

ECE2030A
Introduction to Computer Engineering
Final Exam
Dec 11, 2008

2 hours 50 min exam.

**Close books, close notes. One double-sided letter-size cheat sheet allowed.
NO calculator.**

This exam is given under the Georgia Tech Honor Code System. You must observe and sign the Honor Pledge: "I have neither given nor received aid on this exam." Your print name and signature below signifies your compliance with this honor code.

Name (Print): _____ **SOLUTION** _____

Signature: _____

1. _____ (15 pts)
2. _____ (10 pts)
3. _____ (15 pts)
4. _____ (10 pts)
5. _____ (15 pts)
6. _____ (15 pts)
7. _____ (20 pts)

Total (100 pts) _____

1. (15%) General Questions. (Circle **one** for the answer.)

1.1. Which of the following is the hex number of decimal “-16” using a 5-bit binary?

- (a) 11111
- (b) 10000
- (c) 11110
- (d) Cannot be represented.

1.2. For an integer arithmetic shift, which of the following is **false**?

- (a) When shift to the Right, the sign bit needs to be copied to the MSB.
- (b) When shift to the Right, the result is the floor value of the input divided by 2.
- (c) When shift to the Left, we must check if an overflow/underflow is encountered.
- (d) None of the above.

1.3. What of the following is **false** for a $2^k \times N$ DRAM where $k=22$, $N=16$?

- (a) There are 4M addresses
- (b) The total capacity of this DRAM is 64 Mega-Bytes
- (c) When using a 12-to-4096 row decoder, we need a 1024-to-1 multiplexor in the column decoder output.
- (d) Each data output is 2 bytes

1.4. To implement a 1-bit memory, which of the following will have the least number of transistors?

- (a) 1-bit D Flip-Flop
- (b) 1-bit SRAM
- (c) 1-bit D Latch
- (d) 1-bit DRAM

1.5. Little Endian is used to store a 4-byte data “0x12345678” in memory address 0x00000004, which of the following is **correct**? (assume the data are byte-addressable)

- (a) Memory location 0x00000005 = 0x56
- (b) Memory location 0x00000006 = 0x43
- (c) Memory location 0x00000007 = 0x5
- (d) Memory location 0x00000004 = 0x12

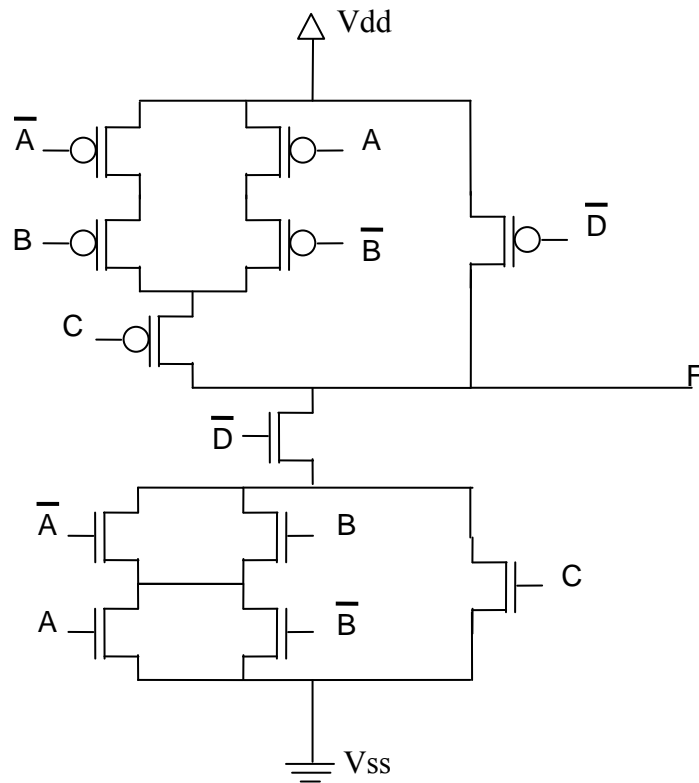
I also gave full credit if you chose (d), as this is the way global data defined in MIPS assembly code considers the leftmost byte as the lowest byte.

