

Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности
Высшая школа технологий искусственного интеллекта
Направление: 02.03.01 «Математика и компьютерные науки»

Отчёт о выполнении лабораторной работы №7
«Система команд, способы адресации и использование
подпрограмм в архитектуре x86-32»

Вариант 14

Выполнил студент
группы 5130201/40002

_____ Семенов И. А.

Проверила
преподаватель

_____ Веробова Н. М.

«_____» _____ 2025г.

Санкт-Петербург
2025

1 Цель работы

Изучить организацию вызова подпрограмм в архитектуре x86-32: механизм передачи параметров через стек, соглашение о вызовах (calling convention), размещение локальных переменных и возврат значений из функций.

2 Задание

Вариант 14: `unsigned char Fn(char, char, unsigned char)`

В функции объявить и использовать локальную переменную типа `char`.

3 Исходный код программы

Листинг 1: Исходный код программы (`main.c`)

```
1 unsigned char ucResult;
2 char cA;
3
4 unsigned char Fn(char, char, unsigned char);
5
6 int main(void) {
7     cA = 'A';
8     ucResult = Fn(cA, 7, 25);
9     return 0;
10 }
11
12 unsigned char Fn(char cParam1, char cParam2, unsigned char ucParam3) {
13     char cLocal;
14     cLocal = cParam1 + cParam2;
15     return (unsigned char)((cLocal * ucParam3) / 10);
16 }
```

3.1 Описание переменных

Переменная	Тип	Размер	Область видимости
ucResult	unsigned char	1 байт	глобальная
cA	char	1 байт	глобальная
cParam1	char	1 байт	параметр функции
cParam2	char	1 байт	параметр функции
ucParam3	unsigned char	1 байт	параметр функции
cLocal	char	1 байт	локальная в Fn

4 Дизассемблированный код

4.1 Функция main

Листинг 2: Дизассемблированный код функции main

```
1 0000118d <main>:  
2      118d:  8d 4c 24 04          lea    ecx,[esp+0x4]  
3      1191:  83 e4 f0          and    esp,0xffffffff0  
4      1194:  ff 71 fc          push   DWORD PTR [ecx-0x4]  
5      1197:  55                push   ebp  
6      1198:  89 e5          mov    ebp,esp  
7      119a:  53                push   ebx  
8      119b:  51                push   ecx  
9      119c:  e8 ef fe ff ff    call   1090 <_x86.get_pc_thunk.bx>  
10     11a1:  81 c3 3b 2e 00 00  add   ebx,0x2e3b  
11     11a7:  c6 83 2e 00 00 00 41  mov    BYTE PTR [ebx+0x2e],0x41  
12     11ae:  0f b6 83 2e 00 00 00  movzx eax,BYTE PTR [ebx+0x2e]  
13     11b5:  0f be c0          movsx  eax,al  
14     11b8:  83 ec 04          sub    esp,0x4  
15     11bb:  6a 19          push   0x19  
16     11bd:  6a 07          push   0x7  
17     11bf:  50                push   eax  
18     11c0:  e8 18 00 00 00    call   11dd <Fn>  
19     11c5:  83 c4 10          add    esp,0x10  
20     11c8:  88 83 2d 00 00 00  mov    BYTE PTR [ebx+0x2d],al  
21     11ce:  b8 00 00 00 00    mov    eax,0x0  
22     11d3:  8d 65 f8          lea    esp,[ebp-0x8]  
23     11d6:  59                pop    ecx  
24     11d7:  5b                pop    ebx  
25     11d8:  5d                pop    ebp  
26     11d9:  8d 61 fc          lea    esp,[ecx-0x4]  
27     11dc:  c3                ret
```

