Table of Contents:

Introduction:

Registers/Latches

In this lab Latches are used to store an 8 bit value, that when clocked, stores the value as memory and will be operated in the ALU Core.

FSM (FINITE STATE MACHINE)

The Finite State Machine, FSM for short, is a sequential circuit that uses a clock, data-in, and reset input. When clocked, the current state (4 bits) will change, along with the output (student number - 4 bits). This component will be used in the 4:16 Decoder and the ALU Core.

4:16 DECODER

The 4:16 Decoder, takes the 4 bit output of the current state and uses it as input. When Encode is 'HIGH', then the current state will be decoded into a 16 bit opcode. This opcode will be used in the ALU Core.

ALU CORE

The ALU Core is the main part of this CPU Unit. It will take the inputs of clock, the 8 bit values of latch A/B, as well as the opcode. Based on the binary value of the opcode (look at FSM for more details), the ALU Core will perform an operation on the two values. The output will be decoded using a BCD decoder so that it can be displayed on a seven segment display.

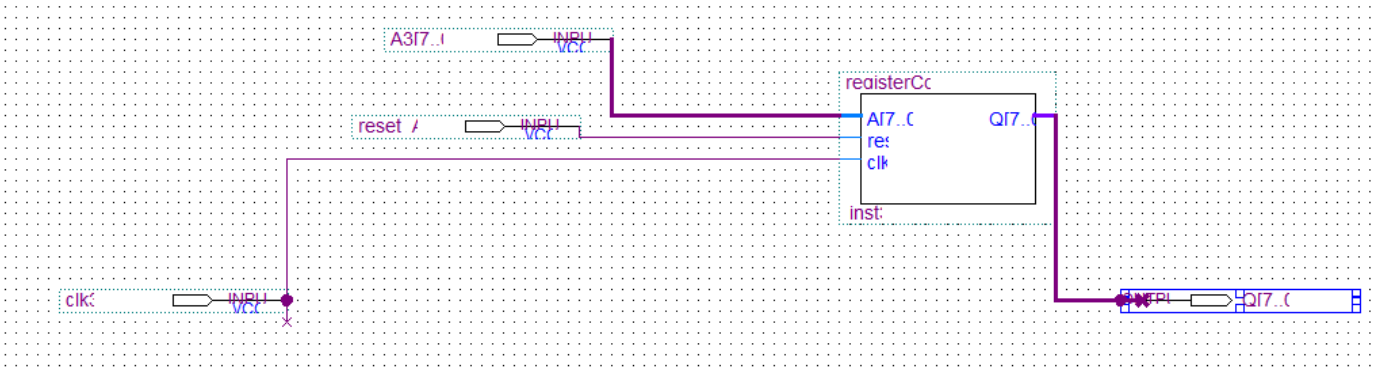
BCD DECODER

The BCD Decoder is used for the seven segment display, it takes a 4 bit input, such as current state or the result of the ALU Core. This will decode the 4 bit value into a 7 segment code in order to view the number on the FPGA board.

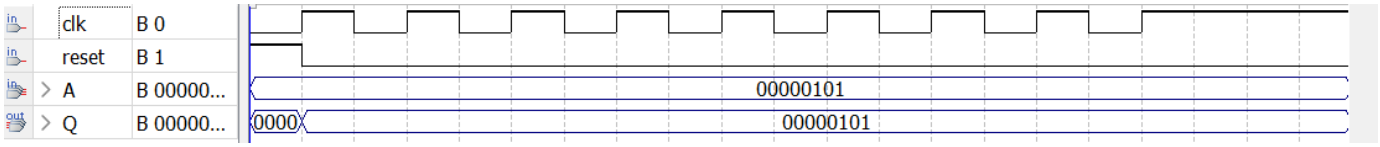
Components (In Depth): Register A, Register B, 4:16 Decoder, FSM

Register A:

The first register is a latch that will be used as input in the ALU Core. This latch will store an 8 bit binary value that the user chooses. If the reset value is ‘High’, then this would reset the binary number to “00000000”. When clocked, the value of the user input is stored in the latch and will be used in the ALU Core.

