Student Name:Yuner Zhang

Student ID:20012100061 H00365046

Programme:B38DB

University:Xidian university

Year:Year 2

Submission Date:20/11/2021

B38DB Digital Design & Programming

Lab 5 – Design of ALU with Seven Segment Display

Part 1: Introduction

In this lab, you are going to learn how to design an ALU with a seven-segment display output. An arithmetic logic unit (ALU) is a part of a CPU that carries out arithmetic and logic operations. A seven-segment display is an electronic component used to display decimal numerals. It allows us to read the results easily by converting the binary results into decimal form. You are going to design an ALU that accepts two 8-bit inputs (A and B) and perform up to 8 different functions based on the operation code (select line of multiplexer).

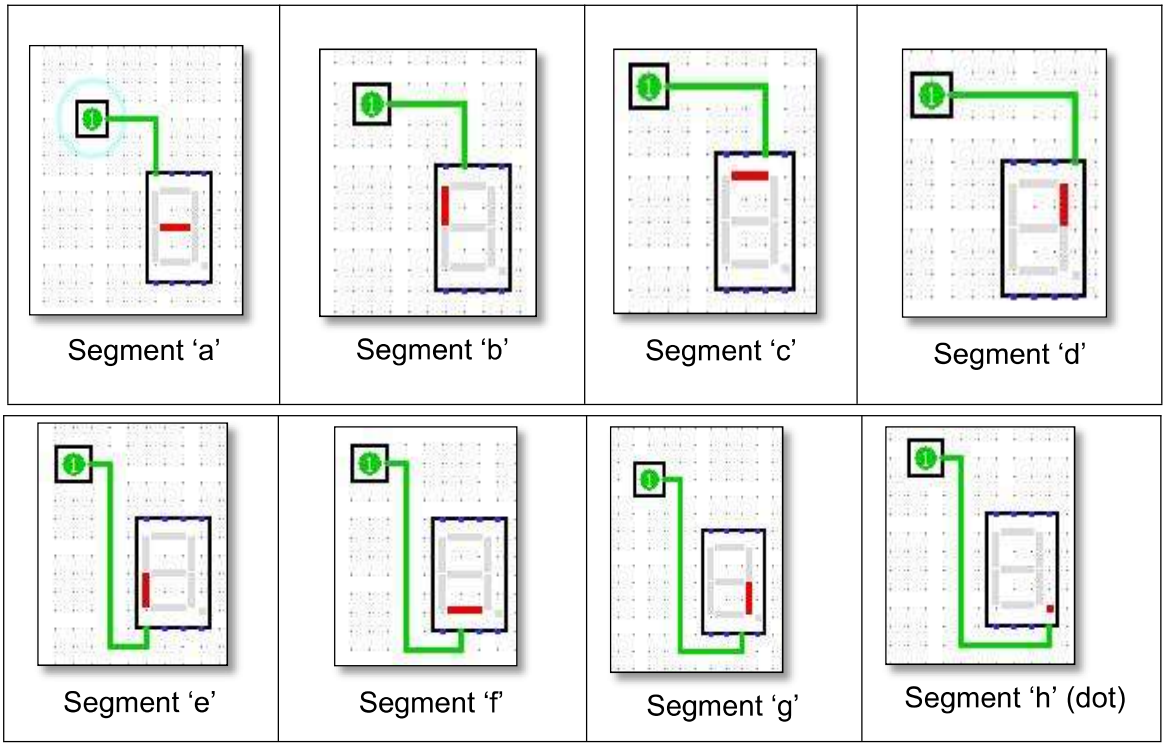
Part 2: The Seven Segment Display in Logisim

The seven-segment display in Logisim is used to simulate LEDS that are arranged in a rectangular fashion to display numbers. By default, the seven-segment display is set to “active-high” mode. This means that anytime an input is high, the segment corresponding to that input node is lit. Figure 1 showcases the seven-segment display when it is unlit.

Figure 1: Seven-segment display in Logisim

The seven-segment display consists of 8 input nodes. Each input node is in blue as can be seen in Figure 1. Each input node corresponds to a segment. The following list showcases which segment corresponds to which input node. You can use this list in Table 1 for your reference when designing circuits that make use of the seven-segment display.

Table 1: List of seven segment display input nodes and their corresponding lit segment



You can identify the number of segments that need to be on (1) and the segments that need to be off (0) to showcase various numbers from 0-9 and “E” when there is an error (number higher than 9 within one display).

Task 1: With reference to the position of seven segments shown in Table 1 complete Table 2 for the numbers 0 to 9 and Character “E” (error).

Character	h	g	f	e	d	c	b	a	Hex
0	0	1	1	1	1	1	1	0	7E
1	0	1	0	0	1	0	0	0	48
2	0	0	1	1	1	1	0	1	3D
3	0	1	1	0	1	1	0	1	6D
4	0	1	0	0	1	0	1	1	4B
5	0	1	1	0	0	1	1	1	67
6	0	1	1	1	0	1	1	1	77
7	0	1	0	0	1	1	0	0	4C
8	0	1	1	1	1	1	1	1	7F
9	0	1	1	0	1	1	1	1	6F
E	0	0	1	1	0	1	1	1	37

You are not expected to design the seven-segment display section from scratch. Instead, a template circuit will be given to you. This can be found on Vision as "*Lab 5 Template.circ*". Figure 2 shows a flowchart which shows how the data flows from the ALU to the seven-segment display. Figure 3 shows the seven-segment display portion of the circuit. You will be designing a portion of the seven-segment decoder. The decoder takes the ALU result as an input and splits it into three numbers to represent the hundreds, tens, and units place of a number.