4.2 Функция Fn

Листинг 3: Дизассемблированный код функции Fn

```
1 000011dd <Fn>:  
2      11dd:  55                push   ebp  
3      11de:  89 e5          mov    ebp,esp  
4      11e0:  83 ec 1c          sub    esp,0x1c  
5      11e3:  e8 48 00 00 00    call   1230 <_x86.get_pc_thunk.ax>  
6      11e8:  05 f4 2d 00 00    add    eax,0x2df4  
7      11ed:  8b 4d 08          mov    ecx,DWORD PTR [ebp+0x8]  
8      11f0:  8b 55 0c          mov    edx,DWORD PTR [ebp+0xc]  
9      11f3:  8b 45 10          mov    eax,DWORD PTR [ebp+0x10]  
10     11f6:  88 4d ec          mov    BYTE PTR [ebp-0x14],cl  
11     11f9:  88 55 e8          mov    BYTE PTR [ebp-0x18],dl  
12     11fc:  88 45 e4          mov    BYTE PTR [ebp-0x1c],al
```

```

13 11ff: 0f b6 55 ec      movzx  edx,BYTE PTR [ebp-0x14]
14 1203: 0f b6 45 e8      movzx  eax,BYTE PTR [ebp-0x18]
15 1207: 01 d0            add    eax,edx
16 1209: 88 45 ff        mov    BYTE PTR [ebp-0x1],al
17 120c: 0f be 55 ff      movsx  edx,BYTE PTR [ebp-0x1]
18 1210: 0f b6 45 e4      movzx  eax,BYTE PTR [ebp-0x1c]
19 1214: 89 d1            mov    ecx,edx
20 1216: 0f af c8        imul   ecx,eax
21 1219: ba 67 66 66 66  mov    edx,0x66666667
22 121e: 89 c8            mov    eax,ecx
23 1220: f7 ea            imul   edx
24 1222: 89 d0            mov    eax,edx
25 1224: c1 f8 02        sar    eax,0x2
26 1227: c1 f9 1f        sar    ecx,0x1f
27 122a: 89 ca            mov    edx,ecx
28 122c: 29 d0            sub    eax,edx
29 122e: c9                leave
30 122f: c3                ret

```

5 Разбор дизассемблера по байтам

5.1 Функция main

- 0x0000118d <+0>: 8d 4c 24 04 lea ecx,[esp+0x4]

Байт-код: 8d 4c 24 04

Сохранение адреса аргументов командной строки в ECX для последующего выравнивания стека.

- 0x00001191 <+4>: 83 e4 f0 and esp,0xffffffff0

Байт-код: 83 e4 f0

Выравнивание стека по границе 16 байт (требование ABI).

- 0x00001194 <+7>: ff 71 fc push DWORD PTR [ecx-0x4]

Байт-код: ff 71 fc

Сохранение адреса возврата в выровненный стек.

- 0x00001197 <+10>: 55 push ebp

Байт-код: 55

Сохранение базового указателя вызывающей функции.

- 0x00001198 <+11>: 89 e5 mov ebp,esp

Байт-код: 89 e5

Установка нового кадра стека.

- 0x0000119a <+13>: 53 push ebx

Байт-код: 53

Сохранение регистра EBX (callee-saved).

- 0x0000119b <+14>: 51 push ecx
Байт-код: 51
Сохранение ECX с адресом аргументов.
- 0x0000119c <+15>: e8 ef fe ff ff call 1090
Байт-код: e8 ef fe ff ff
Вызов служебной функции для получения адреса в EBX (PIC).
- 0x000011a1 <+20>: 81 c3 3b 2e 00 00 add ebx,0x2e3b
Байт-код: 81 c3 3b 2e 00 00
Вычисление базового адреса глобальных данных.
- 0x000011a7 <+26>: c6 83 2e 00 00 00 41 mov BYTE PTR [ebx+0x2e],0x41
Байт-код: c6 83 2e 00 00 00 41
Присваивание сA = 'A' (0x41 = 65 = 'A').
- 0x000011ae <+33>: 0f b6 83 2e 00 00 00 movzx eax,BYTE PTR [ebx+0x2e]
Байт-код: 0f b6 83 2e 00 00 00
Загрузка сA с нулевым расширением до 32 бит.
- 0x000011b5 <+40>: 0f be c0 movsx eax,al
Байт-код: 0f be c0
Знаковое расширение для передачи как char (signed).
- 0x000011b8 <+43>: 83 ec 04 sub esp,0x4
Байт-код: 83 ec 04
Выравнивание стека перед передачей параметров.
- 0x000011bb <+46>: 6a 19 push 0x19
Байт-код: 6a 19
Передача третьего параметра ucParam3 = 25 (0x19).
- 0x000011bd <+48>: 6a 07 push 0x7
Байт-код: 6a 07
Передача второго параметра cParam2 = 7.
- 0x000011bf <+50>: 50 push eax
Байт-код: 50
Передача первого параметра cParam1 = сA ('A').
- 0x000011c0 <+51>: e8 18 00 00 00 call 11dd
Байт-код: e8 18 00 00 00
Вызов функции Fn.
- 0x000011c5 <+56>: 83 c4 10 add esp,0x10
Байт-код: 83 c4 10

Очистка стека: 16 байт (3 параметра по 4 байта + 4 байта выравнивания).

- 0x000011c8 <+59>: 88 83 2d 00 00 00 mov BYTE PTR [ebx+0x2d],al
Байт-код: 88 83 2d 00 00 00
Сохранение результата ucResult = AL.
- 0x000011ce <+65>: b8 00 00 00 00 mov eax,0x0
Байт-код: b8 00 00 00 00
Возврат 0 из main (return 0).
- 0x000011d3 <+70>: 8d 65 f8 lea esp,[ebp-0x8]
Байт-код: 8d 65 f8
Восстановление ESP для эпилога.
- 0x000011d6 <+73>: 59 pop ecx
Байт-код: 59
Восстановление ECX.
- 0x000011d7 <+74>: 5b pop ebx
Байт-код: 5b
Восстановление EBX.
- 0x000011d8 <+75>: 5d pop ebp
Байт-код: 5d
Восстановление EBP.
- 0x000011d9 <+76>: 8d 61 fc lea esp,[ecx-0x4]
Байт-код: 8d 61 fc
Восстановление оригинального ESP.
- 0x000011dc <+79>: c3 ret
Байт-код: c3
Возврат из функции main.

5.2 Функция Fn

- 0x000011dd <+0>: 55 push ebp
Байт-код: 55
Сохранение базового указателя вызывающей функции.
- 0x000011de <+1>: 89 e5 mov ebp,esp
Байт-код: 89 e5
Установка нового кадра стека.

- 0x000011e0 <+3>: 83 ec 1c sub esp,0x1c
Байт-код: 83 ec 1c
 Выделение 28 байт для локальных переменных.
- 0x000011e3 <+6>: e8 48 00 00 00 call 1230
Байт-код: e8 48 00 00 00
 Вызов служебной функции для PIC (получение адреса в EAX).
- 0x000011e8 <+11>: 05 f4 2d 00 00 add eax,0x2df4
Байт-код: 05 f4 2d 00 00
 Вычисление базового адреса глобальных данных.
- 0x000011ed <+16>: 8b 4d 08 mov ecx,DWORD PTR [ebp+0x8]
Байт-код: 8b 4d 08
 Загрузка первого параметра cParam1 в ECX.
- 0x000011f0 <+19>: 8b 55 0c mov edx,DWORD PTR [ebp+0xc]
Байт-код: 8b 55 0c
 Загрузка второго параметра cParam2 в EDX.
- 0x000011f3 <+22>: 8b 45 10 mov eax,DWORD PTR [ebp+0x10]
Байт-код: 8b 45 10
 Загрузка третьего параметра ucParam3 в EAX.
- 0x000011f6 <+25>: 88 4d ec mov BYTE PTR [ebp-0x14],cl
Байт-код: 88 4d ec
 Сохранение cParam1 в локальную область стека.
- 0x000011f9 <+28>: 88 55 e8 mov BYTE PTR [ebp-0x18],dl
Байт-код: 88 55 e8
 Сохранение cParam2 в локальную область стека.
- 0x000011fc <+31>: 88 45 e4 mov BYTE PTR [ebp-0x1c],al
Байт-код: 88 45 e4
 Сохранение ucParam3 в локальную область стека.
- 0x000011ff <+34>: 0f b6 55 ec movzx edx,BYTE PTR [ebp-0x14]
Байт-код: 0f b6 55 ec
 Загрузка cParam1 с нулевым расширением.
- 0x00001203 <+38>: 0f b6 45 e8 movzx eax,BYTE PTR [ebp-0x18]
Байт-код: 0f b6 45 e8
 Загрузка cParam2 с нулевым расширением.
- 0x00001207 <+42>: 01 d0 add eax,edx
Байт-код: 01 d0
 Вычисление cParam1 + cParam2.

- 0x00001209 <+44>: 88 45 ff mov BYTE PTR [ebp-0x1],al
Байт-код: 88 45 ff
Сохранение результата в локальную переменную cLocal.
- 0x0000120c <+47>: 0f be 55 ff movsx edx,BYTE PTR [ebp-0x1]
Байт-код: 0f be 55 ff
Загрузка cLocal со знаковым расширением.
- 0x00001210 <+51>: 0f b6 45 e4 movzx eax,BYTE PTR [ebp-0x1c]
Байт-код: 0f b6 45 e4
Загрузка ucParam3 с нулевым расширением (unsigned).
- 0x00001214 <+55>: 89 d1 mov ecx,edx
Байт-код: 89 d1
Копирование cLocal в ECX.
- 0x00001216 <+57>: 0f af c8 imul ecx, eax
Байт-код: 0f af c8
Умножение: ecx = cLocal * ucParam3.
- 0x00001219 <+60>: ba 67 66 66 66 mov edx,0x66666667
Байт-код: ba 67 66 66 66
Загрузка магической константы для деления на 10.
- 0x0000121e <+65>: 89 c8 mov eax,ecx
Байт-код: 89 c8
Копирование произведения в EAX для деления.
- 0x00001220 <+67>: f7 ea imul edx
Байт-код: f7 ea
Знаковое умножение EAX на EDX, результат в EDX:EAX.
- 0x00001222 <+69>: 89 d0 mov eax,edx
Байт-код: 89 d0
Старшая часть результата (EDX) — это результат деления.
- 0x00001224 <+71>: c1 f8 02 sar eax,0x2
Байт-код: c1 f8 02
Арифметический сдвиг вправо на 2 (корректировка деления).
- 0x00001227 <+74>: c1 f9 1f sar ecx,0x1f
Байт-код: c1 f9 1f
Получение знакового бита исходного числа.
- 0x0000122a <+77>: 89 ca mov edx,ecx
Байт-код: 89 ca
Копирование знакового бита в EDX.

- 0x0000122c <+79>: 29 d0 sub eax, edx
Байт-код: 29 d0
 Корректировка для отрицательных чисел (добавление 1 если отрицательное).
- 0x0000122e <+81>: c9 leave
Байт-код: c9
 Восстановление ESP и EBP (эквивалент mov esp,ebp; pop ebp).
- 0x0000122f <+82>: c3 ret
Байт-код: c3
 Возврат из функции. Результат в AL (младший байт EAX).

6 Анализ организации подпрограммы

6.1 Соглашение о вызовах (cdecl)

В архитектуре x86-32 используется соглашение cdecl:

- Параметры передаются через стек справа налево
- Вызывающая функция очищает стек после вызова
- Возвращаемое значение помещается в регистр EAX (AL для char)
- Регистры EAX, ECX, EDX могут изменяться вызываемой функцией
- Регистры EBX, ESI, EDI, EBP должны сохраняться

6.2 Схема передачи параметров

При вызове Fn(cA, 7, 25) параметры помещаются в стек в обратном порядке:

Листинг 4: Передача параметров в main

```

sub    esp,0x4          ; выравнивание стека
push   0x19             ; ucParam3 = 25 (третий параметр)
push   0x7              ; cParam2 = 7 (второй параметр)
push   eax              ; cParam1 = cA (первый параметр)
call   11dd <Fn>
add    esp,0x10          ; очистка стека (16 байт)

```

6.3 Карта стека функции main

Адрес	Содержимое	Описание
[ebp+0x4]	адрес возврата	4 байта
[ebp]	сохранённый EBP	4 байта
[ebp-0x4]	сохранённый EBX	4 байта
[ebp-0x8]	сохранённый ECX	4 байта

6.4 Карта стека функции Fn

Адрес	Содержимое	Размер
[ebp+0x10]	ucParam3 (третий параметр)	4 байта
[ebp+0xc]	cParam2 (второй параметр)	4 байта
[ebp+0x8]	cParam1 (первый параметр)	4 байта
[ebp+0x4]	адрес возврата	4 байта
[ebp]	сохранённый EBP	4 байта
[ebp-0x1]	cLocal (локальная переменная)	1 байт
[ebp-0x14]	копия cParam1	1 байт
[ebp-0x18]	копия cParam2	1 байт
[ebp-0x1c]	копия ucParam3	1 байт

6.5 Карта глобальных переменных

Адрес	Переменная	Тип
[ebx+0x2d]	ucResult	unsigned char
[ebx+0x2e]	cA	char

6.6 Возврат значения

Функция возвращает `unsigned char` через регистр AL (младший байт EAX). Результат сохраняется в main:

Листинг 5: Сохранение результата

```
mov     BYTE PTR [ebx+0x2d],al    ; ucResult = AL
```

6.7 Расширение типов

- `movzx` — нулевое расширение (для `unsigned char`)
- `movsx` — знаковое расширение (для `char`)

Пример: при загрузке сA для передачи как `char`:

Листинг 6: Расширение типа char

```
movzx  eax,BYTE PTR [ebx+0x2e]      ; нулевое расширение  
movsx  eax,al                      ; знаковое расширение
```

7 Выводы

В ходе выполнения лабораторной работы изучены:

1. Механизм передачи параметров через стек по соглашению cdecl
2. Организация кадра стека (stack frame) с использованием EBP
3. Размещение локальных переменных в стеке относительно EBP
4. Способы адресации для доступа к параметрам и локальным переменным
5. Возврат значений через регистр EAX/AL
6. Расширение типов при передаче параметров меньше 32 бит

Параметры типа `char` и `unsigned char` при передаче через стек расширяются до 32 бит (4 байта) для выравнивания стека.