EEE 120

<u>Simulation Lab 4 Answer Sheet</u>

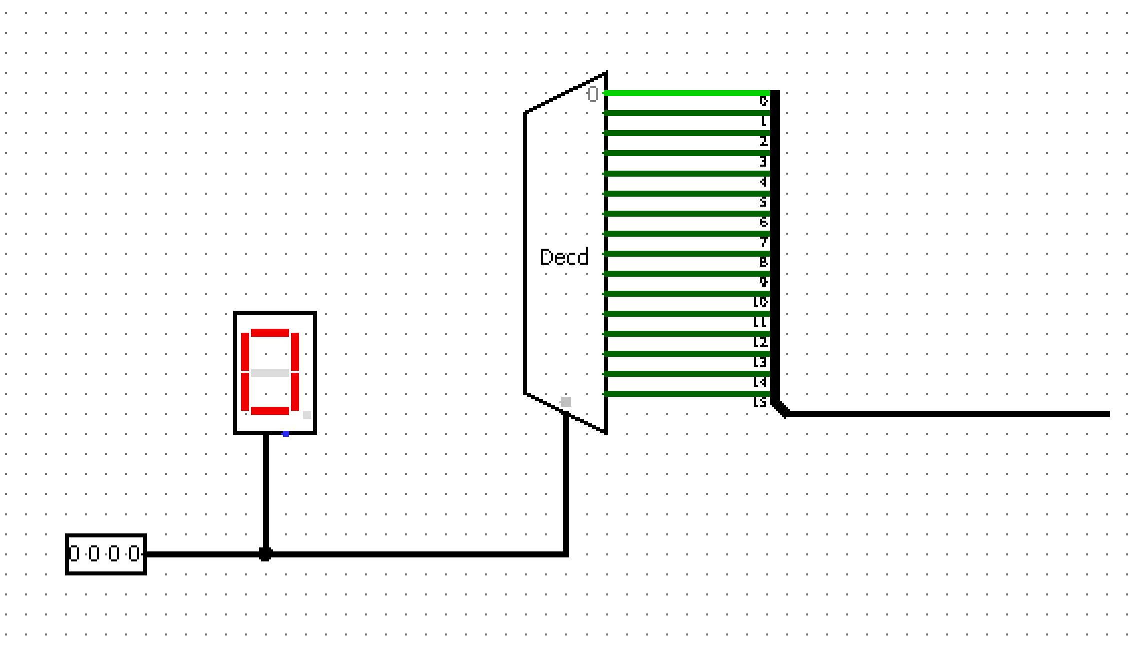
The Microprocessor

Name: Anish Katukam

Date: 3/27/18

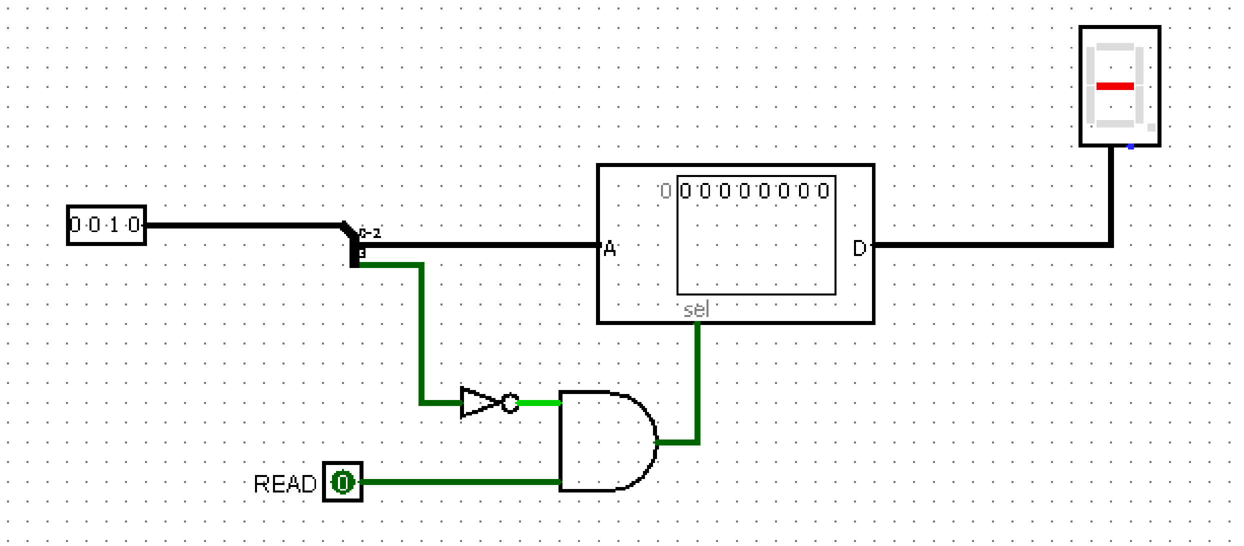
Task 4-1: Build the Addressing Logic

Include a picture of your Logisim addressing logic circuit here:



