Лекция 13

24 марта

Размещение параметров и возвращаемого значения

- sizeof == 4
  - Целиком занимает машинное слово, помещаемое на стек
  - EAX
- sizeof < 4
  - Занимает младшие байты слова на стеке
  - AX, AL
- sizeof == 8 (long long)
  - Два машинных слова в естественном порядке
  - EAX:EDX
- Массивы ≡ указатели
- Структуры
  - ???

Какие есть компиляторы и соглашения о вызове функций?

- Microsoft
- Borland
- Watcom
- Gnu
- Intel

- Digital Mars
- Codeplay

- Модели памяти
  - 16, 32, 64 разряда

- Операционные системы
  - Linux (FreeBSD, …)
  - Windows
  - Mac OS

Agner Fog http://www.agner.org/optimize/

Параметр – структура

```
typedef struct {
   short a;
   short b;
   short c;
} t_short;

short g(t_short x, t_short y) {
   return x.a * x.b - x.c + y.a;
}
```

```
g:
   push    ebp
   mov     ebp, esp
   movzx   eax, word [ebp+10]
   imul    ax,  word [ebp+8]
   sub     ax,  word [ebp+12]
   add     ax,  word [ebp+16]
   pop     ebp
   ret
```

Возвращаемое значение - структура

```
typedef struct {
  int x;
  int y;
  int z;
} triple;

triple f2(int a, int b, int c) {
   triple v = {a, b, c};
   return v;
}

void f(triple *p) {
   *p = f2(1, 2, 3);
}
```

```
f:
  push    ebp                    ; (1)
  mov     ebp, esp               ; (2)
  sub     esp, 24                ; (3)
  mov     eax, 1                 ; (4)
  mov     ecx, 3                 ; (5)
  mov     edx, 2                 ; (6)
  mov     dword [esp+4], eax     ; (7)
  mov     dword [esp+12], ecx    ; (8)
  mov     dword [esp+8], edx     ; (9)
  mov     eax, dword [ebp+8]     ; (10)
  mov     dword [esp], eax       ; (11)
  call    f2                     ; (12)
  sub     esp, 4                 ; (13)
  leave                          ; (14)
  ret                            ; (15)
```

Возвращаемое значение - структура

```
typedef struct {
  int x;
  int y;
  int z;
} triple;

triple f2(int a, int b, int c) {
   triple v = {a, b, c};
   return v;
}

void f(triple *p) {
   *p = f2(1, 2, 3);
}
```

```
f2:
   push    ebp                    ; (1)
   mov     ebp, esp               ; (2)
   mov     eax, dword [ebp+8]     ; (3)
   mov     edx, dword [ebp+20]    ; (4)
   mov     dword [eax+8], edx     ; (5)
   mov     edx, dword [ebp+16]    ; (6)
   mov     dword [eax+4], edx     ; (7)
   mov     edx, dword [ebp+12]    ; (8)
   mov     dword [eax], edx       ; (9)
   pop     ebp                   ; (10)
   ret     4                     ; (11)
```

Функция main

```
#include <stdio.h>

void nullify(int argc, char* argv[]) {
}

int main(int argc, char* argv[]) {
   nullify(argc, argv);
   return 0;
}
```

```
CMAIN:
   lea     ecx, [esp+4]  ; (1)
   and     esp, -16      ; (2)
   push    dword [ecx-4] ; (3)
   push    ebp           ; (4)
   mov     ebp, esp      ; (5)
   push    ecx           ; (6)
   sub     esp, 20       ; (7)
   ; ...

nullify:
        ret
```

Функция main

```
#include <stdio.h>

void nullify(int argc, char* argv[]) {
}

int main(int argc, char* argv[]) {
   nullify(argc, argv);
   return 0;
}
```

```
CMAIN:
   ; …
   mov      eax, dword [ecx+4] ; (8)
   mov      dword [esp+4], eax ; (9)
   mov      eax, dword [ecx]   ; (10)
   mov      dword [esp], eax   ; (11)
   call     nullify            ; (12)
   ; ...

nullify:
         ret
```

Функция main

```
#include <stdio.h>

void nullify(int argc, char* argv[]) {
}

int main(int argc, char* argv[]) {
   nullify(argc, argv);
   return 0;
}
```

```
CMAIN:
   ; …
   mov     eax, 0        ; (13)
   add     esp, 20       ; (14)
   pop     ecx           ; (15)
   pop     ebp           ; (16)
   lea     esp, [ecx-4]  ; (17)
   ret

nullify:
        ret
```

