

# CPSC 240: Computer Organization and Assembly Language

## Assignment 01, Fall Semester 2023

CWID: \_\_\_\_\_ Name: \_\_\_\_\_

1. Download the “CPSC-240 Assignment01.docx” document.
2. Follow the “CPSC-240 Ex01 Hello World.pdf” slide to design a “hello.asm” Assembly program and generate. “hello.o”, “hello.lst”, and “hello” files.
3. Copy and paste the “hello.asm” file into the document.
4. Follow the “CPSC-240 Ex01 Debugger.pdf” slide to debug the “hello” file.
5. When the program runs to line 12, copy and paste the "Register" window into the document.
6. When the program runs to line 18, copy and paste the "Register" window into the document.
7. When running the "x/14db &text" and "x/s &text" commands, copy and paste the "DDD" window (including the gdb panel) into the document to display the memory results.
8. Save the file in pdf format and submit the pdf file to Canvas before deadline.
9. Deadline is 23:59 pm on 09/06/2023.

[Insert hello.asm file here]

```
; ex_hello.asm
; char text[] = "Hello, World!\n"
; cout << text;
```

```
section .data
text db "Hello, World!", 10
```

```
section .text
global _start
```

```
_start:
mov rax, 1
mov rdi, 1
mov rsi, text
mov rdx, 14
syscall
```

```
mov rax, 60
mov rdi, 0
syscall
```

[Insert 1<sup>st</sup> Register window here]

The screenshot shows the DDD (Data Display Debugger) interface. The main window displays the assembly code for 'hello.asm'. A breakpoint is set at line 12, which is highlighted with a red arrow and a 'STOP' icon. The assembly code is as follows:

```
1 ; ex_hello.asm
2 ; char text[] = "Hello, World!\n"
3 ; cout << text;
4
5 section .data
6     text db "Hello, World!", 10
7
8 section .text
9     global _start
10
11 _start:
12     mov rax, 1
13     mov rdi, 1
14     mov rsi, text
15     mov rdx, 14
16     syscall
17
18     mov rax, 60
19     mov rdi, 0
20     syscall
```

The 'Registers' window is open, showing the state of the CPU registers. The registers are listed in a table with their names, values, and comments.

Register	Value	Comment
rax	0x0	0
rbx	0x0	0
rcx	0x0	0
rdx	0x0	0
rsi	0x0	0
rdi	0x0	0
rbp	0x0	0x0
rsp	0x7fffffff220	0x7fffffff220
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	0x401000	0x401000 <_start>
eflags	0x202	[ IF ]
cs	0x33	51
ss	0x2b	43
ds	0x0	0
es	0x0	0
fs	0x0	0
gs	0x0	0

Below the table, there are two radio buttons: 'Integer registers' (selected) and 'All registers'. At the bottom of the window, there are 'Close' and 'Help' buttons.

The status bar at the bottom of the DDD window shows the command '(gdb) run' and the message 'Starting program: /home/899486336/Desktop/hello/hello.asm'. It also indicates the breakpoint: 'Breakpoint 1, \_start () at hello.asm:12'.

[Insert 2<sup>nd</sup> Register window here]

DDD: /home/899486336/Desktop/hello/hello.asm

File Edit View Program Commands Status

() : hello.asm:12

```

1 ; ex_hello.asm
2 ; char text[] = "Hello, World!\n"
3 ; cout << text;
4
5 section .data
6     text db "Hello, World!", 10
7
8 section .text
9     global _start
10
11 _start:
12     mov rax, 1
13     mov rdi, 1
14     mov rsi, text
15     mov rdx, 14
16     syscall
17
18     mov rax, 60
19     mov rdi, 0
20     syscall

```

(gdb) step

(gdb) step

(gdb) step

Hello, World!

(gdb) [

DDD: Registers

Registers

rax	0xe	14
rbx	0x0	0
rcx	0x40101e	4198430
rdx	0xe	14
rsi	0x402000	4202496
rdi	0x1	1
rbp	0x0	0x0
rsp	0x7fffffff220	0x7fffffff220
r8	0x0	0
r9	0x0	0
r10	0x0	0
r11	0x302	770
r12	0x0	0
r13	0x0	0
r14	0x0	0
r15	0x0	0
rip	0x40101e	0x40101e <_start+30>
eflags	0x202	[ IF ]
cs	0x33	51
ss	0x2b	43
ds	0x0	0
es	0x0	0
fs	0x0	0
gs	0x0	0

Integer registers

All registers

Close

Help

[Insert DDD window here]

(gdb) step

(gdb) step

Hello, World!

(gdb) x/14db &text

0x402000:	72	101	108	108	111	44	32	87
0x402008:	111	114	108	100	33	10		

(gdb) x/s &text

0x402000: "Hello, World!\n"

(gdb) [

3