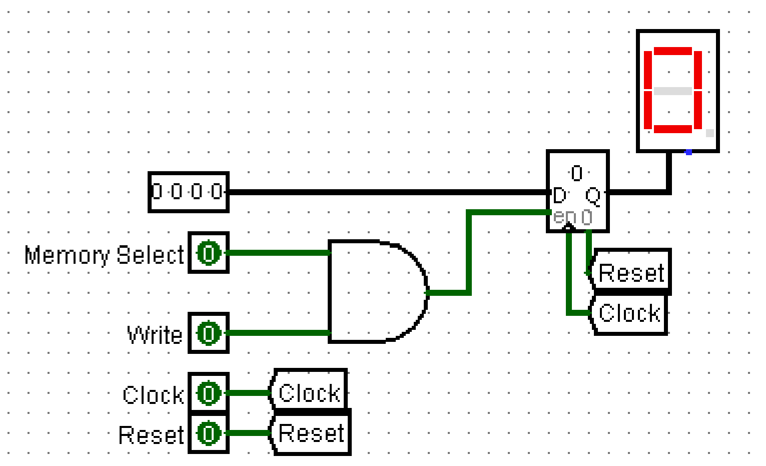
Task 4-2: Build a 4-Bit ROM Memory Cell

Include a picture of your Logisim 4-bit ROM circuit here:



Task 4-3: Build 4-Bit Output Port

Include a picture of your Logisim 4-bit output port circuit here:

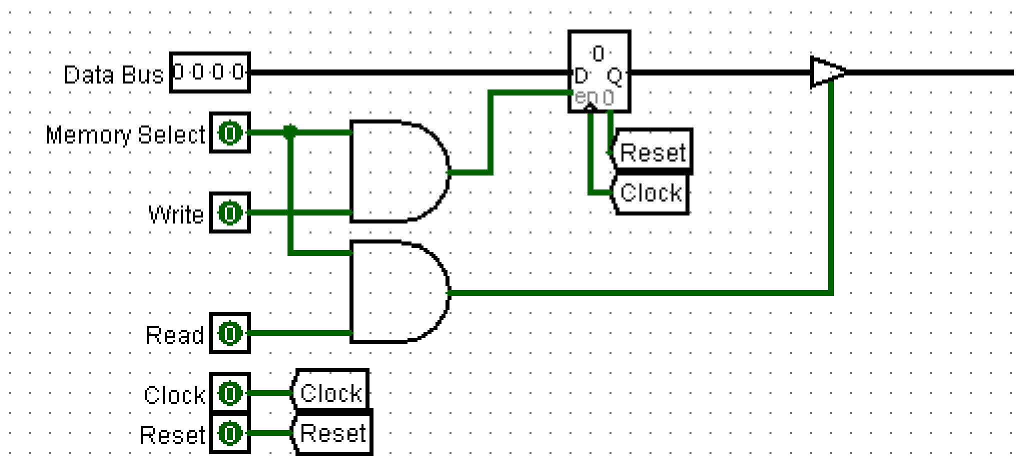


Test your circuit and record the results in **Table** .

Table 1			
Data Bus (4-bit binary)	Write	Memory Select	Q
1101	1	1	d
0001	0	1	d
0001	1	1	1
1100	1	1	C

Task 4-4: Build the 4-Bit RAM Cell

Include a picture of your Logisim 4-bit RAM circuit here:



Test your circuit and record the results in **Table** . Include a picture of your Logisim 4-bit RAM circuit testing set up.

