WANG, Tiffany 260684152
YE, Frank 260689448
GROUP 08 - MacroHard EE

G08 MODULO 13 CIRCUIT

OBJECTIVES

The goal of the first half of the lab is to get familiar with the Quartus design and simulation tools by creating a 7-bit comparator circuit. The second half requires designing, building and testing of a Modulo 13 circuit given a 6 bit input. Design specifications are given in the form of the following equation:

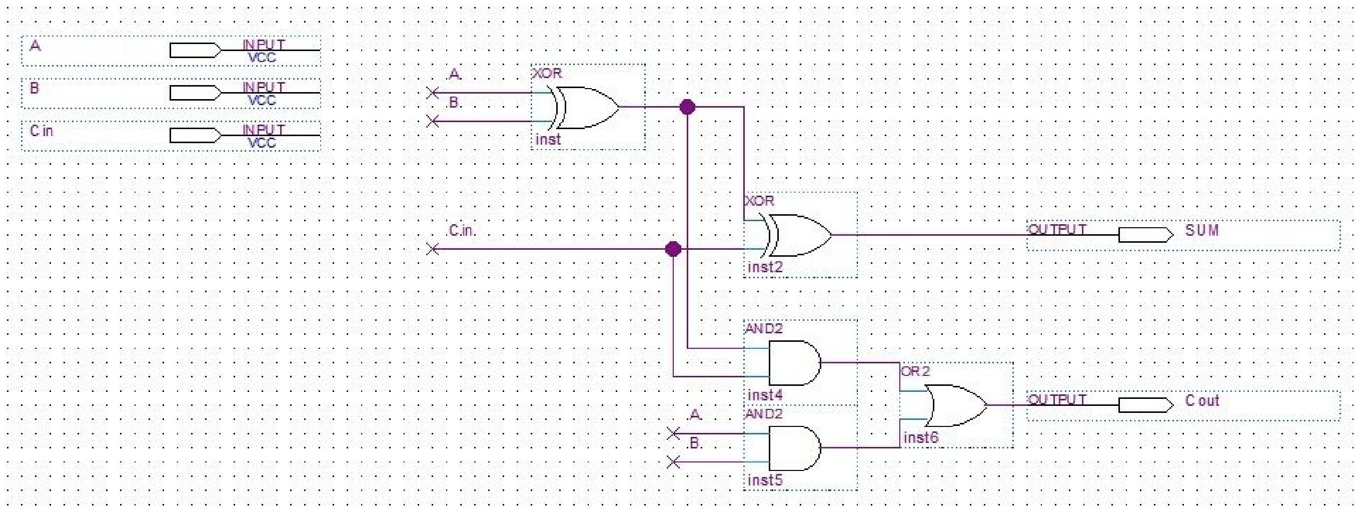
$$A\%13 = A - floor(A*5/64) *13$$

From the expression above, the A%13 circuit is achieved using various sub-circuits (e.g. adders, multipliers, shift registers). In the following report, we will present and justify our design, and prove that it behaves as desired using simulation plots.

