# December 02, 2023

**Design of a Simple General-Purpose Processor**

Ethan Chiu - 501170506

COE 318 - Digital System

Section 03

Instructor: Dr. Reza Sedaghat

Teaching Assistant: Marjan FatehiJananloo

# 1. Introduction

This lab's objective was to use the final four digits of a student's number to simultaneously display that number and the output of various boolean functions. Two latches named StorageUnit, an ALU (Arithmetic and Logic Unit), and a control unit made up of a 4:16 decoder and a Moore type FSM were the circuit components used in this lab.

Similar to storage units, which are used to temporarily store input values, are how latches operate. The ALU receives these input values and uses them to perform the intended boolean operation.

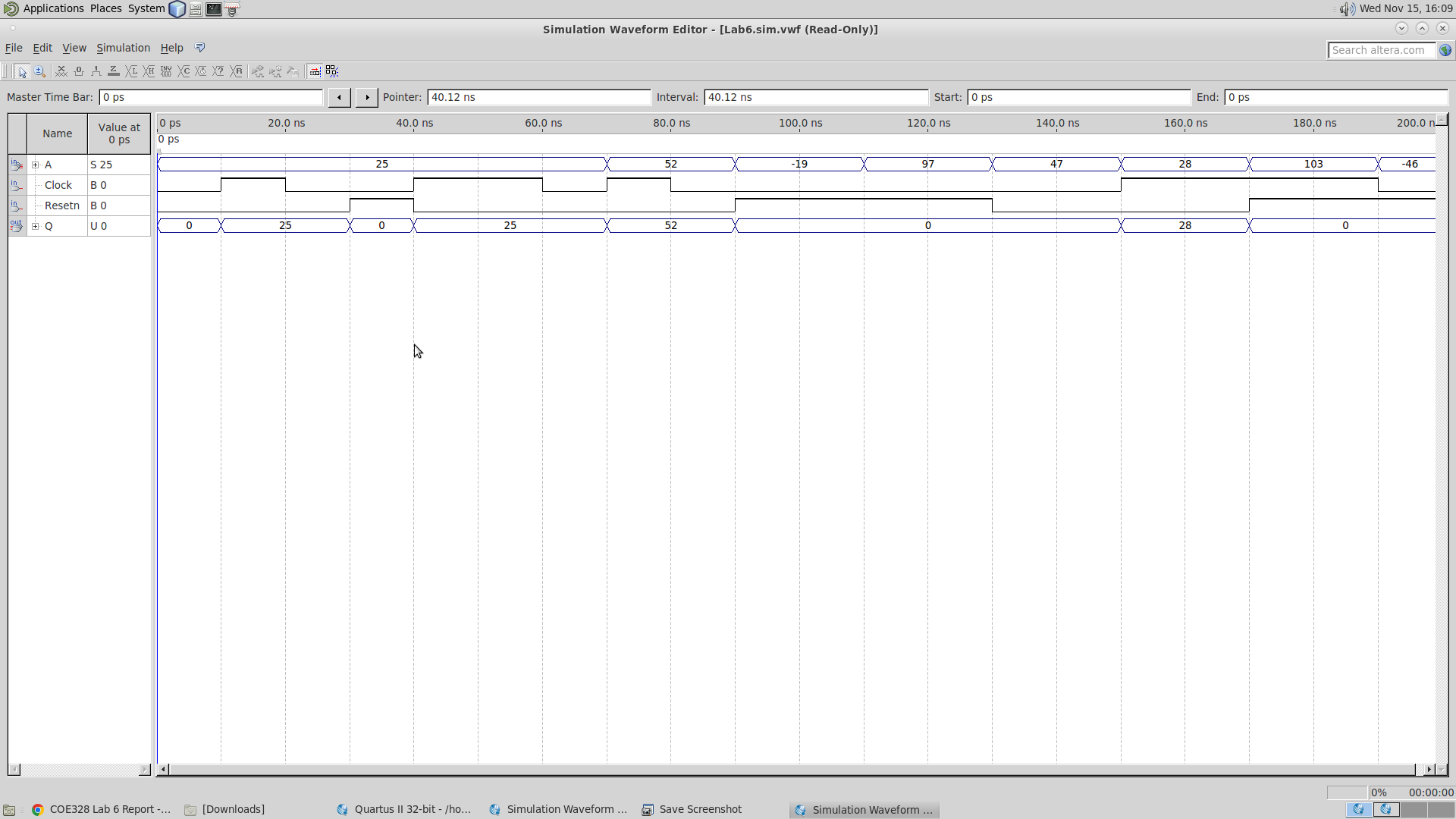
According to its name, the ALU is the part of the system that performs arithmetic and logical operations depending on the microcode given to the current state. Two 7-segment displays then show its 8-bit output.

The two components that make up the control unit (4:16 Decoder and Moore FSM) function as a selector for the ALU, directing which microcode should be applied and generated. As an up-counter, the MOORE FSM cycles through states 0 to 8 in order. The clock signal is used as the input, and the 4-bit current state is produced as the output. The 4x16 decoder receives the output, and the ALU receives the 16-bit output.

