CS 224 Homework 3.5

30 Points

This assignment is designed to give you additional practice on the types of questions that you will encounter on Midterm 2, which will focus entirely on the Y86-64 instructions set and underlying architecture.

Note: Since you will not have access to the Y86-64 simulator during the exam, you should not use it for this assignment, so that you are as prepared as possible for the Midterm.

Grading and Feedback

Pass/Fail: If you make an honest attempt at this homework, you will receive full points. If you don't turn in anything, or turn in a submission that does not represent an honest effort, you will receive a 0.

Once you have submitted your assignment on Canvas, the TAs will look it over briefly to ensure that you made an honest effort to answer all of the questions. If that is the case, you will receive full credit for the assignment and the TA will immediately email you the solutions to this homework, so that you can use them in studying and preparing for the midterm.

1. **[10 points] Y86-64 Assembly and the Program State**

   Consider the following Y86-64 program. The object code is shown on the left, and the assembly code on the right. Your task is to determine what the program state will be after this program is run to completion.

```
0x0000: 30f40001000000000000    | irmovq stack, %rsp
0x000a: 30f70300000000000000    | irmovq 0x3, %rdi
0x0014: 30f60400000000000000    | irmovq 0x4, %rsi
0x001e: 8028000000000000000     | call loopy
0x0027: 00                      | halt
                                |
0x0028:                         | loopy:
0x0028: 2072                    |         rrmovq  %rdi, %rdx
0x002a: 6222                    |         andq    %rdx, %rdx
0x002c: 717500000000000000      |         jle     .L5
0x0035: 2062                    |         rrmovq  %rsi, %rdx
0x0037: 6222                    |         andq    %rdx, %rdx
0x0039: 717500000000000000      |         jle     .L5
0x0042: 30f00000000000000000    |         irmovq  $0, %rax
0x004c: 30f80100000000000000    |         irmovq  $1, %r8
0x0056: 706300000000000000      |         jmp     .L3
0x005f:                         | .L4:
0x005f: 6060                    |         addq    %rsi, %rax
0x0061: 2027                    |         rrmovq  %rdx, %rdi
0x0063:                         | .L3:
0x0063: 2072                    |         rrmovq  %rdi, %rdx
0x0065: 6182                    |         subq    %r8, %rdx
0x0067: 2071                    |         rrmovq  %rdi, %rcx
0x0069: 6211                    |         andq    %rcx, %rcx
0x006b: 765f00000000000000      |         jg      .L4
0x0074: 90                      |         ret
0x0075:                         | .L5:
0x0075: 30f00000000000000000    |         irmovq  $0, %rax
0x007f: 90                      |         ret
                                |
0x0080:                         | .pos 0x100
0x0100:                         | stack:
```

   Please fill in the following details about the program state **after the program has run to completion**.

   (a) Condition Codes - Indicate the value of the three condition codes:

SF		ZF		OF	

(b) Registers - Indicate the value in each of the following registers:

%rax		%rdx		%rcx	
%rsi		%rdi		%rbx	
%rsp		%r8		%r10	

(c) What is the value of the program counter `PC`?

(d) What does the function `loopy` do? What are its inputs? What is its output? What is the relationship of the inputs to the output?

(e) **[10 points] Encoding Y86-64 Instructions**

Consider the following sequence of Y86-64 instructions. You will be determining the byte-level encoding for these instructions, assuming that they are placed in memory starting at memory location `0x000`.

```
irmovq $5, %rax
irmovq $10, %rdx
subq %rax, %rdx
andq %rdx, %rax
halt
```

Please fill in following memory table, placing the encoded byte in each memory cell. Leave any cells blank that are not used by the given instruction sequence. Memory is displayed as in the Y86-64 online simulator, that is, memory addresses increase as we move to the right and down. The address of the first byte shown on each row is given to the left of the row. Each cell of the memory table should contain a single byte (if used in the encoding), or be blank (if not used in the encoding).

Address	Memory							
0x000	30	F0	05	00	00	00	00	00
0x008	00	00	30	F2	0A	00	00	00
0x010	00	00	00	00	61	02	62	20
0x018	00							

Consider the following sequence of bytes, shown in memory starting at address 0x000. Your task is to determine the Y86-64 instruction sequence that is encoded in these bytes. Memory is displayed as in the Y86-64 online simulator, that is, memory addresses increase as we move to the right and down. The address of the first byte shown on each row is given to the left of the row. Each cell of the memory table contains a single byte.

Address	Memory							
0x000	30	f2	03	00	00	00	00	00
0x008	00	00	10	30	f1	01	00	00
0x010	00	00	00	00	00	60	20	61
0x018	12	76	15	00	00	00	00	00
0x020	00	00	00					

What is the sequence of Y86-64 instructions that is encoded in these bytes?

```
0x000:  irmovq $3, %rdx
0x00a:  nop
0x00b:  irmovq $1, %rcx
0x015:  addq %rdx, %rax
0x017:  subq %rcx, %rdx
0x019:  jg 0x015
0x022:  halt
```

After this program has been run to completion, what is the value in register %rax?

%rax = 6