140 + 10

COMP.3050 Test 2 (200 points) 4/8/2019

CLOSED BOOK/ LAPTOP. No calculators, but two pages of notes allowed.

NAME: PHONG VO

Give specific numbers where appropriate, not a general verbal description.
Exam is printed double-sided.

1. The following assembly program (which has line reference numbers attached to help you answer the questions below) will begin executing at location 0:

```
0          lodd start:
1          stod 4093
2          stod 4095
3 top:     lodd 4095
4          subd mask:
5          jneg top
6 print:   lodd char:
7          stod 4094
8          addd c4:
9          stod char:
10         subd chlim:
11         jneg cont:
12         lodd chB:
13         stod char:
14 cont:   lodd 4093
15         subd mask:
16         jneg top:
17 done:   halt
18 start:  8
19 mask:   10
20 chlim:  "G"
21 chB:    "B"
22 char:   "A"
23 prompt: "Please type in a positive integer"
24 c4:     4
```

A	E	B	F	B	F
A	E	B	F	B	F
E	I	F	J	F	J
E	I	F	J	F	J
−	+	−	+	−	+
−	+	−	+	−	+
	B		B		B
	B		B		B

A. (15 pts) Line number 3 loads a value out of the transmitter status register. What value will it read when the UART is ready to transmit?

$1010_{(2)}$ OR 10 ~~8~~ x -15

B. (40 pts) What are the first six characters written to the display?

AEBFBF ✓

40

2. (30 points) The MIC-1 bit format is shown below. You should be familiar with all the fields and how they are used. Also below are 3 MAL instructions. Indicate if a given MAL is valid or invalid for MIC-1, and, if valid, fill in the **DECIMAL** (i.e. bits 1101 are filled as 13) values for each field **in the space provided**. **If invalid, write below the figure why not, but in case you are wrong fill in as many of the fields as you can.**

Register designations are as follows: pc=0 (prog counter) ac=1 (accumulator) sp=2 (stack ptr) ir=3 (instr reg) tir=4 (tmp inst reg) zr=5 (fixed zero) po=6 (plus 1) no=7 (minus 1) amask=8 (addr msk) smask=9 (stack msk) a=10(a scratch) b=11(b scratch) c=12(c scratch) d=13(d scratch) e=14(e scratch) f=15(f scratch)

A. mbr := lshift(band(ir,amask));ir := lshift(band(pc,amask));goto 100;
B. b := rshift(band(a,smask)); if n then goto 70;
C. tir :=lshift(band(tir,mbr));if z then goto 22;

VALID?	AMUX	COND	ALU	SH	MBR	MAR	RD	WR	ENC	C	B	A	ADDR	
N		3		2		10							100	
Y	0	1	1	1	1	0	0	0	0	12	11	10 4	70	X -1
Y	?	1	2	2	?	0	0	0		?	4		?	X -3
		1	2	2	0	0	0	0	1	4	4		22	

AMUX	COND	ALU	SH	MBR,MAR,RD,WR,ENC
0=A latch	0 = no jmp	0 = A + B	0 = no shift	0 = no
1=MBR	1 = jmp if n=1	1 = A and B	1 = shift rt	1 = yes
	2 = jmp if z=1	2 = A	2 = shift lt	
	3 = always jmp	3 = not A		

Invalid rows **Why invalid**

1 ✓ SH and MAR are overflow X -7

19

3. (10 pts) What two Mic-1 instructions must be used to access an argument that has been passed by reference to a function? (Circle the best answer)
   A) lodl addl
   B) lodl lodd
   C) lodl pushi
   D) lodl pop
   E) addl pushi
   F) addl subl
   G) addl swap

4. (10 pts) If a Mic-1 microcode instruction is supposed to modify the SP register, which bit in the microinstruction must be set to 1? (See the microinstruction format on the previous page. Circle the best answer.)
   A) RD
   B) WR
   C) ENC
   D) MBR
   E) MAR
   F) AMUX

5. (40 pts) Refer to the Mic1 microcode and specify ALL the lines of microinstructions after 0, 1, and 2 that are executed to do a CALL **in the order they are executed**. Don't list lines 0, 1, and 2.

28 29 35 38 39 16 X -34

28 40 46 47 48 49

6. (15 pts) In line 3 of the microcode, what is affecting the N flag?
   a) lshift
   b) ir + ir
   c) both

41

7. (40 pts) Suppose a new MACRO instruction is to be added to the set in the book (like you did in Assign 4). This new instruction is called **QUINTUPLE** and will take a value on the top of the stack and will multiply it by 5 (by adding it five times to itself, NOT actually multiplying) and put the new value back on the top of the stack. You should not modify the stack in any other way or modify AC. **Don't give up! This isn't that difficult.** Just get the value off the top of the stack, do the addition five times (you don't need to loop), and put it back. Use existing microcode for guidance for how to write the new microcode. You do **NOT** have to worry about **DECODING its OPCODE**, but will assume that your code begins at line 100 **with the first MAL statement needed after the instruction has been successfully decoded** by earlier parts of the microprogram and has jumped to your first MAL statement. The instruction is in the IR register, if needed. You must make sure that when you finish putting the new number on the top of the stack, your microprogram segment will continue normal machine execution.

```
100 : mar := sp; rd; {read value from memory}
101 : rd;
102 : a := mbr;
103 : b := a;
104 : b := a + b;
105 : b := b + b;
106 : b := a + b;
107 : mbr := b; wr;
108 : wr; goto Ø;
```

40