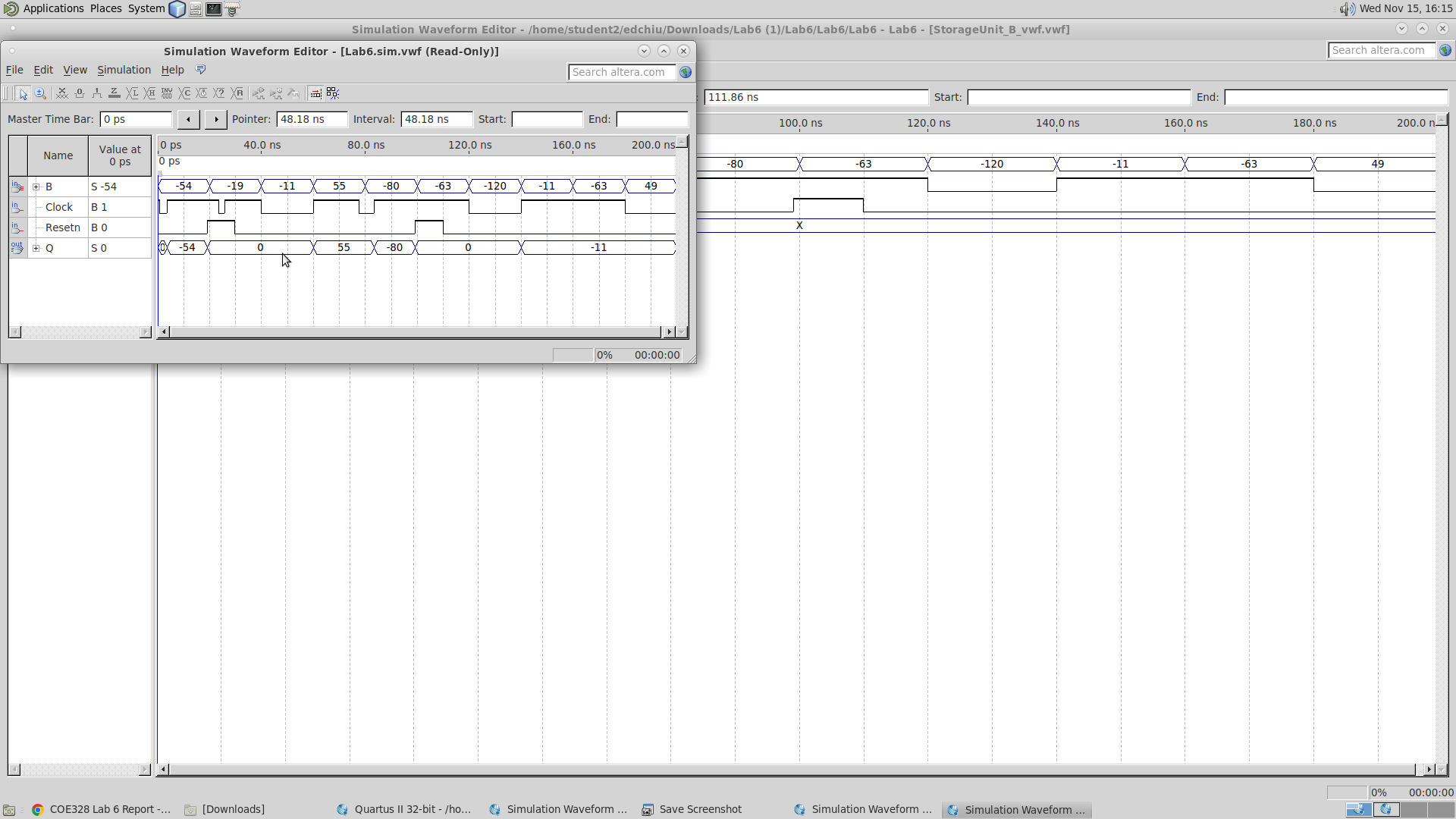
# 

# 2. Components Machine

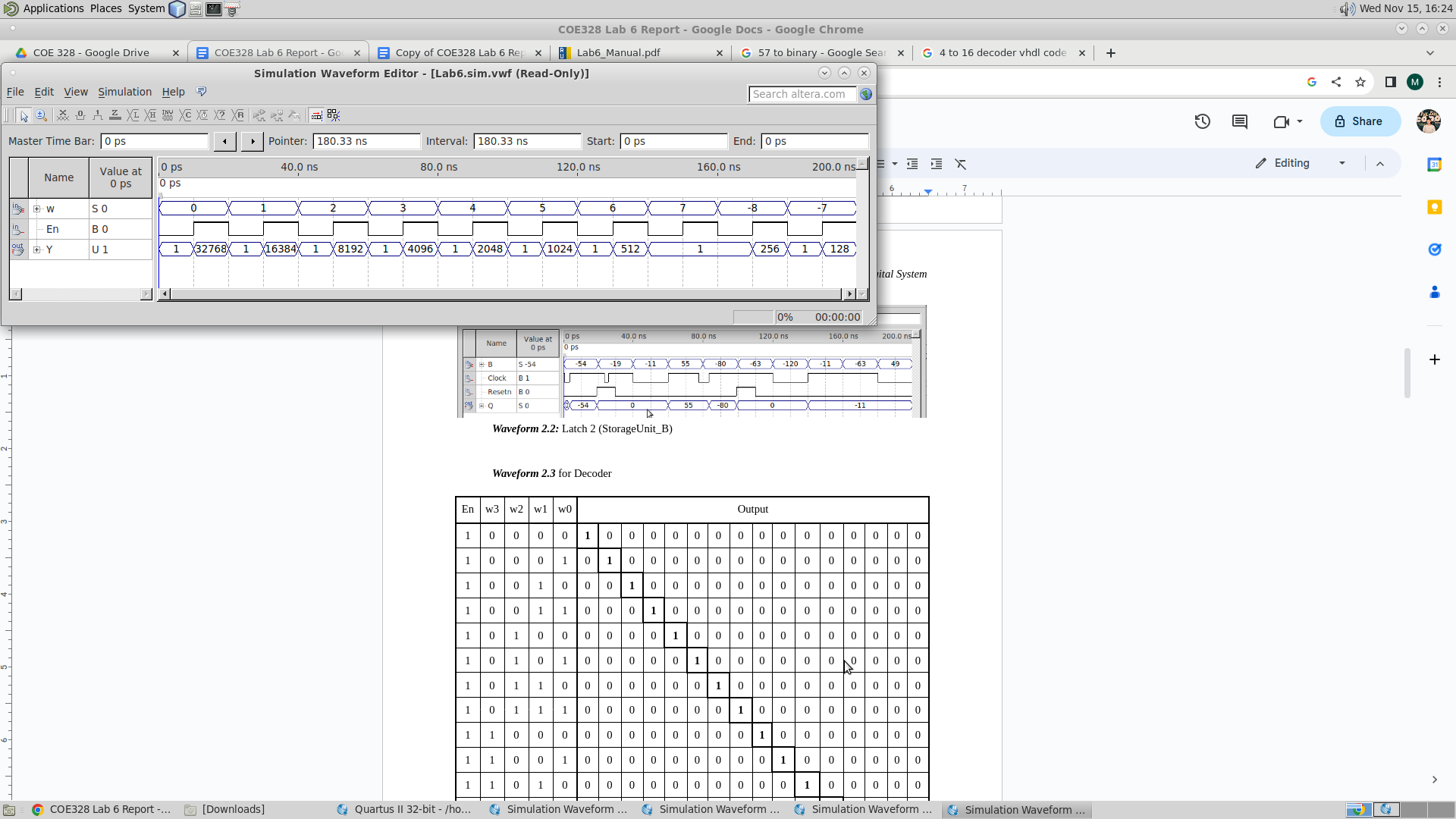
As previously indicated, latches 1 and 2 serve as temporary storage units for input values that are sent to the ALU and utilized in boolean operations. The ALU receives 16 bits from the 4:16 decoder, which are used to determine which operation should be performed. The 4:16 decoder receives the current state value from the Moore FSM, which up-counts from 0 to 8.



