Tony Hong
1005109482
Oct. 28, 2020

CSC258 Prelab (Lab 6)

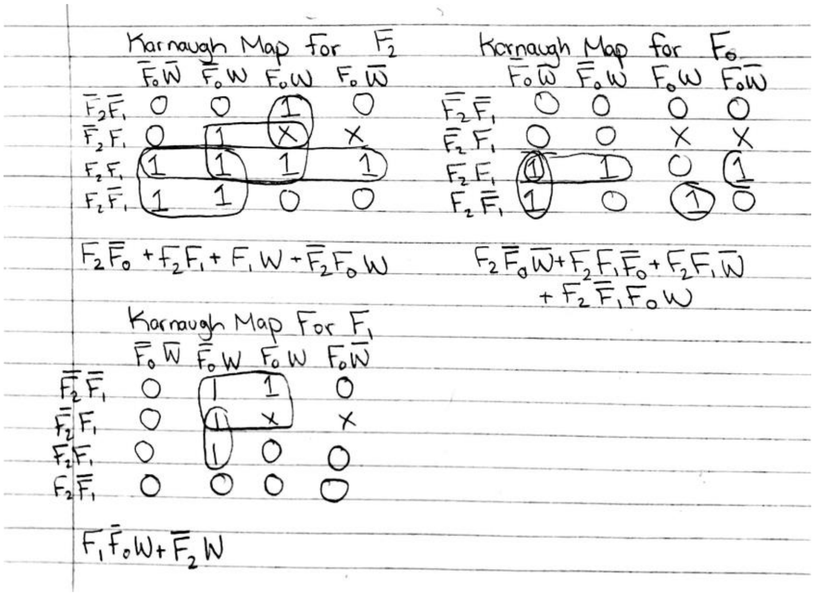
Part 1: Sequence Recognizer

1. Given the starter circuit, the reset signal is an asynchronous reset that is active high. During tests, the reset should be used when we want to test a new sequence of 4 numbers without influence from the previous inputs.
2. Below is the state table and assigned flip flop values for each state

Input State	F2	F1	F0	W (Input)	**F2**	**F1**	**F0**	**Output State**
A	0	0	0	0	0	0	0	A
A	0	0	0	1	0	1	0	B
B	0	1	0	0	0	0	0	A
B	0	1	0	1	1	1	0	C
C	1	1	0	0	1	0	1	E
C	1	1	0	1	1	1	1	D
D	1	1	1	0	1	0	1	E
D	1	1	1	1	1	0	0	F
E	1	0	1	0	0	0	0	A
E	1	0	1	1	0	0	1	G
F	1	0	0	0	1	0	1	E
F	1	0	0	1	1	0	0	F
G	0	0	1	0	0	0	0	A
G	0	0	1	1	1	1	0	C

Note, that some of the transitions change more than 1 bit. It is impossible to design a state table for this FSM where each transition changes a maximum of 1 bit because there are 5 different states that either transition to or transition from E.

Below are the K-maps to determine the circuits that will implement the logic from the state table.



Karnaugh Map for F_2

	$\bar{F}_0\bar{W}$	$\bar{F}_0W$	F_0W	$F_0\bar{W}$
$\bar{F}_2\bar{F}_1$	0	0	1	0
$\bar{F}_2F_1$	0	1	X	X
F_2F_1	1	1	1	1
$F_2\bar{F}_1$	1	1	0	0

$F_2\bar{F}_0 + F_2F_1 + F_1W + \bar{F}_2F_0W$

Karnaugh Map for F_0

	$\bar{F}_0\bar{W}$	$\bar{F}_0W$	F_0W	$F_0\bar{W}$
$\bar{F}_2\bar{F}_1$	0	0	0	0
$\bar{F}_2F_1$	0	0	X	X
F_2F_1	1	1	0	1
$F_2\bar{F}_1$	1	0	1	0