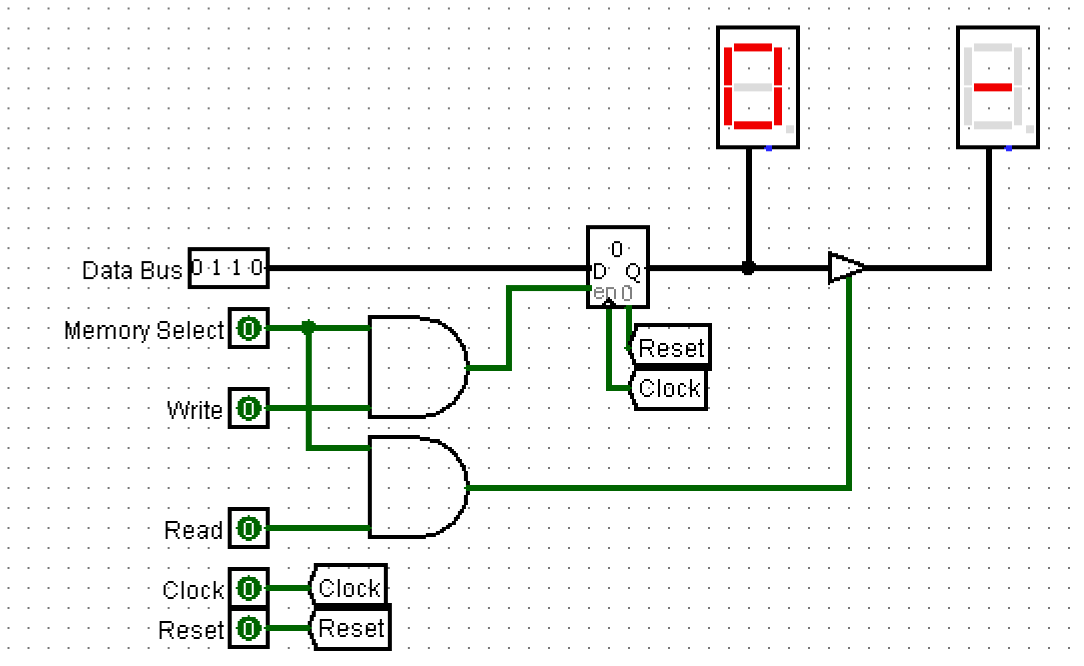
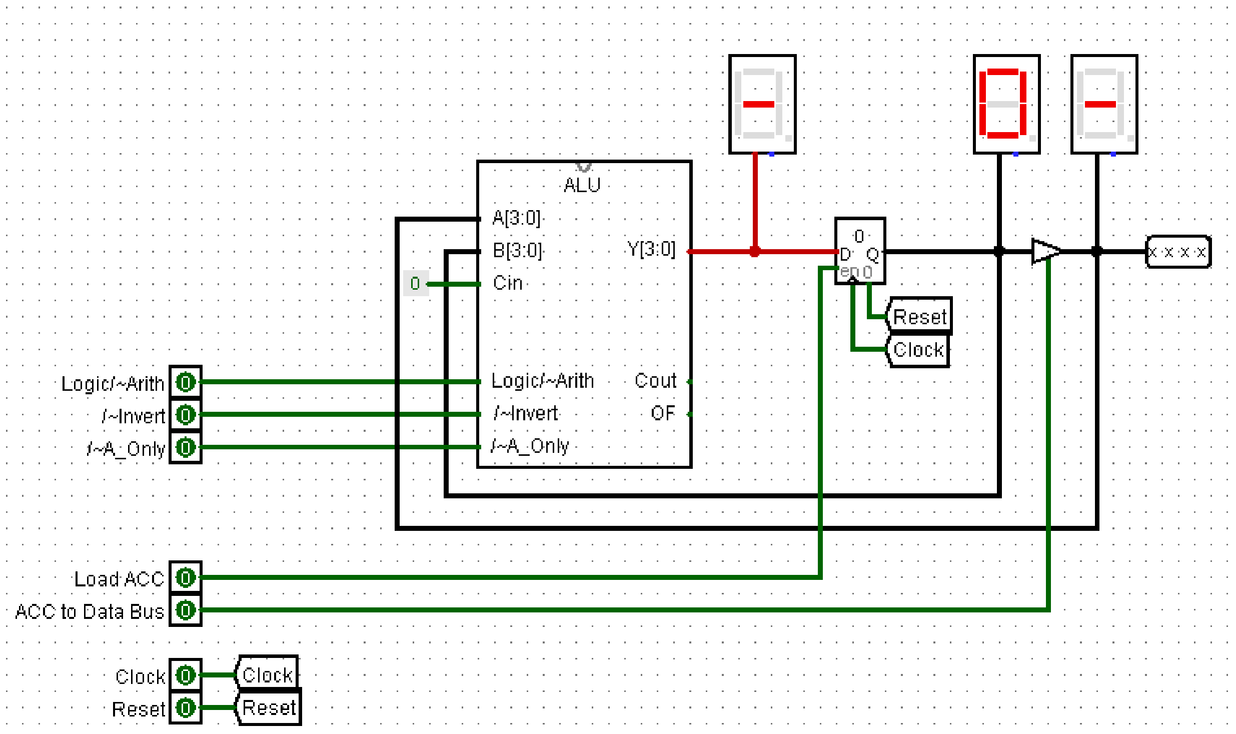


