Лабораторная работа №6

Архитектура вычислительных систем

Горбачев Алексей Антонович

Содержание

1	Цель работы	5				
2	Задание	6				
3	Теоретическое введение	7				
4	Выполнение лабораторной работы	9				
5	Ответы на вопросы:	21				
6	Выводы	22				
Список литературы						

Список иллюстраций

4.1	61.png.																			9
4.2	62.png.																			10
4.3	63.png.																			11
4.4	64.png.																			12
4.5	65.png.																			13
4.6	66.png.																			14
4.7	67.png.																			14
4.8	68.png.							•												14
4.9	69.png.																			15
4.10	610.png							•												16
4.11	611.png			•									•		•					17
4.12	612.png						•											•		18
4.13	613.png							•												19
4 14	614nng																			2.0

Список таблиц

1 Цель работы

Освоить арифметические инструкций языка ассемблера NASM.

2 Задание

Написать программу вычисления выражения. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x, вычислять заданное выражение в зависимости от введенного x, выводить результат вычислений. Вид функции выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создайть исполняемый файл и проверить его работу для значений из 6.3.

3 Теоретическое введение

- 1. Адресация в NASM Существует три основных способа адресации: Регистровая адресация операнды хранятся в регистрах и в команде используются имена этих регистров, например: mov ax,bx. Непосредственная адресация значение операнда задается непосредственно в команде, Например: mov ax,2. Адресация памяти операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.
- 2. Арифметические операции в NASM Схема команды целочисленного сложения add (от англ. addition добавление) выполняет сложение двух операндов и записывает результат по адресу первого операнда. Команда add работает как с числами со знаком, так и без знака.
- 3. Целочисленное вычитание sub Команда целочисленного вычитания sub (от англ. subtraction вычитание) работает аналогично команде add.
- 4. Команды инкремента и декремента Довольно часто при написании программ встречается операция прибавления или вычитания единицы. Прибавление единицы называется инкрементом, а вычитание декрементом. Для этих операций существуют специальные команды: inc (от англ. increment) и dec (от англ. decrement), которые увеличивают и уменьшают на 1 свой операнд.
- 5. Команда изменения знака операнда neg Команда рассматривает свой операнд как число со знаком и меняет знак операнда на противоположный. Операндом может быть регистр или ячейка памяти любого размера.

- 6. Команды умножения mul и imul Умножение и деление, в отличии от сложения и вычитания, для знаковых и беззнаковых чисел производиться по-разному, поэтому существуют различные команды. Для беззнакового умножения используется команда mul (от англ. multiply умножение). Для знакового умножения используется команда imul.
- 7. Команды деления div и idiv Для деления, как и для умножения, существует 2 команды div (от англ. divide деление) и idiv. Для беззнакового умножения используется команда div. Для знакового умножения используется команда idiv.

4 Выполнение лабораторной работы

1. Создаём каталог для программам лабораторной работы No 7, перейдём в него и создаём файл lab6-1.asm

```
aagorbachev@dk3n40 ~ $ cd work
aagorbachev@dk3n40 ~/work $ mkdir arch-pc
aagorbachev@dk3n40 ~/work $ cd arch-pc
aagorbachev@dk3n40 ~/work/arch-pc $ mkdir lab06
aagorbachev@dk3n40 ~/work/arch-pc $ cd lab06
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ touch_lab6-1.asm
```

Рис. 4.1: 61.png

2. Введем в файл lab6-1.asm текст программы из листинга 6.1.

```
GNU HallO /.Z /als/.uk.
%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. 4.2: 62.png

3. Создаём копию файла in_out.asm в каталоге.

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab06
bash: ./lab06: Нет такого файла или каталога
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис. 4.3: 63.png

4. Создадим исполняемый файл и запустим его.