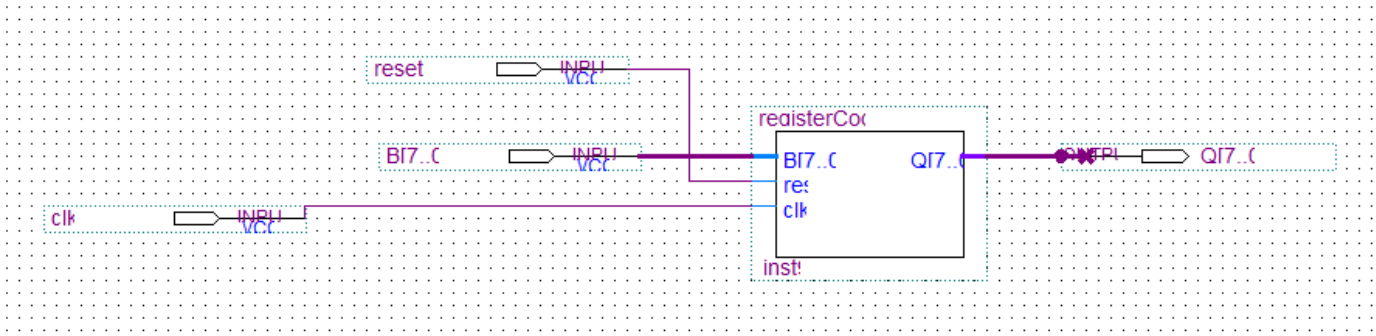


Waveform

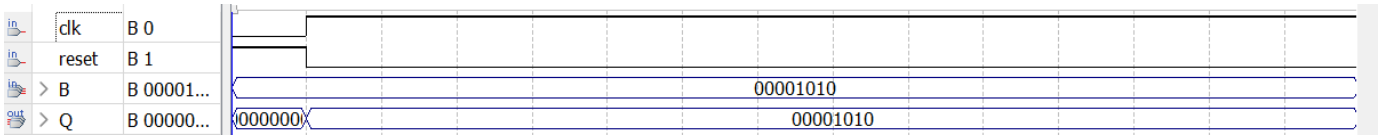


Register B:

Similar to Register A, this register is a latch that will be used as input in the ALU Core. It stores an 8 bit binary value that will be used as input for the ALU Core. When the reset input is ‘High’, then the value is reset to “00000000”. When clocked, the value that the user has inputted will be stored in the latch and will be operated on in the ALU Core.

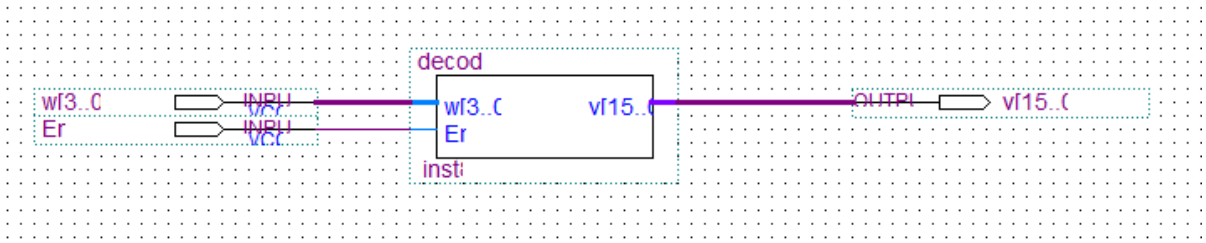


Waveform:

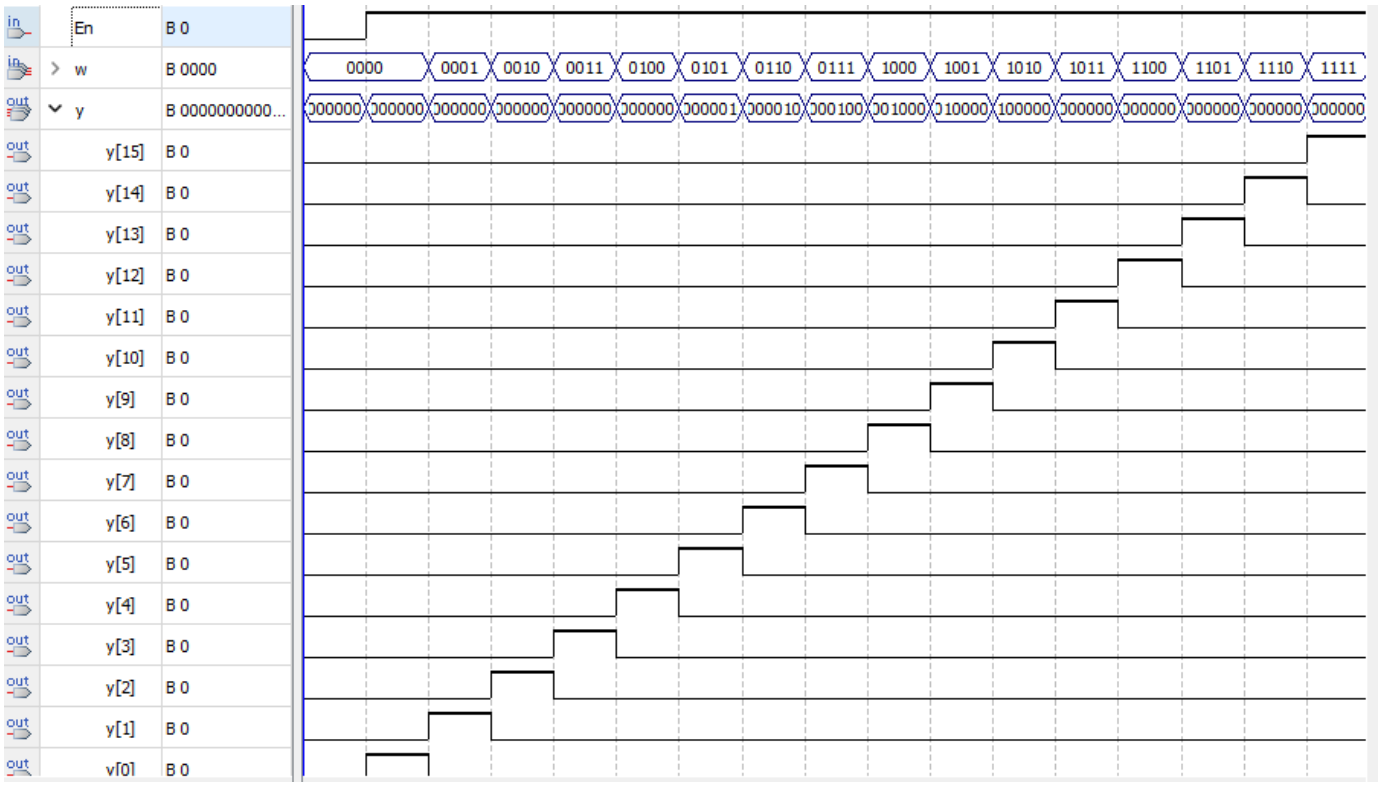


4:16 Decoder

The 4:16 Decoder will be used to decode the 4 bit binary values of the current state and will decode it into a 16 bit binary opcode. This opcode will select the operation needed to perform on register A and B. When the FSM is clocked then the current state changes, thus the decoder selects a different opcode.



Waveform

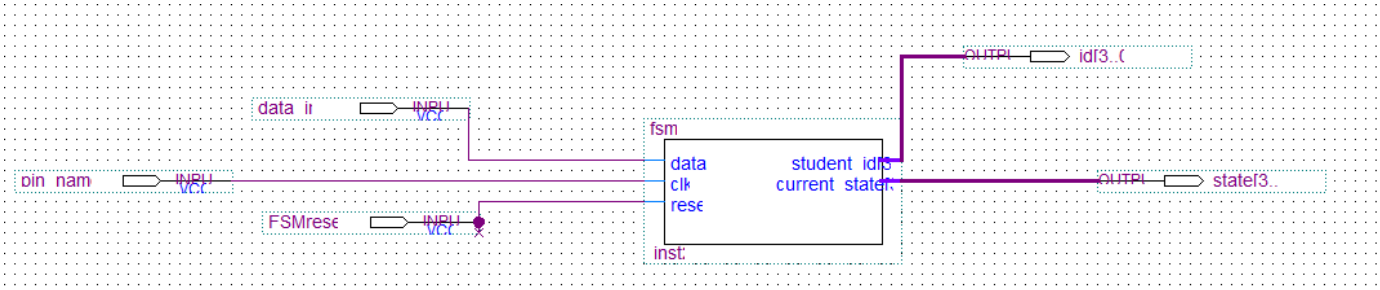

En
B 0
w
B 0000
y
B 0000000000...
y[15]
y[14]
y[13]
y[12]
y[11]
y[10]
y[9]
y[8]
y[7]
y[6]
y[5]
y[4]
y[3]
y[2]
y[1]
y[0]
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