Table 2					
Data Bus (4-bit binary)	Write	Memory Select	Read	Q {between register and buffer}	Data Bus {after buffer}
0111	0	1	0	7	-
0111	0	1	1	7	7
0110	1	1	1	6	6
1111	1	1	0	f	-
1111	0	1	1	f	f
1001	1	1	0	f	-

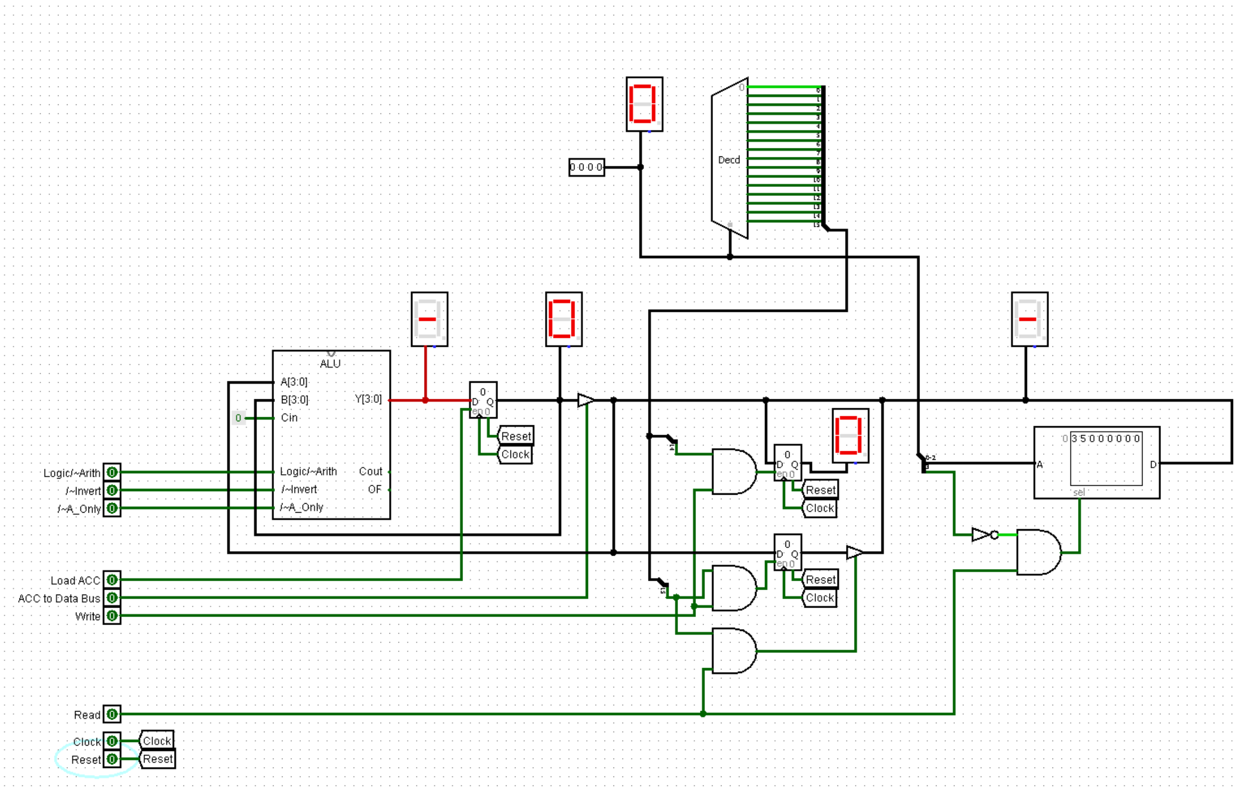
Task 4-5: Build the Brainless Central Processing Unit

Include a picture of your Logisim Brainless Central Processing Unit circuit here:



Task 4-6: Build the Brainless Microprocessor

Include a picture of your Logisim brainless microprocessor circuit here:



Task 4-7: Testing and Controlling the Brainless Microprocessor

Follow steps 1 through 3 outlined in the laboratory manual to test your brainless microprocessor circuit. It might be helpful to review the ALU Function Table from Sim lab 3. Table 3 is an example, for the ADD command, of how to fill out tables to record the values of the control lines during every clock cycle.

Table 3	
Instruction [Add operand to Accumulator (ACC)]	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	Address of operand
Write	0
Read	1
ACC to Data Bus	0
Load ACC	1
/~A_Only	1
/~Invert	1
Logic/~Arith	0

For all of the instructions you performed (i.e. Subtract, Load ACC, etc.) record the values of the control lines during every clock cycle in Table 4 and Table 5.

