EEL3701 – Dr. Gugel **Last Name**_____ **First**_____
Spring 2017 Exam II

Open book/open notes, 90-minutes. Calculators permitted. Do not write on the back side of any of the pages.

Page 1) 18 points Alan
Page 2) 30 points #3 Aaron / Veronica #5
Page 3) 14 points #6 Casey / #7 Kevin
Page 4) 20 points Kyle
Page 5) 18 points #13 Lucas / #14 Daniel
TOTAL Sum → Casey out of 100

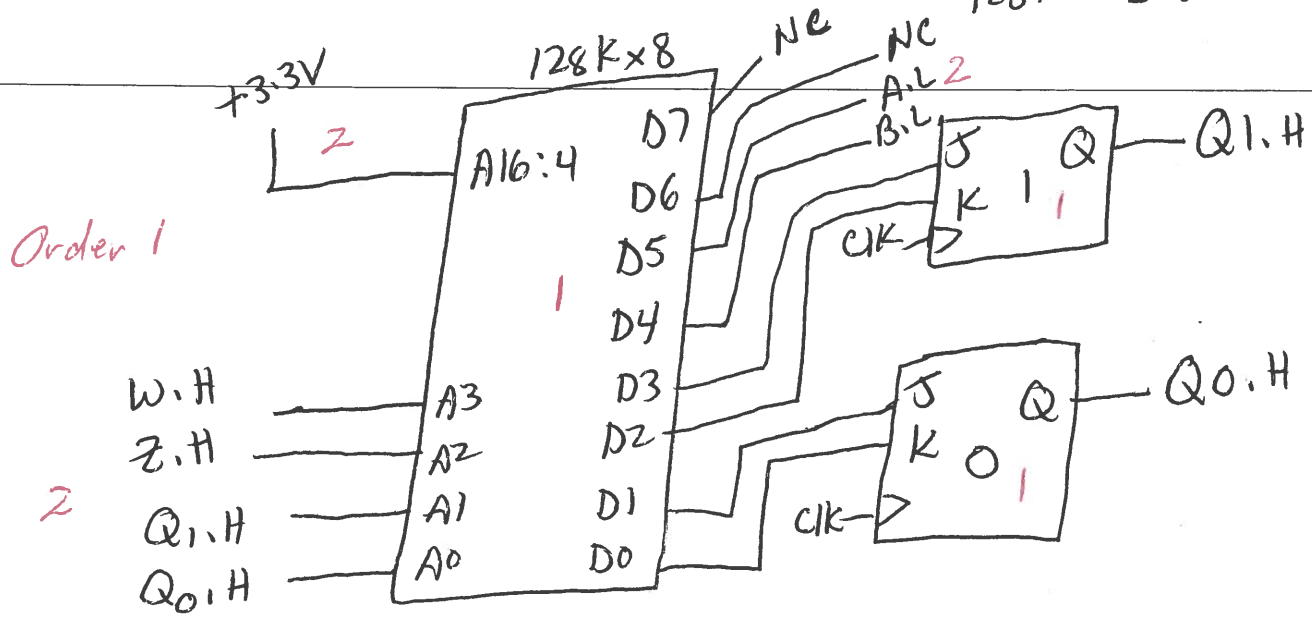
Grade Review Information: *1. Deadline of request for grade review is the day the exam is returned.* 2. Do not make any changes to problems in the test as this will be considered cheating. 3. Write only in this blocked area for a re-grade request. 4. Simply write the problem number that you would like re-graded. ***3 Maximum.***

Alan, Veronica, Kyle, Aaron, Lucas, Casey
Wyatt, Kevin, Cody, Daniel, Caleb
↳ Enter into Canvas ↳ No Job ↳ No job

See the **ASM Flow Chart** in **Appendix A** to answer question 1-5.

1. Assuming that we will implement the ASM Flow Chart in Appendix A with a 128K x 8 EEPROM and JK Flip Flops, draw the complete functional block diagram for the system below. **Label all signals** and assume that all **unused address lines will be tied to 3.3V**. Note: Place the system **state variables** (Qn:Q0) and **JK inputs** in the **least significant bits** of the **address or data bus** where they are used. (10 pt.)

128K = $2^7 \cdot 2^{10}$



2. How many memory locations will need to be programmed in this design and what is the address range of these locations? Also, identify the designated part number of this device. (8 pt.)

2 Number of locations programmed in memory 16 (Decimal number) 2

4 Range of memory to be programmed 1FFF0 - 1FFFF (Hex) 4

2 What is the part number of this device? 28 1024 2

28256 ⇒ 32k 28512 ⇒ 64K

Page Score = 18

3. Draw the complete Next State Table (inputs, state variables, outputs, etc.) below for only the rows that correspond to when the present state is **State 0**. Recall: **JK Flip-Flops** & **EEPROM** implementation. (12 pt.)