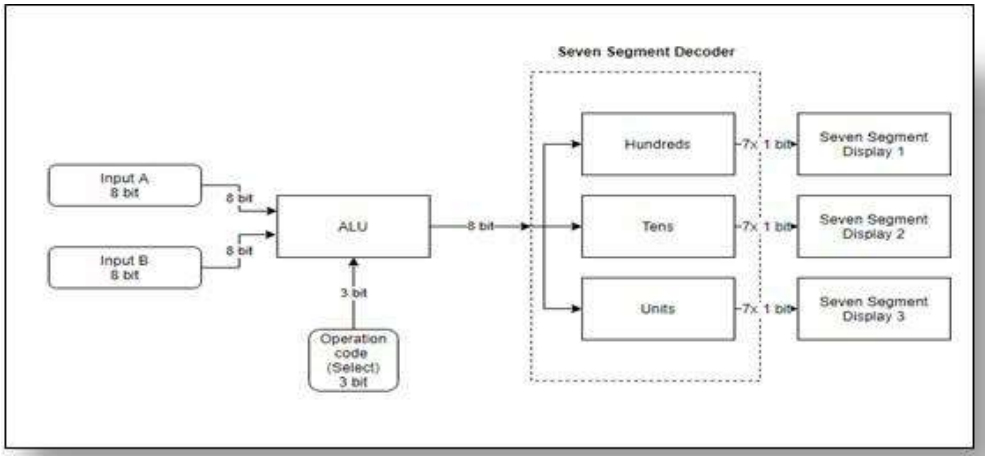


Figure 2: Data flowchart from input to seven segment output

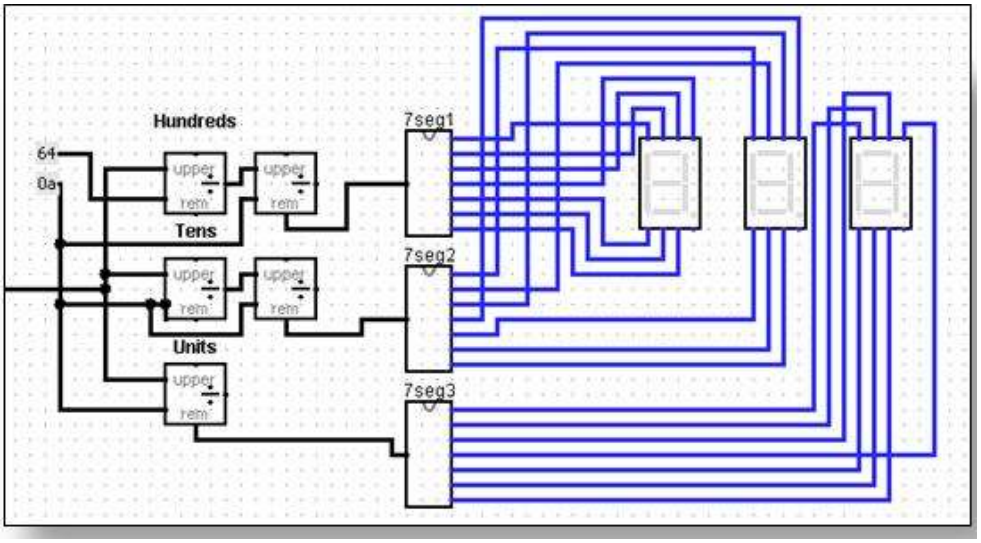


Figure 3: Seven Segment Display Section

Task 2: What is the highest number the seven-segment display with 3 digits can show?

Answer:255.

Inside the given circuit template, you can find the incomplete 7-segment decoder circuit within the object browser under the name “7segdecoder”. Figure 3 showcases this decoder.

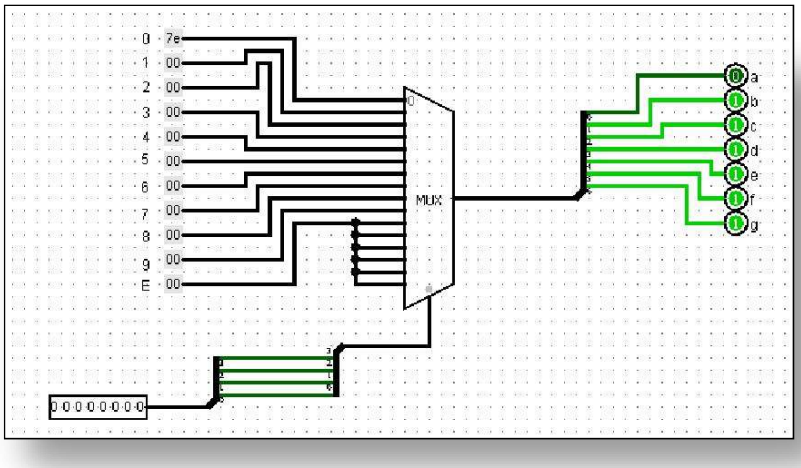
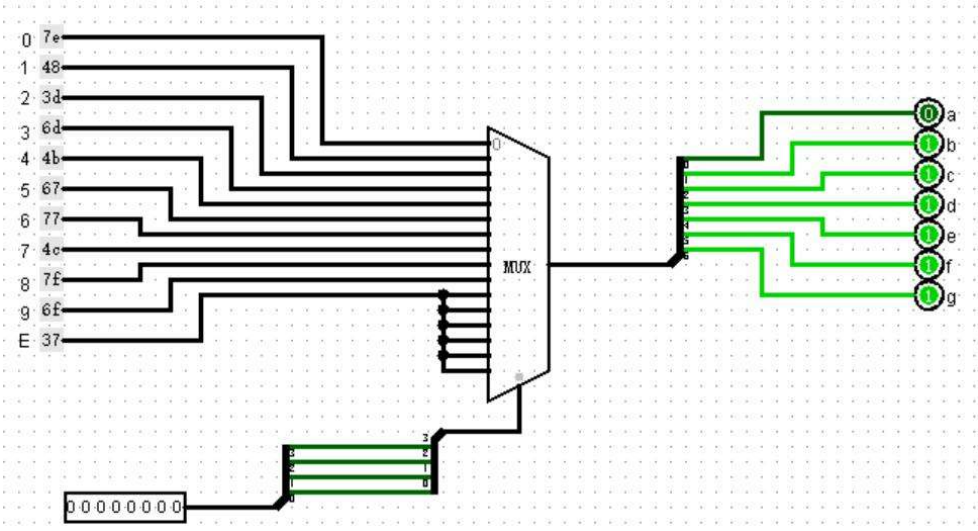