***Waveform 2.1:*** Latch 1 (StorageUnit\_A)



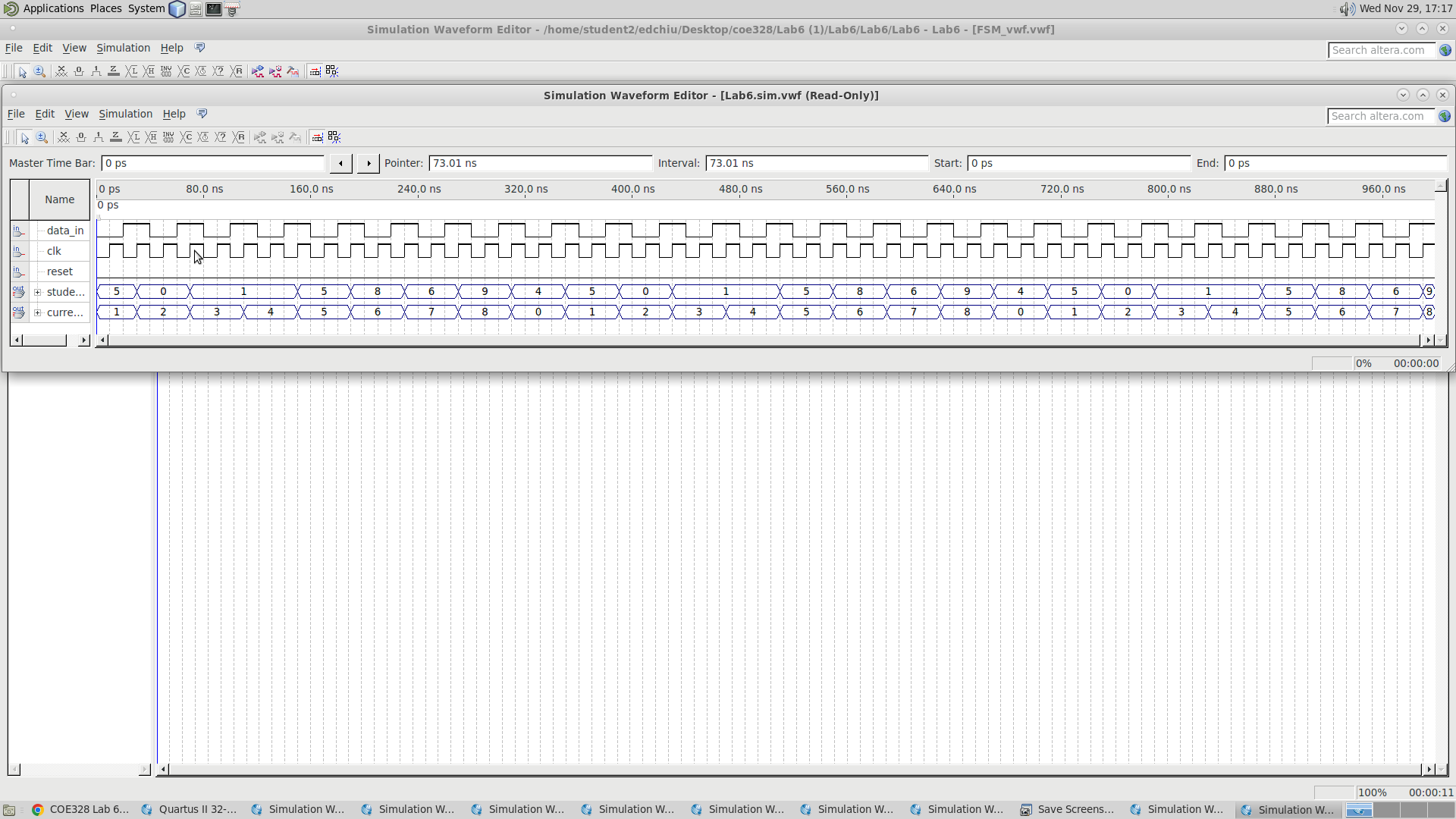
***Waveform 2.2:*** Latch 2 (StorageUnit\_B)



***Waveform 2.3*** for Decoder

| En | w3 | w2 | w1 | w0 | Output | | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 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 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** |

***Truth Table 2.3.1:*** 4:16 Decoder



***Waveform 2.4:*** Finite State Machine (MOORE Machine)

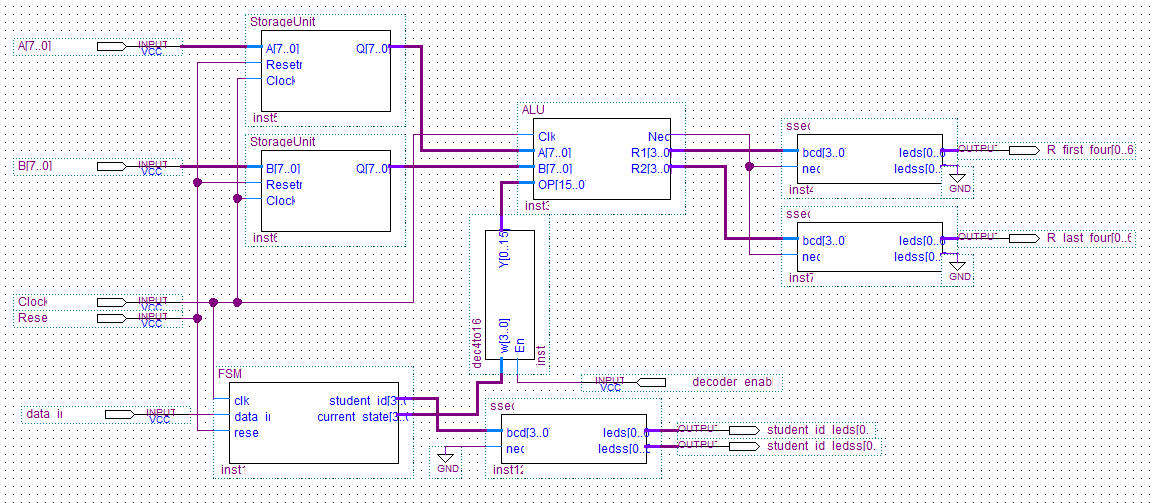
| **Current State** | **Next State** | | **Output** |
| --- | --- | --- | --- |
|  | w = 0 | w = 1 |  |
| 0000 | 0000 | 0001 | 5 |
| 0001 | 0001 | 0010 | 0 |
| 0010 | 0010 | 0011 | 1 |
| 0011 | 0011 | 0100 | 1 |
| 0100 | 0100 | 0101 | 5 |
| 0101 | 0101 | 0110 | 8 |
| 0110 | 0110 | 0111 | 6 |
| 0111 | 0111 | 1000 | 9 |
| 1000 | 1000 | 0000 | 4 |

***State Assigned Table 2.4.1:*** Finite State Machine (MOORE Machine)

# 3. Problem Set

## 3.1 Problem Set 1: ALU Core

The part of the GPU unit that handles all arithmetic and logical operations is called the Arithmetic and Logical Unit (ALU). There were three outputs (Neg, R1, and R2) and five inputs (Clk, A, B, student\_id, and OP) for the ALU in this experiment. This problem set's objective was to change the ALU so that, for the microcode it was given, it performs the proper boolean operation between the 8-bit values A and B and produces the right result. To accomplish this, the ALU selects the action to be performed between A and B using the 16-bit value of the FSM's "current state."