TABLE OF CONTENT

1. CIRCUIT DESCRIPTION

a. 1-bit adder



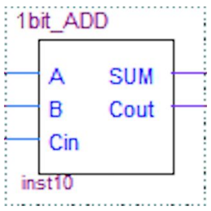
SUM = A ⊕ B

Cout = A • B + (A ⊕ B) • Cin

INPUTS: A, B and Cin

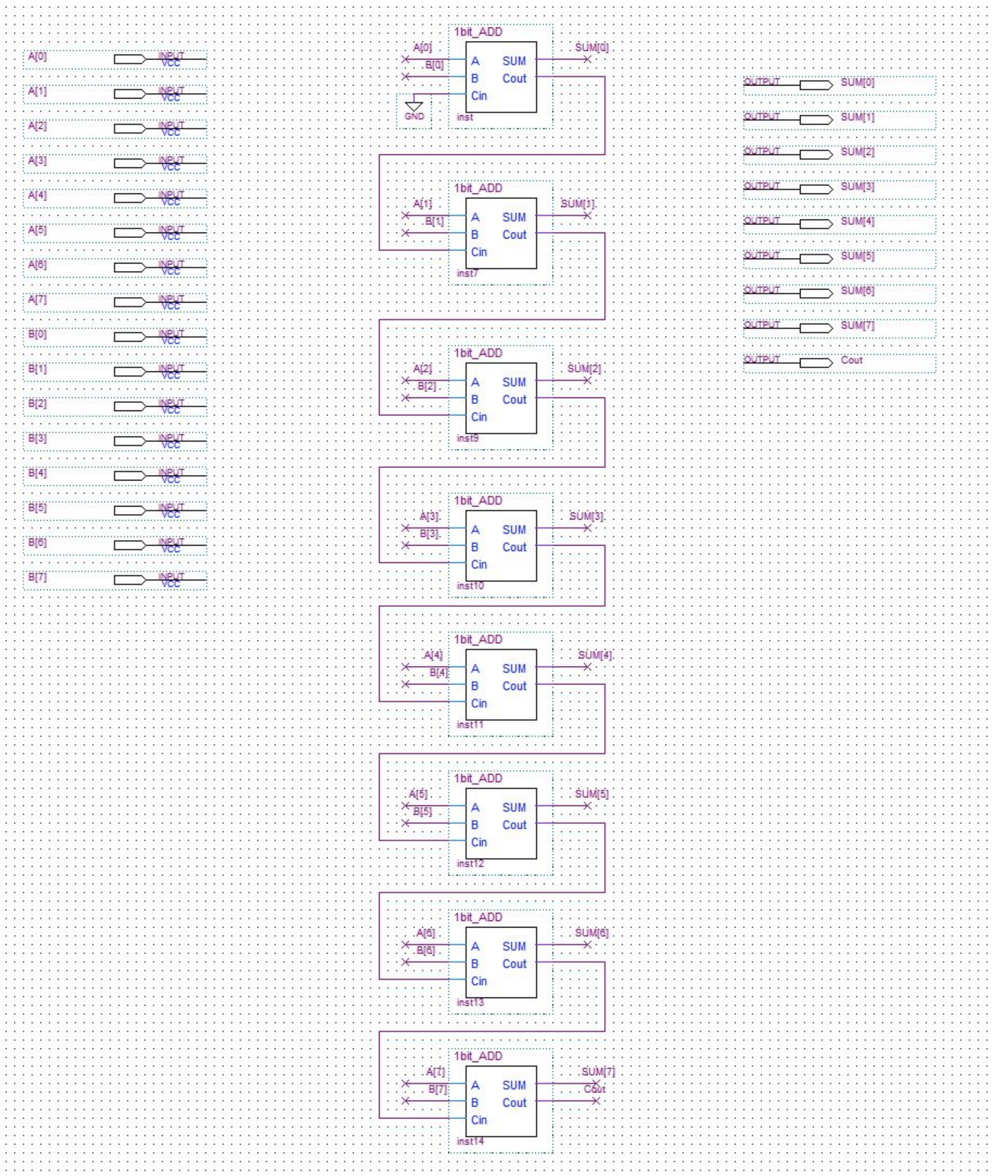
OUTPUTS: SUM, Cout

Symbol diagram:



1-bit adder circuit taking A, B, Cin as 1-bit inputs. It is a simple arithmetic circuit performing the addition of the two inputs, and outputs the SUM. If A+B > 1, then the carry out (Cout) has a value of 1. (NOTE: Cin was added as an input for modularity, used when cascading 1-bit adder to create larger adder/multiplier circuits.