Figure 4: Seven segment display decoder (Incomplete)

Replace the hexadecimal constant values for 0-9 & E (inputs of MUX) using completed Table 2.

Task 3: Screenshot of completed seven segment display decoder:



Part 3: Completing the 8-function ALU

Now that the seven-segment display section is completed. You are to design 8 functions for the ALU section. Figure 4 highlights the incomplete ALU section of the circuit.

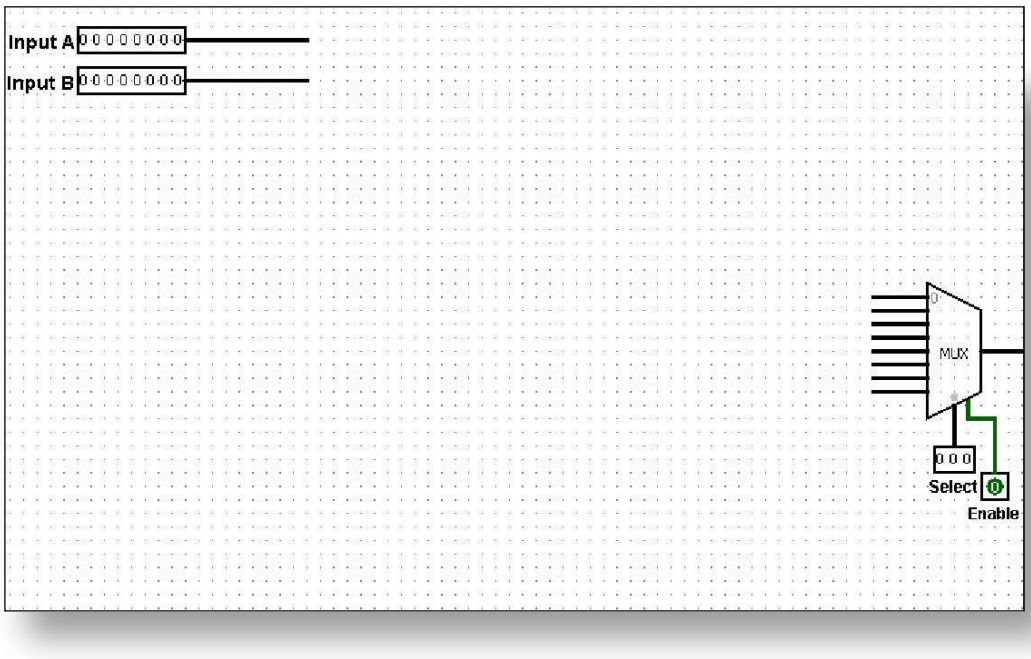
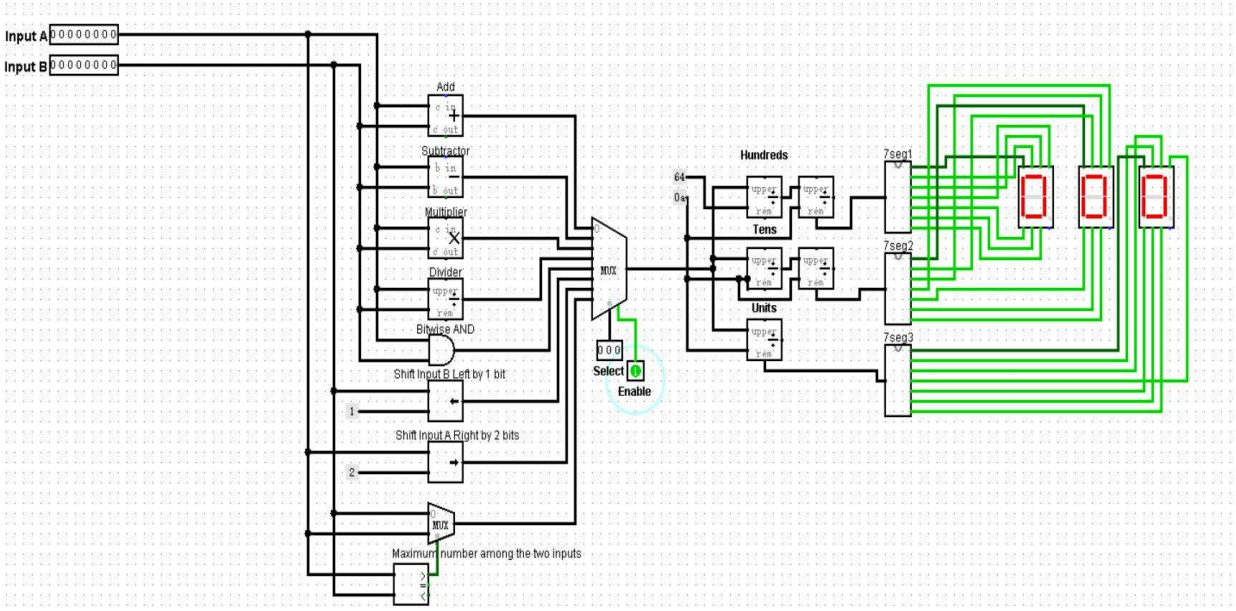