2. (10%) **CMOS circuits.** Draw the CMOS circuits (P-type and N-type) of the following Boolean expression. Please label the inputs as well as the output F in the circuits.

$$F = \overline{\overline{A \oplus B} + C} + D$$

$$F = (A \oplus B) \cdot \overline{C} + D$$

$$F = (A\overline{B} + \overline{A}B) \cdot \overline{C} + D$$



3. (15%) **Combinational Logic Design.**

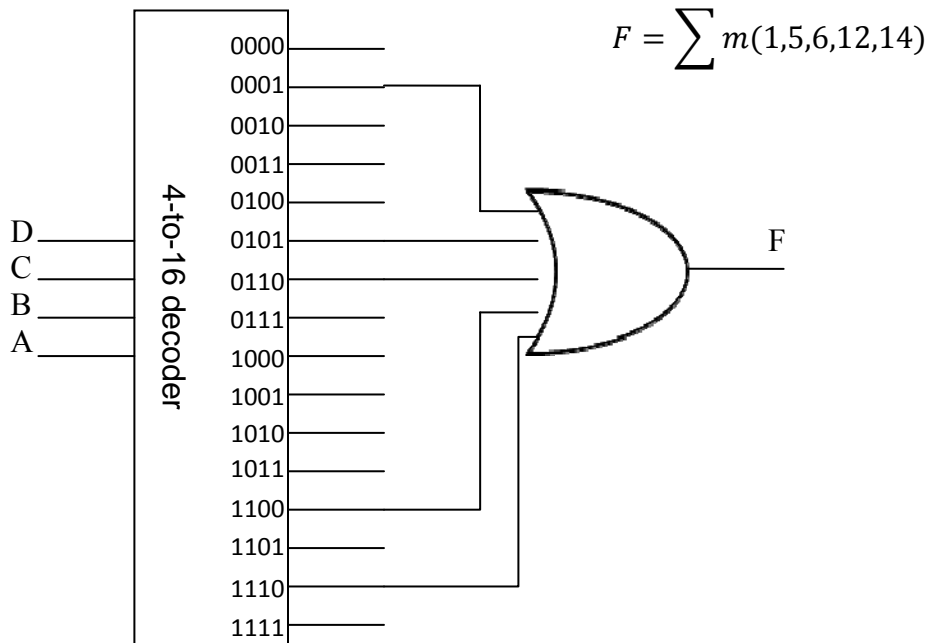
3.1. (5%) Minimize the following canonical POS form using a Karnaugh Map and write down the minimized Boolean equation.

$$F(A, B, C, D) = \prod M(0, 2, 3, 7) + d(4, 8, 9, 10, 11, 13, 15)$$

		CD			
		00	01	11	10
AB	00	0	1	0	0
	01	X	1	0	1
	11	1	X	X	1
	10	X	X	X	X

$$F = B\bar{D} + \bar{C}D$$

3.2. (5%) Use a **4-to-16 Decoder and basic logic gates** to implement the same function above. (You simply draw the decoder as a block, no need to show the internal logic.)



3.3. (5%) Use an **8-to-1 Multiplexor** to implement the same function above. (You simply draw the MUX as a block, no need to show the internal logic.

