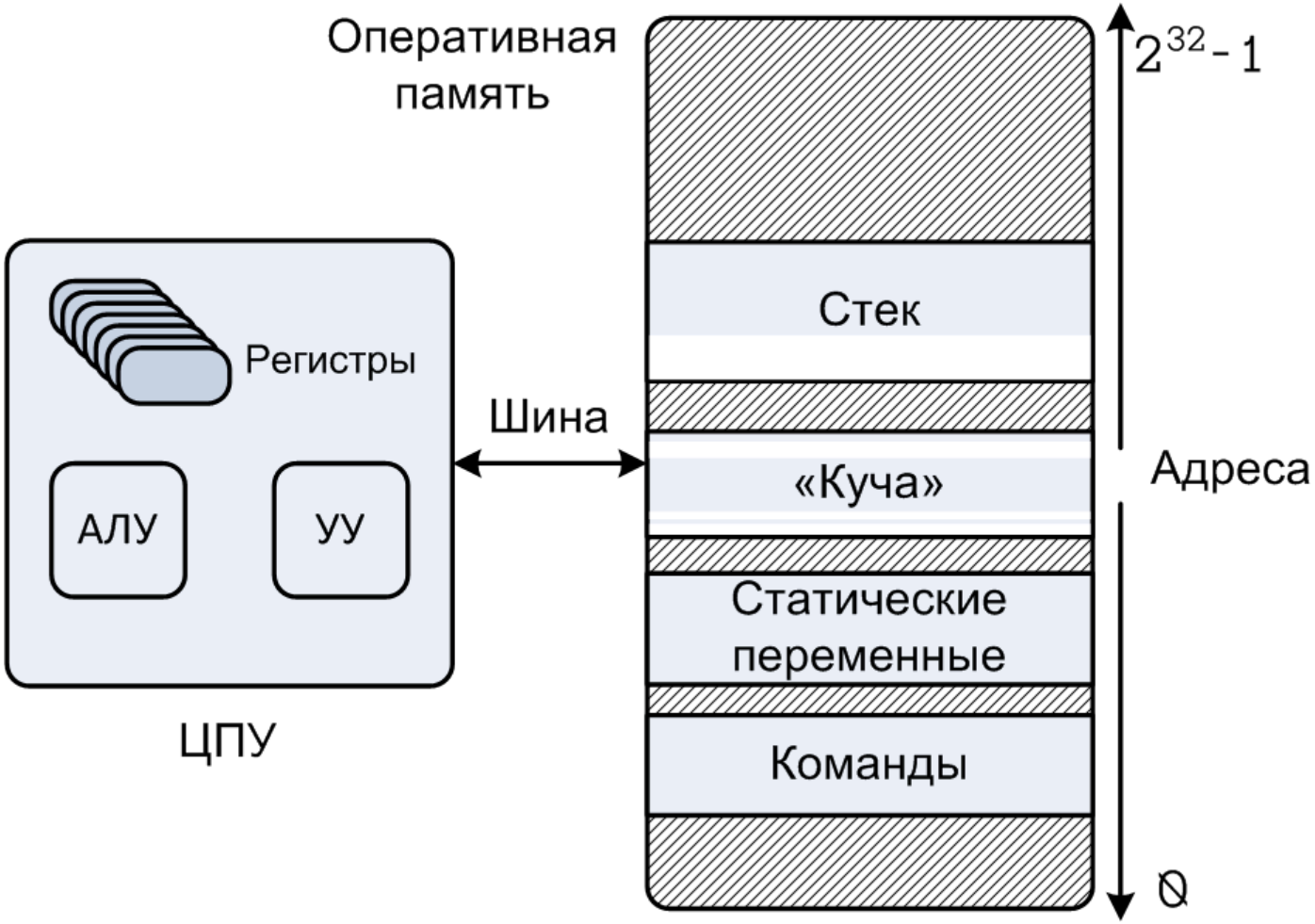
© 2023 МГУ/ВМК/СП

Промежуточные итоги: ключевые термины и понятия

- Регистр, имя регистра, значение регистра, разрядность регистра, машинное слово
- Память, ячейка памяти, адрес ячейки памяти, время доступа к памяти
- Цикл, тактовая частота, счетчик команд
- Машинная команда, код операции, операнд, адресность команды
  Исполнительный (действительный) адрес
  англ. *effective (executive) address*
- Способ адресации
  прямая, непосредственная, косвенная