Figure 5: Incomplete ALU section

You are required to use the given empty space to implement 8 different functions. Each of these functions need to feed into one of the inputs of the MUX (from top to bottom). These functions are (when testing the functions, ensure the mux enable is set to 1):

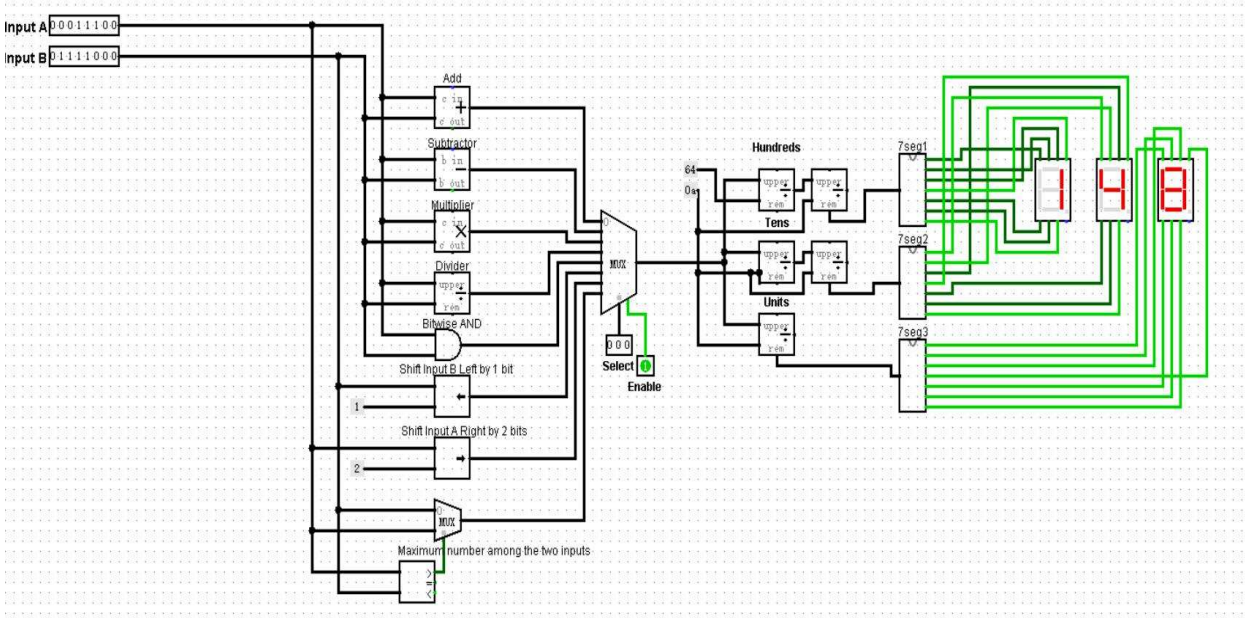
1. Adder
2. Subtractor
3. Multiplier
4. Divider
5. Bitwise AND
6. Shift Input B Left by 1 bit
7. Shift Input A Right by 2 bits
8. Maximum number among the two inputs

Task 4: Screenshot of completed circuit with all 8 functions clearly labeled:

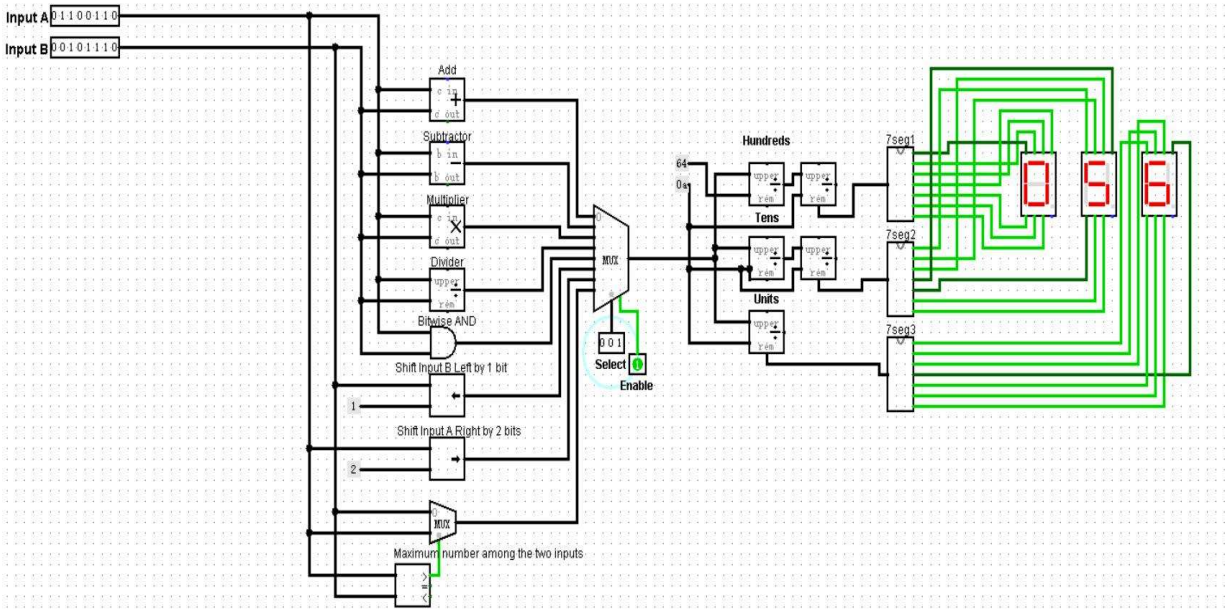


You may show the following screenshots in two separate pictures (one for ALU section and one for seven segment section) for more clarity.

Task 5: Screenshots of adder result with the following inputs, A = 28_{10}, B = 120_{10}

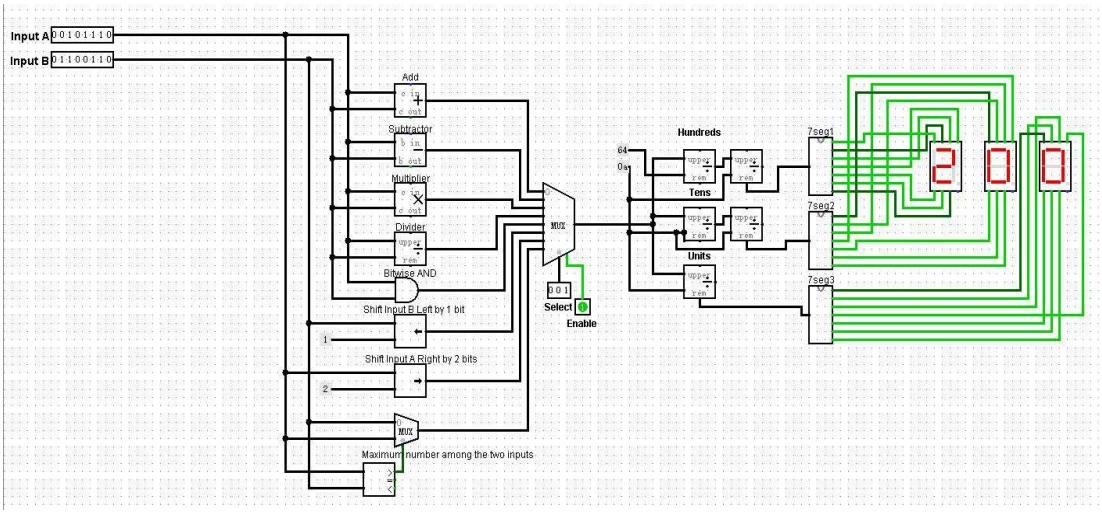


Task 6: Screenshots of subtractor result with the following inputs, A = 102_{10}, B = 46_{10}



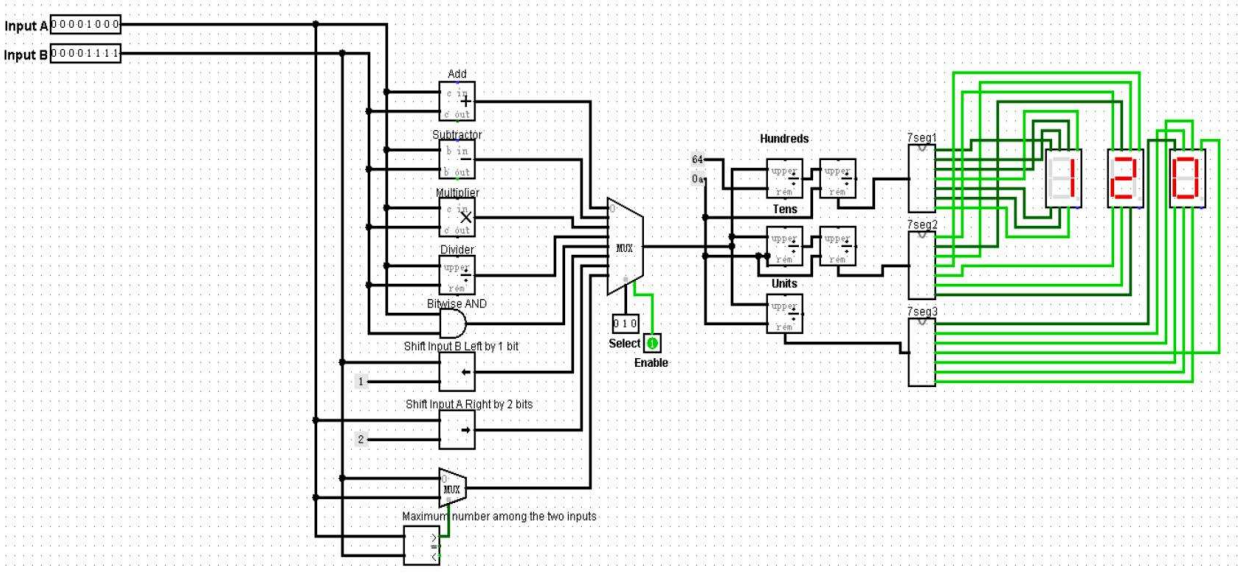
Task 7: With reference to Task 6, **what would be the result if A = 46 and B = 102? Comment on your result**