$F_2\bar{F}_0\bar{W} + F_2F_1\bar{F}_0 + F_2F_1\bar{W}$
$+ F_2\bar{F}_1F_0W$

Karnaugh Map For F_1

	$\bar{F}_0\bar{W}$	$\bar{F}_0W$	F_0W	$F_0\bar{W}$
$\bar{F}_2\bar{F}_1$	0	1	1	0
$\bar{F}_2F_1$	0	1	X	X
F_2F_1	0	1	0	0
$F_2\bar{F}_1$	0	0	0	0

$F_1\bar{F}_0W + \bar{F}_2W$

5. Below outlines a test plan for this FSM.

Prep: set clock tick to 1Hz, Async Reset the state and finally, turn on the Enable for the state.

Test 1: Turn W to 1 and watch 4 clock cycles pass and check if Z becomes 1. Let some more clock cycles pass and check if Z maintains its value of 1.

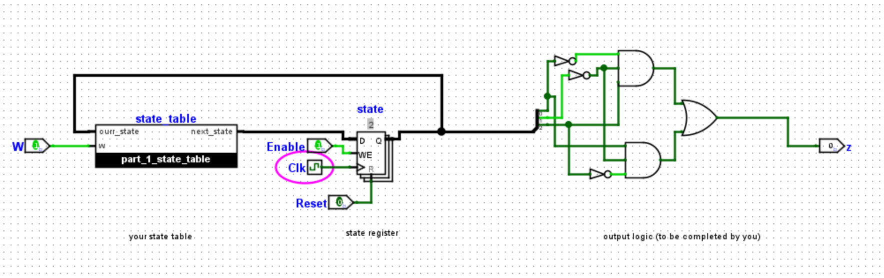
This test shows that the FSM detects the pattern 1111

Test2: Reset the state and turn W to 1 for 3 clock cycles. Turn W to 0 for the next clock cycle and turn W to 1 for the clock cycle after. Check to see that Z is 1 for that clock cycle

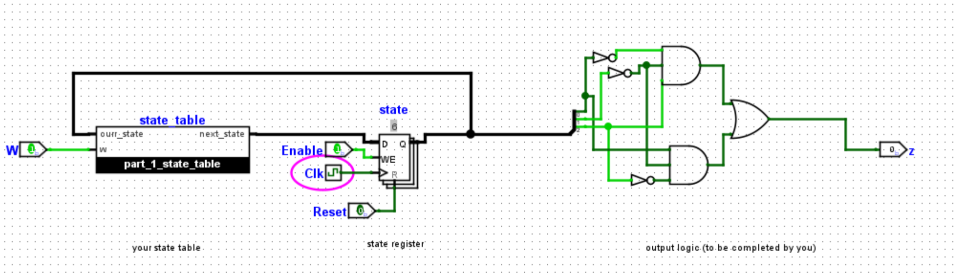
This test shows that the FSM detects the pattern 1101

Below are screen shots showing parts of Test 1.

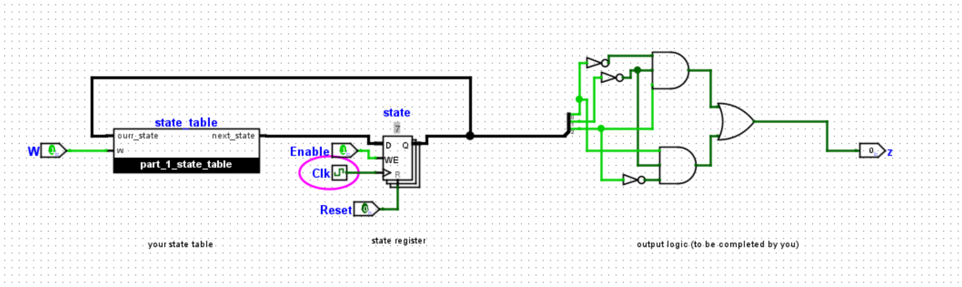
Cycle 1:



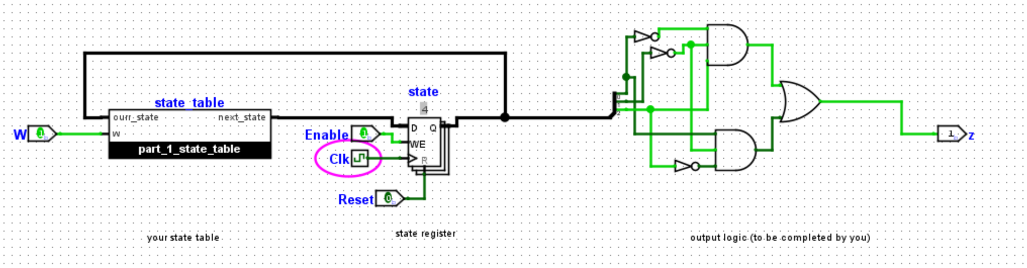
Cycle 2:



Cycle 3:

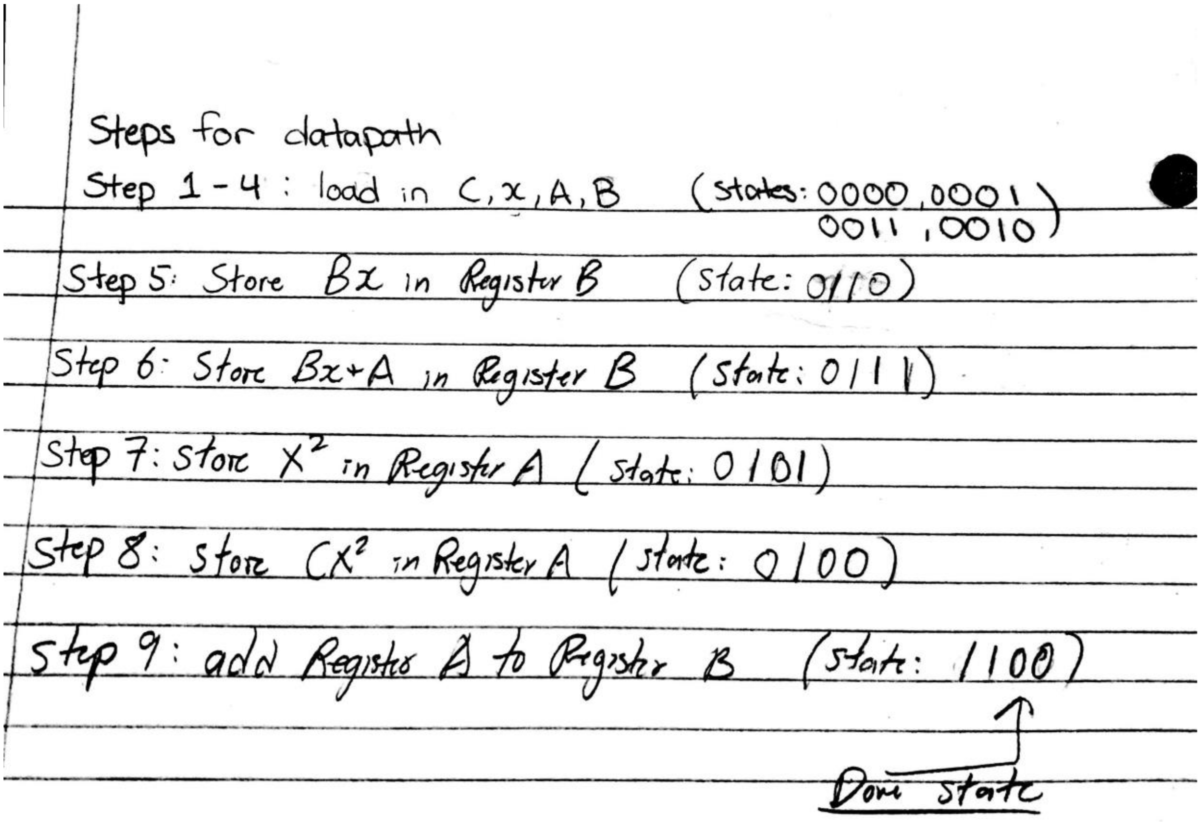


Cycle 4:



Part 2: FSM For $CX^2 + Bx + A$

Below is a plan to complete the required computation along with 4-bit state representations for each step.