Truth Table:

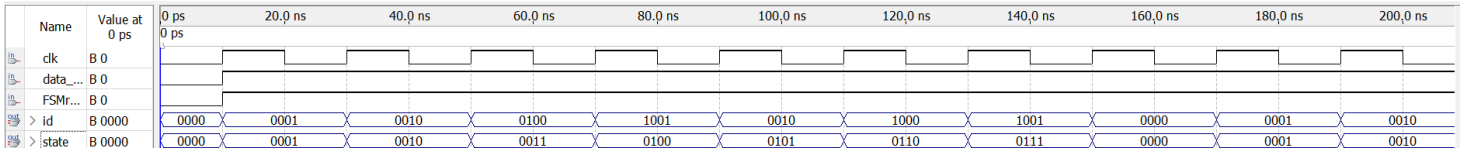
En	Input				Output															
	w_3	w_2	w_1	w_0	y_{15}	y_{14}	y_{13}	y_{12}	y_{11}	y_{10}	y_9	y_8	y_7	y_6	y_5	y_4	y_3	y_2	y_1	y_0
0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Finite State Machine (FSM)

The FSM or Finite State Machine is a crucial component to the CPU Unit. The FSM has a data input, when ‘High’, changes state and when ‘Low’ stays in the same state (Moore Machine). When the FSM is clocked then the current state (4 bit) changes, as well the output (student number 4 - bits). This will change the operation done on A and B once it passes through the 4:16 Decoder.



Waveform:



****The FSM is reset when the input is ‘Low’, this is for easier demonstration on the board.**

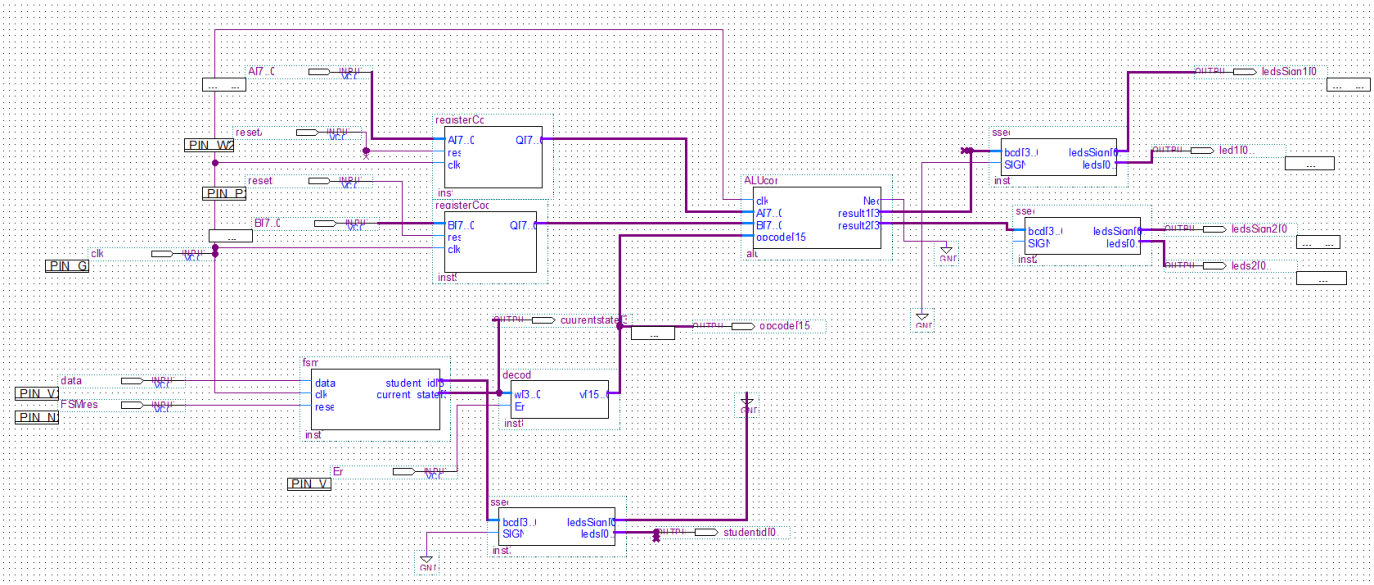
Truth Table:

Present State		Next State		Output z	
State #	$y_2 y_1$	$data\ in = 0$	$data\ in = 1$	$data\ in = 0$	$data\ in = 1$
s0	0000	0000	0001	0000	0001
s1	0001	0001	0010	0001	0010
s2	0010	0010	0011	0010	0100
s3	0011	0011	0100	0100	1001
s4	0100	0100	0101	1001	0010
s5	0101	0101	0110	0010	1000
s6	0110	0110	0111	1000	1001
s7	0111	0111	0000	1001	0000

ALU_1 for Problem Set 1 (ID 501249289)

The purpose of ALU One is to perform operations on two 8 bit binary values (stored). In the first ALU Core we have 4 inputs: clock, register A, register B, and the opcode. The first input, clock (connected to registers and FSM), changes the value of the opcode, thus changing the operation done on values A and B. As stated above the inputs of register A and B are the values being operated on, while the opcode determines the function to use on binary values A and B. We have two outputs, that split the 8 bit binary result of the operation done one A and B (eg. A+B = (8 bits) splits into 2, 4 bit values).

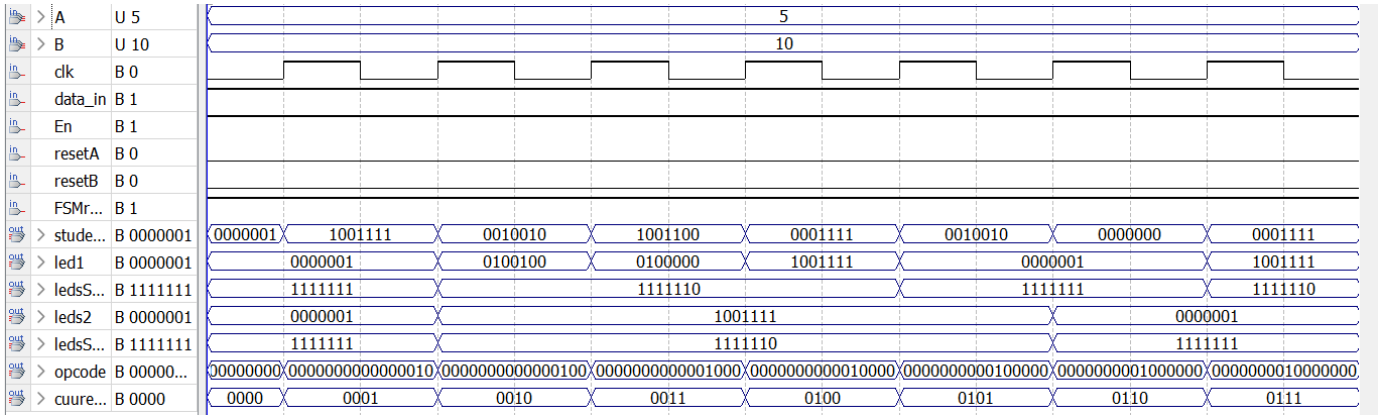
ALU ONE BLOCK SCHEMATIC DIAGRAM



Micro Code ALU ONE:

Function #	Opcode	Function
1	00000001	$sum(A, B)$
2	00000010	$dif(A, B)$
3	00000100	$\overline{A}$
4	00001000	$\overline{A \cdot B}$
5	00010000	$\overline{A + B}$
6	00100000	$A \cdot B$
7	01000000	$A \oplus B$
8	10000000	$A + B$

ALU One Waveform:

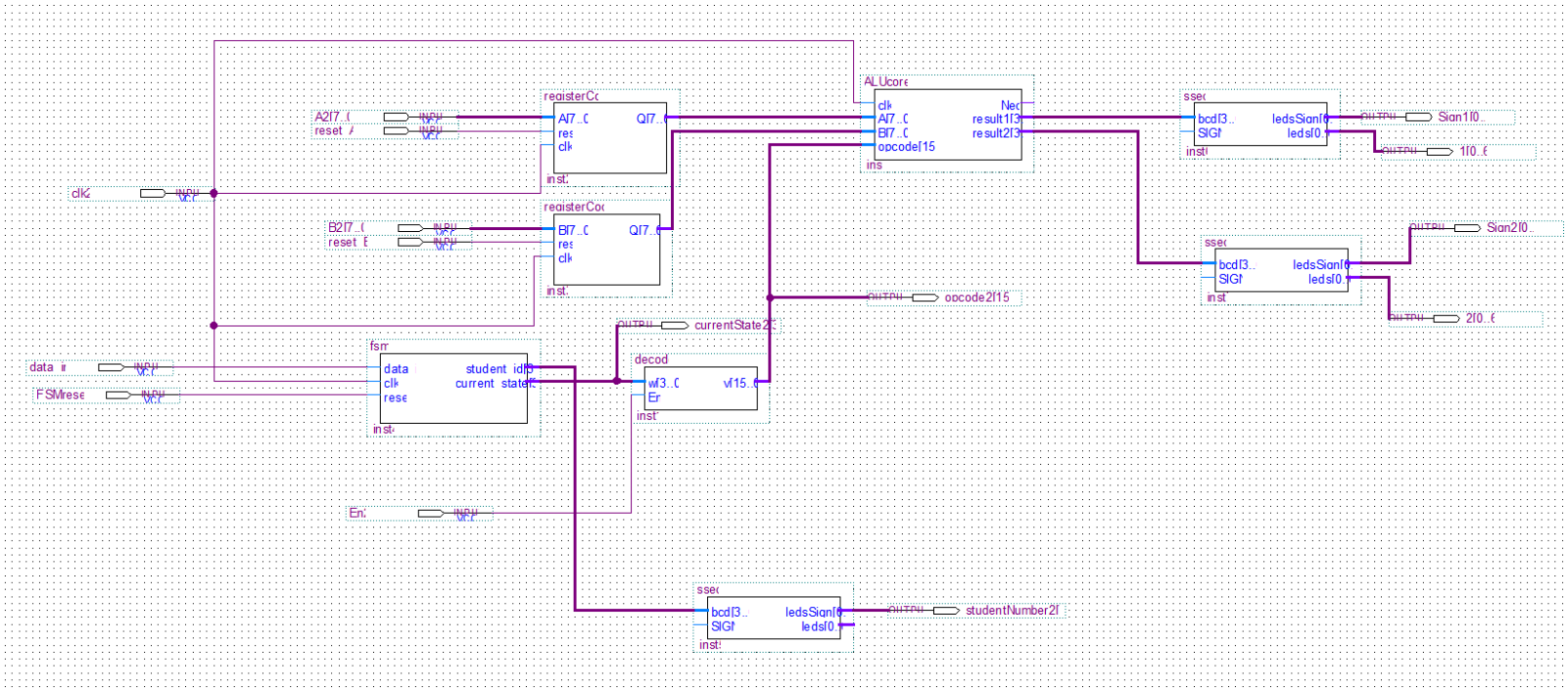


**For seven segments, when the value is zero ‘Low’, LED turns on.

ALU 2 for Problem Set 2 (ID 501249289)

The purpose of ALU 2 is similar to that of ALU 1. It takes the same inputs of clock, registers, and the opcode. The opcode is determined by the current state which is dependent on the FSM, in the waveform this is why there are eight different opcodes for eight different functions. When clocked the opcode changes, as well as the function on value A and value B. In the case of problem set 2, I was assigned to do letter b).