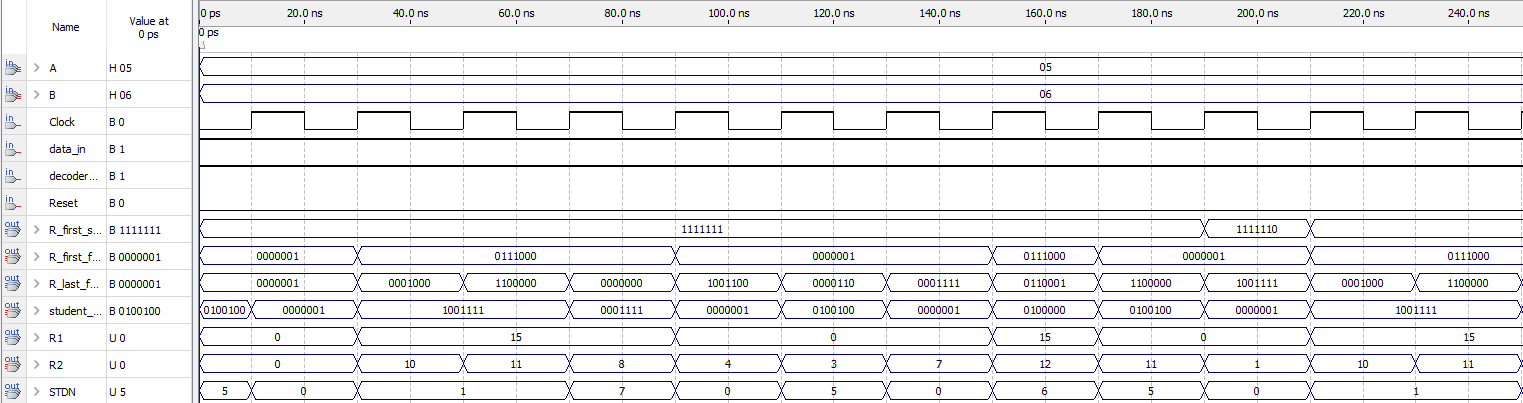


***Figure 3.1.1:***  Block Schematic for Problem 1

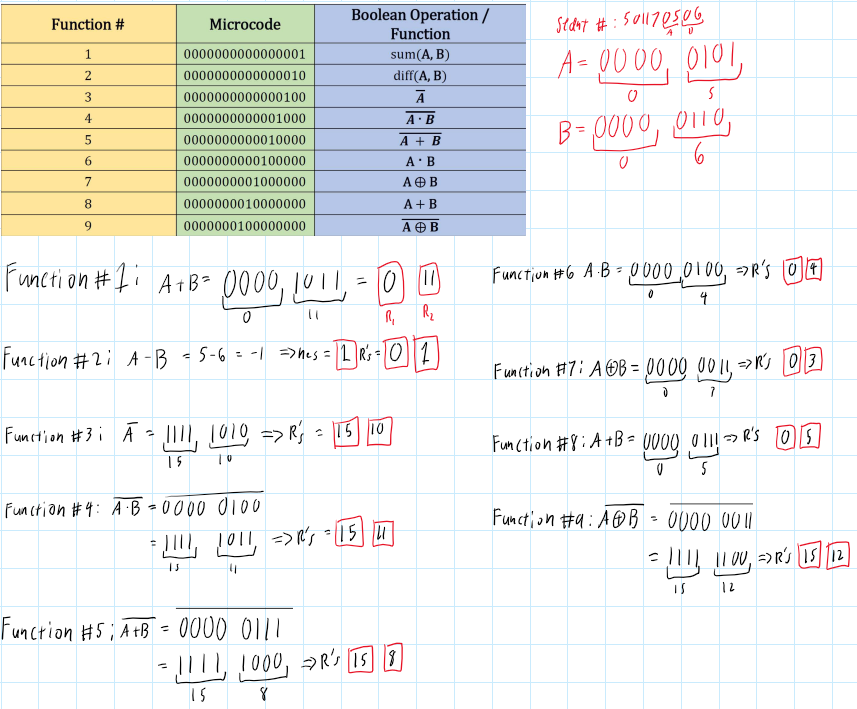
Regarding the intent behind each input and output, the input "Clock" gives an indication of how long each output will be active. The intended values are represented by the inputs "A" and "B," which are based on the student number's final four digits. The intended boolean operation selection is represented by the input "OP". The 8-bit outcome of the boolean operation was divided into two 4-bit values by the outputs "R1" and "R2." The SSEG components receive the values so they may show them.