-1 if Q1 and Q2 out of order (least)

W Z	$Q_1 Q_0$	Q_1^+ Q_0^+	A B	J1 K1	J0 K0
0 0	00	0 1	1 0	0X	1X
0 1	00	0 1	1 0	0X	1X
1 0	00	0 1	1 1	0X	1X
1 1	00	0 0	0 1	0X	0X

OK 0X 1 pt 1pt 1pt 1pt

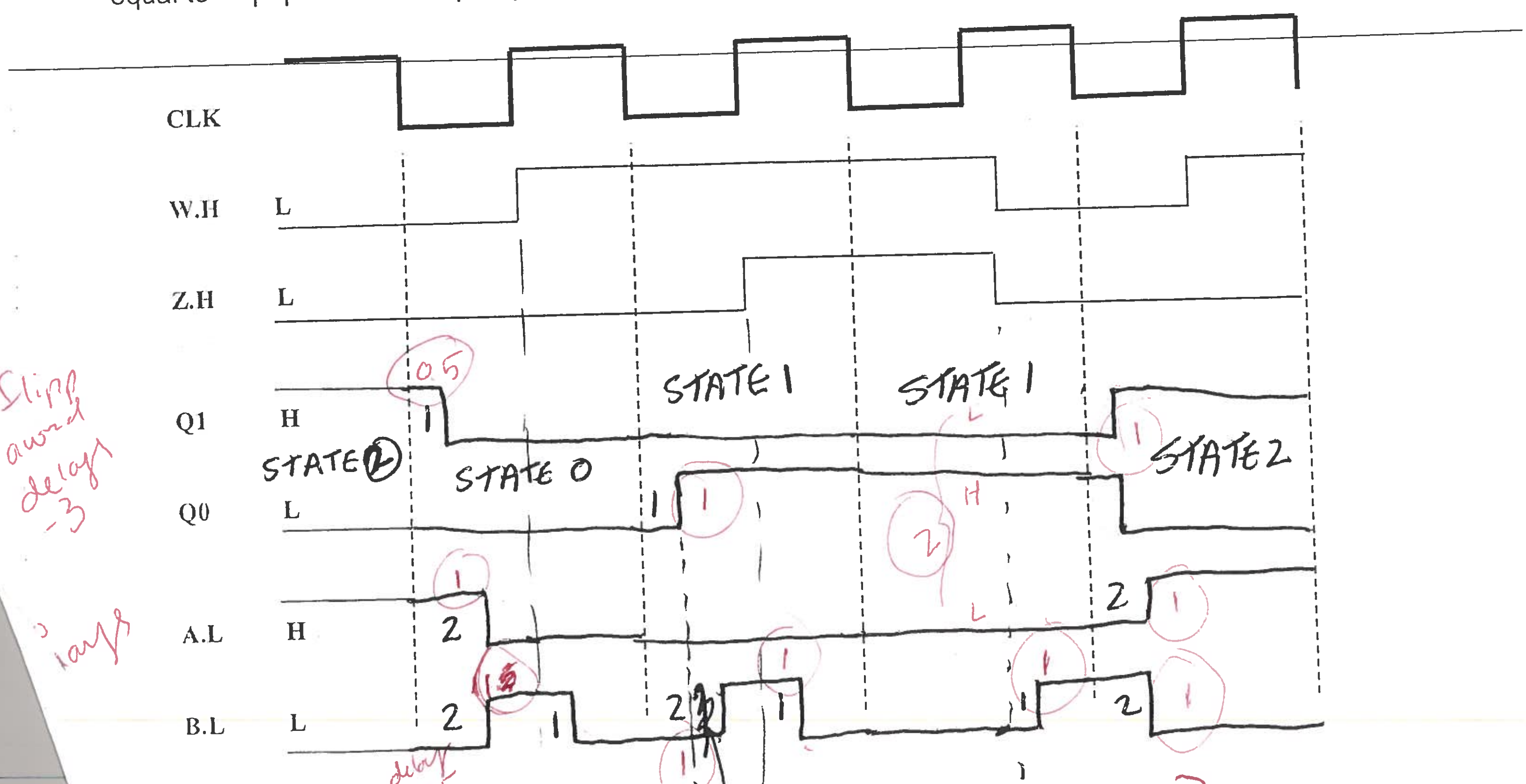
JK	Q^+
00	Q
01	0
10	1
11	$\bar{Q}$

4. Show the EEPROM memory contents (**Address** and **Data** in **Hex**) that needs to be programmed for the locations corresponding to **State 0**. Program **Don't Cares** (X) in Data as **Zeros**. Note: **Signal definitions** are **W.H**, **Z.H**, **A.L**, **B.L**. (6 pt.)

Low; 1pt for data; 0.5 per address

Address (Hex)	Data (Hex)		Hex	if J1,K1,J0,K0,A
1FFF0	L H	LL HL =	12	09
1FFF4	L H	LL HL =	12	09
1FFF8	L L	LL HL =	02	08
1FFFC	H L	LL LL =	20	02

5. Assuming again that we are referencing the ASM Flow Chart in Appendix A. **Fill in** the **Voltage Timing Diagram** below. Note: **Signal definitions** are **W.H, Z.H, A.L, B.L** and **C.L**. Assume all devices have a delay equal to =>| |<= and the flip-flops are falling edge triggered. **Q1:0 = Present State** (12 pt.)



