Київський національний університет імені Тараса Шевченка
факультет радіофізики, електроніки та комп'ютерних систем

Лабораторна робота № 4
Тема: «Вступ до системного програмування на мові асемблера GAS»

Роботу виконав
студент 3 курсу
КІ, МА
Горелов Артем Андрійович

Київ 2021

Мета роботи: ознайомитись із основами синтаксису мови асемблера на x86-64 архітектурі, автоматизацією збірки за допомогою Makefile, відлагоджуванням програм за допомогою відлагоджувача GDB.

Хід виконання роботи:

1. Створіть програму згідно свого варіанту.

1. Потоковий шифр Цезаря

Створіть програму, яка побайтово читає стандартний потік введення, додає до значення байту число 13 (із переповненням) та записує у стандартний потік виведення.

Псевдокод

```
byte b;
main() {
    while(read(stdin, 1, &b) > 0) {
        b += 13;
        write(stdout, 1, &b);
    }
    exit(0);
}
```

Лістинг коду програми:

```
.include "defs.h"
.section .data
b: .byte '0'
.section .text
.global _start
_start:
    loop:
    movq $SYS_READ, % rax
    movq $STDIN, % rdi
    leaq b, % rsi
    movq $1 , % rdx
    syscall
    cmpq $0 , % rax
    je end
    addb $13 , b
    movq $SYS_WRITE, % rax
    movq $STDOUT, % rdi
    movq $b, % rsi
    movq $1 , % rdx
    syscall
    jmp loop
end:
    movq $60 , % rax
    movq $0 , % rdi
syscall
```

2. Створіть Makefile, який за командою `make all` виконає збірку, а за командою `make clean` очистить об'єктні та виконувані файли.

   Правила Makefile:

```
VALUE=lab4
ASFLAGS=--gdwarf-2
LDFLAGS=-static

all: exit
exit: $(VALUE).s
	as $(ASFLAGS) -o $(VALUE).o -c $(VALUE).s
	ld $(LDFLAGS) -o $(VALUE) $(VALUE).o
clean:
	rm -f *.o $(VALUE)
```

   Виклик збірки:

```
[root@localhost lab4_1]# make
as --gdwarf-2 -o norm.o -c norm.s
ld -static -o norm norm.o
[root@localhost lab4_1]# ls
defs.h  exit  exit.o  exit.s  Makefile  norm  norm.o  norm.s
```

3. Продемонструйте результат обробки вашою програмою такого тексту:
   `;X__bJbe_W`

```
[root@localhost lab4_1]# ./norm
;X__bJbe_W
HelloWorld
```

4. Завантажте одержаний виконуваний файл у відлагоджувач за допомогою команди:
   `gdb cipher`

```
[root@localhost lab4_1]# gdb norm
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-120.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/user/lab4_1/norm...done.
(gdb)
```

   Встановіть точку зупинки на початок програми (мітка _start):
   `b _start`

```
(gdb) b _start
Breakpoint 1 at 0x4000b0: file norm.s, line 15.
```

Запустіть програму
`run`

```
(gdb) run
Starting program: /home/user/lab4_1/norm

Breakpoint 1, loop () at norm.s:15
15      movq $SYS READ, % rax
```

Після зупинки виконання програми перегляньте вміст регістрів:
`info registers` або `i r`

```
(gdb) i r
rax            0x0      0
rbx            0x0      0
rcx            0x0      0
rdx            0x0      0
rsi            0x0      0
rdi            0x0      0
rbp            0x0      0x0
rsp            0x7fffffffe090   0x7fffffffe090
r8             0x0      0
r9             0x0      0
r10            0x0      0
r11            0x0      0
r12            0x0      0
r13            0x0      0
r14            0x0      0
r15            0x0      0
rip            0x4000b0 0x4000b0 <loop>
eflags         0x202    [ IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
```

Переходьте до виконання наступної команди:
`next` або `n`

```
(gdb) n
17      movq $STDIN, % rdi
(gdb) n
19      leaq b, % rsi
(gdb) n
21      movq $1 , % rdx
(gdb) n
23      syscall
```

Для виходу із режиму покрокового виконання використовуйте команду `continue` або `c`

```
(gdb) c
Continuing.
```

Для перегляду адресного простору процесу скористайтесь командою `x`: `x/1sb &b`

```
(gdb) x/1sb &b
0x60010d:       "\205"
(gdb) ■
```

Висновок: в даній лабораторній роботі було ознайомлено із мовою асемблера GNU Assembler (GAS) на x86-64 архітектурі та написано програму, яка реалізовує шифр Цезаря, створено Makefile для автоматизації збірки та виконано налагоджування програми через GDB.