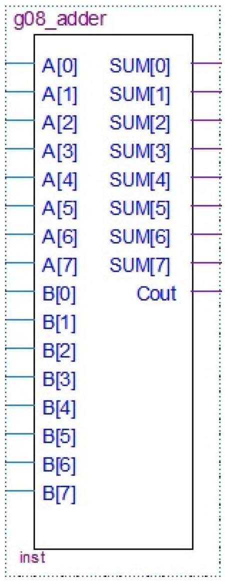
b. 8-bit adder



SUM = A + B

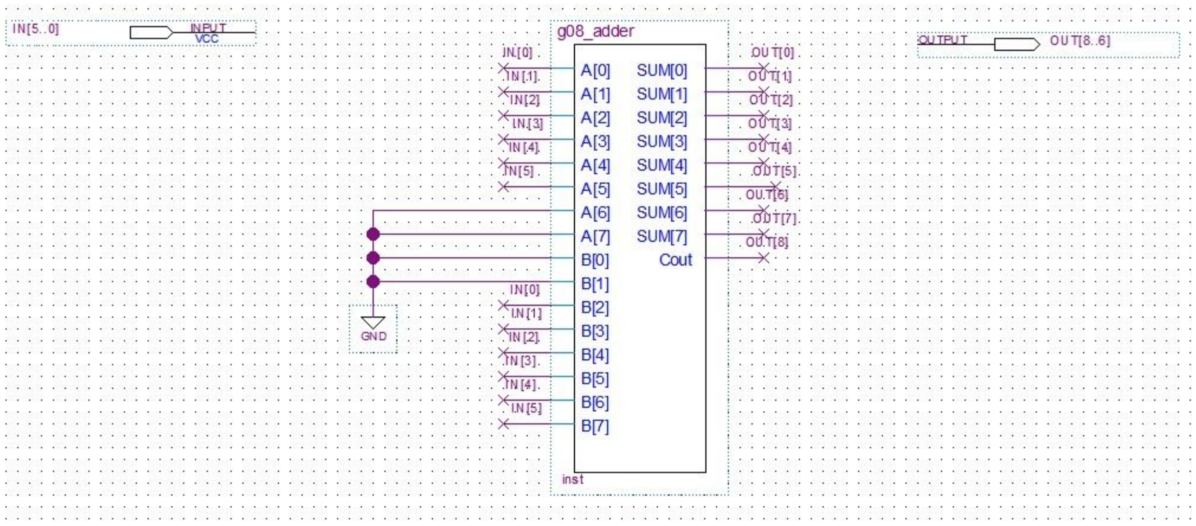
INPUT: A[7..0] and B[7..0]
OUTPUT: SUM[7..0] and Cout

Symbol diagram:



8-bit adder implementing the 1-bit adder using a ripple-carry design. The circuits takes two 8-bit vector inputs and outputs the addition as a 9-bit number (Cout = 1 in case of overflow). The circuit is also reused for multiplication circuits by 101_2 and 1101_2 , as well as in the 2C circuit.

c. Floor-13

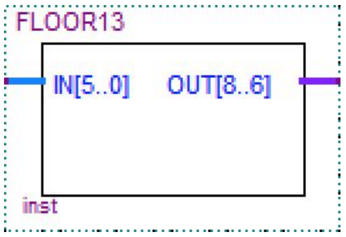


IN[5..0]
INPUT VCC
g08_adder
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7]
B[0] B[1] B[2] B[3] B[4] B[5] B[6] B[7]
SUM[0] SUM[1] SUM[2] SUM[3] SUM[4] SUM[5] SUM[6] SUM[7] Cout
IN[0] IN[1] IN[2] IN[3] IN[4] IN[5]
OUT[0] OUT[1] OUT[2] OUT[3] OUT[4] OUT[5] OUT[6] OUT[7] OUT[8]
GND
inst
OUTPUT OUT[8..6]

OUT = IN * 101_2

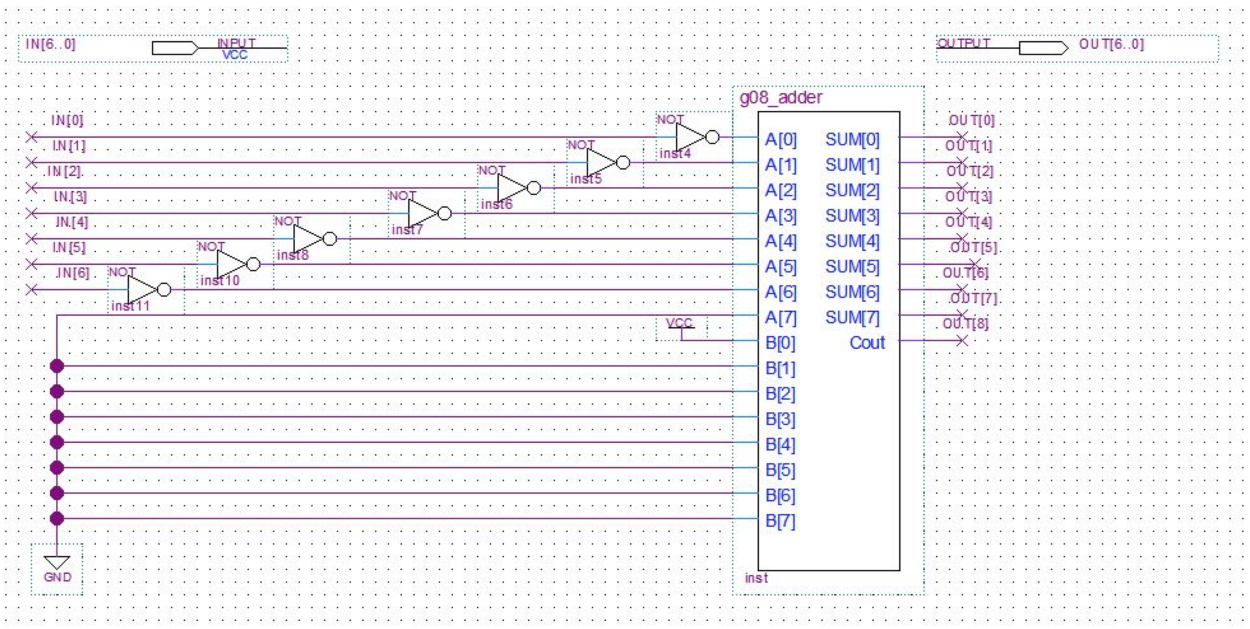
INPUT: IN[5..0]
OUTPUT: OUT[8..6]