| **Function #** | **Microcode** | **Boolean Operation / Function** |
| --- | --- | --- |
| 1 | 0000000000000001 | Sum(A,B) |
| 2 | 0000000000000010 | Diff(A,B) |
| 3 | 0000000000000100 |  |
| 4 | 0000000000001000 |  |
| 5 | 0000000000010000 |  |
| 6 | 0000000000100000 |  |
| 7 | 0000000001000000 |  |
| 8 | 0000000010000000 |  |
| 9 | 0000000100000000 |  |

***Table 3.1.2:*** ALU Core Operations for Problem 1



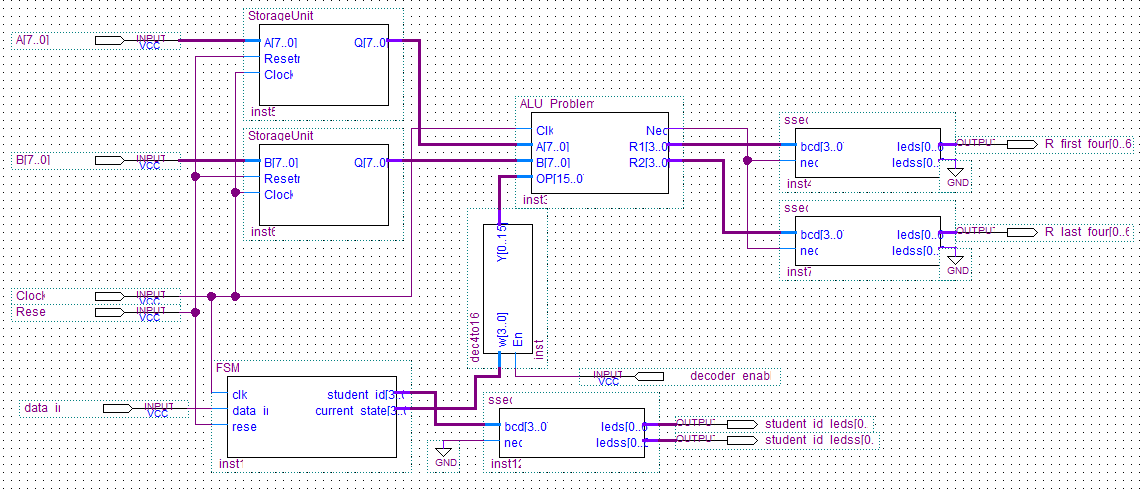
***Waveform 3.1.3:*** Problem 1



***Image:*** Proof of Problem set 1

## 3.2 Problem Set 2: Modified ALU Core (Assigned a)

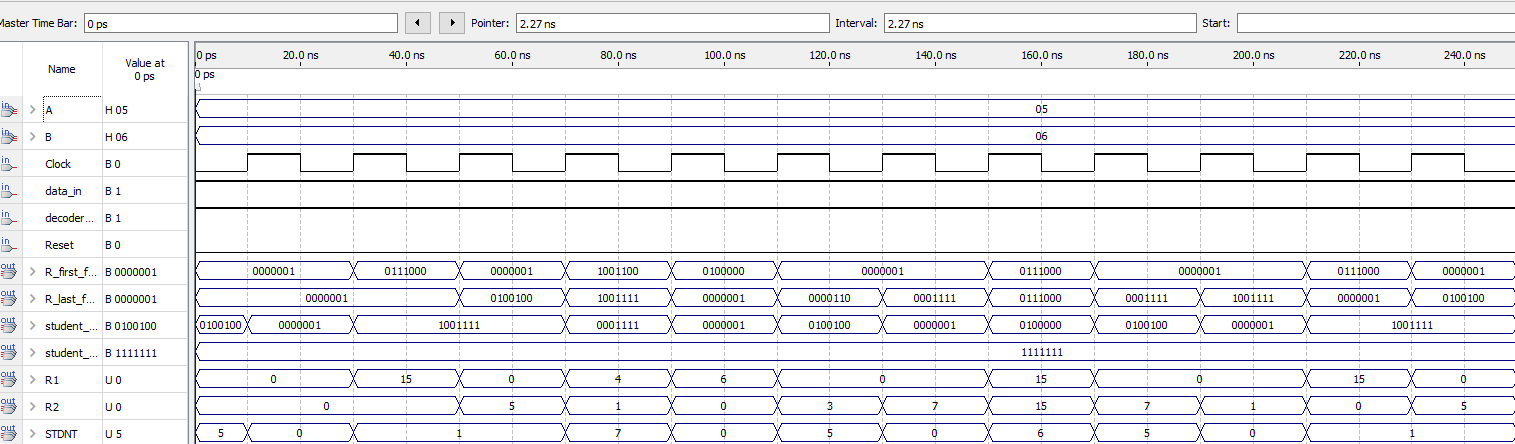
The ALU component in this problem set has the same design as the one in the previous problem set. In this problem set, the assignment of a distinct set of boolean functions to inputs A and B that correspond to the allocated microcode is the aim of the ALU component. For instance, in problem set 1, "Sum of A and B" is displayed by the boolean operation for microcode 0000000000000001. The boolean operation for the same microcode in problem set 2, given the assigned case 'a,' is "Increment A by 2".