Table 4	
Instruction [Load ACC with operand]	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	0000
Write	0
Read	1
ACC to Data Bus	0
Load ACC	1
/~A_Only	0
/~Invert	1
Logic/~Arith	0
Instruction [AND operand with ACC]	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	0000
Write	0
Read	1

ACC to Data Bus	0
Load ACC	1
/~A_Only	1
/~Invert	1
Logic/~Arith	1
Instruction [Store ACC to RAM]	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	1111
Write	1
Read	0
ACC to Data Bus	1
Load ACC	0
/~A_Only	0
/~Invert	0
Logic/~Arith	0
Instruction [Subtract operand from ACC]	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	1111
Write	1
Read	0
ACC to Data Bus	0
Load ACC	1
/~A_Only	1
/~Invert	0
Logic/~Arith	0

Describe any other tests that you performed. NOTE: the laboratory manual gives you a minimum set of items to test

Table 5	
Instruction [Not (operand) to ACC] (1's complement)	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	1111
Write	1
Read	0

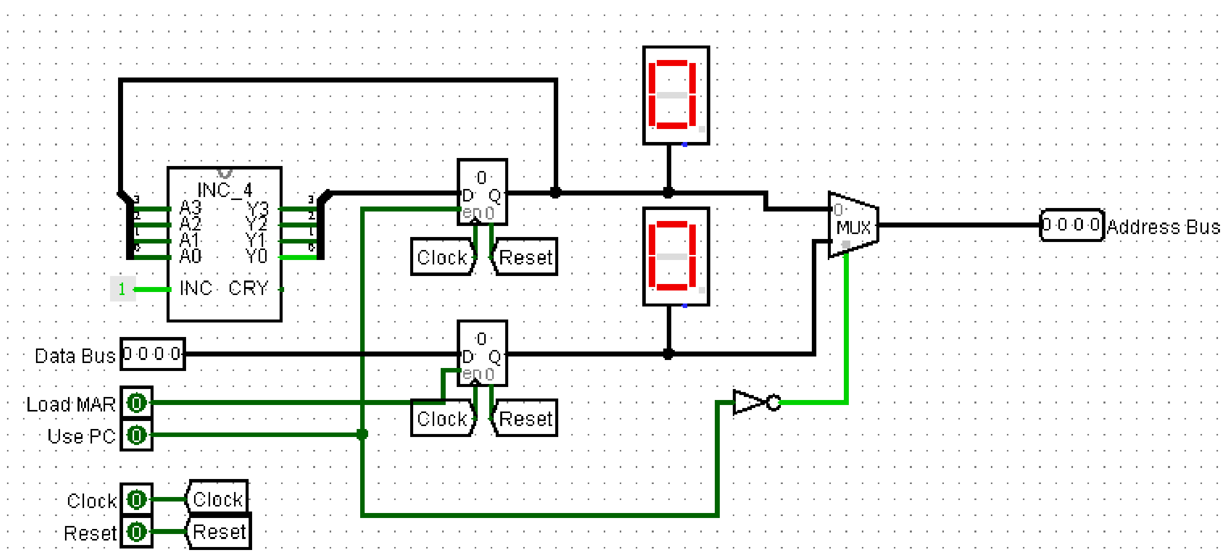
ACC to Data Bus	0
Load ACC	1
/~A_Only	0
/~Invert	0
Logic/~Arith	1
Instruction [Negate(operand) to ACC] (2's complement)	
Control Line	Value
4-bit Binary Keyboard (Address Bus)	1111
Write	1
Read	0
ACC to Data Bus	0
Load ACC	1
/~A_Only	0
/~Invert	0
Logic/~Arith	0

Why do you think the register at the output of the ALU is called the 'accumulator'?

It is where results of the ALU accumulate.

