Отчёт по лабораторной работе №6

Дисциплина: архитектура компьютера

Худдыева Дженнет

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Цель данной лабораторной раюоты - освоение арефметических инструкций языка ассемблера NASM.

2 Задание

1.Символьные и численные данные в NASM. 2.Выполнение арифметических операций в NASM. 3.Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов.Адрес операнда предоставляет место,где хранятся данные, подлежащие обработке.Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация - операнды хранятся в регистрах и в команде используются имена этих регистров,например: mov ax,bx. - Непосредственная адресация - значение операнда задаётся непосредственно в команде,например: mov ax,2. - Адресация памяти - операнд задаёт адрес в памяти.В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде.Кодирование этой информации производится сог- ласно кодовой таблице символов ASCII.ASCII - сокращение от American Standard Code for Information Interchange(Американский стандартный код для обмена информацией).Согласно стандарту ASCII каждый символ кодируется одним байтом.Среди инструкций NASM нет такой,которая выводит числа(не в символьном виде).Поэтому,например,чтобы вывести число,надо предворительно преобразовать его цифры в ASCII-коды этих цифр и выводит на экран эти коды,а не само число.Если же выводить число на экран непосредственно, то экран воспримет его не как число,а как последовательность ASCII-символов-каждый байт число будет воспринят как один ASCII- -символ и выведет на экран эти символы.

4 Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

С помощью утилиты mkdir создаю директорию,в которой буду создавать файлы с программами для лабораторной работы №6 (рис.[4.1]) С помощью утилиты touch создаю файл lab6-1.asm

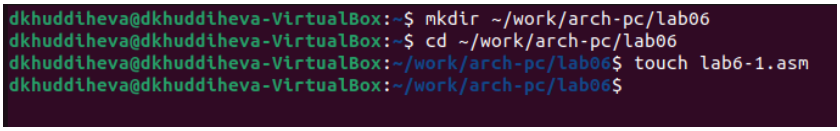

Рис. 4.1: Создание директории

Копирую в текущий каталог файл in_out.asm с помощью утилиты cp (рис.[4.2]).

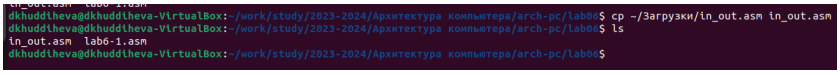

Рис. 4.2: Создание копии файла

Открываю созданный файл lab6-1.asm,вставляю в него программу вывода значения регистра eax (рис.[4.3]).

```
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1:   RESB 80
4
5 SECTION .text
6 GLOBAL _start
7  _start:
8
9  mov eax,'6'
10 mov ebx,'4'
11 add eax,ebx
12 mov [buf1],eax
13 mov eax,buf1
14 call sprintLF
15
16 call quit
```

Рис. 4.3: Редактирование файла

Создаю исполняемый файл программы и запускаю его (рис.[4.4]).Вывод программы: символ j потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.

Рис. 4.4: Запуск исполняемого файла

Изменяю в тексте программы “6” и “4” на цифру 6 и 4 (рис.[4.5]).

```
%include 'in_out.asm'
SECTION .bss
buf1:   RESB 80

SECTION .text
GLOBAL _start
 _start:

 mov eax,6
 mov ebx,4
 add eax,ebx
 mov [buf1],eax
 mov eax,buf1
 call sprintLF

 call quit
```

Рис. 4.5: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его (рис.[4.6]).Теперь вывелся символ 10,это символ перевода строки,этот символ не отображается при выводе на экран.

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ gedit lab6-1.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ nasm -f elf lab6-1.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ./lab6-1
```

Рис. 4.6: Запуск исполняемого файла

Создаю новый файл lab6-2.asm с помощью утилиты touch (рис.[4.7]).

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ touch lab6-2.asm
```

Рис. 4.7: Создание файла

Ввожу в файл текст другой программы для вывода значения регистра eax (рис.[4.8]).