Slipped around delays -3

delays

-delay -2

3 3 3 3

- delay not shown -2

6. In problem #5, if the logic and flip-flops are all implemented in your CPLD what will be the delays in nano-seconds for the state variables Q1:0, Unconditional Outputs and Conditional Outputs? (3 pt.)

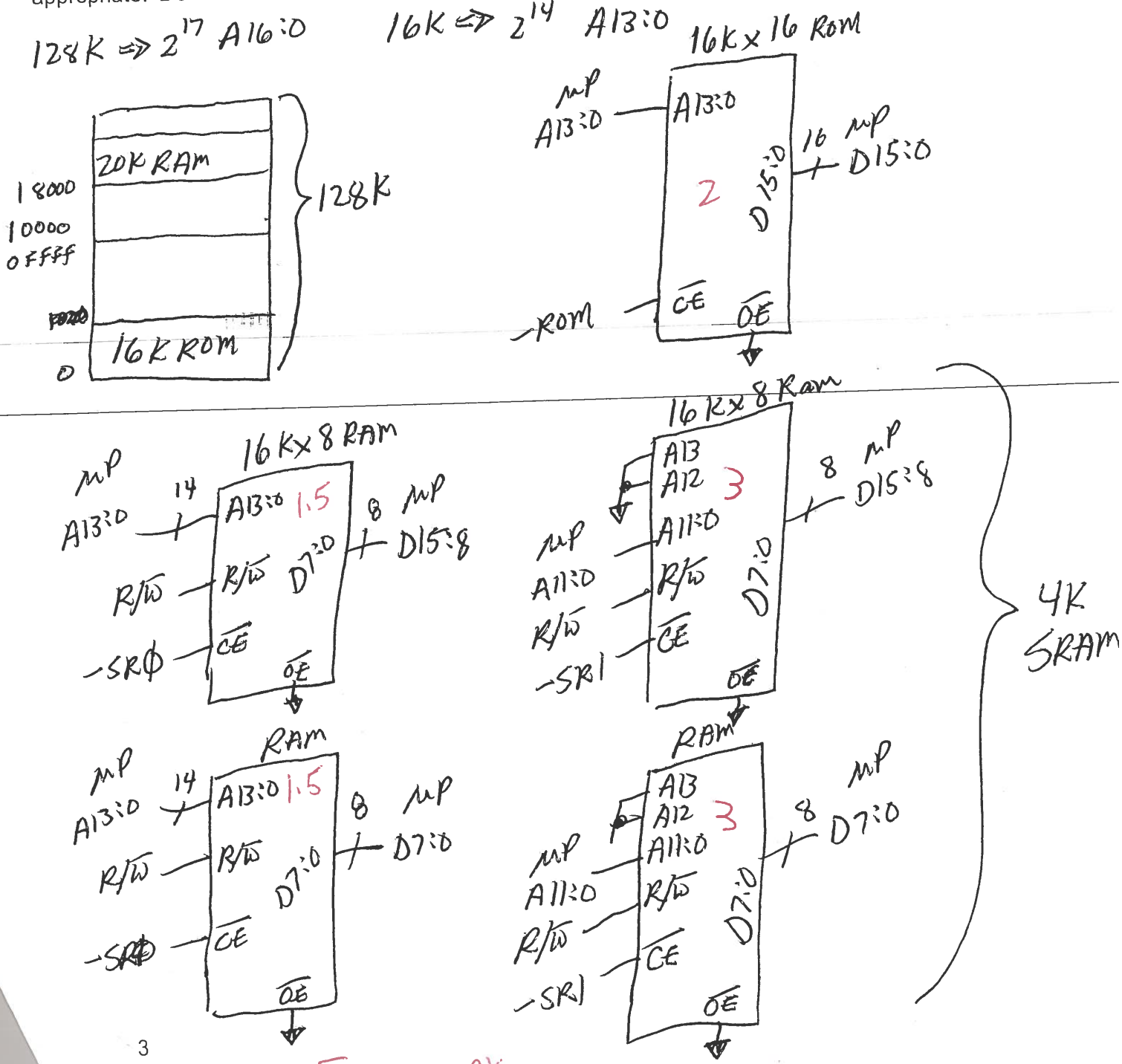
State Variables (Q1:0) Delay 10 nsec

Unconditional Output Delay 20 nsec

Conditional Output Delay 10 - 20 nsec 10 assumes Q1Q0 stable

7. You are given a microprocessor with a **17 bit address bus** and a **16 bit data bus**. The control bus consists of a **RW.H (+Read/-Write)** signal and a low true data strobe (**DS.L**). You are given any number of **16K x 16** ROMs and **16K x 8** SRAMs. Place **20K of RAM** starting at address **18000 Hex** in the system memory map. Place **16K of ROM** in the system memory map with the understanding that the first instruction will be fetched from address **00000 Hex**.

