CPSC 240: Computer Organization and Assembly Language
Assignment 03, Spring Semester 2023

CWID: 886880327 Name: Andrew Saldana

1. Download the "CPSC-240 Assignment03.docx" document.
2. Design the "multiplication.asm" program, and use assembly language to realize the function of the following C++ instructions.

```
unsigned int num1 = 300,000;
unsigned int num2 = 400,000;
unsigned long product = 0;
product = long(num1 * num2);
```

3. Assemble the "multiplication.asm" file and link the "multiplication.o" file to get the "multiplication" executable file.
4. Run the "multiplication" file with the DDD debugger to display the simulation results of num1 and num2, as well as the simulation results of product.
5. Insert source code (multiplication.asm) and simulation results (Terminal Emulator window) of the memory (num1, num2, and prod) in the document. Write an analysis to verify simulation results.
6. Design the "division.asm" program, and use assembly language to realize the function of the following C++ instructions.

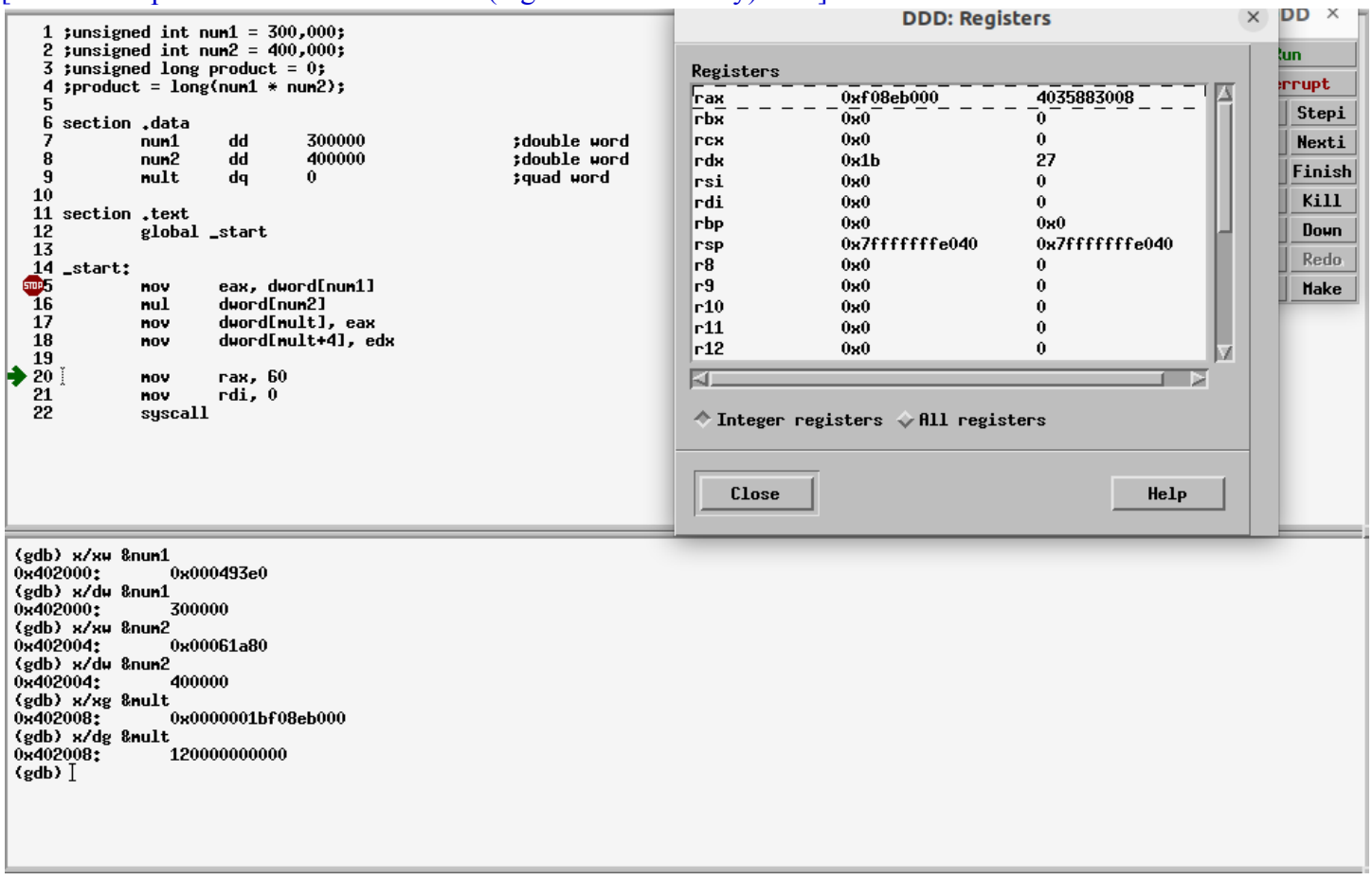
```
unsigned long num1 = 50,000,000,000;
unsigned int num2 = 3,333,333;
unsigned int quotient = 0, remainder = 0;
quotient = num1 / num2;
remainder = num1 % num2;
```