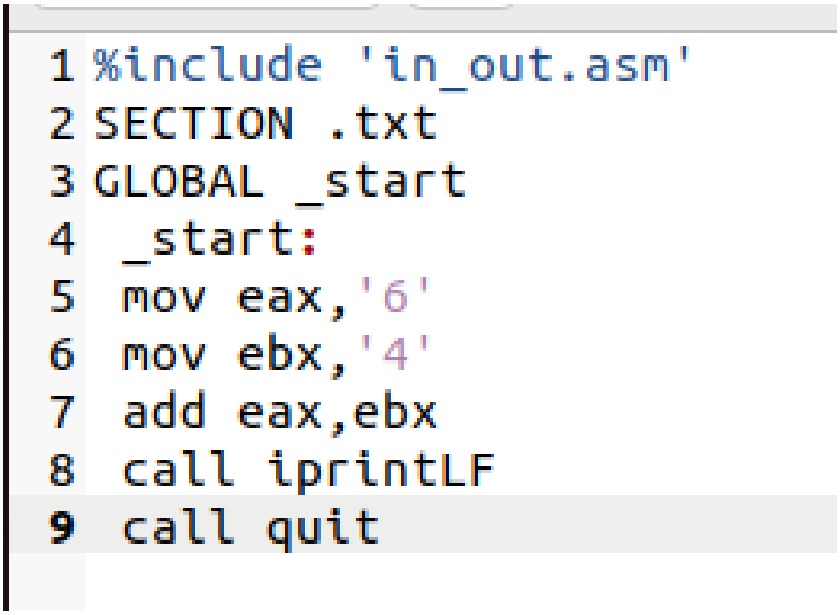

Рис. 4.8: Редактирование файла

Создаю и запускаю исполняемый файл lab6-2 (рис.[4.9]).Теперь вывод числа 106,потому что программа позволяет вывести именно число,а не символ,хотя всё ещё происходит именно сложение кодов символов “6” и “4”.

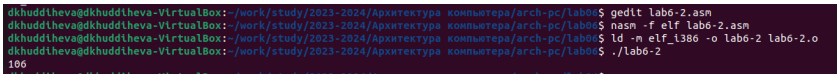

Рис. 4.9: Запуск исполняемого файла

Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на число 6 и 4 (рис.[4.10]).

```
1 %include 'in_out.asm'
2 SECTION .txt
3 GLOBAL _start
4  _start:
5  mov eax,6
6  mov ebx,4
7  add eax,ebx
8  call iprintLF
9  call quit
```

Рис. 4.10: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис.[4.11]).Теперь программа складывает не соответствующие символом коды в системе ASCII,а сами числа,поэтому вывод 10.

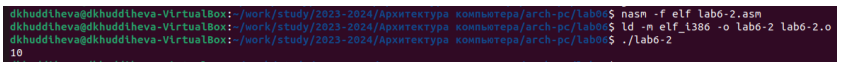

Рис. 4.11: Запуск исполняемого файла

Заменяю в тексте программы функцию iprintLF на iprint (рис.[4.12]).

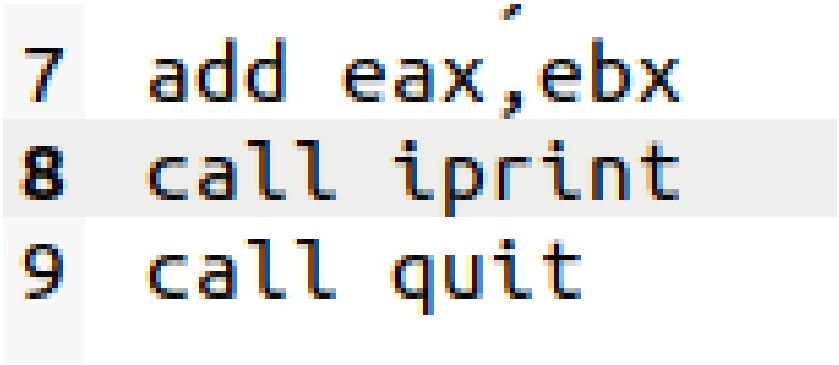

Рис. 4.12: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис.[4.13]).Вывод не изменился,потому что символ переноса строки не отображался, когда программа исполнялась с функцией iprintLF,а iprint не добавляет к выводу символ переноса строки, в отличие от iprintLF.