Show the required Rom & Ram memory devices below. Label all signals and use bus nomenclature where appropriate. ***Do not show the decode equations below.*** (11 pt.)



OE missing OK

8. Show the required **decode logic equation** for the **ROM device**. Note: The decode signal name should match the one that you specified/drew for your ROM memory above. (4 pt.)

$-Rom = \overline{A16} \cdot \overline{A15} \cdot \overline{A14} \cdot R/\overline{W} \cdot DS$ (2, 1)

0|00xx|xxxx|xxxx|xxxx — 16K

↑ R/W also okay

9. Show the required **decode logic equations** for the **RAM devices**. Your decode signal names should match those that you drew for the SRAM memory in problem #7. (6 pt.)

$-SR0 = A16 \cdot A15 \cdot \overline{A14} \cdot DS$ (2, 1)

1|10xx|xxxx|xxxx|xxxx — 16K

$-SR1 = A16 \cdot A15 \cdot A14 \cdot \overline{A13} \cdot \overline{A12} \cdot DS$ (2, 1)

1|11x00|xxxx|xxxx|xxxx — 4K

10. What is the address range for **ROM** in the memory map? 0-3FFF (1, 2) Hex (3 pt.)

11. What is the address range for the **RAM** in the memory map? 18000 - 1CFFF (1, 2) Hex (3 pt.)

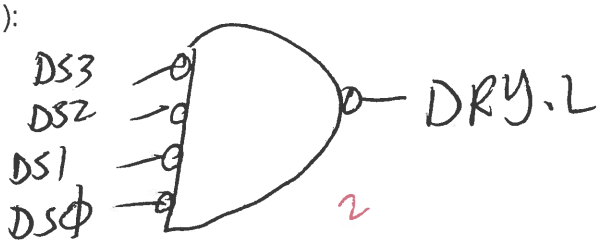
12. When we ***remove and program the ROM*** what is the address should be programmed with the **first instruction** and where should the **last program instruction** be placed in the ROM? (2 pt.)

1st instruction address 0 (1) Hex Last instruction address 3FFF (1) Hex

---

14. A student is designing a controller for a ***clothes dryer*** that has **(4) low true drying sensors DS3:0**. When a sensor senses moisture it outputs a H and when no moisture is detected a L is output. First, design a circuit to detect when ***all sensors*** indicate a 'dry condition'. This new signal is called **DRY.L.**

Show the **circuit** to create **DRY.L** (2 pt.):



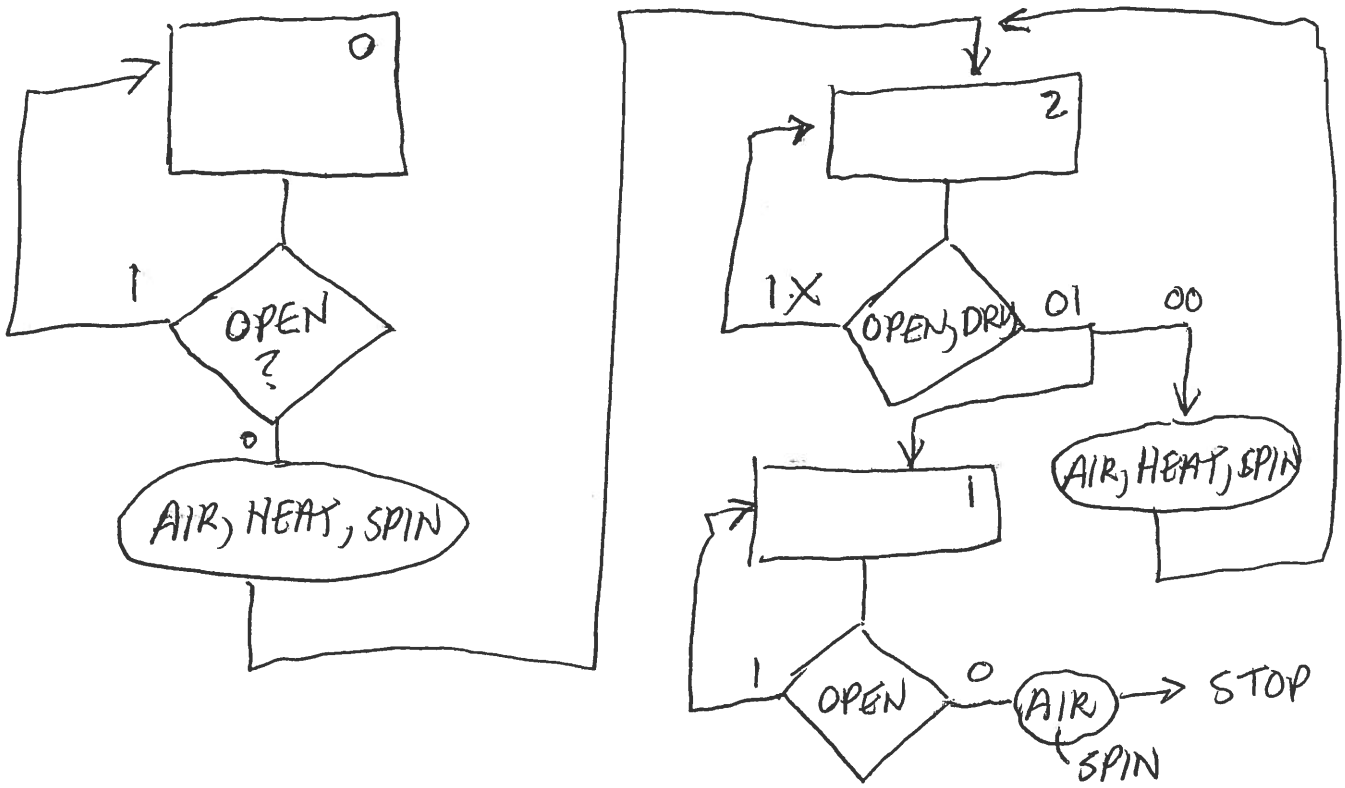
Next, use **DRY** and another low true input Door Open (**OPEN**) to control the high true outputs: **AIR**, **HEAT** and **SPIN**. **AIR** turns on an air pump to inject fresh air, **HEAT** turns on a heater to heat the incoming air and **SPIN** turns on a motor that spins the main dryer drum. Inputs are: **DRY**, **OPEN** Outputs are: **AIR**, **HEAT** and **SPIN**