Answer:



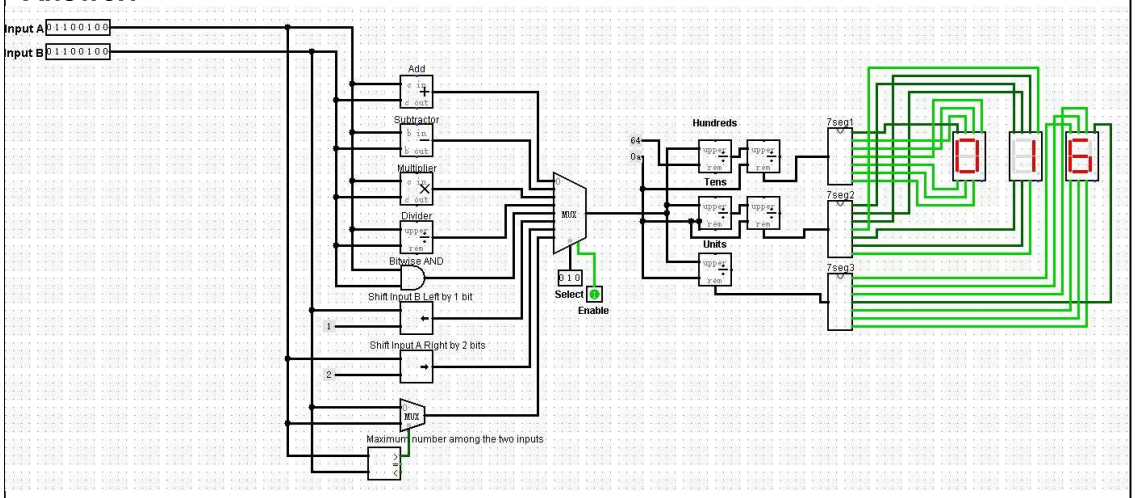
Because 46 minus 102 is a negative number, minus 56. But the last eight bits of -56 in binary are 1100,1000,which is the same result as the positive number 200. So the answer is equal to 200.

Task 8: Screenshots of Multiplier result with the following inputs, A = 8_{10}, B = 15_{10}



Task 9: **(a) What would be the result if A = 100 and B = 100? Comment on your result**

Answer:



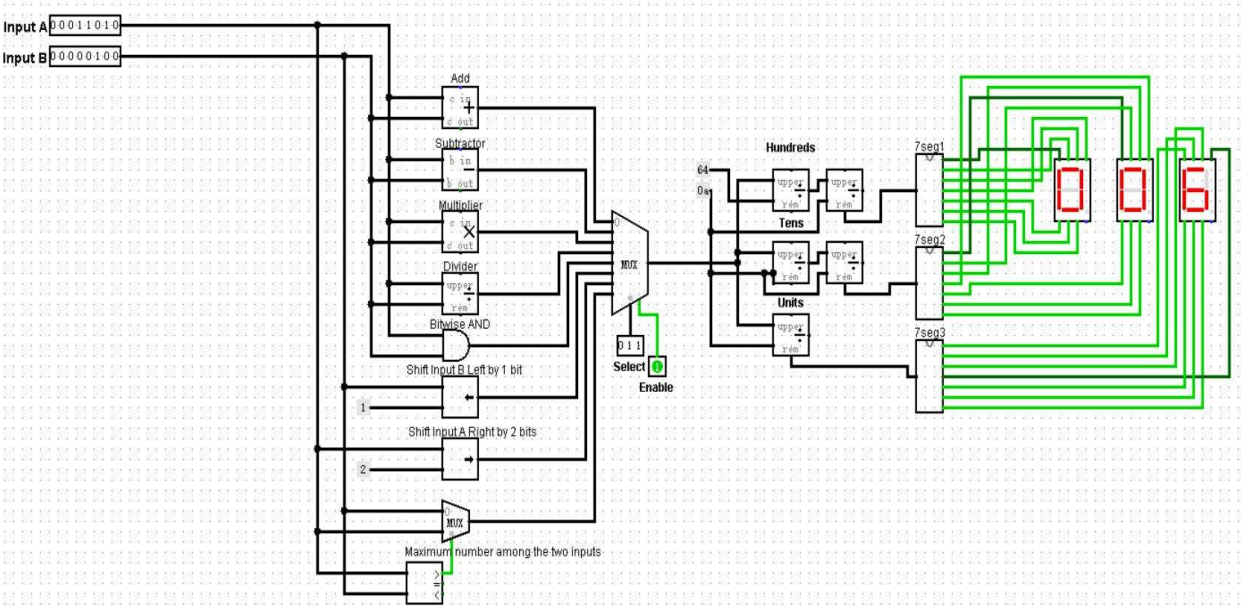
Because 100 multiply 100 is 10000,which in binary is 10 0111 0001 0000. But the last eight bits of 10000 in binary are 0001 0000,which is the same result as the positive number 16. So the answer is equal to 16.

(b) Give a possible suggestion on how you could incorporate higher results.