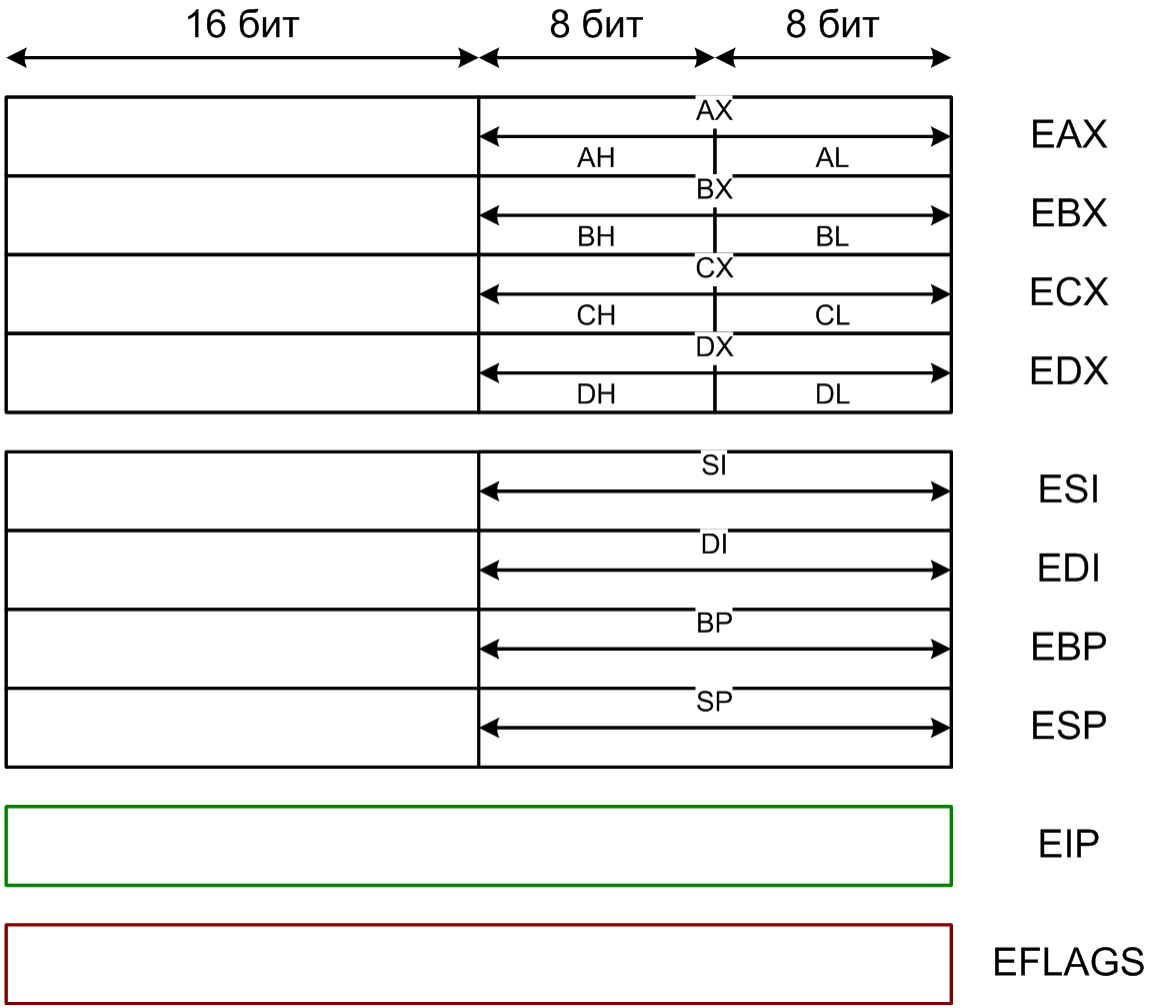
© 2023 МГУ/ВМК/СП

Машина, на которой работает пользовательская программа (архитектура IA-32)