Symbol diagram:



This circuit implements the 8-bit adder to perform the floor(A/13) operation through a series of shifting and adding.
Since 1/13 ≈ 5/64, our result will be an approximation of floor(A*5/64) which is obtained by A * 101_2 / 100000.
A[7..0] represents the A*1 six-bit non-shifted input, while B[7..0] represents the A*100 six-bit input that is shifted left by two. The final output is shifted right by six and disregards all decimals, therefore we are only taking into account the three most significant bits OUT[8..6]. Thus, the shifting is performed by simply taking the 3 MSB at the output and carefully connecting the input pins in order to avoid building extra shift-register circuitry.

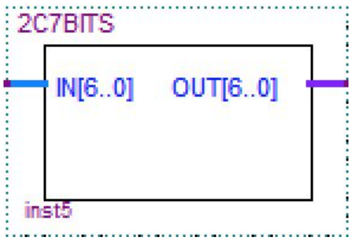
d. 7-bit two's complement



OUT = NOT IN + 1

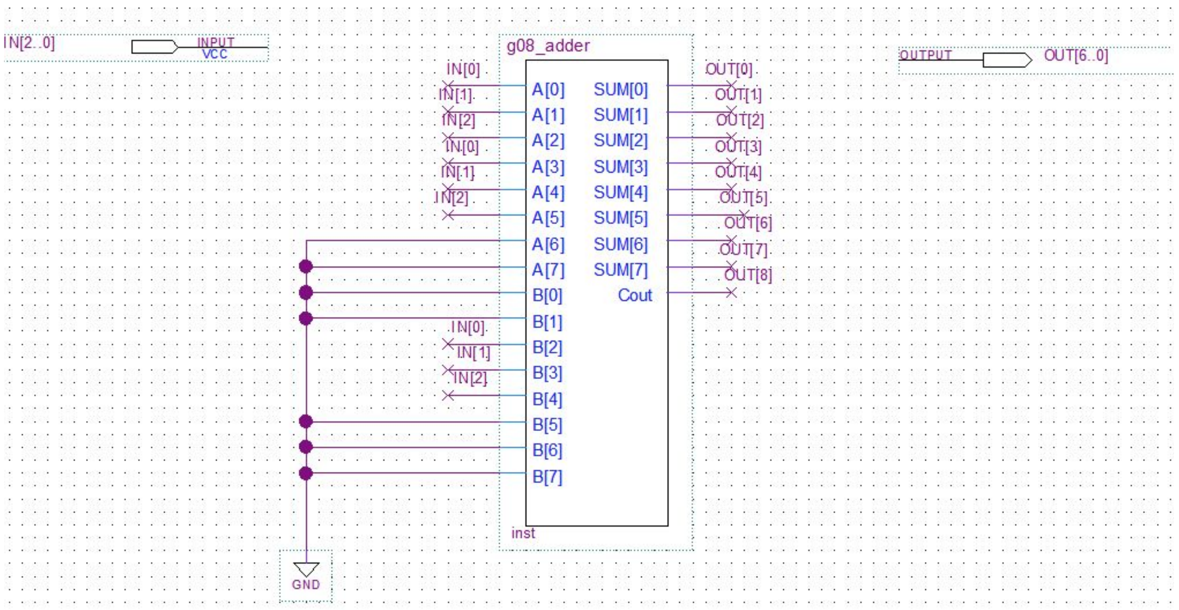
INPUT: IN[6..0]
OUTPUT: OUT[6..0]

Symbol diagram:



This circuit implements the 8-bit adder circuit to perform a two's complement operation. It inverts the 7 bit input values, and adds 1 to it (to convert it to 2C). We use this circuit in order to turn floor(A/13) * 13 into a subtrahend.