$$F(A, B, C, D) = B\bar{D} + \bar{C}D$$

I will use A B C as the selection bits of the mux, therefore,

$$F(0,0,0,D) = D$$

$$F(0,0,1,D) = 0$$

$$F(0,1,0,D) = 1$$

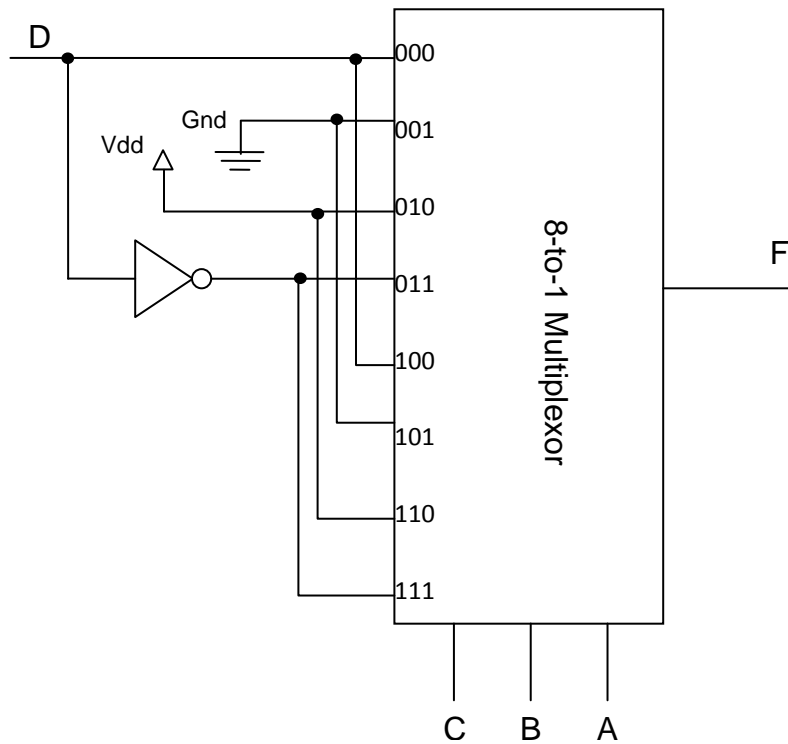
$$F(0,1,1,D) = \bar{D}$$

$$F(1,0,0,D) = D$$

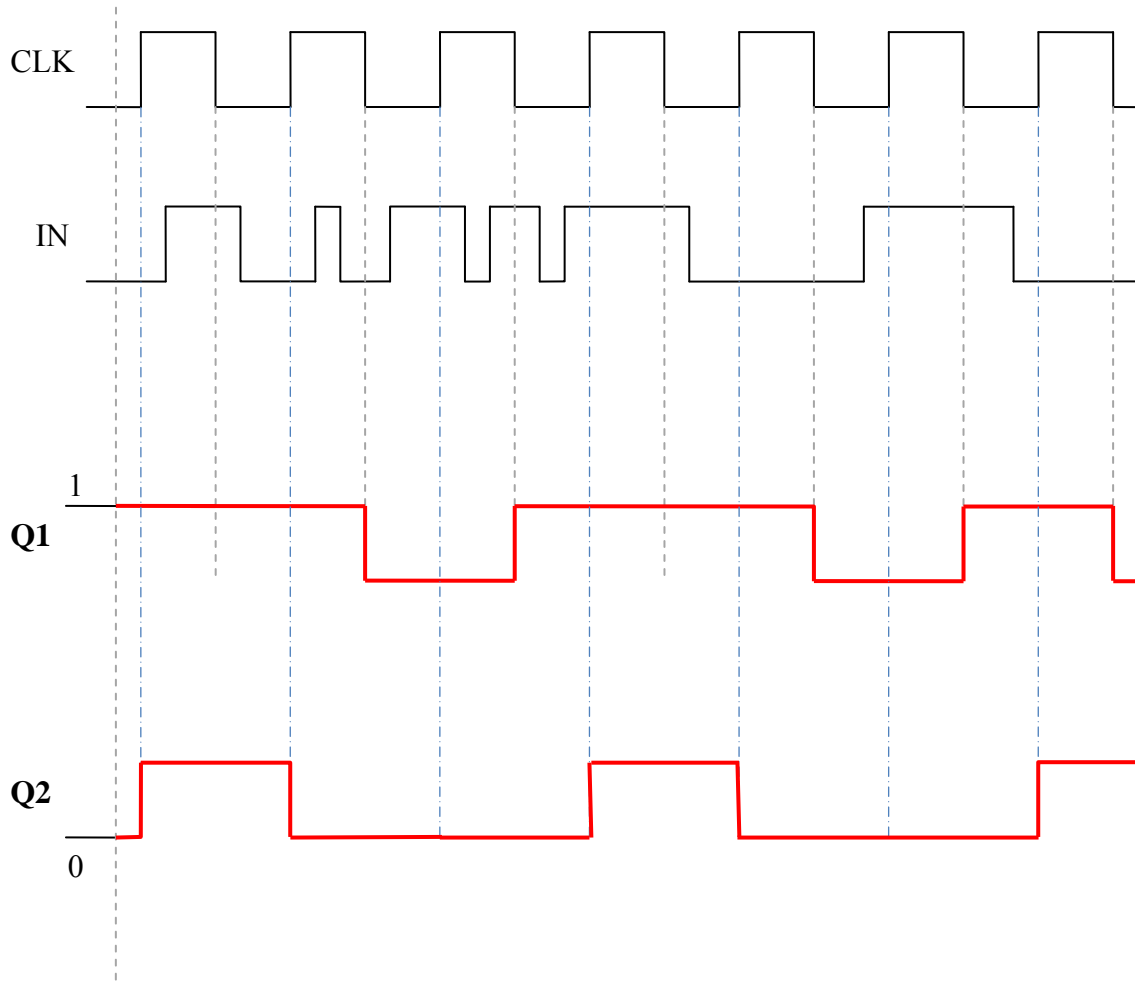
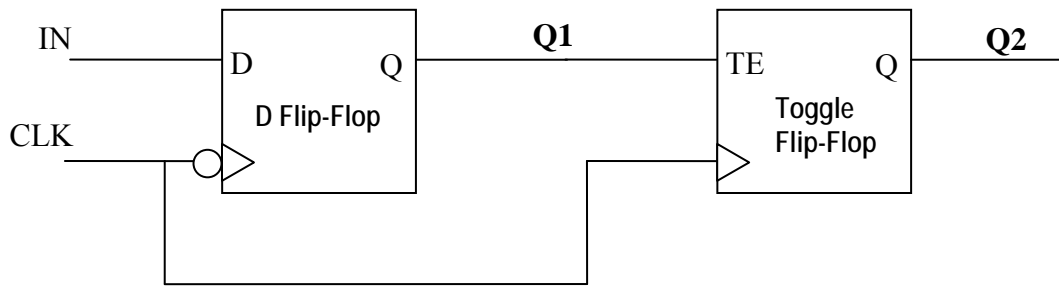
$$F(1,0,1,D) = 0$$