Вспоминаем Си-машину…

© 2023 МГУ/ВМК/СП



Типы операндов

- Регистр r
- Память m
- Константа i

Размеры операндов

- byte 8
- word 16
- dword 32
- qword 64

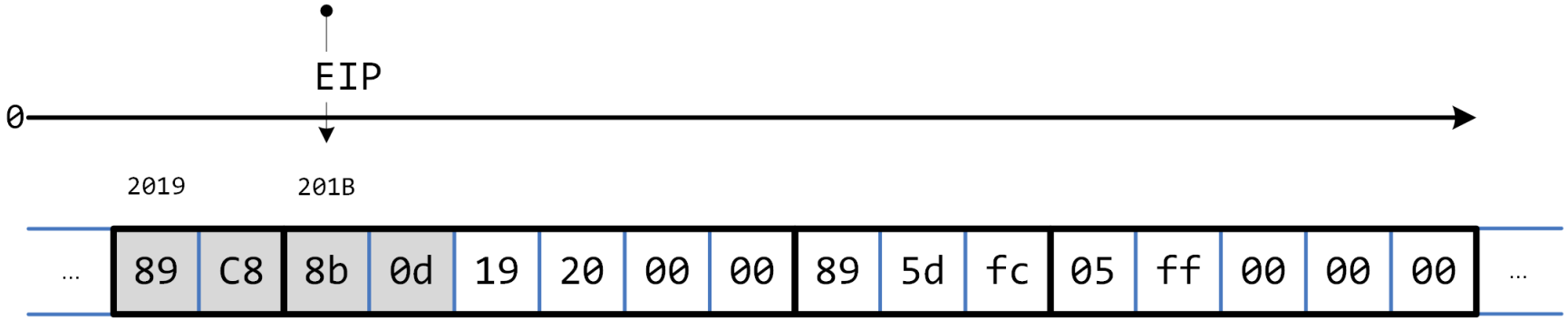
Где сегментные регистры?

```
89 c8
8b 0d 19 20 00 00
89 5d fc
05 ff 00 00 00
```

```
mov eax, ecx
mov ecx, dword [0x2019]
mov dword [ebp-4], ebx
add eax, 0xff
```

© 2023 МГУ/ВМК/СП

Порядок размещения байт в памяти




```
mov ecx, dword [0x2019]
```

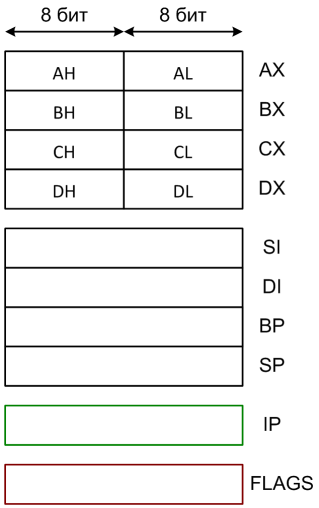
Два наиболее распространенных подхода

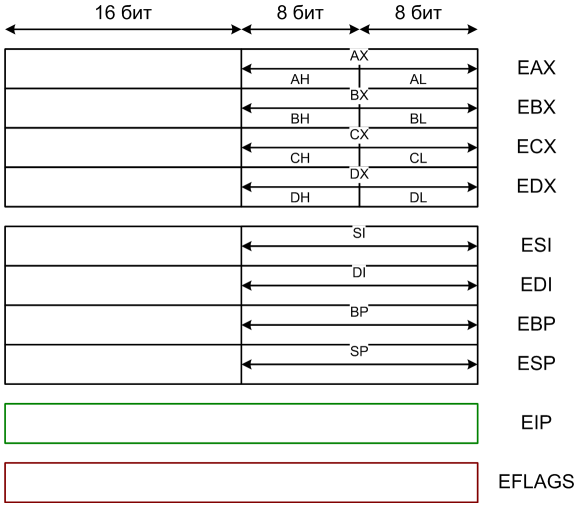
- Порядок от младшего к старшему (*англ. little-endian*) ***используется в IA-32***