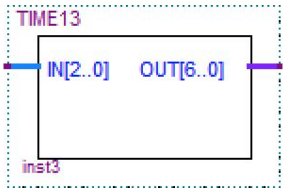
e. TIME13



OUT = IN * 1000 + IN * 100 + IN * 1

INPUT: IN[2..0]
OUTPUT: OUT[6..0]

Symbol diagram:

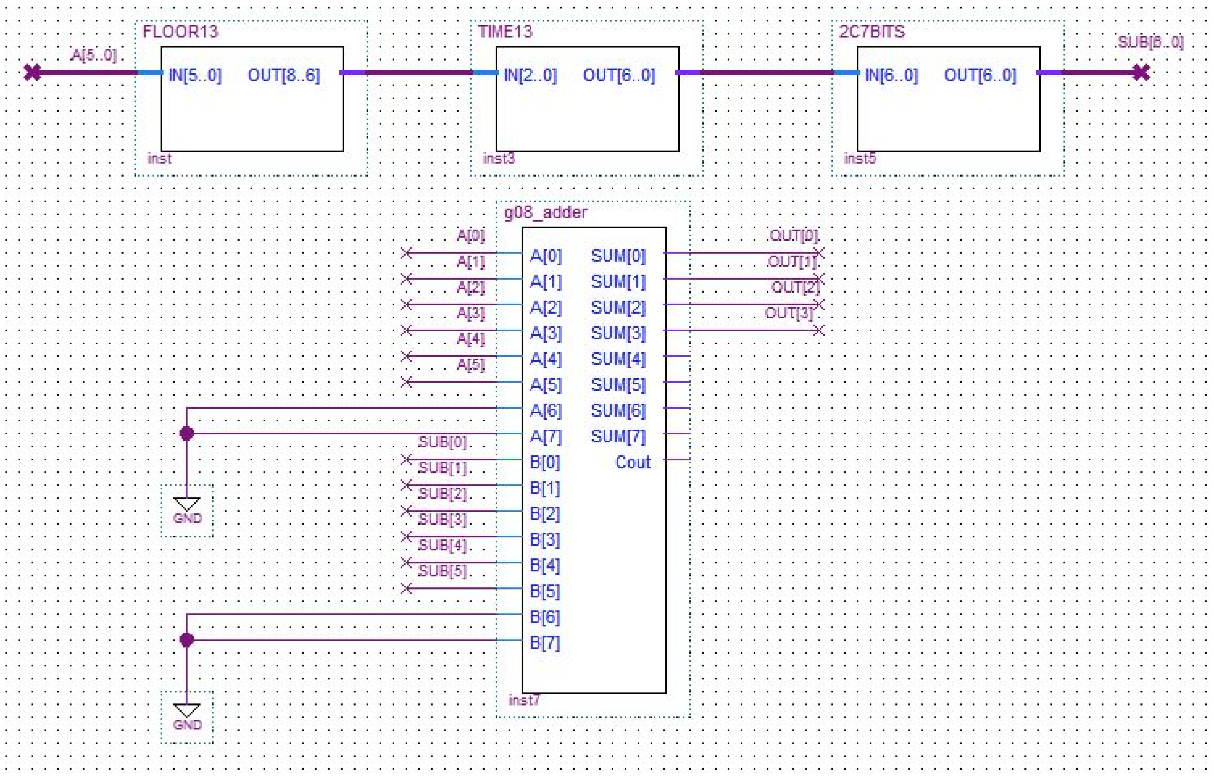


This circuit, similar to the Floor-13 circuit, except that instead we multiply the 3-bit input by 1101_2. The output displays the result as a 7-bit number, in order to account for the largest possible input (111_2). The operation is equivalent to the following:

OUT = left_shift_3 (IN) + left_shift_2(IN) + no_shift(IN)

Similarly to the Floor-13 circuit, the input bits were carefully connected to the input pins of the circuit in order to function accordingly to the equation above, but without extra shift register circuitry.