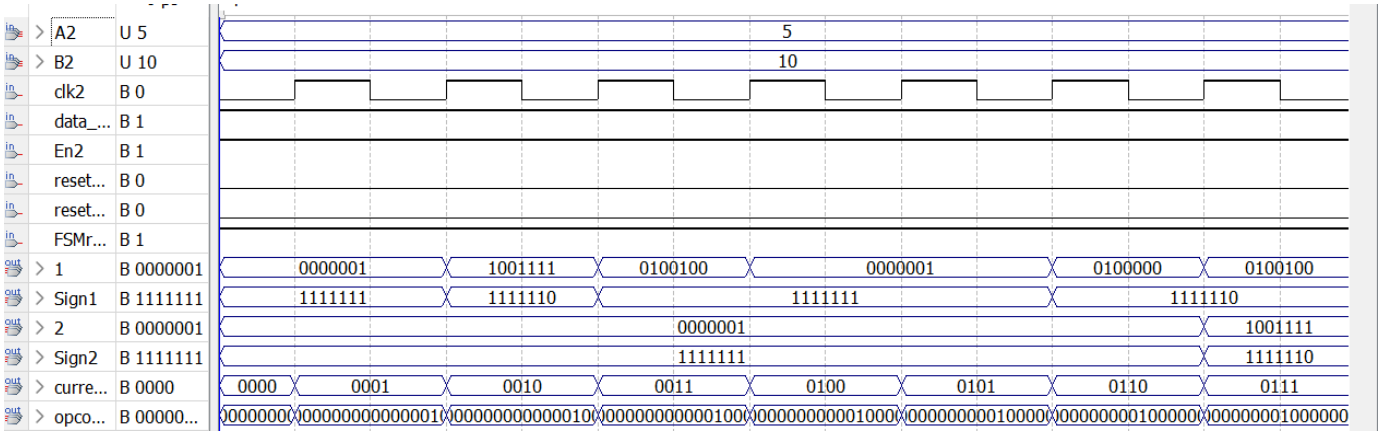
ALU ONE BLOCK SCHEMATIC DIAGRAM



Micro Code ALU 2 - Problem Set 2 b)

Function #	Operation/Function	Opcode
1	Swap the lower and upper 4 bits of A	00000001
2	Produce the result of ORing A and B	00000010
3	Decrement B by 5	00000100
4	Invert all bits of A	00001000
5	Invert the bit-significance order of A	00010000
6	Find the greater value of A and B and produce the results	00100000
7	Produce the difference between A and B	01000000
8	Produce the result of XNORin A and B	10000000