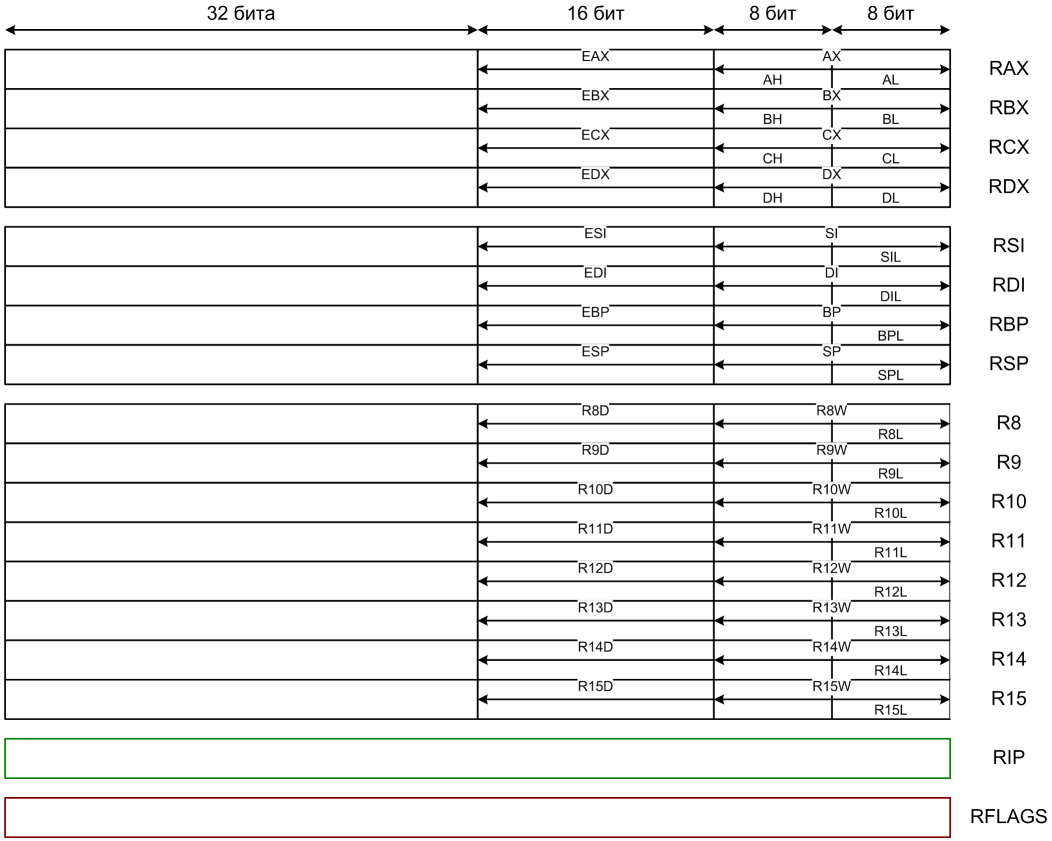
- Порядок от старшего к младшему (*англ. big-endian*) как правило используется в процессорах, предназначенных для обработки сетевых данных