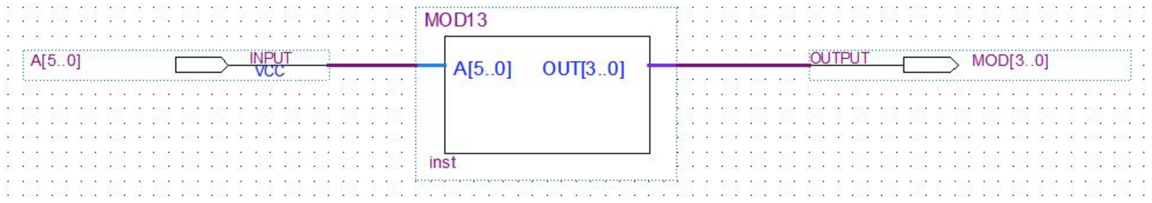
f. Modulo 13 (Top-level entity)



OUT = A - floor(A*5/64) * 13

INPUT: IN[5..0]
OUTPUT: OUT[6..0]

Symbol diagram:



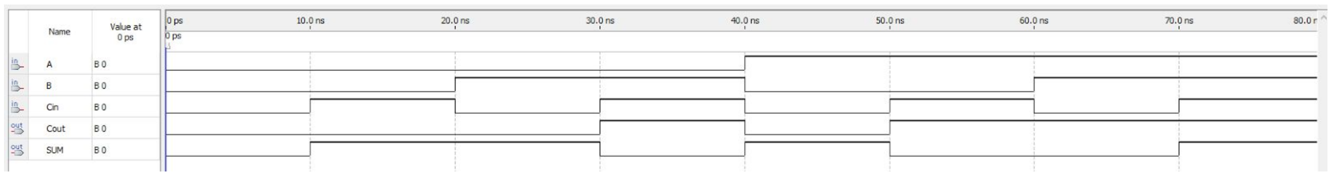
Finally, the Modulo 13 top-level circuit combines the sub-circuits mentioned previously in order to perform the %13 (mod-13) operation. As shown in the schematic, it takes 6-bit input A, takes the floor of it, multiplies the result by 13 uses the multiplied number as the subtrahend in the final 8-bit adder circuit, giving a final 4-bit output.

2. TESTING

a. 1-bit adder

This is the base component of the Modulo 13 circuit, forming the adder and multiplication circuits. The truth table below describes expected behavior.

A	B	Cin	SUM	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

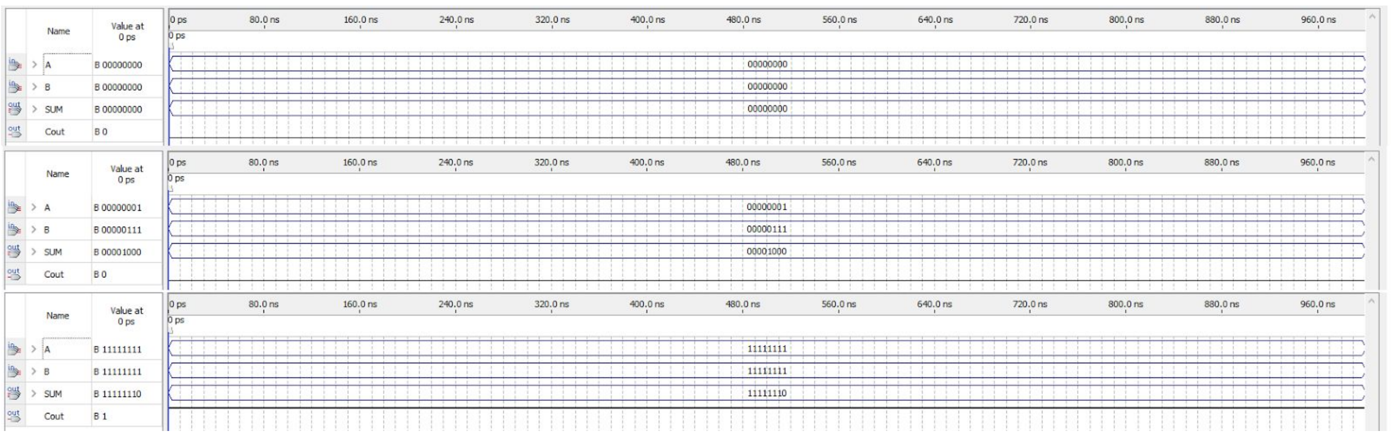


b. 8-bit adder

While the adder is the basic circuit unit in our modulo-13 circuit, the 8-bit adder is the circuit is ubiquitous due to its reuse in the 7-bit 2C, multiplier and floor-13 circuits. Due to the 256 possible input combinations, testing was simplified by choosing specific test cases to evaluate our circuit. Below are the preliminary tests run to verify desired behaviour:

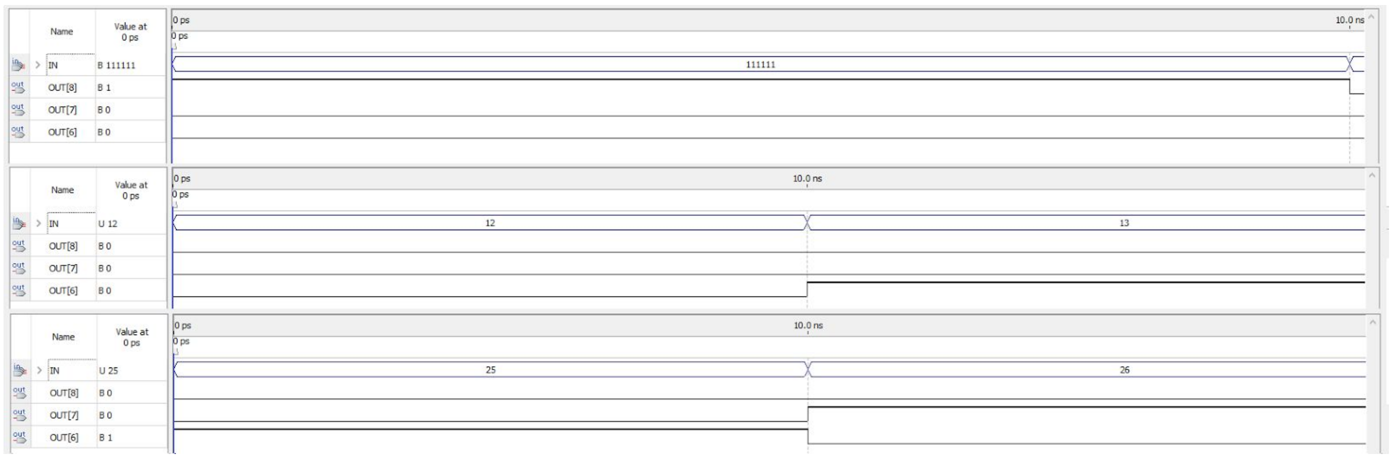
- Base case: 0000000 + 0000000 = 0000000
- Carried over C_out: 00000001 + 00000111
- Largest possible input: 11111111 + 11111111 = 111111110

NOTE: Although the largest possible inputs do not occur in the functional operation of the modulo 13 circuit, testing was still done on these cases in order to keep subcircuits modular for future reuse. Moreover, test-cases for the multiplier, 2C, floor 13 and modulo 13 provide more test-cases for the 8-bit adder circuit, and confirm its functionality.



c. Floor 13

Given the 8-bit adder tests, we focused on the boundary input cases for this circuit. The largest possible input is 111111 which outputs 100. The output is expected to increase by one bit for every multiple of 13, so we tested the behaviour of the circuit when going from 12 to 13, as well as 25 to 26, which is expected to increase from 000 to 001 and 001 to 010 respectively.

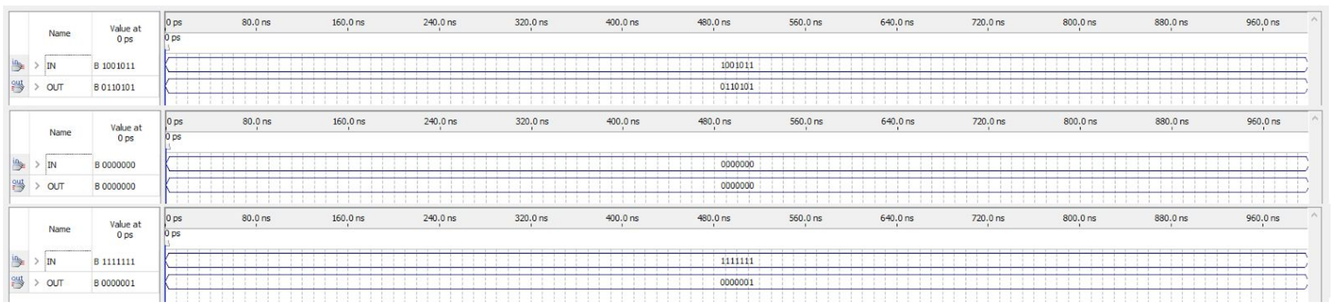


d. 7-bit two's complement

Since the 8-bit adder, 1-bit adder, floor 13 cases have been considered, we considered the boundary cases, as well as one arbitrary case (the inverted inputs do not have to be tested, and we just add 1 to the inverted input, and addition has been tested in the 8-bit adder).