Answer:

Add more 7-segment decoders to represent higher bits.

Task 10: Screenshots of Divider result with the following inputs, A = 26_{10}, B = 4_{10}



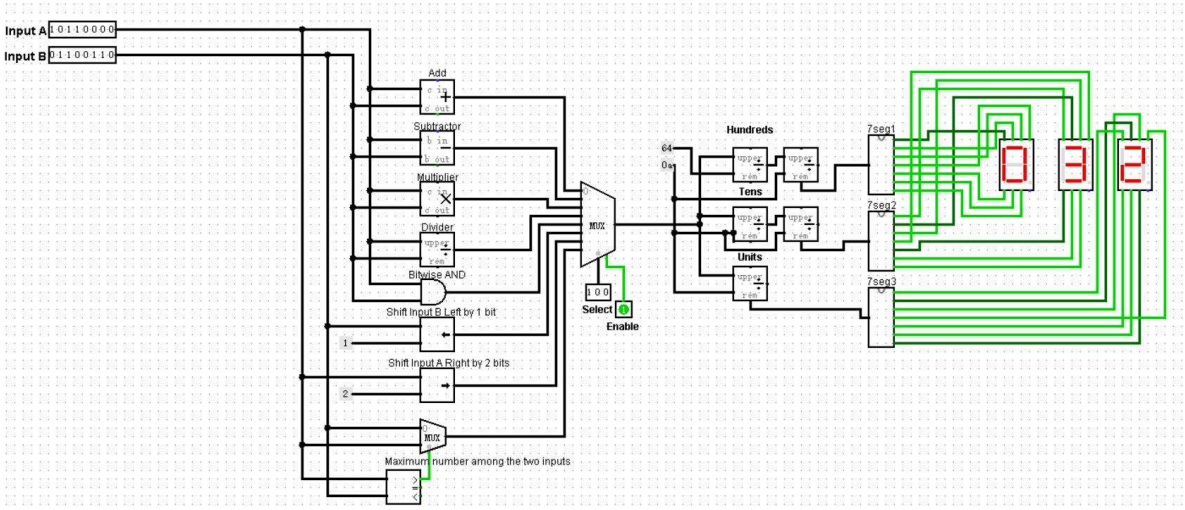
Task 11: What should the actual result be? Comment on the result you obtained in the screenshot.

Answer:

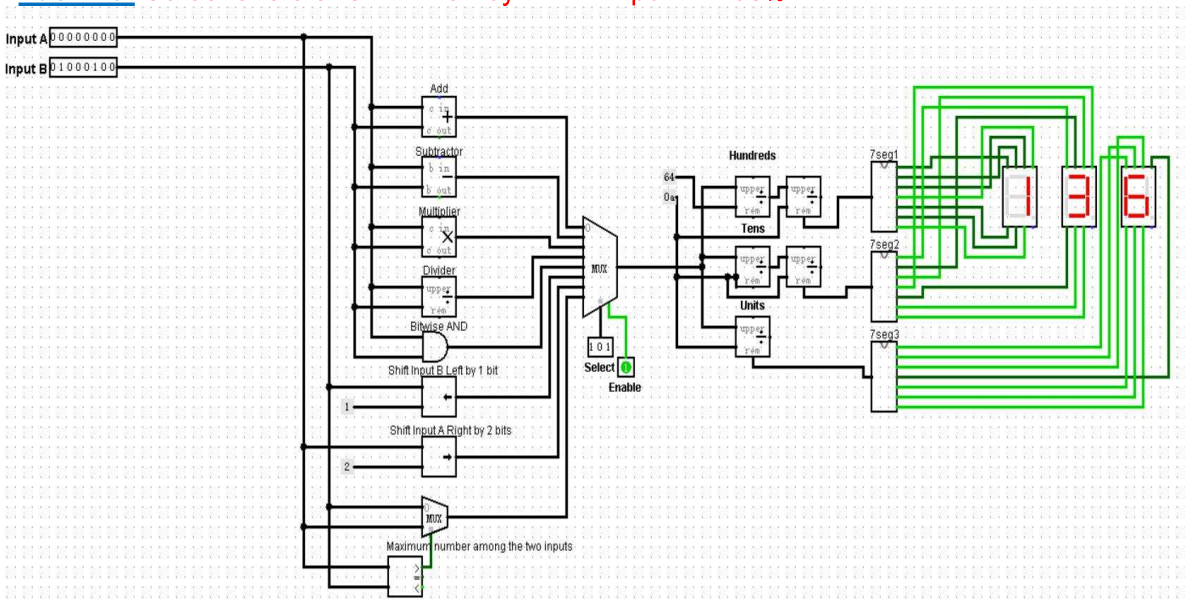
The actual result should be 6.5.

The 7-segment decoder can only read the last eight bits of the integer, so the final result is 6.