Steps for datapath

Step 1 - 4 : load in C, x, A, B (states: 0000, 0001
0011, 0010)

Step 5: Store Bx in Register B (State: 0110)

Step 6: Store Bx+A in Register B (State: 0111)

Step 7: Store X^2 in Register A (State: 0101)

Step 8: Store CX^2 in Register A (state: 0100)

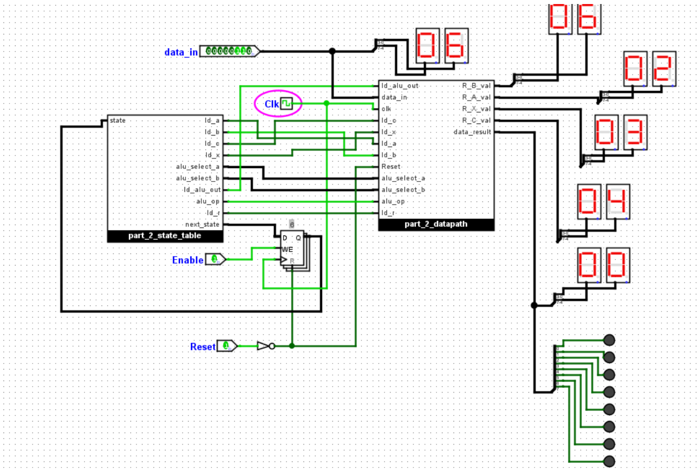
Step 9: add Register A to Register B (State: 1100)

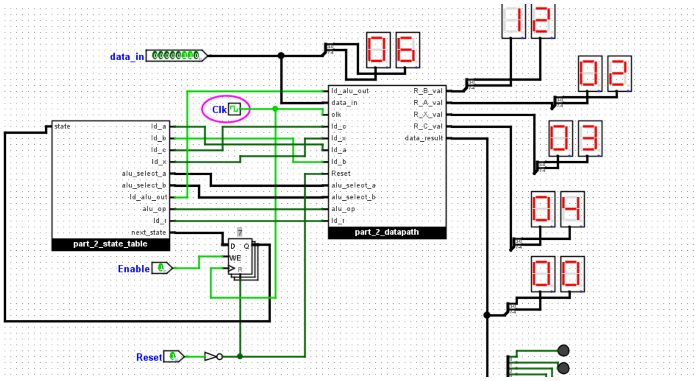
Done state

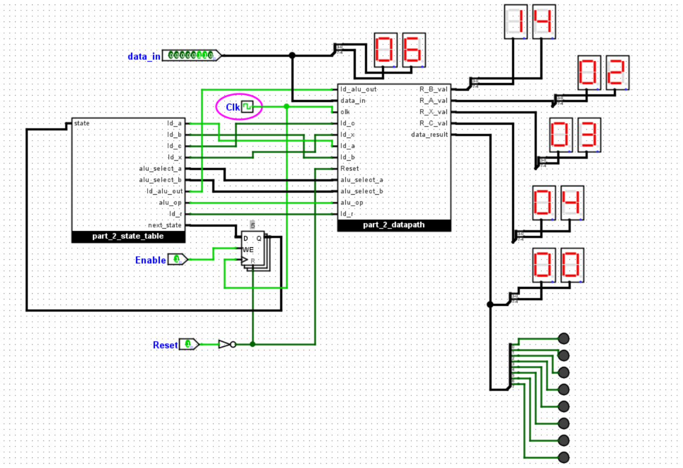
Below is a state table for all states of my controller and the signals for each ALU component