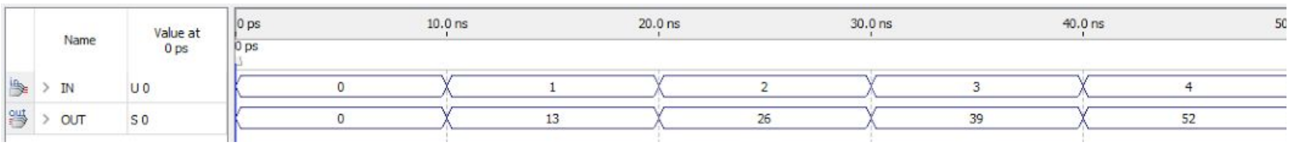
Below are the input test cases:

- 000000 → 000000
- 111111 → 000001



e. TIME13

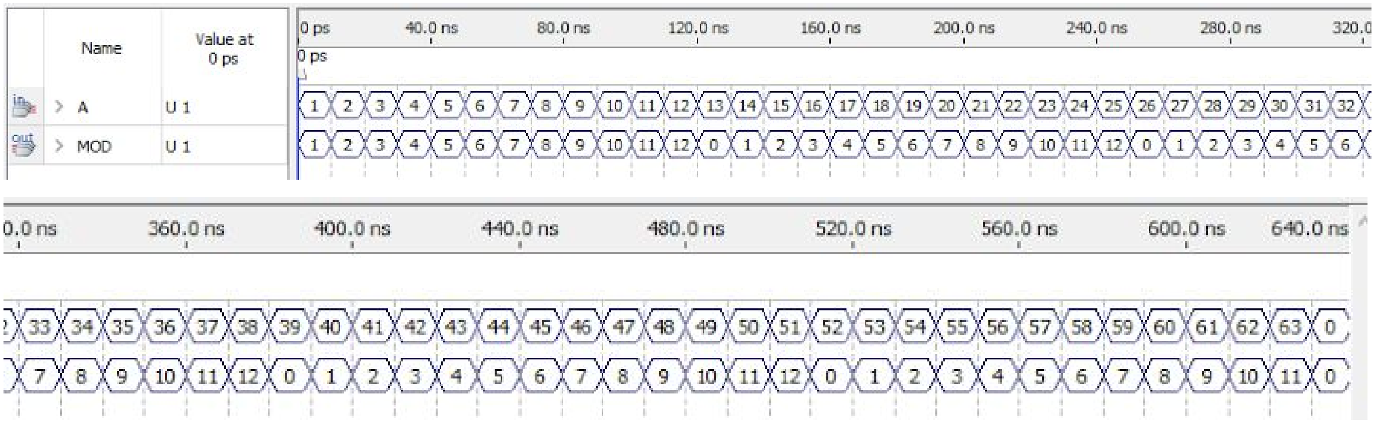
Since multiplication was also based on the 8-bit adder circuit, and that the floor 13 circuit already tested multiplication by 101_2 , a simple arbitrary test value was used to confirm desired behavior.



f. Modulo 13

Modulo 13 is the top-level entity of the lab. If not all possible test cases were covered in the previous circuits, errors would be evident in the modulo 13 circuit. In fact, we ensured that it performed correctly for the full range of input (from 0 to 63) by testing every possible input with the following criteria:

- The output was between 0 and 12 (< 00001101_2)
- As the input was incremented, the output periodically outputted 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 (increasing % 13 values).


Name
Value at 0 ps
A U 1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 0
MOD U 1
1 2 3 4 5 6 7 8 9 10 11 12 0 1 2 3 4 5 6 7 8 9 10 11 12 0 1 2 3 4 5 6 7 8 9 10 11 12 0 1 2 3 4 5 6 7 8 9 10 11 12 0 1 2 3 4 5 6 7 8 9 10 11 0