Assuming the clock has a period = 5 minutes, the controller should have the following specifications:

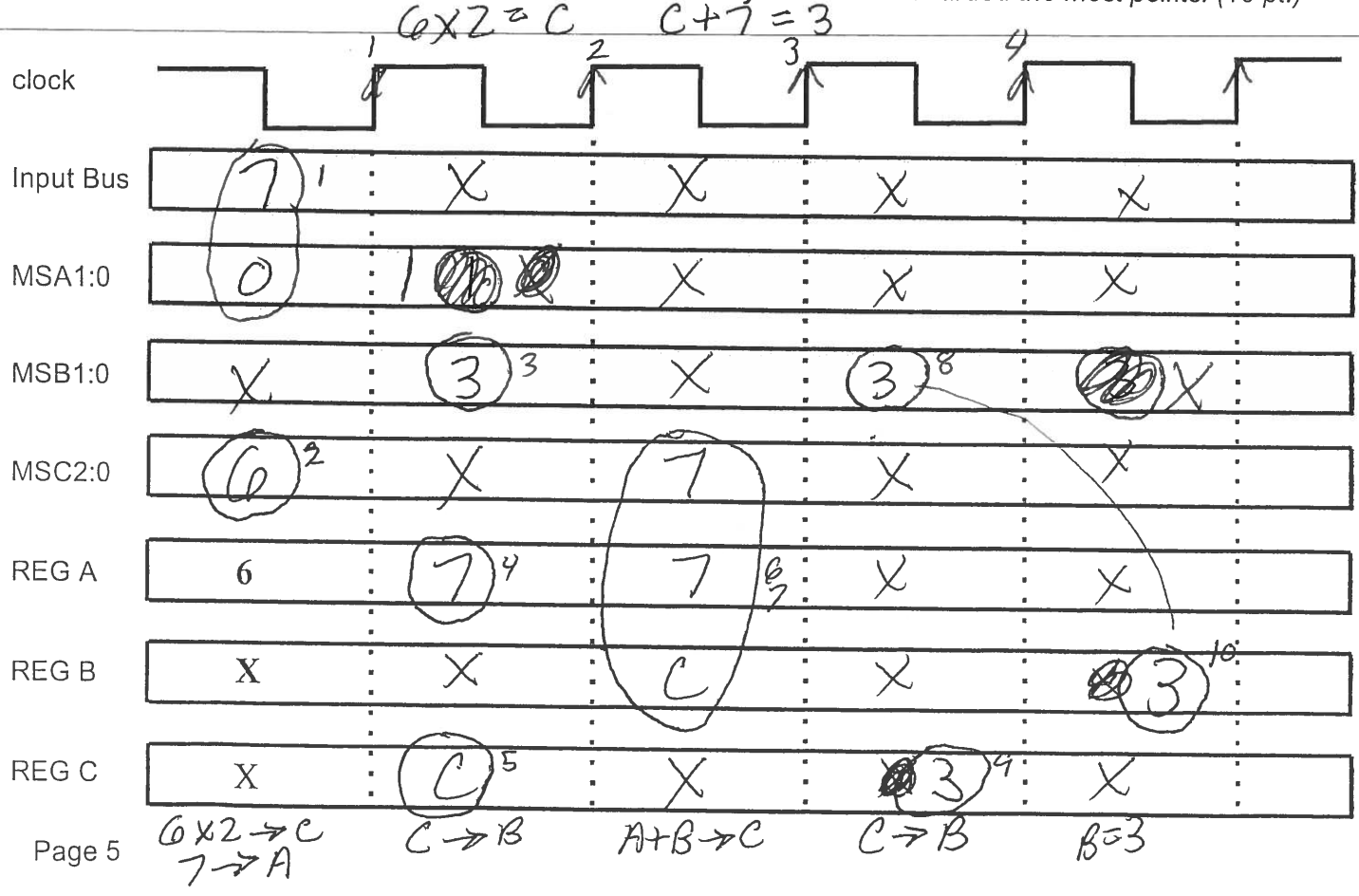
1. Inject and heat air for 5 minutes while spinning the clothes.
2. Next, continue drying/spinning the clothes until **all** (4) dry sensors show the clothes are dry.
3. Once the clothes are dry, cool for 5 minutes where the clothes spun and cooled only with fresh air.
4. ***At any point during the drying cycle, if the door is opened (OPEN = T), turn off the heat and stop spinning the dryer drum immediately for safety purposes.*** Also, extend the dry time for as long as the door is open.

