UNIVERSITY OF TORONTO

Faculty of Arts and Science

April 2015 Examinations

CSC258H1S: Computer Organization

Duration: 3 hours

Permitted Aids: one ruler, one highlighter

Last Name: _______________________________________

First Name: _______________________________________

Student Number: ___________________________________

Instructors: **Steve Engels (L0101)**

Larry Zhang (L0201)

Myrto Papadopoulou (L5101)

Mark Breakdown

Part A:	/ 34
Part B:	/ 26
Part C:	/ 17
Part D:	/ 27
Part E:	/ 12
Part F:	/ 24
Bonus:	/ 1
Total:	**/ 140**

Instructions:

- **Write your name on every page of this exam.**
- **Do not open this exam until you hear the signal to start.**
- Have your student ID on your desk.
- No aids permitted other than writing tools. Keep all bags and notes far from your desk before the exam begins.
- There are 6 questions on 16 pages. When you hear the signal to start, make sure that your exam is complete before you begin.
- Read over the entire exam before starting.
- If you use any space for rough work or have to use the overflow page, clearly indicate the section(s) that you want marked.

Part A: Short Answer (34 marks)

Answer the following questions in the space provided. When providing a written answer, write **as clearly and legibly as possible**. Marks will not be awarded to unreadable answers.

1. Which ones of the following hexadecimal addresses are valid PC values when translated into 32-bit binary addresses? Circle all that apply. **(2 marks)**

a) `E45B23C1` **b)** `4C32A330`

c) `9B31FF1C` **d)** `1234ABCD`

2. Which is of the following registers always stores a value that is divisible by 4? Circle all that apply. **(2 marks)**

a) `$zero` **b)** `$t0`

c) `$sp` **d)** `$ra`

3. In an R-type instruction, how many bits are reserved for the shift amount? **(1 mark)**

a) `5` **b)** `6`

c) `16` **d)** `26`

4. In an I-type instruction, what operations might need to be performed on the immediate value? Circle all that apply. **(2 marks)**

a) sign extension **b)** zero extension

c) ones extension **d)** inversion

5. Which of the following 4-bit signed binary arithmetic operations will cause an overflow? Circle all that apply. **(2 marks)**

a) `1001 + 0100` **b)** `1010 + 1110`

c) `1111 + 0001` **d)** `0100 + 0100`

6. If a one-hot decoder has a binary output of `1000000000000000`, what would the input be? **(2 marks)**

7. What is the maximum distance that a branch instruction can jump (in bytes)? **(1 mark)**

a) 2^{16} **b)** 2^{18}

c) 2^{26} **d)** 2^{28}

8. What is the maximum distance that a jump instruction can jump (in bytes)? **(1 mark)**

a) 2^{16} **b)** 2^{18}

c) 2^{26} **d)** 2^{28}

9. What is the output of a tri-state buffer when both inputs are zero? **(1 mark)**

10. How many bytes can a register unit store, with 6 address bits and 32-bit words? **(2 marks)**

11. How many bits does a register unit need to address each register, if it stores 1024 bytes in 32-bit integers? **(2 marks)**

12. To load the value `1101101100110110` into a 16-bit shift register, it takes X clock cycles. To load the same value into a 16-bit load register, it takes Y clock cycles. What are the values of X and Y? **(2 marks)**

X = ______________ **Y = ______________**

13. Which of the following assembly code segments would implement the pseudo-instruction `bge $s, $t, Label`? Circle all that apply. **(2 marks)**

```
slt $d, $s, $t
beq $d, $zero, Label
```

```
slt $d, $t, $s
beq $d, $zero, Label
```

```
beq $s, $t, Label
slt $d, $s, $t
bne $d, $zero, Label
```

```
beq $s, $t, Label
slt $d, $t, $s
bne $d, $zero, Label
```

14. Which of the following Verilog statements cannot appear inside an `always` block? Circle all that apply. **(2 marks)**

a) `a = b;` **b)** `assign a = b;`

c) `a <= b;` **d)** `not (a, b);`

15. In Lab 1, you had to implement a 5-to-1 multiplexer with 2-to-1 multiplexers. What is the minimum number of 2-to-1 multiplexers that you would need to implement a 15-to-1 multiplexer? **(1 mark)**

