

Computer Science Department
California State University, Fullerton

CPSC 240-01/02 Computer Organization and Assembly Language

Quiz 01

12:00 pm to 1:15 pm

Monday, October 7, 2024

Student Name: Riley Blacklock

Last 4 digits of ID: 4539

Note:

- University regulations on academic honesty will be strictly enforced.
- You have 75 minutes to complete this Quiz.
- Open books, slides and sample programs.
- Turn off or turn vibration your cell phone.
- Use “yasm” assembler to assemble the source code.
- Use “ld” linker to link the object code
- Use “ddd” debugger to simulate the executable code.
- Each student can only submit solution once, and secondary submissions will not be graded. If you have submitting problems, please inform your instructor before you leave the classroom.
- Any content submitted after the due date will be regarded as a make-up quiz.

Quiz 01

1. Download the “CPSC-240-01 Quiz 01.docx” document.
2. Convert the following C/C++ variable declarations and arithmetic operations to x86-64 assembly language. Use the “yasm” assembler to assemble the program, the “ld” linker to link the object code, and the “ddd” debugger to simulate the executable code.

NOTE: variable sizes and program functions should be equivalent to C/C++ instructions.

```
signed short num1 = +30000;    //16-bit signed variable
signed short num2 = +20000;    //16-bit signed variable
signed short num3 = -3333;     //16-bit signed variable
signed int sum = 0             //32-bit signed variable
signed short quo = 0;          //16-bit signed variable
signed short rmd = 0;          //16-bit signed variable
sum = int(num1 + num2);
quo = sum / num3;
rmd = sum % num3;
```

3. After assembling and linking, run the DDD/GDB debugger to display the simulation results of the **decimal values** of `num1`, `num2`, `num3`, `sum`, `quo`, and `rmd` in GDB panel before terminate program.
4. Insert source code and the simulation results (GDB panel) to the bottom of the document.
5. Save the file in pdf or docx format and submit the pdf or docx file to Canvas before the deadline.
6. Deadline is 1:15 pm on 10/07/2024.

[Copy and paste your assembly source code here:]

```
; quiz1.asm;
; signed short num1 = +30000;
; signed short num2 = +20000;
; signed short num3 = -3333;
; signed int sum = 0;
; signed short quo = 0;
; signed short rmd = 0;
; sum = int(num1 + num2);
```

```
; quo = sum / num3;  
; rmd = sum % num3;
```

```
section .data
```

```
SYS_exit    equ 60
```

```
EXIT_SUCCESS    equ 0
```

```
num1        dw 30000
```

```
num2        dw 20000
```

```
num3        dw -3333
```

```
sum         dd 0
```

```
quo         dw 0
```

```
rmd         dw 0
```

```
section .text
```

```
    global _start
```

```
_start:
```

```
    mov     ax, word[num1]
```

```
    add ax, word[num2]
```

```
    adc dx, 0
```

```
    mov     [sum], ax
```

```
    mov     [sum+2], dx
```

```
    mov     ax, [sum]
```

```
    mov     dx, [sum+2]
```

```
    mov     bx, word[num3]
```

```
    idiv bx
```

```
    mov     word[quo], ax
```

```
    mov     word[rmd], dx
```

```

mov    rax, SYS_exit
mov    rdi, EXIT_SUCCESS
syscall

```

[Attach GDB window with all memory data here:]

The screenshot shows the DDD window with the following content:

Assembly Code:

```

14 EXIT_SUCCESS      equ 0
15 num1              dw 30000
16 num2              dw 20000
17 num3              dw -3333
18 sum               dd 0
19 quo               dw 0
20 rmd               dw 0
21
22 section .text
23     global _start
24 _start:
25     mov     ax, word[num1]
26     add     ax, word[num2]
27     adc     dx, 0
28
29     mov     [sum], ax
30     mov     [sum+2], dx
31
32     mov     ax, [sum]
33     mov     dx, [sum+2]
34     mov     bx, word[num3]
35     idiv    bx
36
37     mov     word[quo], ax
38     mov     word[rmd], dx
39
40     mov     rax, SYS_exit
41     mov     rdi, EXIT_SUCCESS
42     syscall

```

Memory Data (gdb):

```

(gdb) x/hd &num1
0x402000: 30000
(gdb) x/hd &num2
0x402002: 20000
(gdb) x/ww &sum
0x402006: 50000
(gdb) x/hd &num3
0x402004: -3333
(gdb) x/hd &quo
0x40200a: -15
(gdb) x/hd &rmd
0x40200c: 5

```

The status bar at the bottom shows: **0x40200c: 5**