Лекция 7

29 февраля

Типы данных языка Си

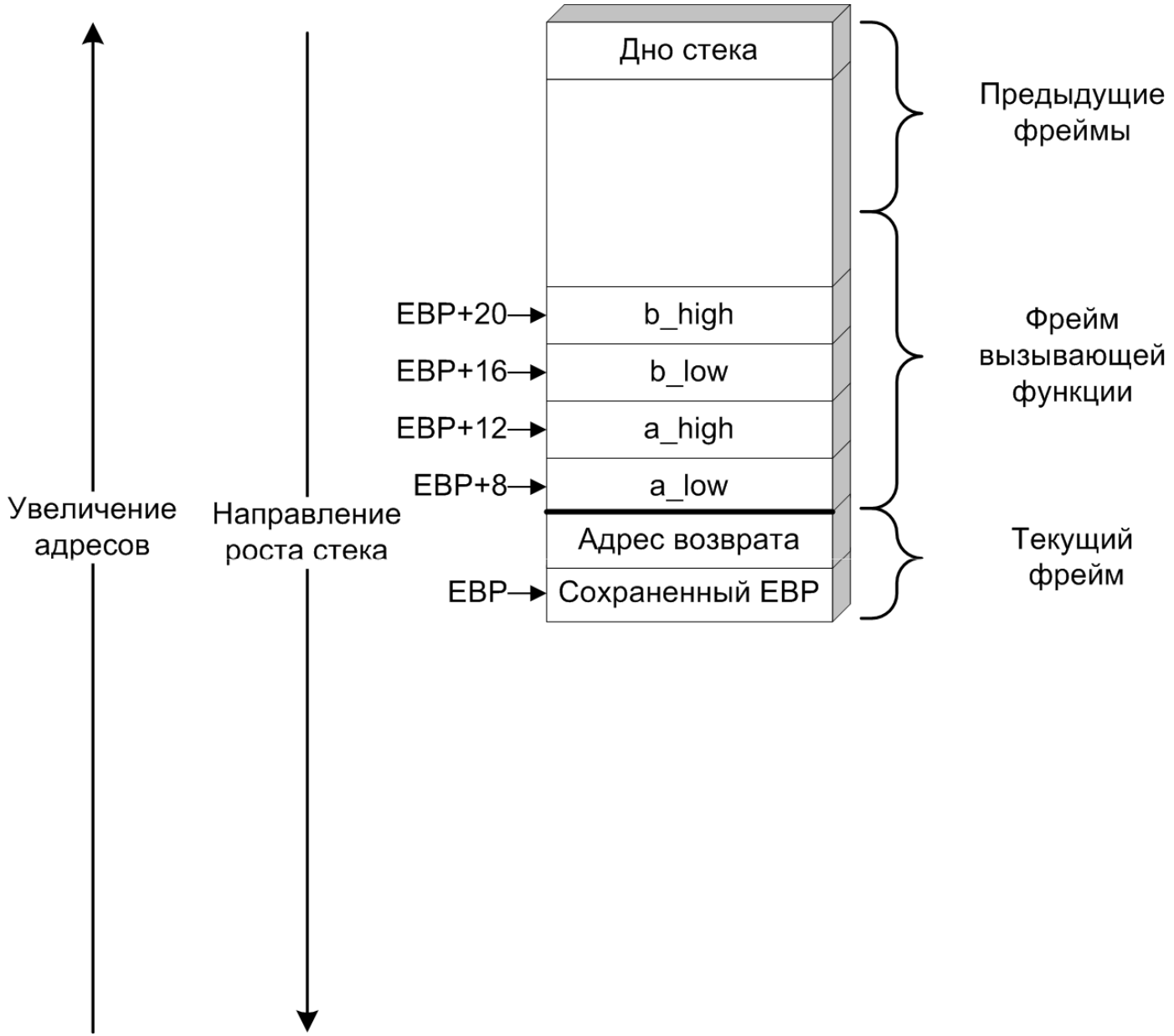
- **`char`**
- **Стандартные знаковые целочисленные типы**
  - **`signed char`**
  - **`short int`**
  - **`int`**
  - **`long int`**
  - **`long long int`**
- **Стандартные беззнаковые целочисленные типы**
  - **`_Bool`**
- **Перечисление**