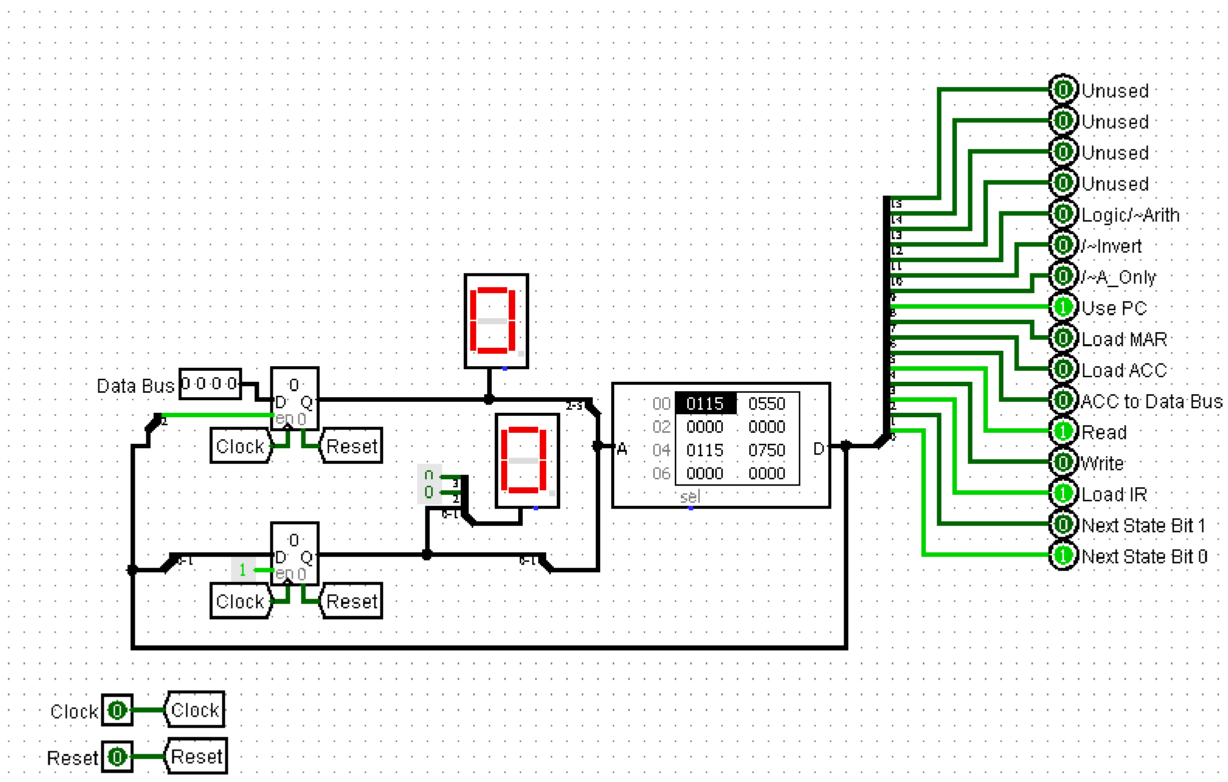
Task 4-8: Build the Memory-Address-Generation Circuit

Include a picture of your Logisim memory address generation circuit here:



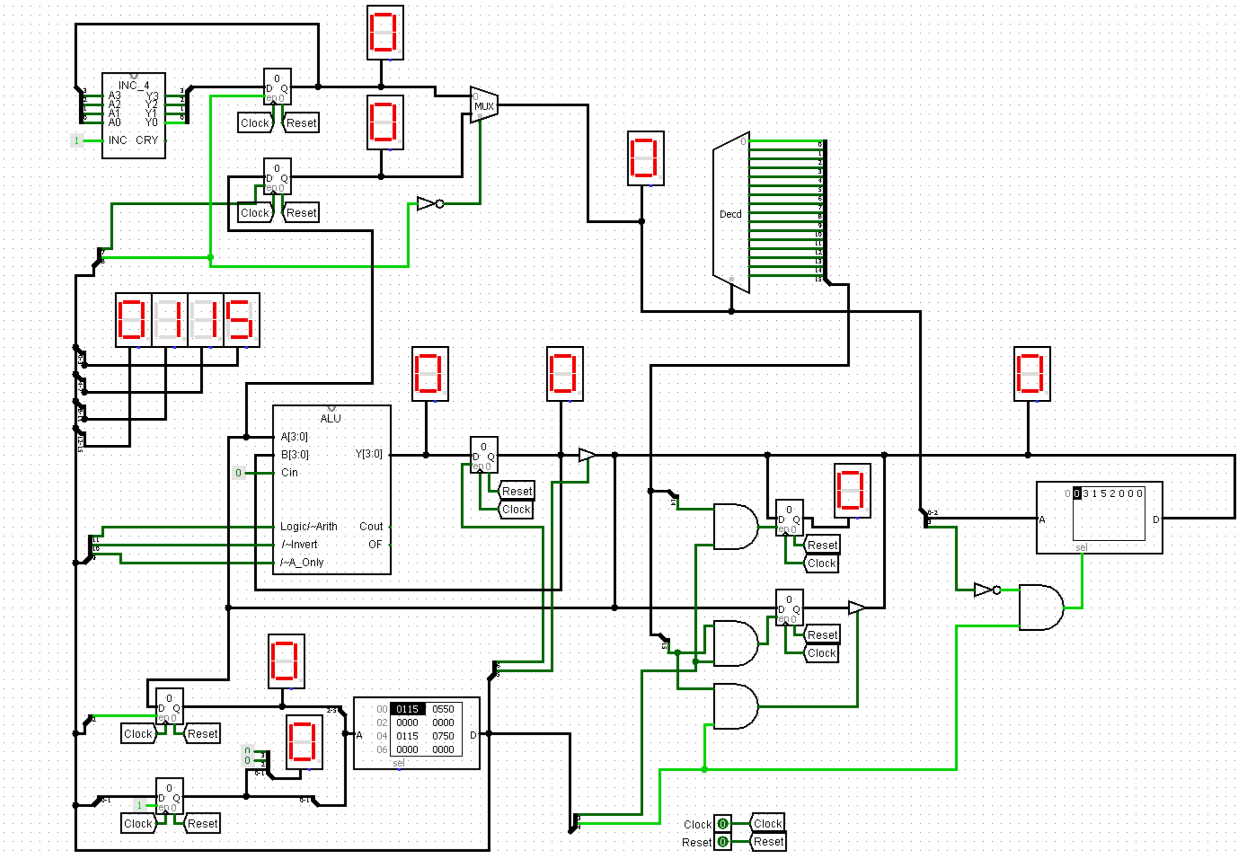
Task 4-9: Build the Controller Circuit

Include a picture of your Logisim controller circuit here:



Task 4-10: Build the Complete Microprocessor Circuit

Include a picture of your Logisim complete microprocessor circuit, with controller, here:



Task 4-11: Write and Execute a Simple Program for Your Microprocessor

Write the program given in your laboratory manual into the appropriate memory locations. Observe the operation of each step of your program (i.e. observe the values of the control lines and record whether data is being moved properly according to those control line settings). Did you get an 8 stored into the accumulator with you initial test?

If not, what error(s) did you find during your debugging process?

Yep! Worked on the first try.

Task 4-12: Add the 'AND', 'Zero', 'Subtract', and 'Store ACC' Instructions

Use Table 6 and Table 7 to enter your values into the microinstruction definition table for each of the four instructions asked for in the laboratory manual. Be sure to label the name of each and every instruction.

Table 6									
	Instruction	AND				ZERO			
	Opcode	3				4			
	Pres. State	00	01	02	03	00	01	02	03
Description	**Pin number**								
Next State Bits	1-0	1	0			1	0		
Load IR	2	1	0			1	0		
Write	3	0	0			0	0		
Read	4	1	1			1	0		
ACC to Data Bus	5	0	0			0	1		
Load ACC	6	0	1			0	1		
Load MAR	7	0	0			0	0		
Use PC	8	1	1			1	0		
/~A_only	9	0	1			0	1		
/~Invert	10	0	1			0	0		
Logic/~Arith	11	0	1			0	0		
X	12	0	0			0	0		
X	13	0	0			0	0		
X	14	0	0			0	0		
X	15	0	0			0	0		
	HEX equiv	0115	0f50			0115	0260		

Table 7									
	Instruction	SUBTRACT				STORE ACC			
	Opcode	5				6			
	Pres. State	00	01	02	03	00	01	02	03
Description	**Pin number**								
Next State Bits	1-0	1	0			1	1	0	
Load IR	2	1	0			1	0	0	
Write	3	0	0			0	0	1	
Read	4	1	1			1	1	0	
ACC to Data Bus	5	0	0			0	0	1	

Load ACC	6	0	1			0	0	0	
Load MAR	7	0	0			0	1	0	
Use PC	8	1	1			1	1	0	
/~A_only	9	0	1			0	0	0	
/~Invert	10	0	0			0	0	0	
Logic/~Arith	11	0	0			0	0	0	
X	12	0	0			0	0	0	
X	13	0	0			0	0	0	
X	14	0	0			0	0	0	
X	15	0	0			0	0	0	
	HEX equiv	0115	0350			0115	0192	0028	

Test your instructions by writing and executing programs. Record at least **four** programs and the output of each program in tables like that of Table 8.

Table 8		
Program #0 (*Example*: ADD = 3+5)		
Address	Value	Operation (In English)
0	0	The 'Load ACC' Opcode
1	3	The number '3' to be loaded into the Accumulator
2	1	The 'Add to ACC' Opcode
3	5	The number '5' to be added to the Accumulator
4	2	The 'Stop' Opcode
What was the final output of your program? ___8__		
Was the program successful? YES		
If not what error(s) did you find in your circuit?		