7. Assemble the "division.asm" file and link the "division.o" file to get the "division" executable file.
8. Run the "division" file with the DDD debugger to display the simulation results of num1 and num2, as well as the simulation results of quotient and remainder.
9. Insert source code (division.asm) and simulation results (Terminal Emulator window) of the memory (num1, num2, quotient, and remainder) in the document. Write an analysis to verify simulation results.
10. Save the file in pdf format and submit the pdf file to Canvas before 23:59 pm on 02/22/2023.

[Insert multiplication assembly source code here]

```
;unsigned int num1 = 300,000;
;unsigned int num2 = 400,000;
;unsigned long product = 0;
;product = long(num1 * num2);

section .data
        num1     dd      300000              ;double word        ;num1 = 0004 93E0h
        num2     dd      400000              ;double word        ;num2 = 0006 1A80h
        mult     dq      0                   ;quad word          ;mult = 0000 0000 0000 0000h

section .text
        global _start

_start:
        mov     eax, dword[num1]                                ;eax = num1 = 0004 93E0h
        mul     dword[num2]                                     ;edx:eax = eax * num2 = 0000 001Bh:F08E B000h
        mov     dword[mult], eax                                ;mult = eax = F08E B000h
        mov     dword[mult+4], edx                              ;mult+4 = edx = 0000 001Bh

        mov     rax, 60                                        ;terminate excuting process
        mov     rdi, 0                                         ;exit status
        syscall                                                ;calling system services
```

[Insert multiplication simulation results (register and memory) here]



[Insert multiplication simulation result analysis here]

300,000 in hex

300,000 / 16 = 18750 R 0
18750 / 16 = 1171 R 14
1171 / 16 = 73 R 3
73 / 16 = 4 R 9
4 / 16 = 0 R 4

hex in 32 bit

num1 = 0004 93E0 h

400,000 in hex

400000 / 16 = 25000 R 0
25000 / 16 = 1562 R 8
1562 / 16 = 97 R 10
97 / 16 = 6 R 1
6 / 16 = 0 R 6

hex in 32 bit

num2 = 0006 1A80h

300000 x 400000 = 120,000,000,000

120,000,000,000 in hex

120000000000 / 16 = 7500000000 R 0
7500000000 / 16 = 468750000 R 0
468750000 / 16 = 29296875 R 0
29296875 / 16 = 1831054 R 11
1831054 / 16 = 114440 R 14
114440 / 16 = 7152 R 8
7152 / 16 = 447 R 0
447 / 16 = 27 R 15
27 / 16 = 1 R 11
1 / 16 = 0 R 1