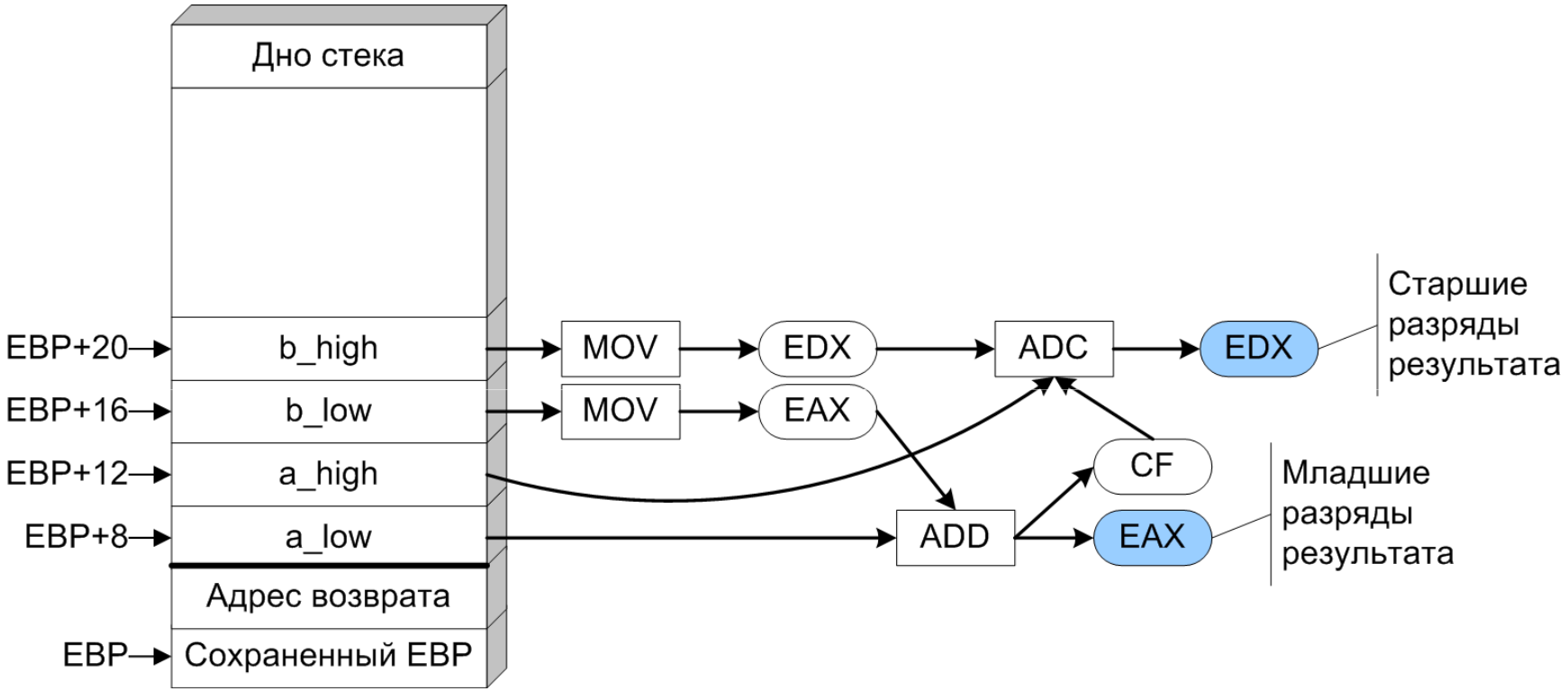
- Типы чисел с плавающей точкой
  - `float`
  - `double`
  - `long double`
  - `_Complex`
- Производные типы
  - *Массивы*
  - Структуры
  - Объединения
  - **Указатели**
  - Указатели на функции

Сложение long long int

```
long long f1(long long a, long long b) {
   long long c;
   c = a + b;
   return c;
}
```

```
; начало функции пропущено
        mov     eax, dword [ebp+16] ; (1)
        mov     edx, dword [ebp+20] ; (2)
        add     eax, dword [ebp+8]  ; (3)
        adc     edx, dword [ebp+12] ; (4)
; конец функции пропущен
```