Waveform

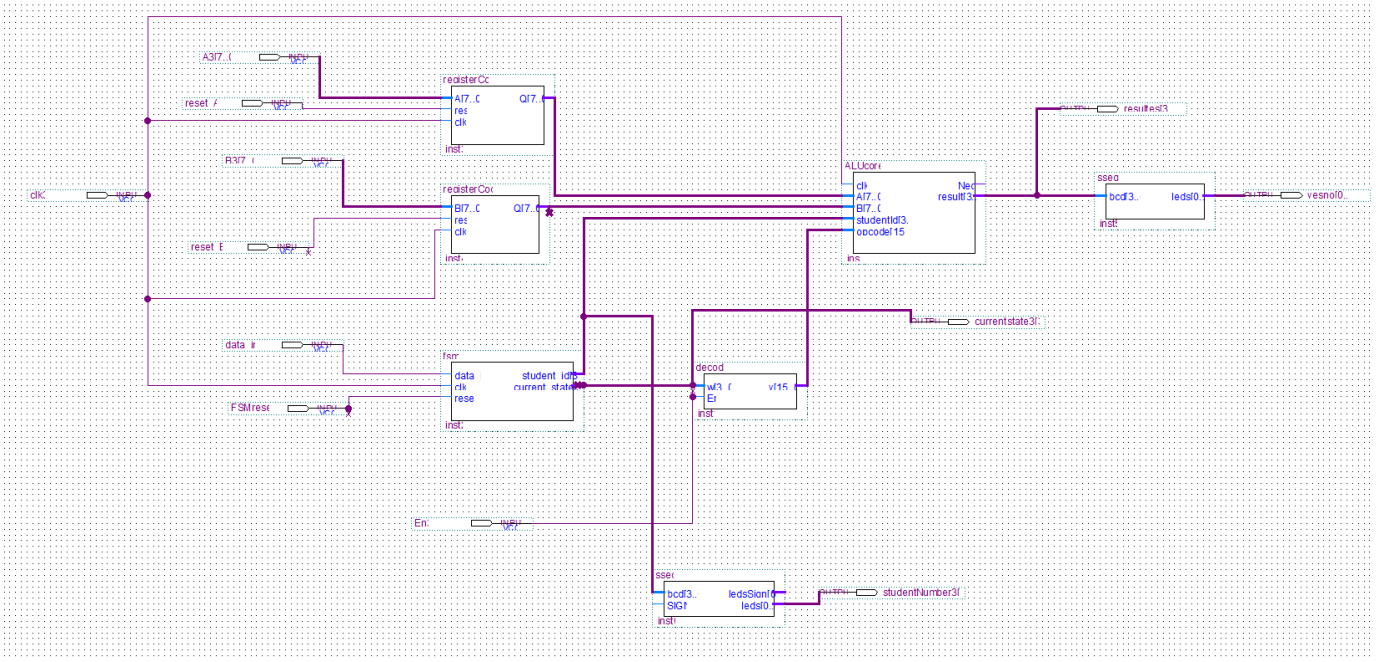


**For seven segments, when the value is zero ‘Low’, LED turns on.

ALU_3 for Problem Set 3 of the Lab 6 Procedure (ID 501249289)

ALU Three is very similar to ALU One and ALU Two, except we are taking five inputs instead. The new input will be the 4 bit value of the student number. When ALU Three is clocked, the student number changes as well as the opcode. Since ALU Three's purpose is to produce a result of whether the student number is odd or even, then the same function will be used for each opcode value. The output will be sent into a BCD decoder to display 'y' if the student number is even, 'n' if otherwise.

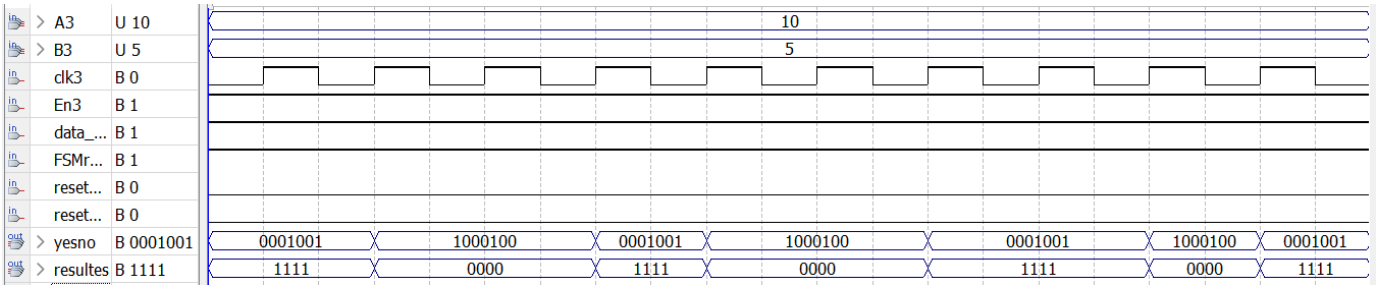
ALU THREE BLOCK SCHEMATIC DIAGRAM:



Micro Code ALU 3 - Problem Set 3 b)

Function #	Operation/Function	Opcode	Student #
1	If student ID is even result “0000” otherwise “1111”	00000001	0 -even
2	If student ID is even result “0000” otherwise “1111”	00000010	1 - odd
3	If student ID is even result “0000” otherwise “1111”	00000100	2 - even
4	If student ID is even result “0000” otherwise “1111”	00001000	4 - even
5	If student ID is even result “0000” otherwise “1111”	00010000	9 - odd
6	If student ID is even result “0000” otherwise “1111”	00100000	2 - even
7	If student ID is even result “0000” otherwise “1111”	01000000	8 - even
8	If student ID is even result “0000” otherwise “1111”	10000000	9 - odd

Waveform ALU 3



**For seven segments, when the value is zero ‘Low’, LED turns on.

Conclusion

To conclude this lab, we explored the basic components of a CPU Unit, such as registers/latches, Finite state machines, decoders, as well as the ALU Core. We can perform functions based on what is stored in the registers and the clock input is essential to the functioning of the CPU unit. This clock function changes the states of the FSM and the function/operation used on the registers. Overall, this lab went in depth of the concepts of the course COE 328: Digital Systems.