Пример вызова malloc

```
#include <stdlib.h>

struct chain;

typedef struct chain {
   int val;
   struct chain *next;
} t_chain, *p_chain;
```

```
p_chain insert(p_chain p, int val) {
   if ((0 == p) || (p->val > val)) {
      p_chain np =
         (p_chain)malloc(sizeof(t_chain));
      np->val = val;
      np->next = p;
      return np;
   } else {
      p->next = insert(p, val);
      return p;
   }
}
```

Пример вызова malloc

```
p_chain insert(p_chain p, int val) {
   if ((0 == p) || (p->val > val)) {
      p_chain np =
         (p_chain)malloc(sizeof(t_chain));
      np->val = val;
      np->next = p;
      return np;
   } else {
      p->next = insert(p, val);
      return p;
   }
}
```

```
%include 'io.inc'

section .text

CEXTERN malloc

insert:
   push    ebp
   mov     ebp, esp
   sub     esp, 24
   ; ...
```

Пример вызова malloc

```
p_chain insert(p_chain p, int val) {
   if ((0 == p) || (p->val > val)) {
      p_chain np =
         (p_chain)malloc(sizeof(t_chain));
      np->val = val;
      np->next = p;
      return np;
   } else {
      p->next = insert(p, val);
      return p;
   }
}
```

```
insert:
   ; ...
   mov      dword [ebp-4], esi
   mov      esi, dword [ebp+8]
   mov      dword [ebp-8], ebx
   mov      ebx, dword [ebp+12]
   ; ...
```

Пример вызова malloc

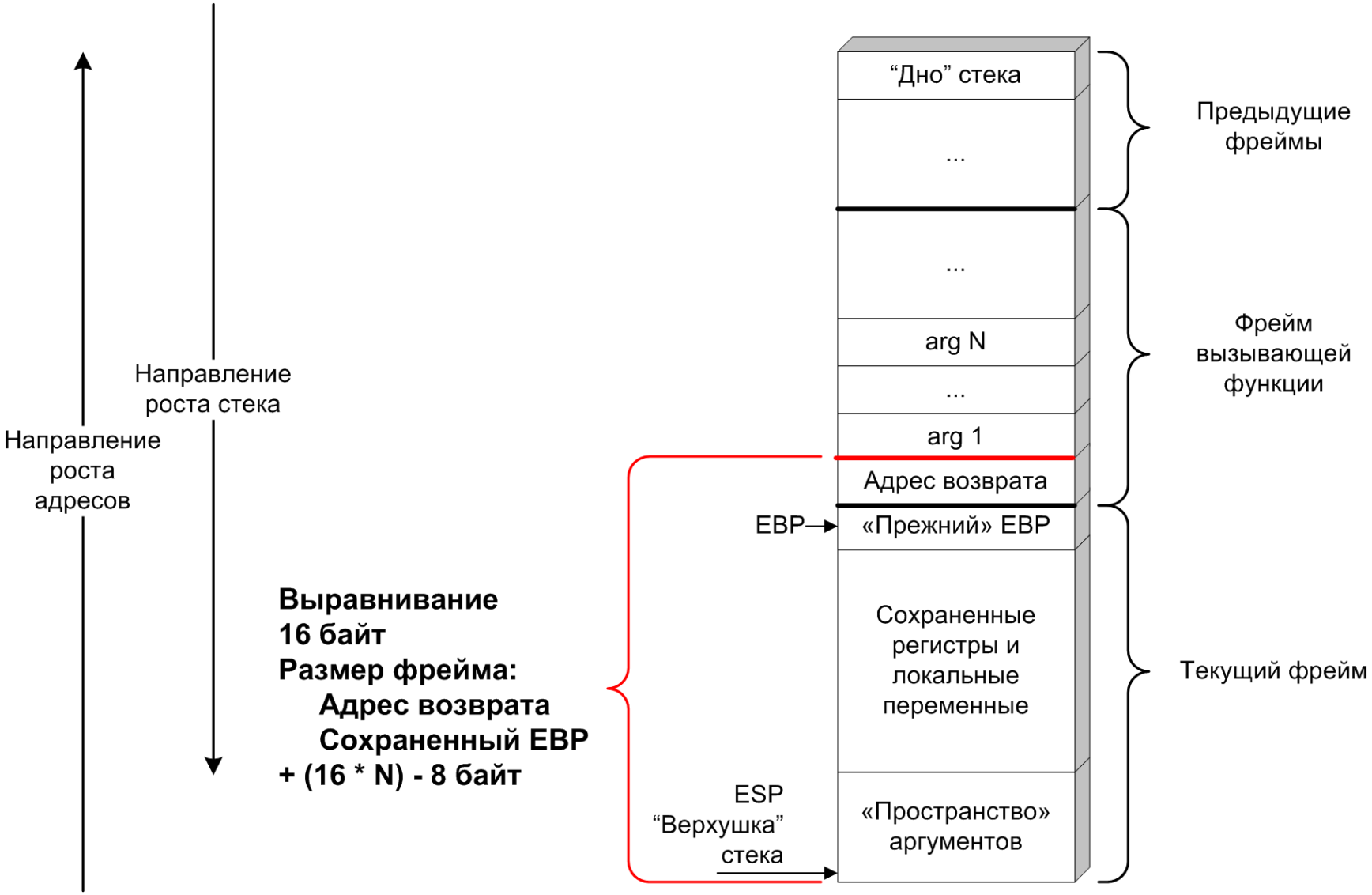
```
p_chain insert(p_chain p, int val) {
   if ((0 == p) || (p->val > val)) {
      p_chain np =
         (p_chain)malloc(sizeof(t_chain));
      np->val = val;
      np->next = p;
      return np;
   } else {
      p->next = insert(p, val);
      return p;
   }
}
```

```
   test    esi, esi
   je      .L2
   cmp     dword [esi], ebx
   jle     .L3
.L2:
   mov     dword [esp], 8
   call    malloc
   mov     dword [eax], ebx
   mov     dword [eax+4], esi
   mov     ebx, dword [ebp-8]
   mov     esi, dword [ebp-4]
   mov     esp, ebp
   pop     ebp
   ret
.L3:
```