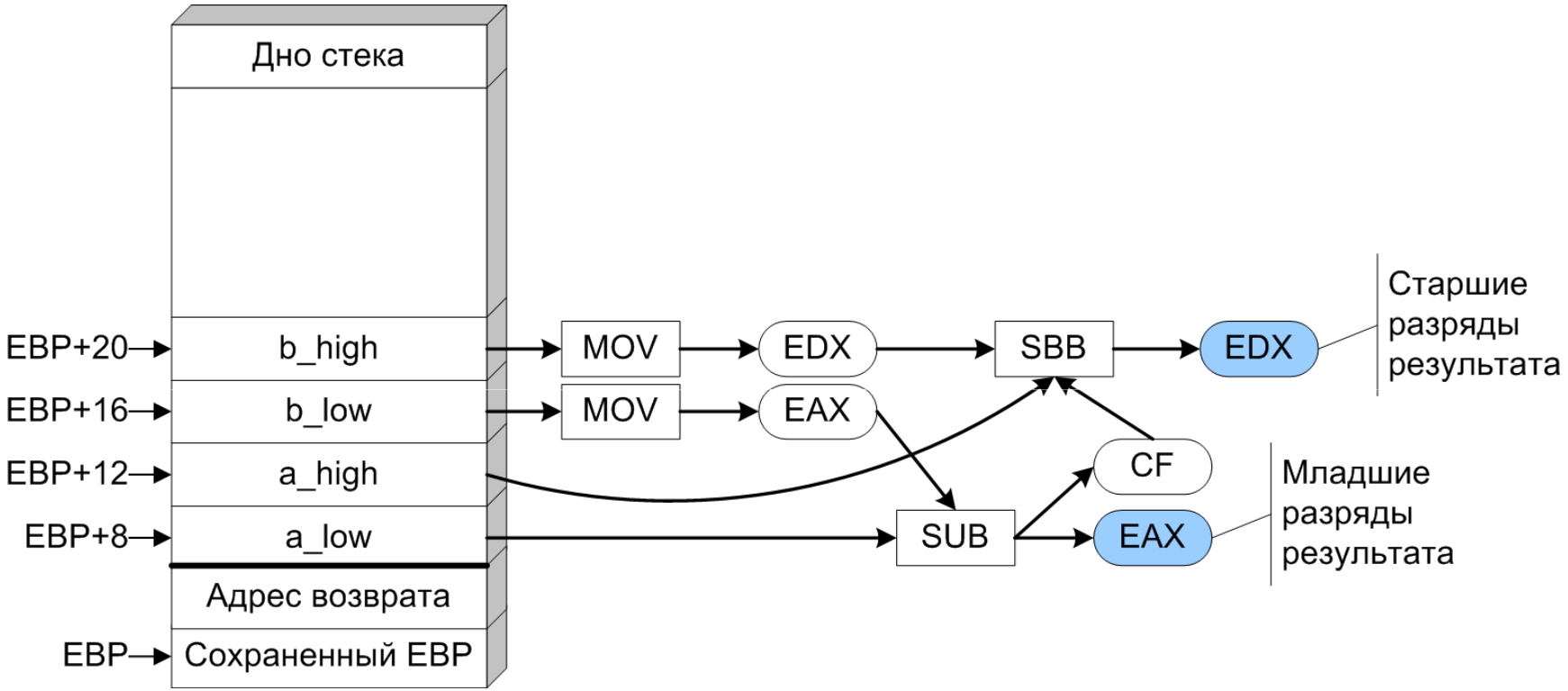




Вычитание long long int

```
long long f3(long long a, long long b) {
   long long c;
   c = a - b;
   return c;
}
```

```
; начало функции пропущено
        mov     eax, dword [ebp+8]  ; (1)
        mov     edx, dword [ebp+12] ; (2)
        sub     eax, dword [ebp+16] ; (3)
        sbb     edx, dword [ebp+20] ; (4)
; конец функции пропущен
```