2

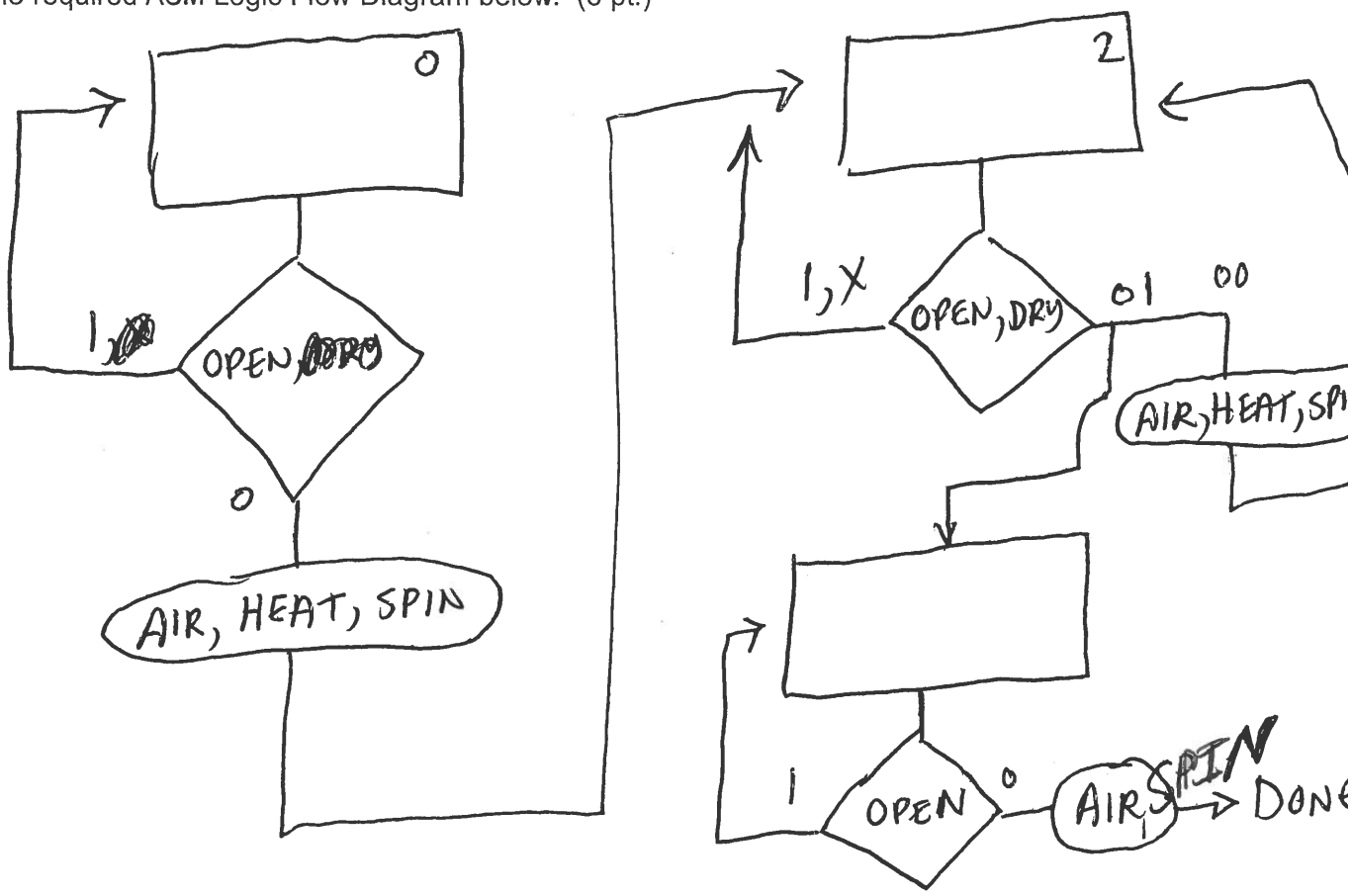
13. Show the required ASM Logic Flow Diagram below. (8 pt.)