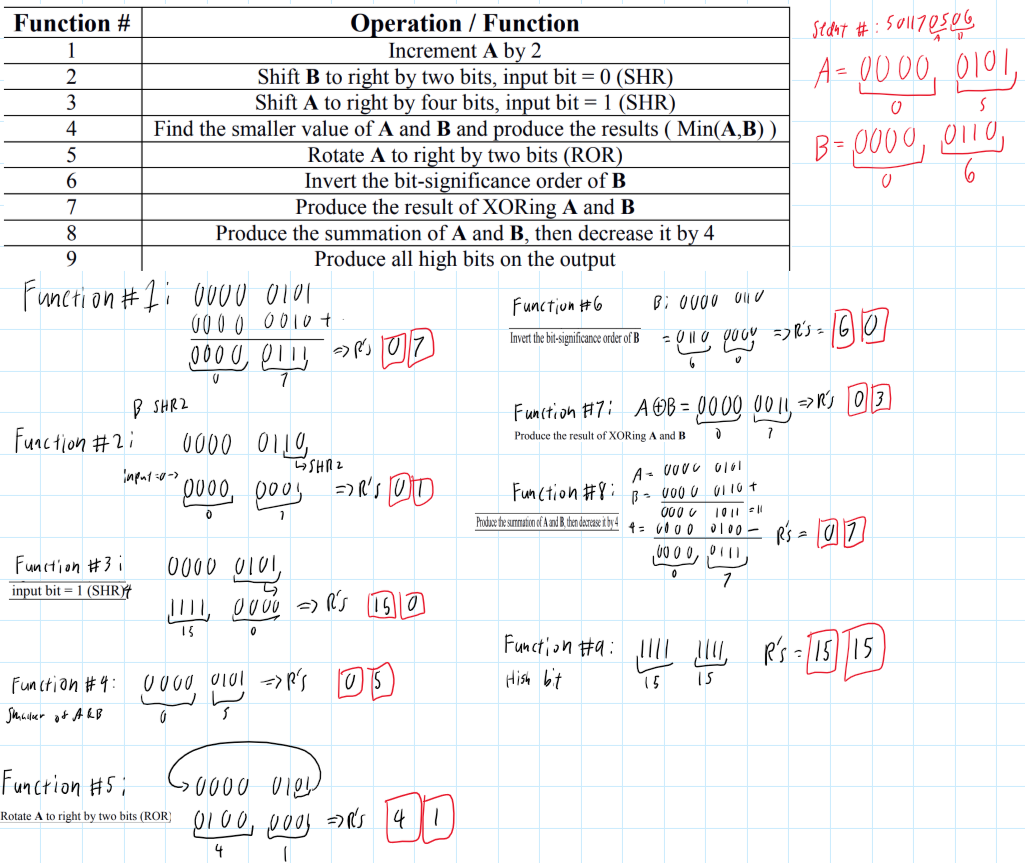
***Figure 3.2.1:***  Block Schematic for Problem 2

| **Function #** | **Microcode** | **Boolean Operation / Function** |
| --- | --- | --- |
| 1 | 0000000000000001 | Increment A by 2 |
| 2 | 0000000000000010 | Shift B to right by two bits, input bit = 0 (SHR) |
| 3 | 0000000000000100 | Shift A to right by four bits, input bit = 1 (SHR) |
| 4 | 0000000000001000 | Find the smaller value of A and B and produce the results ( Min(A,B) |
| 5 | 0000000000010000 | Rotate A to right by two bits (ROR) |
| 6 | 0000000000100000 | Invert the bit-significance order of B |
| 7 | 0000000001000000 | Produce the result of XORing A and B |
| 8 | 0000000010000000 | Produce the summation of A and B, decrease it by 4 |
| 9 | 0000000100000000 | Produce all high bits on the output |