Рис. 4.13: Запуск исполняемого файла

4.2 Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm с помощью утилиты touch (рис.[4.14]).

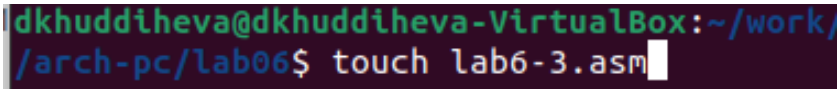

Рис. 4.14: Создание файла

Ввожу в созданный файл текст программы для вычисления значения выражения f(x)=(5*2+3)/3 (рис.[4.15]).

```
%include 'in_out.asm'
SECTION .data
div:  DB 'Результат: ',0
rem:  DB 'Остаток от деления: ',0
SECTION .txt
GLOBAL _start
 _start:
; ---- Вычисление выражения
mov eax,5   ; EAX=5
mov ebx,2   ; EBX=2
mul ebx     ; EAX=EAX*EBX
add eax,3    ; EAX=EAX+3
xor edx,edx   ; обнуляем EDX для корректной работы div
mov ebx,3   ; EBX=3
div ebx    ; EAX=EAX/3,  EDX=остаток от деления

mov edi,eax   ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран

mov eax,div   ; вызов подпрограммы печати
call sprint    ; сообщения 'Результат: '
mov eax,edi    ; вызов подпрограммы печати значения
call iprintLF   ; из 'edx' (остаток) в виде символов

mov eax,rem    ; вызоа подпрограммы печати
call sprint   ; сообщения 'Остаток от деления: '
mov eax,edx   ; вызов подпрограммы печати значения
call iprintLF    ; из 'edx' (остаток) в виде символов
call quit   ; вызов подпрограммы завершения
```

Рис. 4.15: Редактирование файла

Создаю исполняемый файл и запускаю его (рис.[4.16]).

```
dkhuddlheva@dkhuddlheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ nasm -f elf lab6-3.asm
dkhuddlheva@dkhuddlheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
dkhuddlheva@dkhuddlheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
```

Рис. 4.16: Запуск исполняемого файла

Изменяю программу так,чтобы она вычисляла значение выражения (рис.[4.17]).

```
%include 'in_out.asm'
SECTION .data
div:  DB 'Результат: ',0
rem:  DB 'Остаток от деления: ',0
SECTION .txt
GLOBAL _start
 _start:
; ---- Вычисление выражения
mov eax,4   ; EAX=4
mov ebx,6   ; EBX=6
mul ebx     ; EAX=EAX*EBX
add eax,2    ; EAX=EAX+2
xor edx,edx   ; обнуляем EDX для корректной работы div
mov ebx,5   ; EBX=5
div ebx    ; EAX=EAX/5,  EDX=остаток от деления

mov edi,eax   ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран

mov eax,div   ; вызов подпрограммы печати
call sprint    ; сообщения 'Результат: '
mov eax,edi    ; вызов подпрограммы печати значения
call iprintLF   ; из 'edx' (остаток) в виде символов

mov eax,rem    ; вызоа подпрограммы печати
call sprint   ; сообщения 'Остаток от деления: '
mov eax,edx   ; вызов подпрограммы печати значения
call iprintLF    ; из 'edx' (остаток) в виде символов
call quit   ; вызов подпрограммы завершения
```

Рис. 4.17: Изменение программы

Создаю и запускаю новый исполняемый файл (рис.[4.18]).