14. Given the ALU in Appendix B, Fill in the signals below to compute (6 x 2) + 7 with the final answer in **Reg. B**. *Note: Use **X for unknown Register contents** and **X for don't care Inputs. Zero propagation delays**, all answers in **Hex** and the solution with the **least number of cycles** will be awarded the most points. (10 pt.)*

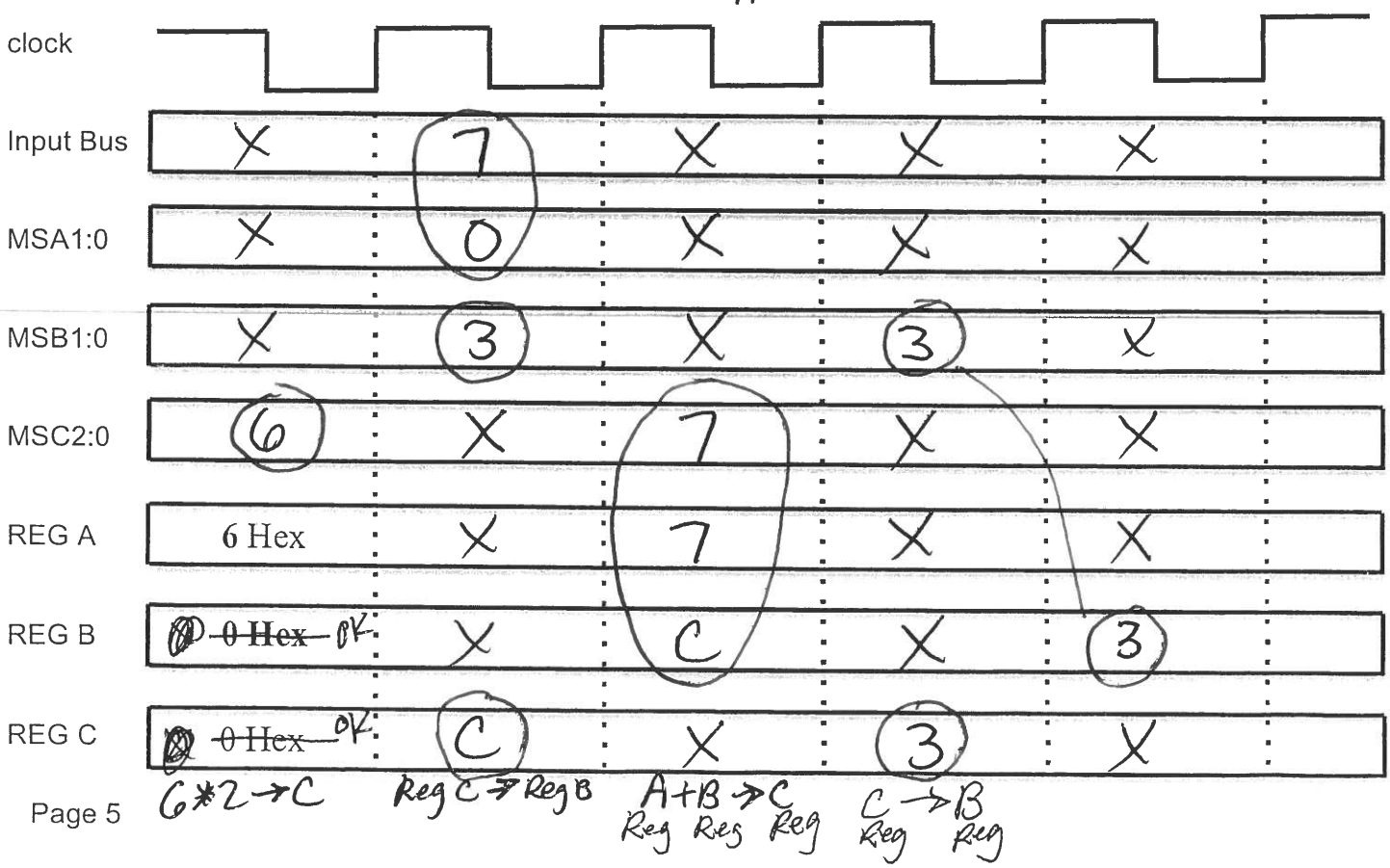


13. Show the required ASM Logic Flow Diagram below. (8 pt.)

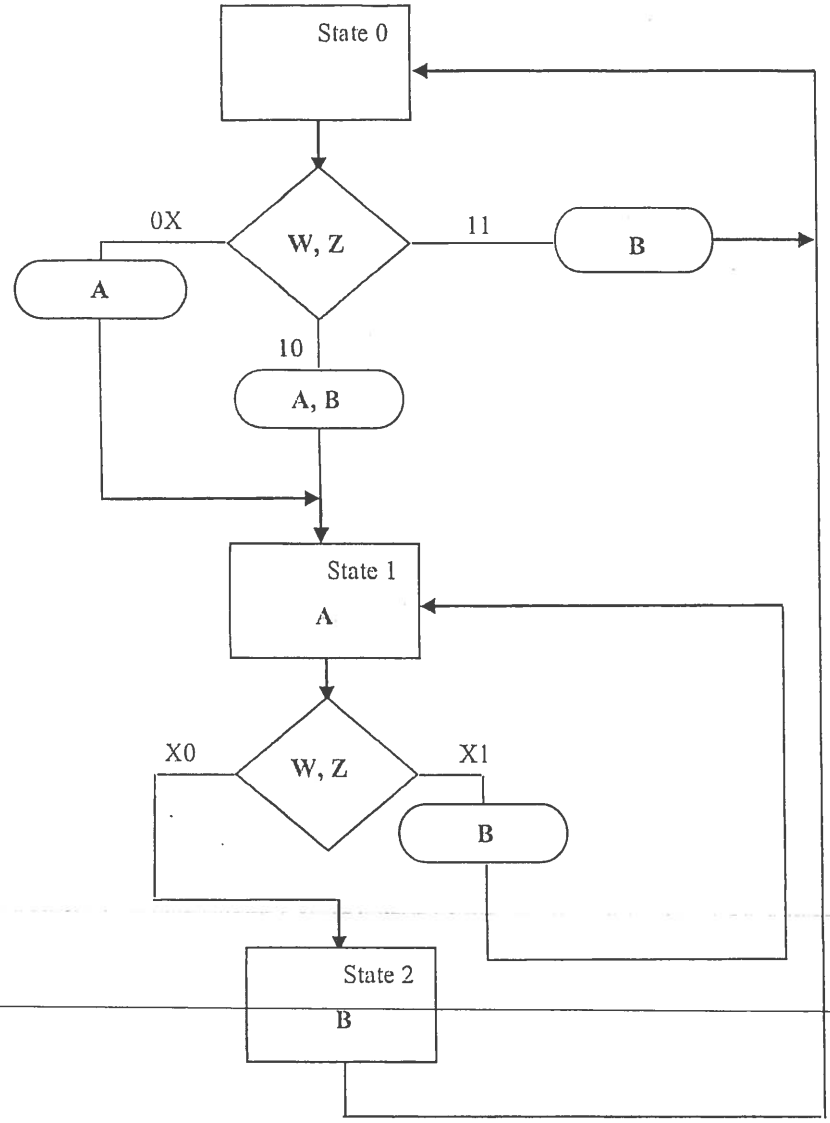


14. Given the ALU in Appendix B, Fill in the signals below for (6*2) + 7 with the final answer in **Register B**.
*Note: Use **X for unknown Register contents** and **X for don't care Inputs. Zero propagation delays**, all answers in **Hex** and the solution with the **least number of cycles** will be awarded the most points. (10 pt.)*