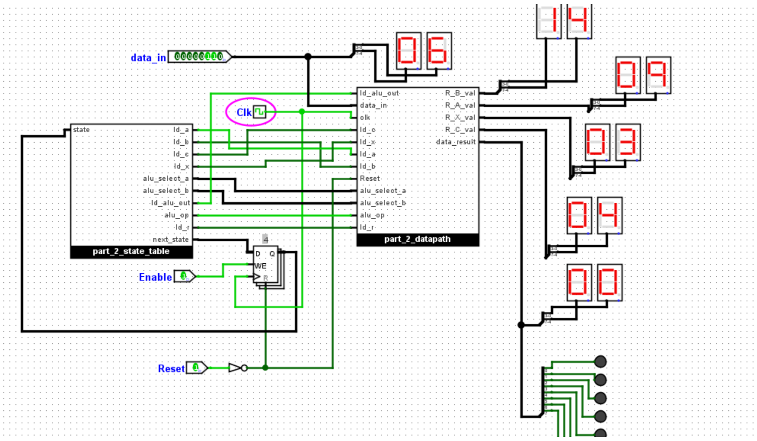
State	ld_a	ld_b	ld_c	ld_x	alu_select_a	alu_select_b	ld_alu_out	alu_op	ld_r	next_state
0000	0	0	1	0	00	00	0	0	0	0001
0001	0	0	0	1	00	00	0	0	0	0011
0011	1	0	0	0	00	00	0	0	0	0010
0010	0	1	0	0	00	00	0	0	0	0110
0110	0	1	0	0	01	11	1	1	0	0111
0111	0	1	0	0	01	00	1	0	0	0101
0101	1	0	0	0	11	11	1	1	0	0100
0100	1	0	0	0	00	10	1	1	0	1100
1100	0	0	0	0	00	01	0	0	1	1100

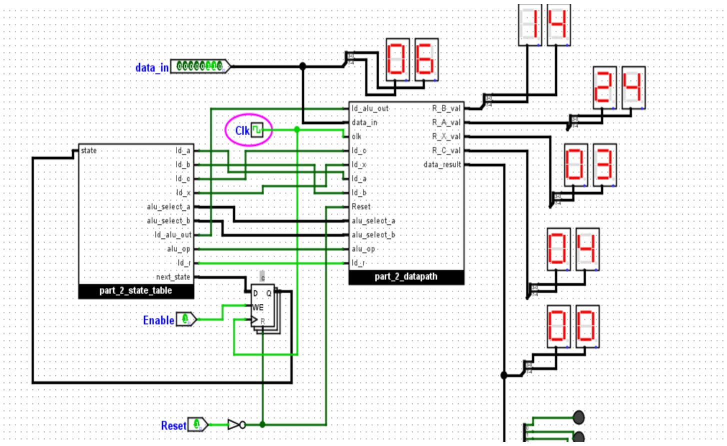
Test and Screenshots: C=4, X=3, B=6, A=2 Expected Output: 56

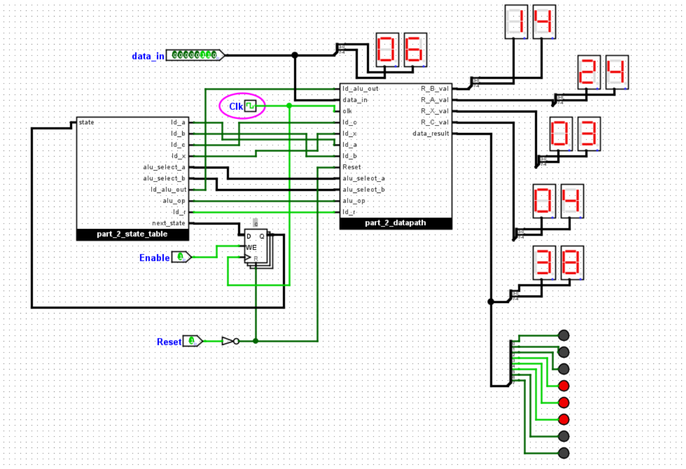












As seen in the final screenshot, the Output is indeed 56. (3*16 + 8)