$$F(1,1,0,D) = 1$$

$$F(1,1,1,D) = \bar{D}$$

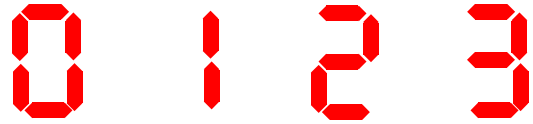
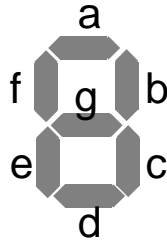


4. (10%) **Flip-Flops**. Finish the timing diagrams for Q1 and Q2. Assume the initial state of Q1 is 1 and that of Q2 is 0.

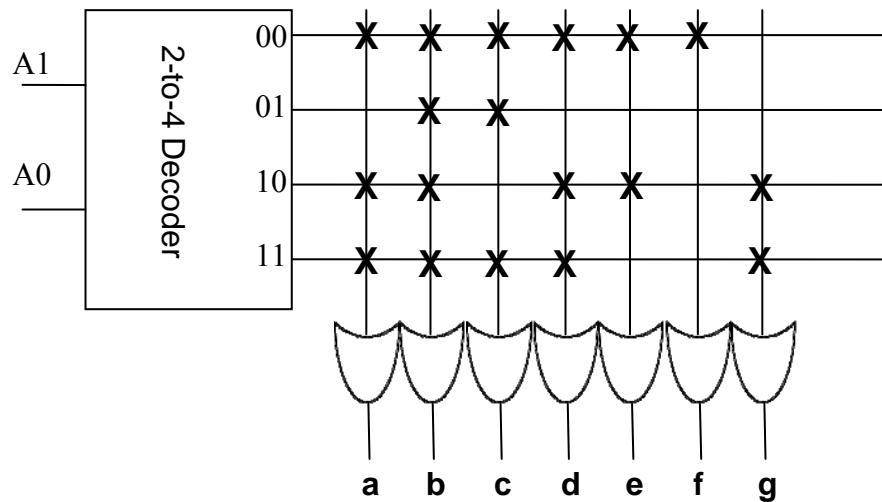


5. (15%) **Read-Only Memory.**

5.1. (7%) Design a 7-segment display using a $2^2 \times 7$ ROM. (2 Inputs: A1 and A0, 7 Outputs: a to g.) You only need to display 0, 1, 2, 3, therefore, only a 2-to-4 decoder is needed on the input side. You don't need to draw the details of the decoder, simply draw it as a block. But you need to show your programmed OR plane of the ROM. No need to optimize OR plane in this part. Please label your inputs and outputs properly.