© 2023 МГУ/ВМК/СП

8086 → IA-32→ Intel64





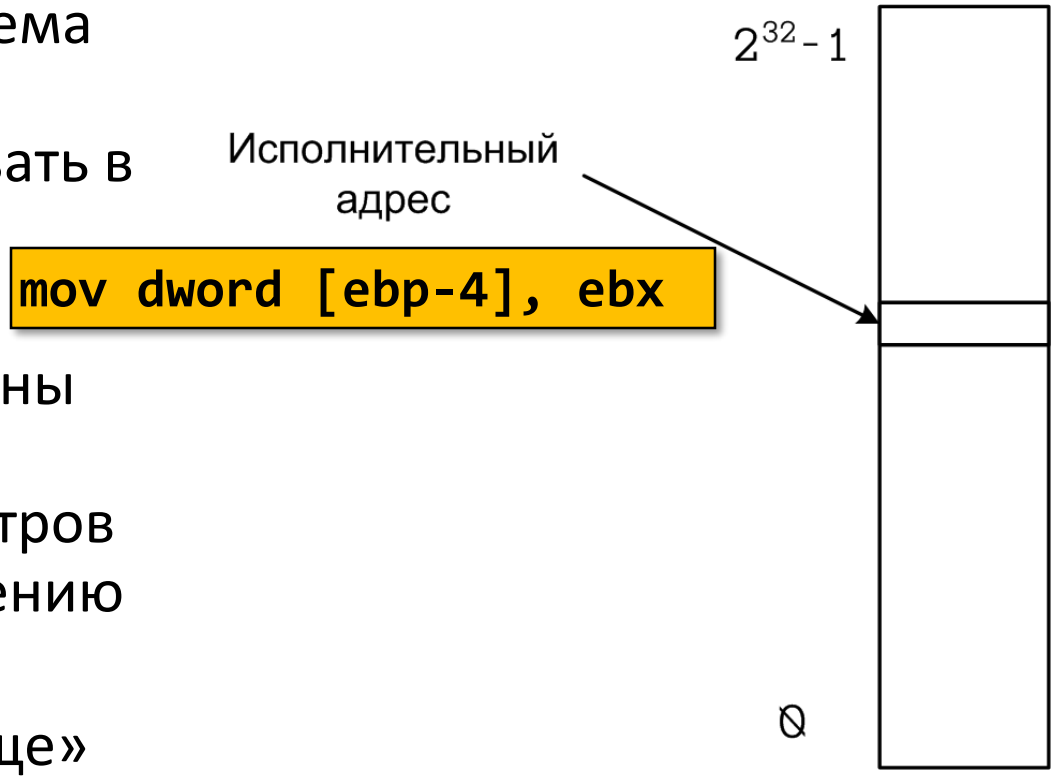


Какие еще названия встречаются?
x86, x86_64, AMD64
IA-64 совсем другая архитектура

© 2023 МГУ/ВМК/СП

«Положительные» особенности IA-32

- Особенности аппаратуры и операционная система (Windows/Linux) позволяют использовать в программах модель плоской памяти
- Значительно ослаблены ограничения на использование регистров в командах по сравнению с 8086
- «Количественно проще» чем Intel64



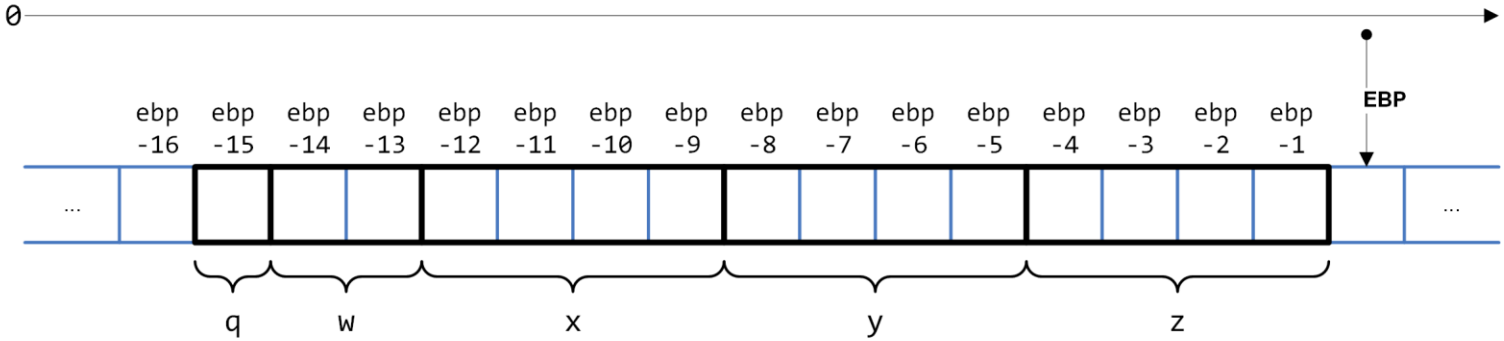
© 2023 МГУ/ВМК/СП

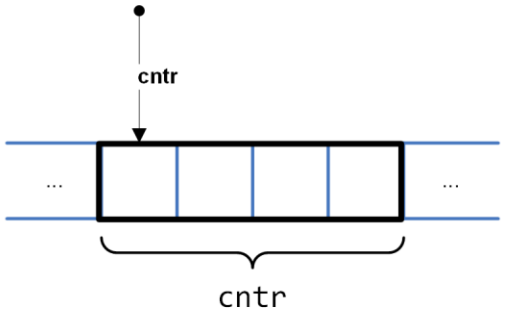
```
void f() {
   static int cntr = 0;      // 1
   int x = 2, y = 1, z = 0;  // 2
   unsigned short w = 100;   // 3
   signed char q = 10;       // 4
   ++cntr;                   // 5
   z = -x + q * w *y - w;    // 6
}
```

```
section .bss
; Резервирование 4 байт памяти
   cntr  resd  1
section .text
   global f
; Точка входа в программу
f:
   push    ebp
   mov     ebp, esp
   sub     esp, 16
   mov     dword [ebp-12], 2     ; (1)
   mov     dword [ebp-8], 1      ; (2)
   mov     dword [ebp-4], 0      ; (3)
   mov     word [ebp-14], 100    ; (4)
   mov     byte [ebp-15], 10     ; (5)
   add     dword [cntr], 1       ; (6)
   movsx   eax, byte [ebp-15]    ; (7)
   movzx   edx, word [ebp-14]    ; (8)
   imul    eax, edx              ; (9)
   imul    eax, dword [ebp-8]    ; (10)
   sub     eax, dword [ebp-12]   ; (11)
   sub     eax, edx              ; (12)
   mov     dword [ebp-4], eax    ; (13)
   leave
   ret
```