Program # (AND 2 and 3)		
Address	Value	Operation (In English)
0	0	Load ACC
1	2	Constant
2	3	ADD with ACC
3	3	Constant
4	2	Stop
What was the final output of your program? _2____		
Was the program successful? Yes		
If not what error(s) did you find in your circuit?		

Program # (Zero)		
Address	Value	Operation (In English)
0	0	Load ACC
1	2	Load 2 into ACC
2	4	Zero ACC
3	2	Stop
What was the final output of your program? 0		
Was the program successful? Yes		
If not what error(s) did you find in your circuit?		

Program # (Subtract 2-1)		
Address	Value	Operation (In English)
0	0	Load ACC
1	2	Load 2 into ACC
2	5	Subtract ACC
3	1	Constant
4	2	Stop
What was the final output of your program? 1		
Was the program successful? Yes		
If not what error(s) did you find in your circuit?		

Program # (Store 1 to RAM)		
Address	Value	Operation (In English)
0	0	Load ACC
1	1	Constant
2	6	Store ACC to RAM
3	15	Memory Address
4	2	Stop
What was the final output of your program? 1 stored in RAM		
Was the program successful? Yes		
If not what error(s) did you find in your circuit?		

Task 4-13: Invent Your Own Instruction (Extra Credit)

Fill in the following two tables for your invented instruction.

	Instruction									
	Opcode	7								
	Pres. State	00	01	02	03	00	01	02	03
Description	**Pin number**								
Next State Bits	1-0								
Load IR	2								
Write	3								
Read	4								
ACC to Data Bus	5								
Load ACC	6								
Load MAR	7								
Use PC	8								
/~A_only	9								
/~Invert	10								
Logic/~Arith	11								
X	12								
X	13								
X	14								
X	15								
	HEX equiv								

Program # ( )		
Address	Value	Operation (In English)
What was the final output of your program? ______		
Was the program successful? Yes or No		
If not what error(s) did you find in your circuit?		

SIMULATION LAB 4: LAB REPORT GRADE SHEET

Name ________________________________

Grading Criteria	Max Points	Points Lost
Template		
Neatness, Clarity, and Concision	2	
Description of Assigned Tasks, Work Performed & Outcomes Met		
Task 4-1: Build the Addressing Logic	2	
Task 4-2: Build a 4-Bit ROM Memory with eight cells	3	
Task 4-3: Build 4-Bit Output Port	2	
Task 4-4: Build the 4-Bit RAM Cell	2	
Task 4-5: Build the Brainless Central Processing Unit	5	
Task 4-6: Build the Brainless Microprocessor	6	
Task 4-7: Test and Control the Brainless Microprocessor	8	
Task 4-8: Build the Memory-Address-Generation Circuit	3	
Task 4-9: Build the Controller Circuit	6	
Task 4-10: Build the Complete Microprocessor Circuit	6	
Task 4-11: Write and Execute a Simple Program for your Microprocessor	3	
Task 4-12: Add the ‘AND’, ‘Zero’, ‘Subtract’, and ‘Store ACC’ Instructions	12	
Task 4-13: Invent Your Own Instruction	(3 extra points)	
Self-Assessment Worksheet (The content of the self-assessment worksheet will not be graded. Full credit is given for including the completed worksheet.)	(2 extra points)	
Lab Score (60 points total)	**Points Lost**	
	Late Lab	
	Lab Score	

SELF-ASSESSMENT WORKSHEET

Put 'X's' in the table below indicating how strongly you agree or disagree that the outcomes of the assigned tasks were achieved. Use '5' to indicate that you 'strongly agree', '3' to indicate that you are 'neutral', and '1' to indicate that you 'strongly disagree'. Use 'NA', 'Not Applicable', when the tasks you performed did not elicit this outcome. Credit will be given for including this worksheet with your lab report; however, your **<u>responses</u> will not be graded**. They are for your instructor's information only.

Table __: Self-Assessment of Outcomes for Simulation Lab 4: The Microprocessor

After completing the assigned tasks and report, I am able to:	**5**	**4**	**3**	**2**	**1**	**NA**
Build, debug and control a simulation of a ROM, RAM and an output port.						
Build and debug a simulation of a microprocessor that is absent a controller.						
Act as the controller for an elementary microprocessor.						
Design a PROM-based controller for an elementary microprocessor.						
Create an instruction set for an elementary microprocessor.						
Use the language of your instruction set to create a program and enter it into memory.						
Execute a program on your simulated microprocessor.						

Write below any suggestions you have for improving this laboratory exercise so that the stated learning outcomes are achieved.