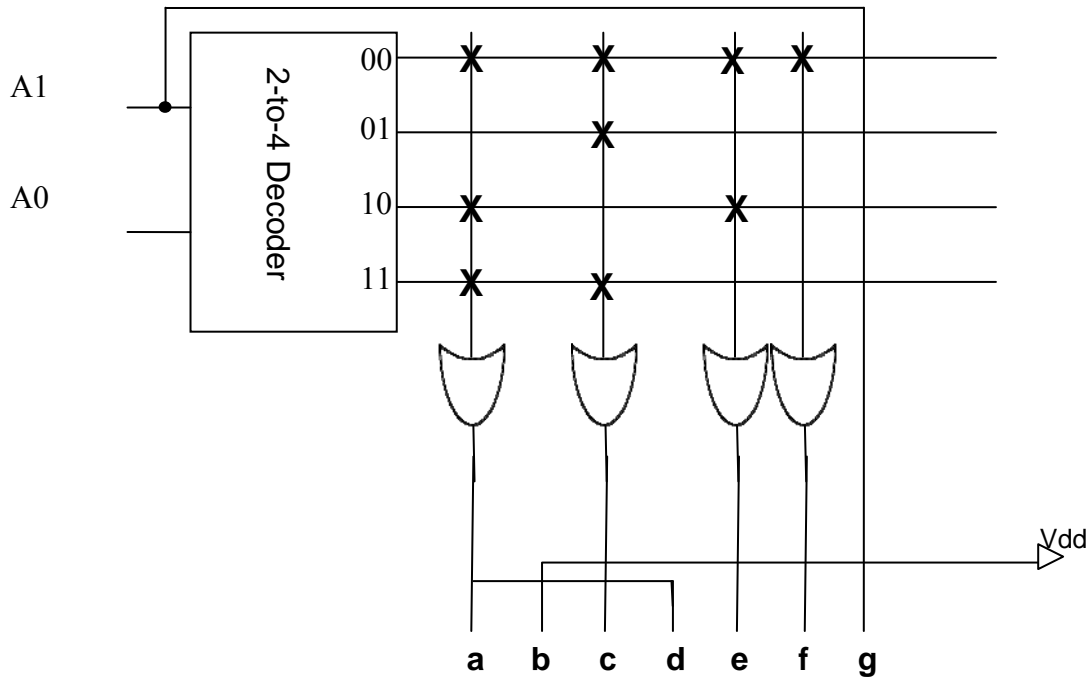


A1 = 0	A1 = 0	A1 = 1	A1 = 1
A0 = 0	A0 = 1	A0 = 0	A0 = 1

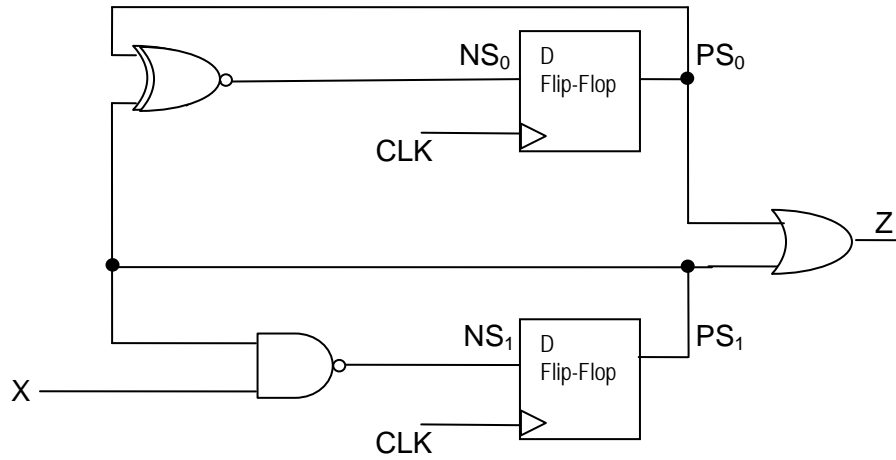


5.2. (8%) The display ROM you designed above can be further optimized and reduced to use a smaller ROM with only 4 OR gates (e.g., $2^2 \times 4$ ROM). Please show the necessary wirings for the output to achieve this optimized design.

One version of the solutions is shown below. (Other viable solutions: You can connect "f" directly to the 00 output without an OR gate or you can connect "e" to an inverted A0 without an OR gate. The bottom line is that if you can get rid of 3 OR gates, you will receive full credit for this problem.)



6. (15%) **Finite State Machine.** Giving the following implementation of a finite state machine, please answer the following questions by reverse engineering it.



6.1. (5%) Derive the Boolean expressions for the state output variables NS_0 , NS_1 and the output Z .

$$NS_0 = \overline{PS_0 \oplus PS_1}$$

$$NS_1 = \overline{X \cdot PS_1}$$

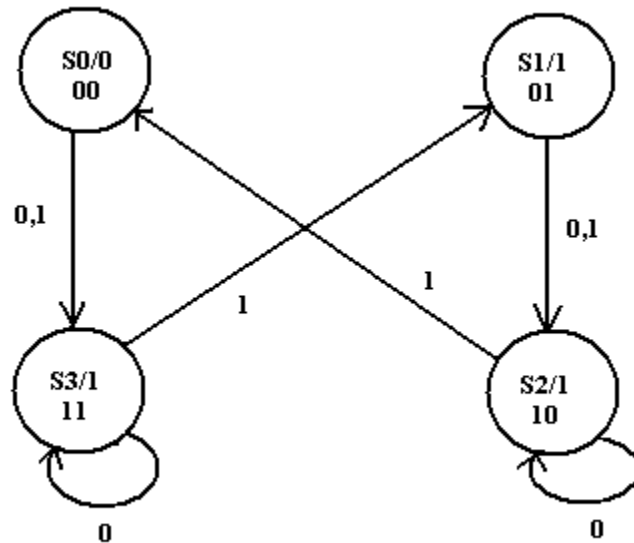
$$Z = PS_0 + PS_1$$

6.2. (5%) Drive the State Table for the output variables.