16. Which of the following are equivalent to $X\overline{Y}$? Circle all that apply. **(2 marks)**

a) $\overline{(\overline{X} + Y)}$ **b)** $X(\overline{Y}Z + \overline{Y}\overline{Z}X)$

c) $\overline{(X + \overline{Y})}$ **d)** $X\overline{Y}Z + \overline{X}\overline{Y}\overline{Z}$

17. The MOSFET is made up of three types of material, listed below. Sort these materials according to the conductivity, from most conductive (1) to least conductive (3). **(2 marks)**

Metal ____ **Oxide ____** **Semiconductor ____**

18. Other than the proposal, there were 5 parts of Lab 6 that you needed to do. What were those parts? Fill in your answer in the spaces below. **(5 marks)**

a) ________________

b) ________________

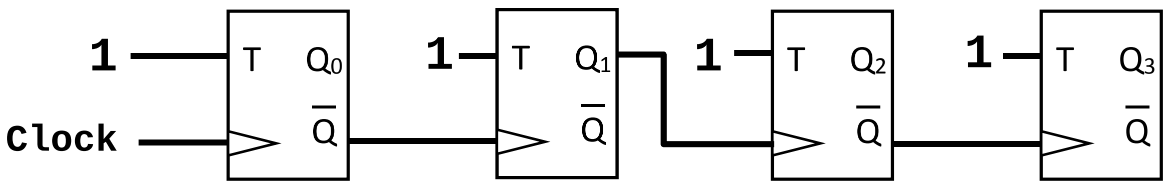
c) ________________

d) ________________

e) ________________

Part B: Design and Analysis (26 marks)

1. Consider the flip-flop diagram below. What values will be on Q_3, Q_2, Q_1, and Q_0, after each consecutive clock cycle? Write your answer in the spaces provided, one space per clock cycle. **(6 marks)**



Q_3 Q_2 Q_1 Q_0

Initial value (before first clock pulse): 0000 ________

0001 ________

____

____

____

____

____

____

____

____

____

____

____

____

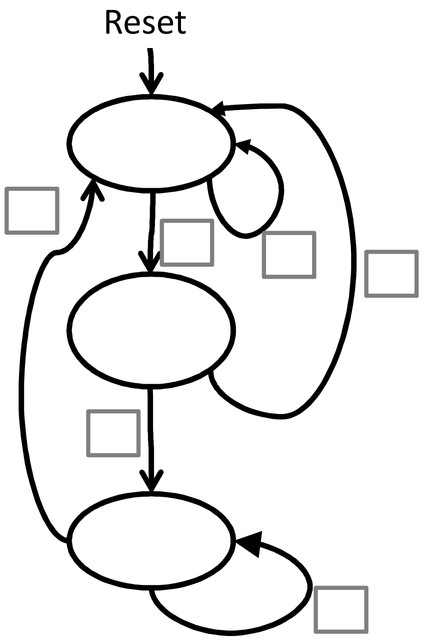
____

____

2. Consider the Verilog code on the left, which implements a Finite State Machine. Refer to this code when answering the questions on the right. **(14 marks)**

```
module fsm(clock, reset, X, Z);

  input clock, reset, X;
  output reg Z;
  reg [2:0] state, next_state;

  always @ (posedge clock)
     if (reset)
        state <= 1;
     else
        state <= next_state;


  always @ (state)
     case (state)
       2: Z = 1;
       default: Z = 0;
     endcase

  always @ (state, X)
     case (state)
       1: if (!X)
            next_state = 4;
           else
            next_state = 1;
       2: if (X)
            next_state = 1;
          else
            next_state = 2;
       4: if (!X)
            next_state = 2;
          else
            next_state = 1;
       default: next_state = 1;
     endcase
endmodule
```

i) Complete the state diagram of this FSM by filling in each state with the correct state number, and labeling each transition with the matching values for X (0 or 1). Ignore any transitions due to reset. **(9 marks)**



ii) Given the above state transition diagram, fill in the output for each state. **(3 marks)**