Task 12: Screenshots of Bitwise AND function with A = 176_{10} B = 102_{10}



Task 13: Screenshots of Shift B left by 1 with input B = 68_{10}

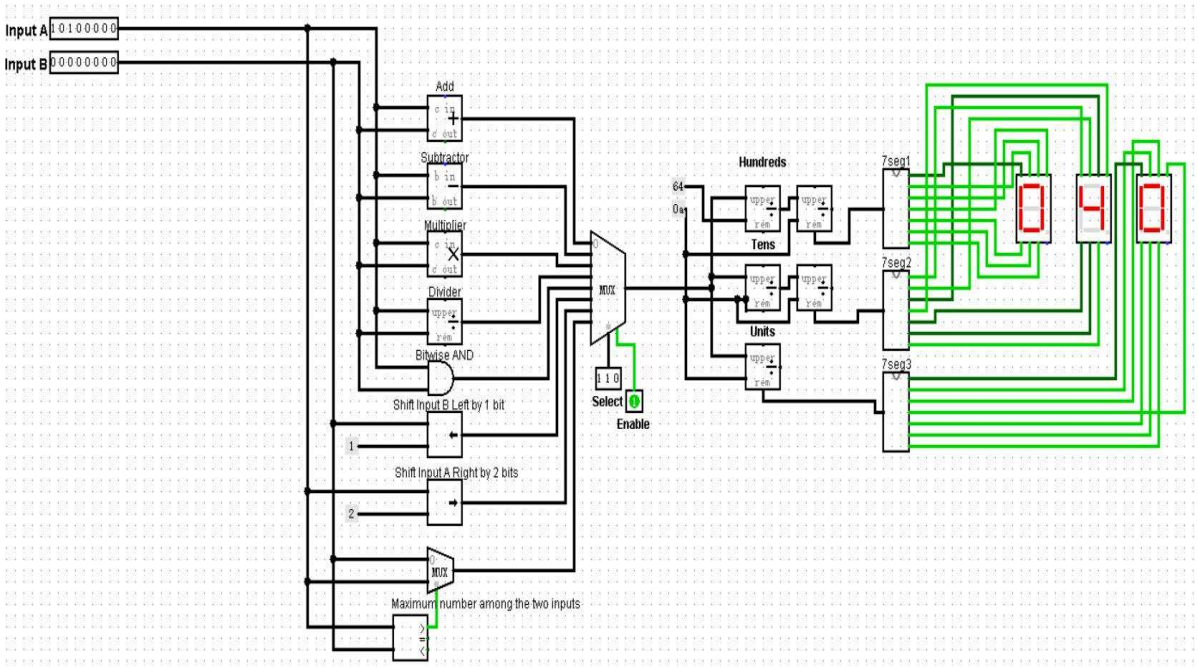


Task 14: What is this shift function equivalent to arithmetically?

Answer:

The shift to the left by one is the arithmetical equivalent of multiplying the original value by two.

Task 15: Screenshots of Shift A Right by 2 with input A = 160_{10}

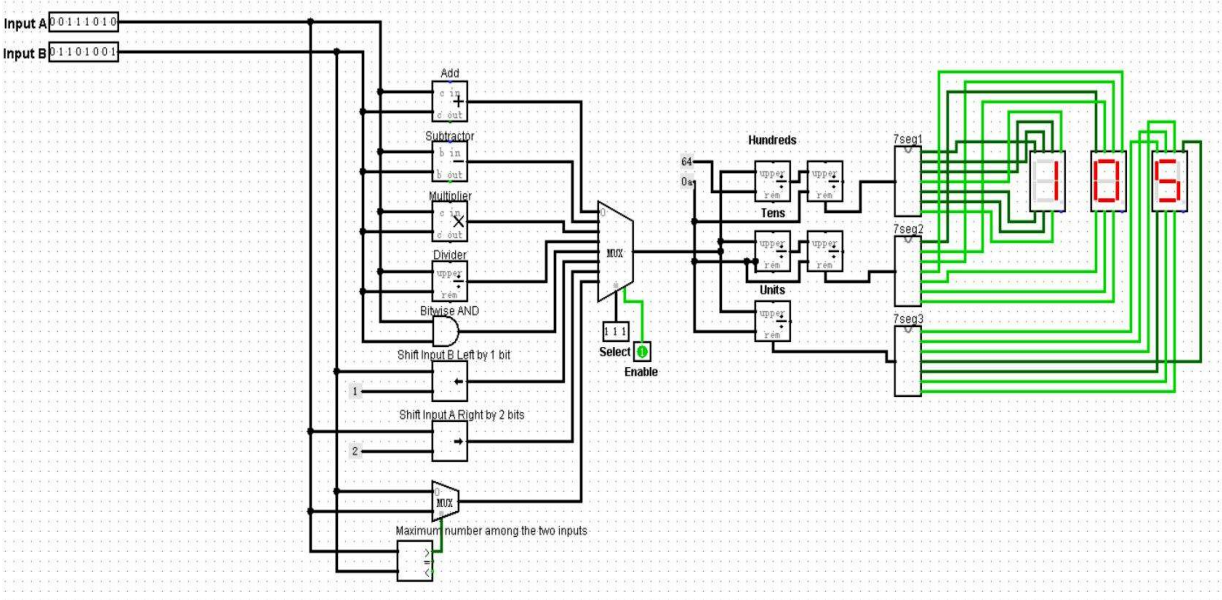


Task 16: What is this shift function equivalent to arithmetically?

Answer:

The shift to the right by one is the arithmetical equivalent of dividing the original value by two.Hence,the shift to the right by two is the same as dividing by four.

Task 17: Screenshots of higher value between both inputs A and B with inputs A = 58_{10} , B = 105_{10}



Part 4: Conclusion

In this lab, you have learnt how a seven-segment display functions and how a decoder functions to light the correct segments based on the character input. You have also learnt how to design an 8-function ALU which successfully carries out various functions. Moreover, you have also learnt about the design limitations of having a certain bit length for your design and how you could possibly tackle such problems in future designs.