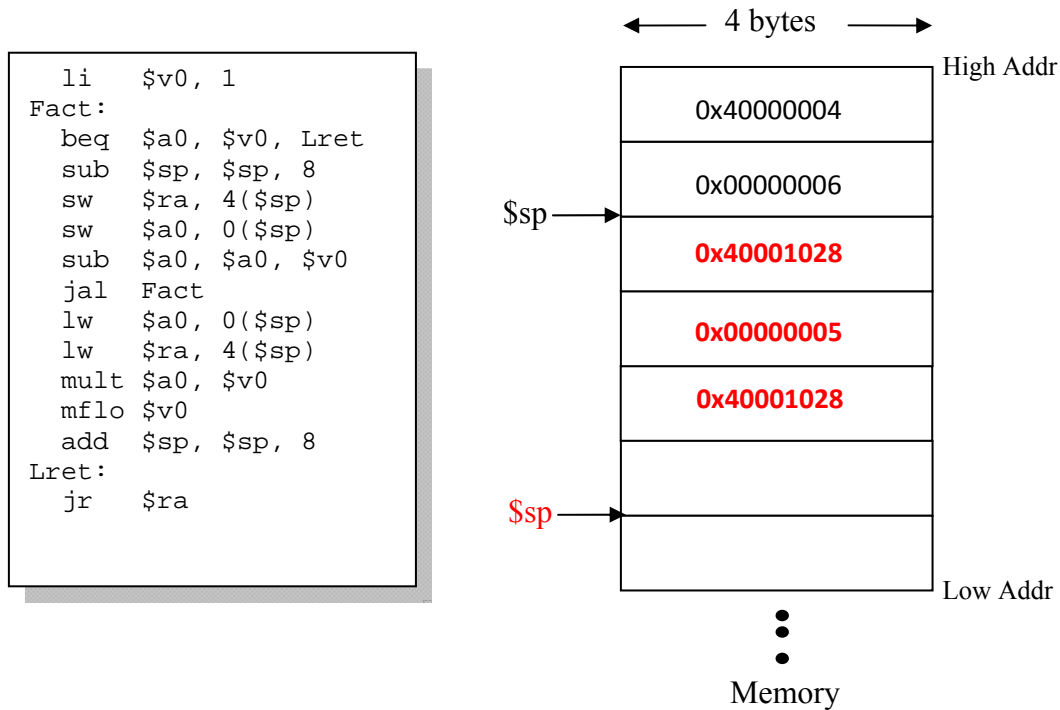
PS_1	PS_0	X	NS_1	NS_0	Z
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	1	0	1
1	0	1	0	0	1
1	1	0	1	1	1
1	1	1	0	1	1

6.3. (5%) Draw the State Diagram.

- S0 : $PS_1 PS_0 = 00$
- S1 : $PS_1 PS_0 = 01$
- S2 : $PS_1 PS_0 = 10$
- S3 : $PS_1 PS_0 = 11$



7. (20%) **MIPS ISA.** Given a MIPS program on the left. Assume the current program counter $\$pc = 0x40001024$ which is the address of the **jal** instruction and the stack pointer $\$sp = 0xFFFF0080$. Also given is the corresponding memory snapshot on the right. Let's execute another **10** instructions (the jal included), please (1) fill the updated memory content on the right; (2) write down the values of the following registers: **$\$a0$, $\$v0$, $\$ra$, $\$sp$, and $\$pc$** (Each instruction is 4-byte wide. You may not need to fill all the blank entries in the memory. Please show minimum work so it does not look like you are guessing the answers.)



At the beginning, we have to figure out the value of $\$a0$ which is the only unknown among all registers. Since the last **sw** instruction (2 instructions before the current **jal** instruction) had stored $\$a0$ at memory location $0(\$sp)$, if we examine the stack, we see **0x6** is stored in this location. Also, the last **sub** instruction decremented $\$a$ (note that $\$v0=1$ set by the first **li** instruction), thus $\$a0$ is currently **0x5**. We will execute 10 instructions now.

1. $\$pc=0x40001024$ jal → jump to Fact and set $\$ra$ to $\$pc+4 = 0x40001028$
2. $\$pc=0x40001010$ beq → condition failed
3. $\$pc=0x40001014$ sub → $\$sp = 0xFFFF0078$
4. $\$pc=0x40001018$ sw → store **0x40001028** on stack ($\$ra$)
5. $\$pc=0x4000101c$ sw → store **0x00000005** on stack ($\$a0$)
6. $\$pc=0x40001020$ sub → $\$a0 = 0x4$
7. $\$pc=0x40001024$ jal → jump to Fact and set $\$ra$ to $\$pc+4= 0x40001028$
8. $\$pc=0x40001010$ beq → condition failed
9. $\$pc=0x40001014$ sub → $\$sp = 0xFFFF0070$
10. $\$pc=0x40001018$ sw → store **0x40001028** on stack ($\$ra$)

$\$a0=0x4$, $\$v0=1$, $\$ra=0x40001028$, $\$sp=0xFFFF0070$, $\$pc=0x4000101c$