State	Output
1	
2	
4	

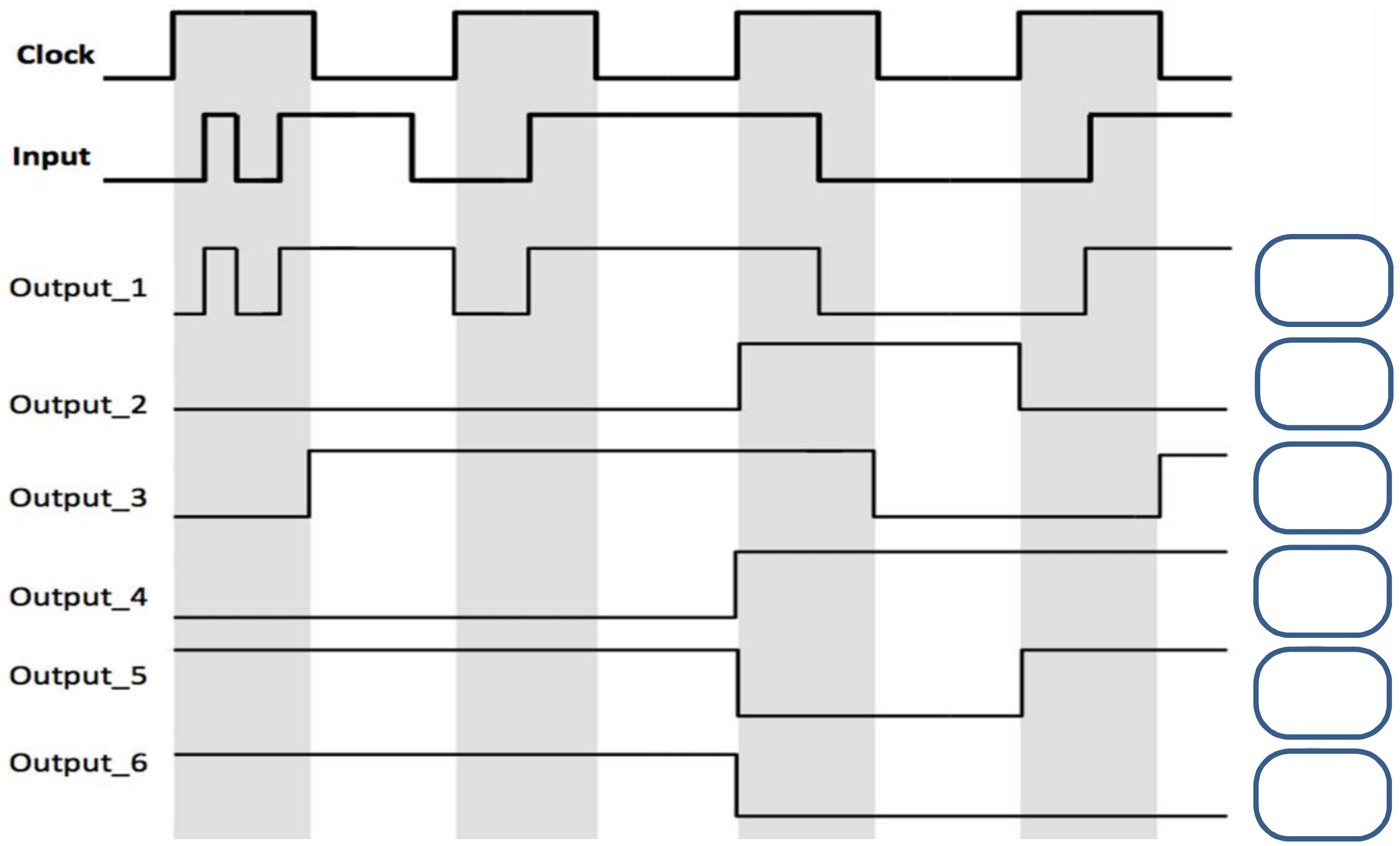
iii) Given the sequence of inputs 0, 1, 0, which will be the resulting FSM state? **(1 mark)**