Hex in 64 bit

mult = 0000 001B F0&E B000h

edx eax

[Insert division assembly source code here]

```
;unsigned long num1 = 50,000,000,000;
;unsigned int num2 = 3,333,333;
;unsigned int quotient = 0, remainder = 0;
;quotient = num1 / num2;
;remainder = num1 % num2;

section .data
        num1      dq      50000000000                   ;num1 = 0000 000B A43B 7400h
        num2      dd      3333333                       ;num2 = 0032 DCD5h
        quotient  dd      0                             ;quotient = 0000 0000h
        remainder dd      0                             ;remainder = 0000 0000h

section .text
        global _start

_start:
        mov     edx, dword[num1+4]                ;edx = num1+4 = 0000 000Bh
        mov     eax, dword[num1+0]                ;eax = num1+0 = A43B 7400h

        div     dword[num2]                       ;eax = eax/num2 = 0000 3A98 = 15000
                                                  ;edx = eax%num2 = 0000 1388h = 5000

        mov     dword[quotient], eax              ;quotient = eax = 0000 3A98h = 15000
        mov     dword[remainder], edx             ;remainder = edx = 0000 1388h = 5000

        mov     rax, 60                           ;terminate excuting process
        mov     rdi, 0                            ;exit status
        syscall                                   ;calling system services
```

[Insert division simulation results(register and memory) here]

```
 2 ;unsigned int num2 = 3,333,333;
 3 ;unsigned int quotient = 0, remainder = 0;
 4 ;quotient = num1 / num2;
 5 ;remainder = num1 % num2;
 6
 7 section .data
 8         num1      dq      50000000000
 9         num2      dd      3333333
10         quotient  dd      0
11         remainder dd      0
12
13 section .text
14         global _start
15
16 _start:
17         mov     edx, dword[num1+4]
18         mov     eax, dword[num1+0]
19
20         div     dword[num2]
21
22
23         mov     dword[quotient], eax
24         mov     dword[remainder], edx
25
26         mov     rax, 60
27         mov     rdi, 0
28         syscall
```

DDD: Registers

Registers

rax	0x3a98	15000
rbx	0x0	0
rcx	0x0	0
rdx	0x1388	5000
rsi	0x0	0
rdi	0x0	0
rbp	0x0	0x0
rsp	0x7fffffffe070	0x7fffffffe070
r8	0x0	0
r9	0x0	0

Integer registers All registers

Close Help

```
(gdb) x/xg &num1
0x402000:       0x0000000ba43b7400
(gdb) x/dg &num1
0x402000:       50000000000
(gdb) x/xw &num2
0x402008:       0x0032dcd5
(gdb) x/dw &num2
0x402008:       3333333
(gdb) x/xw &quotient
0x40200c:       0x00003a98
(gdb) x/dw &quotient
0x40200c:       15000
(gdb) x/xw &remainder
0x402010:       0x00001388
(gdb) x/dw &remainder
0x402010:       5000
(gdb)
```

[Insert division simulation result analysis here]

50,000,000,000 in hex

50000000000 / 16	=	3125000000	R 0
3125000000 / 16	=	195312500	R 0
195312500 / 16	=	12207031	R 4
12207031 / 16	=	762939	R 7
762939 / 16	=	47683	R 11
47683 / 16	=	2980	R 3
2980 / 16	=	186	R 4
186 / 16	=	11	R 10
10 / 16	=	0	R 10

hex in 64 bit

num 1 = 0000 00A4 A43B 7400 h

3333333 in hex

3333333 / 16	=	208333	R 5
208333 / 16	=	13020	R 13
13020 / 16	=	813	R 12
813 / 16	=	50	R 13
50 / 16	=	3	R 2
3 / 16	=	0	R 3

hex in 32 bit

num 2 = 0032 DCD5 h

50000000000 / 333333 = 15000 R 5000

15000 in hex

15000 / 16	=	937	R 8
937 / 16	=	58	R 9
58 / 16	=	3	R 10
3 / 16	=	0	R 3

hex in 32 bit

quotient = 0000 3A98 h

5000 in hex

5000 / 16	=	312	R 8
312 / 16	=	19	R 8
19 / 16	=	1	R 3
1 / 16	=	0	R 1

hex in 32 bit

remainder = 0000 1388 h