© 2023 МГУ/ВМК/СП

Размещение переменных в памяти

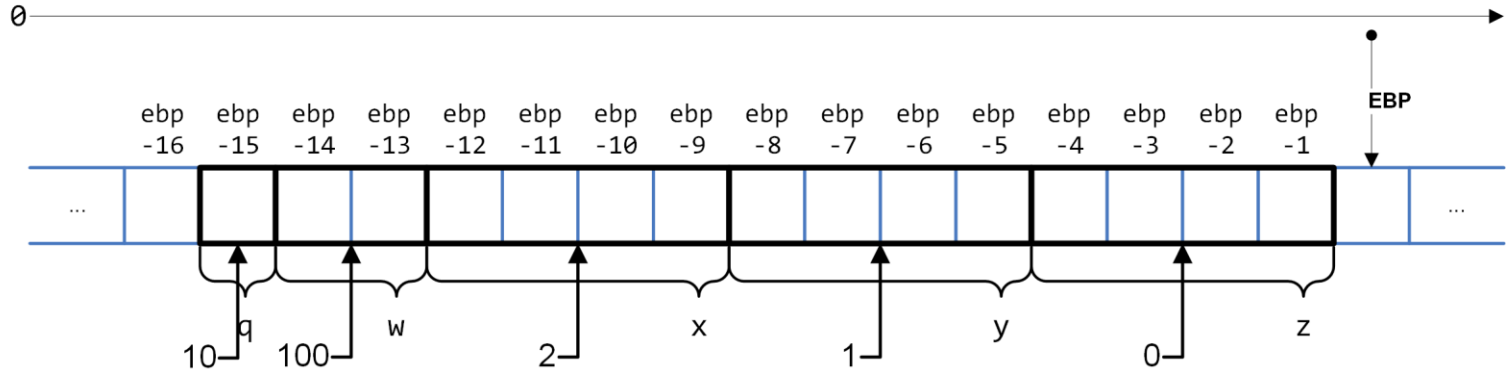


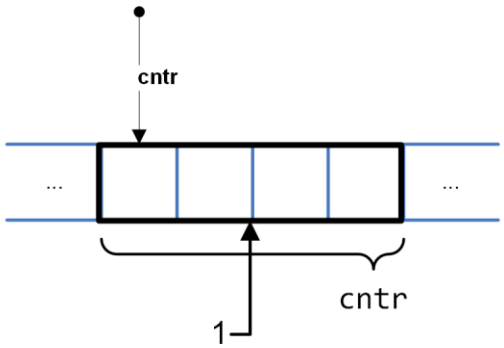


Модели памяти в языке Си

- Автоматическая
  - Каждый вызов функции должен обладать своим комплектом автоматических локальных переменных. Адрес переменной должен определяться динамически, во время работы программы.
- Статическая
  - Один экземпляр переменной. Адрес может быть определен статически, во время компиляции.
- Динамическая
  - Функции стандартной библиотеки языка Си: `malloc` и др.

© 2023 МГУ/ВМК/СП






```
...
   mov     dword [ebp-12], 2    ; (1)
   mov     dword [ebp-8], 1     ; (2)
   mov     dword [ebp-4], 0     ; (3)
   mov     word [ebp-14], 100   ; (4)
   mov     byte [ebp-15], 10    ; (5)
   add     dword [cntr], 1      ; (6)
   movsx   eax, byte [ebp-15]   ; (7)
   movzx   edx, word [ebp-14]   ; (8)
   imul    eax, edx             ; (9)
   imul    eax, dword [ebp-8]   ; (10)
   sub     eax, dword [ebp-12]  ; (11)
   sub     eax, edx             ; (12)
   mov     dword [ebp-4], eax   ; (13)
...
```

```
void f() {
   static int cntr = 0;       // 1
   int x = 2, y = 1, z = 0;   // 2
   unsigned short w = 100;    // 3
   signed char q = 10;        // 4
   ++cntr;                    // 5
   z = -x + q * w *y - w;     // 6
}
```