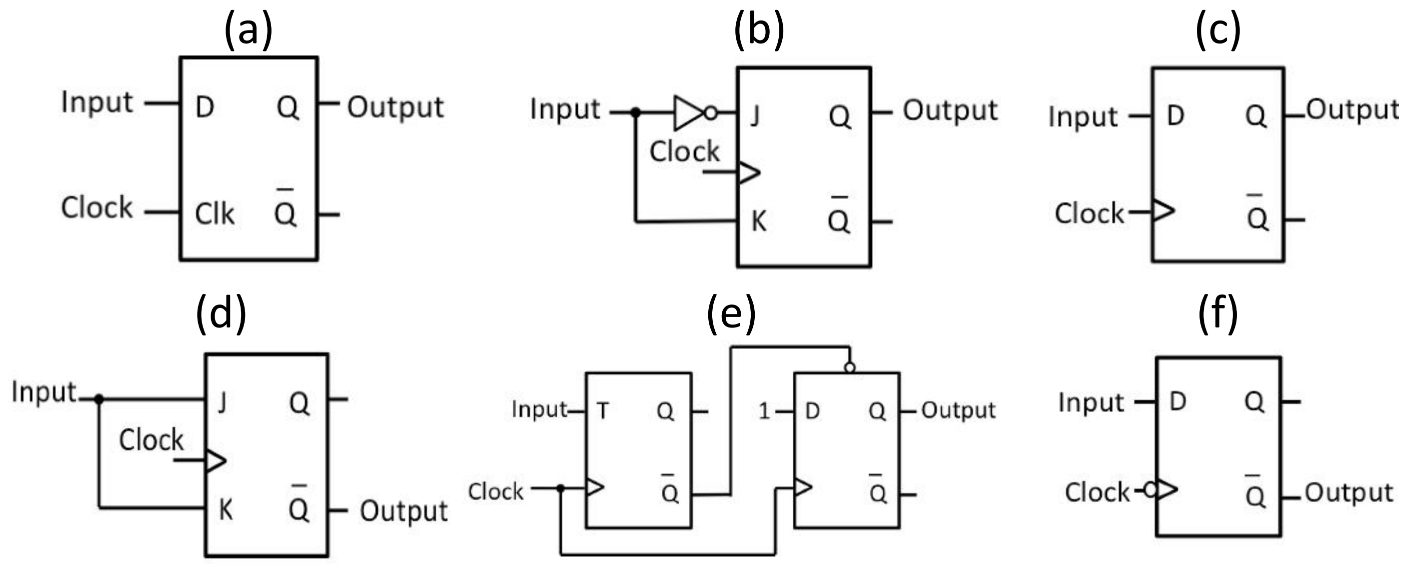
______

iv) What type of FSM is this? **(1 mark)**

a) Mealy **b) Moore**

3. For every output waveform below, specify in the spaces on the right which sequential circuits at the bottom would generate this output behaviour. If you think there is no match, write "None". Assume functional simulation and that the initial Q state, when unknown, was zero. **(6 marks)**





Part C: Processor Operations (17 marks)

1. When Booth's Algorithm is performed on the binary inputs A=1011 and B=1001, the values for A and P change at each step of the algorithm. The framework is provided below, with a few values filled in for you. Fill in the rest, according to the steps shown in class. **(5 marks)**

Initial Values:	A=	**10110**	B=	**1001**	-B=	**0111**

Step #1:

A =	10110	**Initial P value =**	0000 0000
		New P value =	0111 0000

Step #2:

A =	11011	**Initial P value =**	
		New P value =	

Step #3:

A =	11101	**Initial P value =**	
		New P value =	

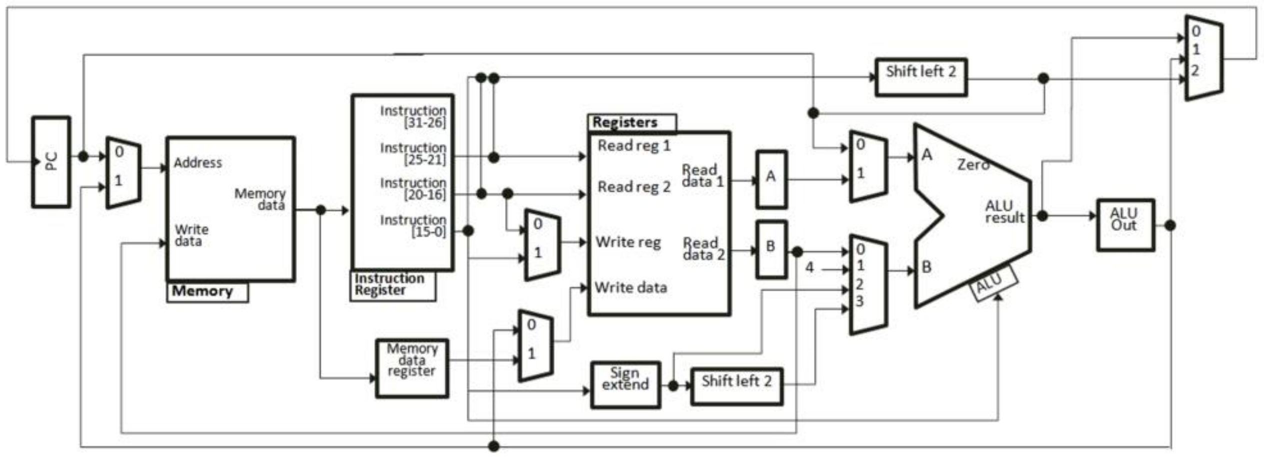
Step #4:

A =	11110	**Initial P value =**	
		New P value =	

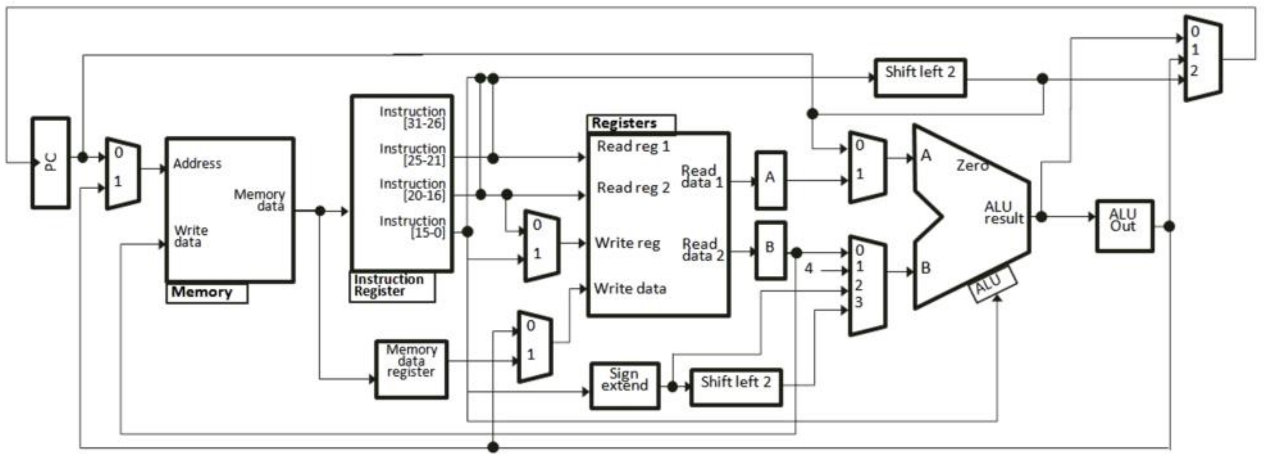
Final P value (binary) =		**Final P value (decimal) =**	

2. Consider the datapaths below. For each of the following operations, highlight the path that the data needs to take, from start to finish. **(12 marks)**

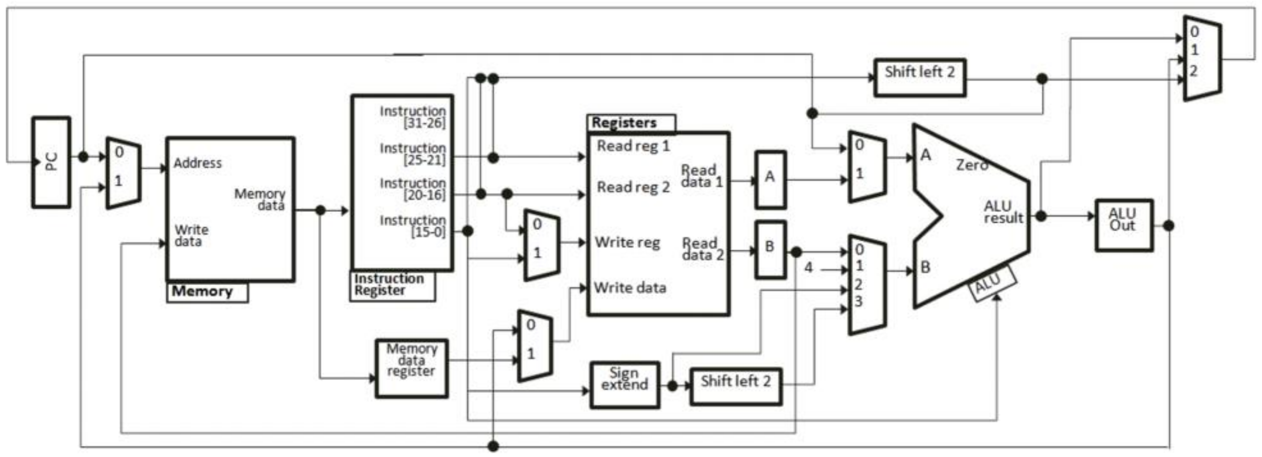
a) Store `$ra` into the program counter.



b) Store the program counter in `$ra`.



c) Push the value stored in `$a0` onto the stack.