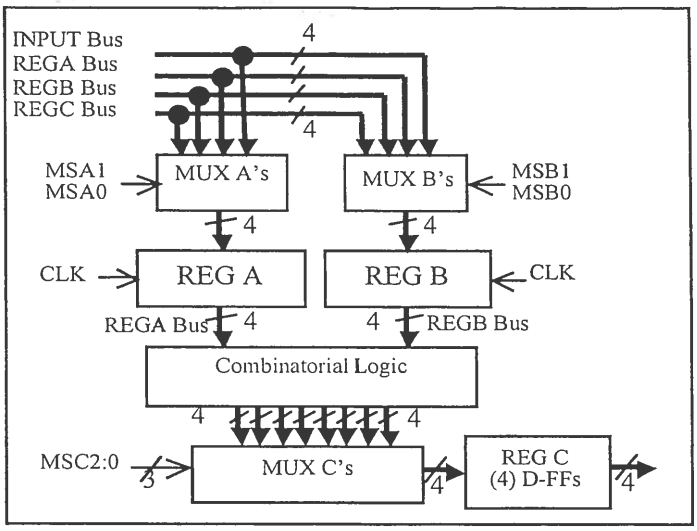
1100 + 0111 = 0011



Appendix A. ASM Diagram (W.H, Z.H, A.L, B.L)



Appendix B. ALU with Registered Output Bus



MSA1/MSB1	MSA0/MSB0	Bus Selected as Input to REGA/B
0	0	INPUT Bus (Input3:0)
0	1	REGA Bus (REGA3:0)
1	0	REGB Bus (REGB3:0)
1	1	REGC Bus (REGC3:0)

MSC2:0 (Most Significant Bit is on the left)

000	=>	complement of REGA, result in REGC
001	=>	REGA AND REGB, result in REGC
010	=>	REGA OR REGB, result in REGC
011	=>	REGA to REGC
100	=>	REGB to REGC
101	=>	shift REGA right one bit, result in REGC
110	=>	shift REGA left one bit, result in REGC
111	=>	REGA ADDED To REGB, result in REGC