Пример вызова malloc

```
p_chain insert(p_chain p, int val) {
   if ((0 == p) || (p->val > val)) {
      p_chain np =
         (p_chain)malloc(sizeof(t_chain));
      np->val = val;
      np->next = p;
      return np;
   } else {
      p->next = insert(p, val);
      return p;
   }
}
```

```
   ; …
.L3:
   mov     dword [esp+4], ebx
   mov     dword [esp], esi
   call    insert
   mov     dword [esi+4], eax
   mov     eax, esi
   mov     ebx, dword [ebp-8]
   mov     esi, dword [ebp-4]
   mov     esp, ebp
   pop     ebp
   ret
```



Стандартная библиотека языка Си

- 24 заголовочных файла
- stdlib.h
  - Преобразование типов: atoi, strtod, …
  - Генерация псевдослучайных последовательностей
  - Выделение и освобождение памяти
  - Сортировка и поиск
  - Математика
- stdio.h
  - Функции для файловых операций
  - Функции для операций ввода-вывода
- string.h
- …

STDCALL

```
#include <stdio.h>

__attribute__((stdcall))
int sum(int x, int y);

int main() {
   int a = 1, b = 2, c;
   c = sum(a, b);
   printf("%d\n", c);
   return 0;
}

__attribute__((stdcall))
int sum(int x, int y) {
   int t = x + y;
   return t;
}
```

```
sum:
   push    ebp
   mov     ebp, esp
   sub     esp, 16
   mov     edx, dword [ebp+12]
   mov     eax, dword [ebp+8]
   add     eax, edx
   mov     dword [ebp-4], eax
   mov     eax, dword [ebp-4]
   leave
   ret     8
```

STDCALL

```
#include <stdio.h>

__attribute__((stdcall))
int sum(int x, int y);

int main() {
   int a = 1, b = 2, c;
   c = sum(a, b);
   printf("%d\n", c);
   return 0;
}

__attribute__((stdcall))
int sum(int x, int y) {
   int t = x + y;
   return t;
}
```

```
CMAIN:
   ; …
   mov     eax, dword [ebp-12]
   mov     dword [esp+4], eax
   mov     eax, dword [ebp-16]
   mov     dword [esp], eax
   call    sum
   sub     esp, 8
   mov     dword [ebp-8], eax
   ; …
```

FASTCALL

```
#include <stdio.h>

__attribute__((fastcall)) int
sub(int x, int y);

int main() {
   int c = sub(1, 2);
   printf("%d\n", c);
   return 0;
}

__attribute__((fastcall)) int
sub(int x, int y) {
   int t = x - y;
   return t;
}
```

```
CMAIN:
   ; …
   mov      edx, 2
   mov      ecx, 1
   call     sub
   mov      dword [esp+20], eax
   ; …

sub:
   sub      ecx, edx
   mov      eax, ecx
   ret
```