***Table 3.2.2:*** ALU Core Operations for Problem 2



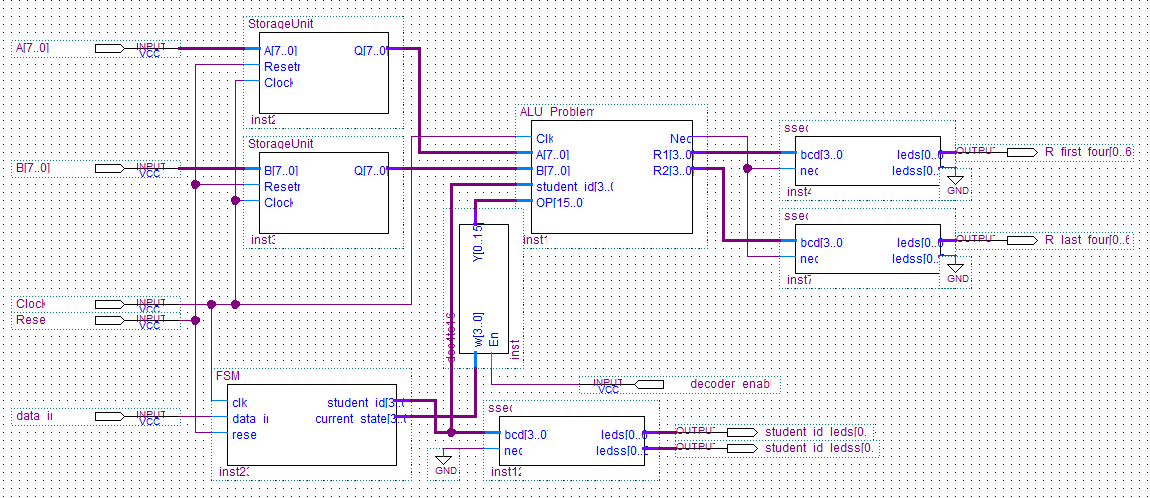
***Waveform 3.1.3:*** Problem 2



***Image:*** Proof of Problem Set 2

## 3.3 Problem Set 3: Modified Control Unit (FSM) (Assigned a)

In problem set 3, the assignment was to change the ALU so that it could determine if the student number's digits were odd or even more than nine clock cycles. On a single seven-segment display, "y" is shown if the student number's digit is odd, and "n" is shown if it is even. In order to accomplish this, the ALU code was expanded to include conditional statements and the input "student\_id." Rather than performing a boolean function on the current state, which is sent to the ALU between A and B, the ALU only verifies if the input value "student\_id" is odd or even.

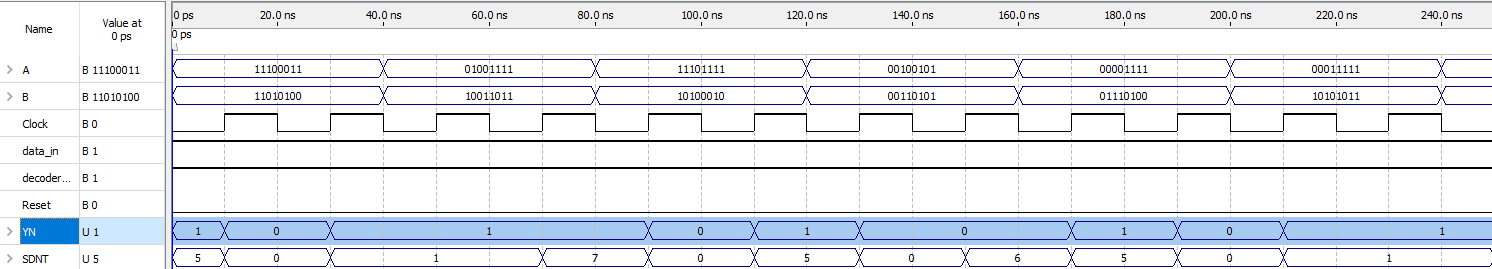


***Figure 3.3.1:***  Block Schematic for Problem 3

Regarding the function of the inputs and outputs, while they are not very important in Problem Set 3, inputs "A" and "B" indicate the intended values based on the final four digits of the student number. The student number to be tested for odd or even is chosen by the input "OP." Finally, "Clock" gives the duration of each output's active state. The seven-segment display can show either "y" or "n" when output "R1" is utilized. As previously stated, on a single seven-segment display, "y" is displayed if the student number's digit is odd, and "n" is displayed if it is even.

| **Function #** | **Microcode** | **Boolean Operation / Function** |
| --- | --- | --- |
| 1 | 0000000000000001 | 5/y |
| 2 | 0000000000000010 | 0/n |
| 3 | 0000000000000100 | 1/y |
| 4 | 0000000000001000 | 1/y |
| 5 | 0000000000010000 | 7/y |
| 6 | 0000000000100000 | 0/n |
| 7 | 0000000001000000 | 5/y |
| 8 | 0000000010000000 | 0/n |
| 9 | 0000000100000000 | 6/n |

***Table 3.3.2:*** ALU Core Operations for Problem 3 (odd/y and even/n)



***Waveform 3.3.3:*** Problem 3

# 4. Conclusions

This lab's goal was to build basic general-purpose processors, with three sets of challenges outlining the necessary specifications. The goal was to develop several ALU cores that could conduct boolean functions and modifications on inputs "A" and "B" and show the results on a seven-segment display in order to solve these problem sets. This lab assessed concepts covered throughout the semester because it was the last one. The final designs utilized in this lab incorporate elements from earlier laboratories, including the decoder, finite state machine, and seven-segment display. However, it also included new parts, such as the latches. It was necessary to fully understand how each component functions and interacts with the others in order to finish this lab.