J*cc*	Условие	Описание
JE	ZF	Равно / Ноль
JNE	~ZF	Не равно / Не ноль
JS	SF	Отрицательное число
JNS	~SF	Неотрицательное число
JG	~(SF^OF)&~ZF	Больше (знаковые числа)
JGE	~(SF^OF)	Больше либо равно (знаковые числа)
JL	(SF^OF)	Меньше (знаковые числа)
JLE	(SF^OF)|ZF	Меньше либо равно (знаковые числа)
JA	~CF&~ZF	Больше (числа без знака)
JB	CF	Меньше (числа без знака)

```
int absdiff(int x, int y) {
    int result;
    if (x > y) {
      result = x-y;
    } else {
      result = y-x;
    }
    return result;
}
```

```
absdiff:
   push  ebp
   mov   ebp, esp
   mov   edx, dword [8 + ebp]  ; (1)
   mov   eax, dword [12 + ebp] ; (2)
   cmp   edx, eax              ; (3)
   jle   .L6                   ; (4)
   sub   edx, eax              ; (5)
   mov   eax, edx              ; (6)
   jmp   .L7                   ; (7)
.L6:                           ; (8)
   sub eax, edx                ; (9)
.L7:                           ; (10)
   pop ebp
   ret
```

```
int goto_ad(int x, int y) {
  int result;
  if (x <= y) goto Else;
  result = x-y;
  goto Exit;
Else:
  result = y-x;
Exit:
  return result;
}
```

```
absdiff:
   push  ebp
   mov   ebp, esp
   mov   edx, dword [8 + ebp]  ; (1)
   mov   eax, dword [12 + ebp] ; (2)
   cmp   edx, eax              ; (3)
   jle   .L6                   ; (4)
   sub   edx, eax              ; (5)
   mov   eax, edx              ; (6)
   jmp   .L7                   ; (7)
.L6:                           ; (8)
   sub eax, edx                ; (9)
.L7:                           ; (10)
   pop ebp
   ret
```

Условная передача данных

```
val = Test ? Then_Expr : Else_Expr;

val = x>y ? x-y : y-x;
```

```
   nt = !(Test);
   if (nt) goto Else;
   val = Then_Expr;
   goto Done;
Else:
   val = Else_Expr;
Done:
   ...
```

```
tmp_val = Then_Expr;
val = Else_Expr;
t = Test;
if (t) val = tmp_val;
```

```
int absdiff(int x, int y) {
    int result;
    if (x > y) {
      result = x-y;
    } else {
      result = y-x;
    }
    return result;
}
```

Регистр	Значение
edi	x
esi	y

```
absdiff:
   ...
   mov edx, edi
   sub edx, esi     ; tmp_val:edx = x-y
   mov eax, esi
   sub eax, edi     ; result:eax = y-x
   cmp edi, esi     ; Compare x:y
   cmovg eax, edx   ; If >, result:eax = tmp_val:edx
   ...
```

Оператор do-while

```
int pcount_do(unsigned x) {
  int result = 0;
  do {
    result += x & 0x1;
    x >>= 1;
  } while (x);
  return result;
}
```

```
int pcount_do(unsigned x) {
  int result = 0;
loop:
  result += x & 0x1;
  x >>= 1;
  if (x)
    goto loop;
  return result;
}
```

Оператор do-while

Регистр	Значение
edx	**x**
ecx	**result**

```
int pcount_do(unsigned x) {
  int result = 0;
loop:
  result += x & 0x1;
  x >>= 1;
  if (x)
    goto loop;
  return result;
}
```

```
   mov ecx, 0     ;   result = 0
.L2:              ; loop:
   mov eax, edx
   and eax, 1     ;   t = x & 1
   add ecx, eax   ;   result += t
   shr edx, 1     ;   x >>= 1
   jne .L2        ;   If !0, goto loop
```

Оператор while

```
int pcount_while(unsigned x) {
  int result = 0;
  while (x) {
    result += x & 0x1;
    x >>= 1;
  }
  return result;
}
```

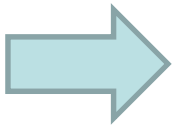
```
int pcount_do(unsigned x) {
  int result = 0;
  if (!x) goto done;
loop:
  result += x & 0x1;
  x >>= 1;
  if (x)
    goto loop;
done:
  return result;
}
```

```
int pcount_do(unsigned x) {
  int result = 0;
loop:
  if (!x) goto done;
  result += x & 0x1;
  x >>= 1;
  goto loop;
done:
  return result;
}
```

Оператор for

```
#define WSIZE 8*sizeof(int)

int pcount_for(unsigned x) {
  int i;
  int result = 0;
  for (i = 0; i < WSIZE; i++) {
    unsigned mask = 1 << i;
    result += (x & mask) != 0;
  }
  return result;
}
```



```
int pcount_for_gt(unsigned x) {
  int i;
  int result = 0;
  i = 0;
  if (!(i < WSIZE))
    goto done;
 loop:
  {
    unsigned mask = 1 << i;
    result += (x & mask) != 0;
  }
  i++;
  if (i < WSIZE)
    goto loop;
 done:
  return result;
}
```