Part D: Processor Instructions (27 marks)

1. Each assembly language instructions below matches a machine code instruction at the bottom. However, the opcodes have been removed from each machine code instruction. Match the assembly language instructions to their corresponding machine code instructions, and label each machine code instruction with its missing opcode. **(12 marks)**

A) `or $t8, $t0, $t1`	D) `lw $v0, -1($sp)`
B) `jalr $v0`	E) `srlv $a0, $a1, $a2`
C) `lbu $t1, -30($t0)`	F) `addiu $ra, $zero, 6143`

i. ☐ 11101000101111111111111111

Assembly code instruction: ___________________

ii. ☐ 00110001010010011101000110

Assembly code instruction: ___________________

iii. ☐ 00000111110001011111111111

Assembly code instruction: ___________________

iv. ☐ 11000010000100100000100101

Assembly code instruction: ___________________

v. ☐ 00010111110000000000001001

Assembly code instruction: ___________________

vi. ☐ 01000010011111111111100001

Assembly code instruction: ___________________

2. For each of the processor tasks below, indicate what the values of the following control unit signals will be by filling in the boxes next to each signal with the signal values. **(15 marks)**

- If a control signal doesn't affect the operation, fill in its value with an X.
- For ALUOp, full marks will only be given for binary values. If you don't know what the values are, just write what kind of operation is taking place instead.

Add 4 to the stack pointer value.

PCWrite		PCWriteCond		IorD		MemRead		MemWrite	
MemToReg		IRWrite		PCSource		ALUOp			
ALUSrcA		ALUSrcB		RegWrite		RegDst			

Add 100 to the program counter, and store the result in $ra.

PCWrite		PCWriteCond		IorD		MemRead		MemWrite	
MemToReg		IRWrite		PCSource		ALUOp			
ALUSrcA		ALUSrcB		RegWrite		RegDst			

Fetch and load the next instruction from memory.

PCWrite		PCWriteCond		IorD		MemRead		MemWrite	
MemToReg		IRWrite		PCSource		ALUOp			
ALUSrcA		ALUSrcB		RegWrite		RegDst			

Part E: Verilog (12 marks)

Consider the piece of Verilog code on the right.

1. In a sentence or less, describe what operation this code performs. **(4 marks)**

2. Based on your answer above, what do the `i` and `o` reg values represent? **(2 marks)**

i: ______________________________

o: ______________________________

```
module one (o, s, t, d, i, m, w);

input [31:0] m;
input w;
output reg [5:0] o;
output reg [4:0] s, t, d;
output reg [15:0] i;
reg [31:0] c;

always @ (*)
  begin
    if (w)  c <= m;
    o <= c[31:26];
    s <= c[25:21];
    t <= c[20:16];
    d <= c[15:11];
    i <= c[15:0];
  end
endmodule
```

Consider the piece of Verilog code on the right.

3. In one sentence, describe what function this code performs. **(4 marks)**

4. Given your answer to part 1, what functions do input signals `e` and `w` perform? **(2 marks)**

```
module two (c, e, w, a, di, do);
  input   c, w, e;
  input  [7:0] a;
  input  [3:0] di;
  output [3:0] do;

  reg    [3:0] M [255:0];
  reg    [3:0] do;

  always @(posedge c)
  begin
    if (e) begin
       if (w)
          M[a] <= di;
       else
          do <= M[a];
    end
  end
endmodule
```

Part F: Assembly Language (24 marks)