```
include 'in_out.asm'
  CTION .bss
     RESB 80
 ECTION .text
 LOBAL _start
 start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

Рис. 4.4: 64.png

5. Далее изменим текст программы и вместо символов, запишем в регистры числа. Исправим текст программы.

```
GNU HallO /.Z /als/.uk.
%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: 65.png

6. Создадим исполняемый файл и запустим его (6-1).

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc

aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-1

aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc
```

Рис. 4.6: 66.png

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ cd
aagorbachev@dk3n40 ~ $ touch ~/work/arch-pc/lab06/lab6-2.asm
aagorbachev@dk3n40 ~ $ cd work
aagorbachev@dk3n40 ~/work $ cd arch-pc
aagorbachev@dk3n40 ~/work/arch-pc $ cd lab06
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
ld: предупреждение: невозможно найти символ входа _start; начальный адрес не устанавливает
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc

aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-2
```

Рис. 4.7: 67.png

7. Создадим файл lab6-2.asm в каталоге. Введем в него текст программы из листинга 6.2 и запустим его.

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-2
10
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc
```

Рис. 4.8: 68.png

8. Изменим символы на числа в lab6-2. Создадим исполняемый файл и запустим его.

```
GNO Hallo 7.2 /als/.uk.sci.pru.euu.ru/Hollie/a/a/aagorbachev
 Программа вычисления выражения
%include 'in_out.asm' ; подключение внешнего файла
  CTION .data
v: DB 'Результат: ',0
      <mark>В</mark> 'Остаток от деления: ',0
em: DB 'Octa
SECTION .text
    AL _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
            X=EAX/3, EDX=остаток от деления
div ebx ; E/
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 'edi' в виде символов
mov eax,rem ; вызов 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. 4.9: 69.png

9. Создадим файл lab6-3.asm в каталоге. Введем в файл lab6-3.asm текст программы из листинга 6.3

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-3.asm \\ aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ cd work
bash: cd: work: Нет такого файла или каталога
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ cd arch-pc
bash: cd: arch-pc: Нет такого файла или каталога
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ cd work/arch-pc
bash: cd: work/arch-pc: Нет такого файла или каталога
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ cd
aagorbachev@dk3n40 ~ $ cd work
aagorbachev@dk3n40 ~/work $ cd arch-pc
aagorbachev@dk3n40 ~/work/arch-pc $ cd lab06
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
ld: предупреждение: невозможно найти символ входа _start; начальный адрес не устанавливается aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ mc
```

Рис. 4.10: 610.png

10. Создадим исполняемый файл и запустим его.

```
GNO HAHO 7.2 /ars/.uk.scr.pru.euu.ru/Hollie/a/a/aagorbacher
 Программа вычисления выражения
%include 'in_out.asm' ; подключение внешнего файла
  CTION .data
v: DB 'Результат: ',0
rem: DB 'Ocia
SECTION .text
      <mark>Љ</mark> 'Остаток от деления: ',0
    AL _start
; ---- Вычисление выражения
mov eax,4 ; EAX=5
mov ebx,6; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,2 : EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5; EBX=3
            X=EAX/3, EDX=остаток от деления
div ebx ; E/
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 'edi' в виде символов
mov eax, rem ; вызов 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. 4.11: 611.png

11. Введем в файл lab6-3 программу вычисления выражения.

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
```

Рис. 4.12: 612.png

12. Создадим исполняемый файл и запустим его для вычисления выражения.

```
; Программа вычисления варианта
%include 'in_out.asm'
   TION .data
nsg: DB 'Введите № студенческого билета: ',0
       'Ваш вариант: ',0
  CTION .bss
        80
 SECTION .text
SLOBAL _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. 4.13: 613.png

13. Создадим файл variant.asm в каталоге ~/work/arch-pc/lab06:

```
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132230799
Ваш вариант: 20
aagorbachev@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 4.14: 614png

- 14. Вводим номер студенческого и получаем вариант для выполнения задания
- 15. Составляем программу для нашего варианта lab6-4 (Самостоятельная работа).
- 16. Запускаем программу и вводим два числа из условия, убеждаемся что программа работает верно.

5 Ответы на вопросы:

- 1. строки листинга 7.4 отвечают за вывод на экран сообщения 'Ваш вариант: mov eax и rem call sprint;
- 2. mov ecx,х запись входной переменной в регистр ecx; mov edx, 80 запись размера переменной в регистр edx; call sread вызов процедуры чтения данных;
- 3. call atoi функция преобразующая ASCII код символа в целое число и записывающая результат в регистр eax;
- 4. xor edx, edx mov ebx, 20 div ebx, inc edx;
- 5. div ebx ebx;
- 6. inc используется для увеличения операнда на единицу;
- 7. Следующие строки листинга отвечают за вывод на экран результата вычислений mov eax, rem call sprint mov eax, edx call iprintLF.

6 Выводы

В ходе выполнения данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM.

Список литературы