

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7__

дисциплина: Архитектура компьютера

Студент: Мурылев Иван Валерьевич

Группа: НПИбд-03-25

МОСКВА

2025 г.

Содержание

Цель работы

Теоретическое введение

Выполнение работы

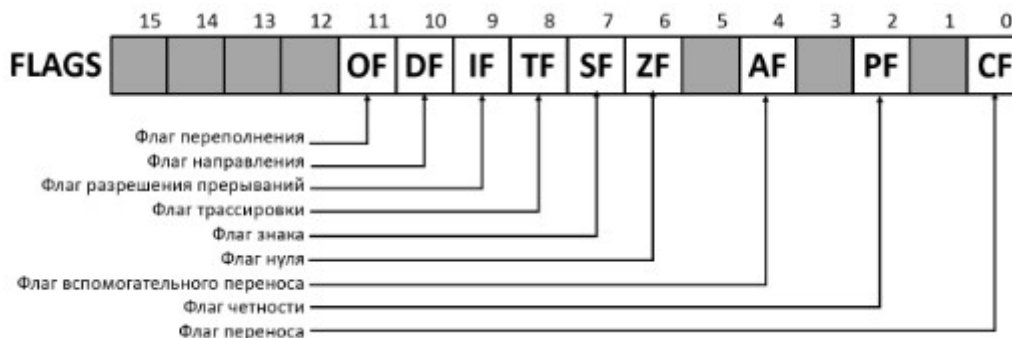
Вывод

Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

Теоретическое введение

Флаг – это бит, принимающий значение 1 («флаг установлен»), если выполнено некоторое условие, и значение 0 («флаг сброшен») в противном случае. Флаги работают независимо друг от друга, и лишь для удобства они помещены в единый регистр — регистр флагов, отражающий текущее состояние процессора. В следующей таблице указано положение битовых флагов в регистре флагов.



Флаги состояния (биты 0, 2, 4, 6, 7 и 11) отражают результат выполнения арифметических инструкций, таких как ADD, SUB, MUL, DIV.

Выполнение работы

Первые этапы пропускаются, так как в репозитории уже созданы нужные файлы.

После создания файла lab07.asm и внесения в него текста листинга 7.1 производим трансляцию и компоновку. Затем запуск.

```
[ivmurihlev@personal report]$ nasm -f elf lab07-1.asm
lab07-1.asm:1: error: unable to open include file 'in_out.asm': No such file or directory
[ivmurihlev@personal report]$ nasm -f elf lab07-1.asm
[ivmurihlev@personal report]$ ld -m elf_i386 lab07-1.o -o lab07-1
[ivmurihlev@personal report]$ ./lab07-1
Сообщение № 2
Сообщение № 3
```

Затем создаем файлы lab02.asm lab03.asm и вносим в них тексты листингов 7.2 и 7.3 соответственно. Затем транслируем, компонуем и запускаем.

```
lab07-2.asm:100: error: invalid combination of opcode and operands
[ivmurihlev@personal report]$ nasm -f elf -l lab07-2 lab07-2.asm
[ivmurihlev@personal report]$ ld -m elf_i386 lab07-2.o -o lab07-2
[ivmurihlev@personal report]$ ./lab07-2
Сообщение № 2
Сообщение № 1
```

```

[ivmurihlev@personal report]$ nasm -f elf lab07-3.asm
[ivmurihlev@personal report]$ ld -m elf_i386 lab07-3.o -o lab07-3
[ivmurihlev@personal report]$ ./lab07-3
Введите В: 3
Наибольшее число: 50
[ivmurihlev@personal report]$ ./lab07-3
Введите В: 700
Наибольшее число: 700
[ivmurihlev@personal report]$

```

Проверим листинг и отобразим содержимое файла lab07-3-1.lst, полученного в ходе трансляции через команду *nasm -f elf -f lab07-3-1.lst lab07-3.asm* .

```

[ivmurihlev@personal report]$ nasm -f elf -l lab07-3-1.lst lab07-3.asm
[ivmurihlev@personal report]$ cat lab07-3-1.lst
1                               %include 'in_out.asm'
1                               <1> ;----- slen -----
2                               <1> ; Функция вычисления длины сообщения
3                               <1> slen:
4 00000000 53                    <1>     push    ebx
5 00000001 89C3                  <1>     mov     ebx, eax
6                               <1>
7                               <1> nextchar:
8 00000003 803800                <1>     cmp     byte [eax], 0
9 00000006 7403                  <1>     jz      finished
10 00000008 40                   <1>     inc     eax
11 00000009 EBF8                  <1>     jmp     nextchar
12                               <1>
13                               <1> finished:
14 0000000B 29D8                  <1>     sub     eax, ebx
15 0000000D 5B                   <1>     pop     ebx
16 0000000E C3                   <1>     ret
17                               <1>
18                               <1>
19                               <1> ;----- sprint -----
20                               <1> ; Функция печати сообщения
21                               <1> ; входные данные: mov eax,<message>
22                               <1> sprint:
23 0000000F 52                    <1>     push    edx
24 00000010 51                    <1>     push    ecx
25 00000011 53                    <1>     push    ebx
26 00000012 50                    <1>     push    eax
27 00000013 E8E8FFFFFF            <1>     call    slen
28                               <1>
29 00000018 89C2                  <1>     mov     edx, eax
30 0000001A 58                    <1>     pop     eax
31                               <1>
32 0000001B 89C1                  <1>     mov     ecx, eax
33 0000001D BB01000000             <1>     mov     ebx, 1
34 00000022 B804000000             <1>     mov     eax, 4
35 00000027 CD80                  <1>     int     80h
36                               <1>

```

Удалим второй операнд в 28 строке и проведем операцию по созданию листинга как в случае с листингом lab07-3-1.lst, но для lab07-3-2.lst

```
; ----- Сравниваем 'A' и 'C' (как символы)
cmp esx, | ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B'
mov esx, [C] ; иначе 'esx = C'
```

```
[ivmurihlev@personal report]$ nasm -f elf -l lab07-3-2.lst lab07-3.asm
lab07-3.asm:28: error: invalid combination of opcode and operands
[ivmurihlev@personal report]$
```

Как видно программа выдает ошибку и просто пропускает строку в листинге.

```
26 00000012 50          <1>      push    eax
27 00000013 E8E8FFFFFF    <1>      call   slen
28                                <1>      |
29 00000018 89C2          <1>      mov     edx, eax
30 0000001A 58              <1>      pop     eax
31                                <1>
```

В самостоятельной части в предыдущем мой вариант 7:

```
7          45,67,15
```

Создам файл lab07-4-1.asm и получаем после исполнения файла программа работает.

```
[ivmurihlev@personal report]$ nasm -f elf lab07-4-1.asm
[ivmurihlev@personal report]$ ld -m elf_i386 lab07-4-1.o -o lab07-4-1
[ivmurihlev@personal report]$ ./lab07-4-1
Наибольшее число: 67
```

Затем завершаем задание lab07-4-2.asm. После трансляции и компоновки запускаем, получаем следующее

```
[ivmurihlev@personal report]$ nasm -f elf lab07-4-2.asm
[ivmurihlev@personal report]$ ld -m elf_i386 lab07-4-2.o -o lab07-4-2
[ivmurihlev@personal report]$ ./lab07-4-2
Введите x: 2
Введите a: 2
final : 12
[ivmurihlev@personal report]$ ./lab07-4-2
Введите x: 3
Введите a: 4
final : 7
```

Вывод

Переходы в Ассамблере напоминают указатели в более высоко уровневых языках программирования. Был изучен механизм переходов и способы отладки.