1. In the spaces provided below, write the assembly language instruction(s) that perform the following tasks. Full marks will only be given for one-instruction answers. **(12 marks total)**

a) Set the value stored in `$s0` to one eighth of its original value. **(3 marks)**

b) Invert all the bits of the value stored in `$a0`. **(3 marks)**

c) Pop a character off the stack and store it in `$t0`. **(3 marks)**

d) Set the program counter to zero. **(3 marks)**

2. In the space provided, write the resulting `list` values for each assembly program. **(12 marks)**

```
          .data
len:      .word     5
list:     .word     -4, -1, 0, 1, 4
          .text
main:     addi $s0, $zero, list
          addi $s1, $zero, len
          lw $t1, 0($s1)
top:      lw $t0, 0($s0)
          add $t0, $t0, $t0
          sw $t0, 0($s0)
          addi $t1, $t1, -1
          addi $s0, $s0, 4
          bne $t1, $zero, top
end:      jr $ra
```

```
          .data
len:      .word     5
list:     .word     -4, -1, 0, 1, 4
          .text
main:     addi $s0, $zero, list
          addi $s1, $zero, len
          lw $t1, 0($s1)
top:      lw $t0, 0($s0)
          sub $t0, $t0, $t0
          sw $t0, 0($s0)
          addi $t1, $t1, -1
          addi $s0, $s0, 4
          bne $t0, $zero, top
end:      jr $ra
```

```
          .data
len:      .word     5
list:     .word     -4, -1, 0, 1, 4
          .text
main:     addi $s0, $zero, list
          addi $s1, $zero, len
          lw $t1, 0($s1)
top:      lw $t0, 0($s0)
          addi $t1, $t1, -1
          addi $s0, $s0, 4
          add $t0, $t0, $t0
          sw $t0, 0($s0)
          bne $t1, $zero, top
end:      jr $ra
```

```
          .data
len:      .word     5
list:     .word     -4, -1, 0, 1, 4
          .text
main:     addi $s0, $zero, list
          addi $s1, $zero, len
          lw $t1, 0($s1)
          lw $t0, 0($s0)
top:      addi $t1, $t1, -1
          add $t0, $t0, $t1
          bne $t1, $zero, top
          addi $s0, $s0, 4
          sw $t0, 0($s0)
end:      jr $zero
```

Reference Information

ALU arithmetic input table:

Select		Input	Operation	
S_1	S_0	Y	$C_{in}=0$	$C_{in}=1$
0	0	All 0s	G=A	G=A+1
0	1	B	G=A+B	G=A+B+1
1	0	B	G=A-B-1	G=A-B
1	1	All 1s	G=A-1	G=A

Register assignments:

Register values : Processor role

- Register 0 ($zero): reserved value.
- Register 1 ($at): reserved for the assembler.
- Registers 2-3 ($v0, $v1): return values
- Registers 4-7 ($a0-$a3): function arguments
- Registers 8-15, 24-25 ($t0-$t9): temporaries
- Registers 16-23 ($s0-$s7): saved temporaries
- Registers 28-31 ($gp, $sp, $fp, $ra)

Bonus Question: (1 mark)

Draw something in the space below.

Instruction table:

Instruction	Type	Op/Func	Syntax
add	R	100000	$d, $s, $t
addu	R	100001	$d, $s, $t
addi	I	001000	$t, $s, i
addiu	I	001001	$t, $s, i
div	R	011010	$s, $t
divu	R	011011	$s, $t
mult	R	011000	$s, $t
multu	R	011001	$s, $t
sub	R	100010	$d, $s, $t
subu	R	100011	$d, $s, $t
and	R	100100	$d, $s, $t
andi	I	001100	$t, $s, i
nor	R	100111	$d, $s, $t
or	R	100101	$d, $s, $t
ori	I	001101	$t, $s, i
xor	R	100110	$d, $s, $t
xori	I	001110	$t, $s, i
sll	R	000000	$d, $t, a
sllv	R	000100	$d, $t, $s
sra	R	000011	$d, $t, a
srav	R	000111	$d, $t, $s
srl	R	000010	$d, $t, a
srlv	R	000110	$d, $t, $s
beq	I	000100	$s, $t, label
bgtz	I	000111	$s, label
blez	I	000110	$s, label
bne	I	000101	$s, $t, label
j	J	000010	label
jal	J	000011	label
jalr	R	001001	$s
jr	R	001000	$s
lb	I	100000	$t, i ($s)
lbu	I	100100	$t, i ($s)
lh	I	100001	$t, i ($s)
lhu	I	100101	$t, i ($s)
lw	I	100011	$t, i ($s)
sb	I	101000	$t, i ($s)
sh	I	101001	$t, i ($s)
sw	I	101011	$t, i ($s)
trap	I	001100	i
mflo	R	010010	$d

This page is left blank intentionally for answer overflows.

Total Marks = 140

Total Pages = 16

End of exam