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ gedit lab6-3.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ nasm -f elf lab6-3.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
```

Рис. 4.18: Запуск исполняемого файла

Создаю файл variant.asm с помощью утилиты touch (рис.[4.19]).

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06$ touch variant.asm
```

Рис. 4.19: Создание файла

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис.[4.20]).

```
%include 'in_out.asm'
SECTION .data
msg:  DB 'Введите № студенческого билета: ',0
rem:  DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .txt
GLOBAL _start
 _start:
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax,x   ; вызов подпрограммы преобразования
call atoi    ;ASCII кода в число, `eax=x`
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit
```

Рис. 4.20: Редактирование файла

Создаю и запускаю исполняемый файл (рис.[4.21]).Ввожу номер своего студ.билета с клавиатуры,программа вывела что мой вариант-17

```
dkhuddtheva@dkhuddtheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf variant.asm
dkhuddtheva@dkhuddtheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o variant variant.o
dkhuddtheva@dkhuddtheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./variant
Введите № студенческого билета:
1032235056
Ваш вариант: 17
```

Рис. 4.21: Запуск исполняемого файла

4.2.1 Ответы на вопросы по программе

1.За вывод сообщения “Ваш вариант” отвечают строки кода:

```
mov eax,rem
call sprint
```

2.Инструкция mov ecx, x используется,чтобы положит адрес вводимой строки x в регистр ecx mov edx, 80 - запись в регистр edx длины вводимой строки call sread - вызов программы из внешнего файла,обеспечивающей ввод сообщения с клавиатуры

3.call atoi используется для вызова подпрограммы из внешнего файла,которая преобразует ASCII -код символа в целое и записывает результат в регистр eax

4.За вычисления варианта отвечают строки:

```
xor edx,edx  ; обнуление edx для корректной работы div
mov ebx,20  ; ebx=20
div ebx  ; eax=eax/20, edx-остаток от деления
inc edx  ; edx=edx+1
```

5.При выполнении инструкции div ebx остаток от деления записывается в регистр edx

6.Инструкция inc edx увеличивает регистра edx на 1

7.За вывод на экран результатов вычислений отвечают строки:

```
mov eax,edx
call iprintLF
```

4.3 Выполнение заданий для самостоятельной работы

Создаю файл lab6-4.asm с помощью утилиты touch (рис.[4.22])

Рис. 4.22: Создание файла

Открываю созданной файл для редактирования,ввожу в него текст программы для вычисления значения выражения 18*(x+1)/6.Это выражение было под вариантом 17 (рис.[4.23])

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; секция инициированных данных
msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss ; секция не инициированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный размер - 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
; ---- Вычисление выражения
mov eax, msg ; запись адреса выводимиого сообщения в eax
call sprint ; вызов подпрограммы печати сообщения
mov ecx, x ; запись адреса переменной в ecx
mov edx, 80 ; запись длины вводимого значения в edx
call sread ; вызов подпрограммы ввода сообщения
mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, eax=x
add eax,1; eax = eax+1 = x+1
mov ebx,18; запись значения 18 в регистр ebx
mul ebx; EAX=EAX*EBX = (x+1)*18
mov ebx,6; запись значения 6 в регистр ebx
div ebx;eax=eax/6=((x+1)*18)/6
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprint ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 4.23: Написание программы

Создаю и запускаю исполняемый файл (рис.[4.24])

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-202
/arch-pc/lab06$ nasm -f elf lab6-4.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-202
/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-202
/arch-pc/lab06$ ./lab6-4
Введите значение переменной x: 3
Результат: 12dkhuddiheva@dkhuddiheva-VirtualBox:~/work/s
pa компьютера
/arch-pc/lab06$
```

Рис. 4.24: Запуск исполняемого файла

Листинг 4.1. Программа для вычисления значения выражения 18*(x+1)/6.

5 Выводы

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM