

BME 393L: Lab 2

4. In lab Procedure

4.1.1 Downloading Digital Circuit on FPGA

The VHDL code from Lab 1, First_circuit.vhd, was successfully downloaded onto the board, and each input combination was verified by testing all possible input combinations and looking at the output states of the LEDRs and LEDGs.

4.1.2 Primitive Gates on Breadboard

AND Gate

The AND circuits as shown in Figure 4.1 from the lab manual were constructed using the following parts:

AND Gate IC part number: SN74HC08N

Resistor part number: RN60D8060FB14

LED part number: MCL514GD

It should be noted that the resistance of the resistor used is not exactly 1 kOhm, but was the most similar considering what was available in the lab kit (806 Ohm).

The constructed circuits are shown in **Figure 1** and **Figure 2** below:

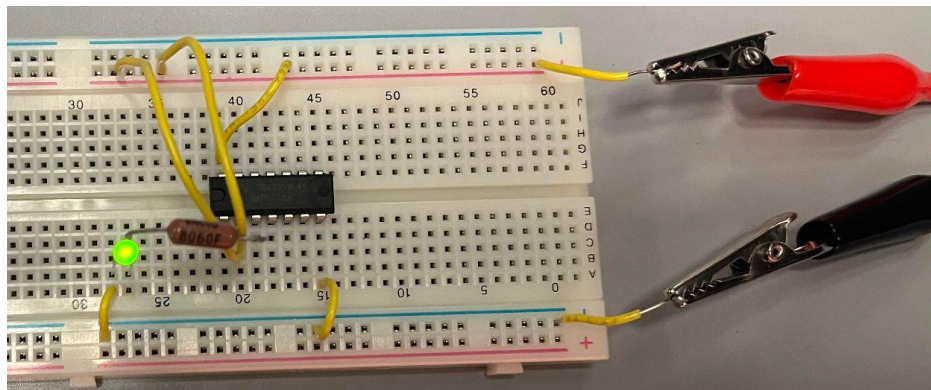


Figure 1: The AND circuit A constructed based on Figure 4.1 from the manual

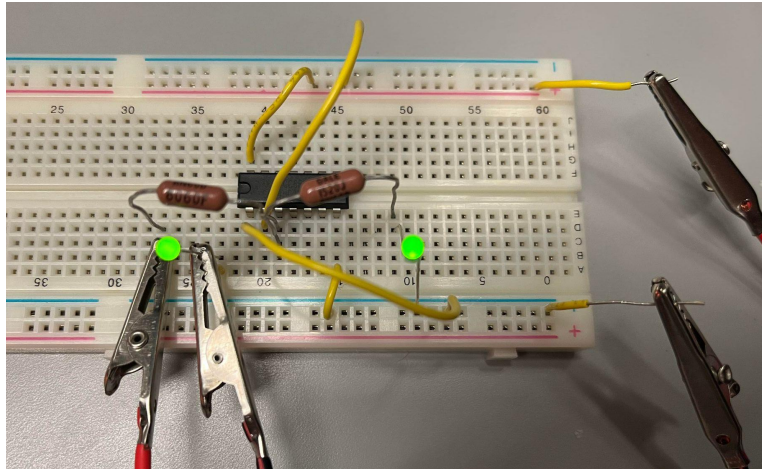


Figure 2: The AND circuit B constructed based on Figure 4.1 from the manual

The voltage measurements collected from these two circuits are shown in **Table 1** below.

Table 1: Voltage Measurements for AND gate

Item	Voltages in 1 branch	Voltages in 2 branches
V_1	2.90	2.33
V_2	1.95	1.95
V_t	$V_1+V_2=4.85$	$V_1+V_2=4.28$

Truth tables were constructed from the inputs and outputs of AND gate circuit A and B. This is recorded in **Table 2** and **Table 3** below.

Table 2: Truth Table for the AND gate circuit A

Input A	Input B	Output
0	0	0
0	1	0
1	0	0
1	1	1

Table 3: Truth Table for the AND gate circuit B

Input A	Input B	Output LED A	Output LED B
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

Does it comply with an AND gate truth table shown here?

Yes, the truth tables for the AND gate for circuit A and B both comply with the AND gate truth table.

Why V_t is not the same for the two cases of one-resistor versus two-resistors?

Here, we can assume that the resistance of the LED is insignificant relative to the resistance of the resistor. By adding another resistor-LED pair in parallel, the equivalent resistance at the output decreases. With this decreased resistance, the gate tries to keep the output voltage at 5V, by increasing the output current. However, as the gate is non ideal, it has a small internal resistance at the output. As the current at the output is increased, and the value of the internal resistance remains constant, the voltage drop across the internal resistor increases relative to the one-resistor case, meaning that the actual voltage drop across each branch (V_t) decreases.

OR Gate

The OR circuits as shown in Figure 4.2 from the lab manual were constructed using the following parts:

OR Gate IC part number: SN74HC32N

Resistor part number: RN60D8060FB14

LED part number: MCL514GD

It should be noted that the resistance of resistor used is not exactly 1 kOhm, but was the most similar from what was in the lab kit (806 Ohm)

The constructed circuits are shown in **Figure 3** and **Figure 4** below:

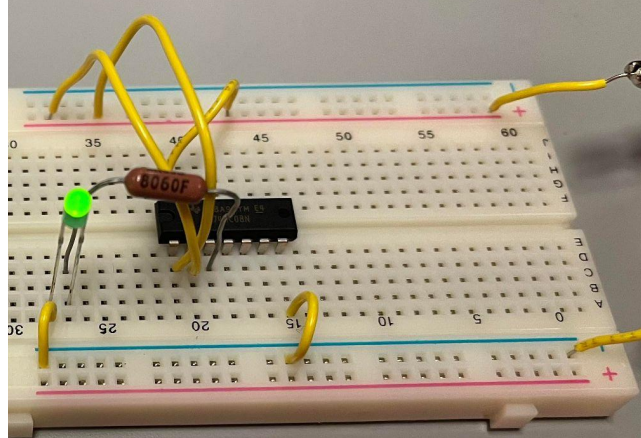


Figure 3: The OR circuit A constructed based on Figure 4.2 from the manual

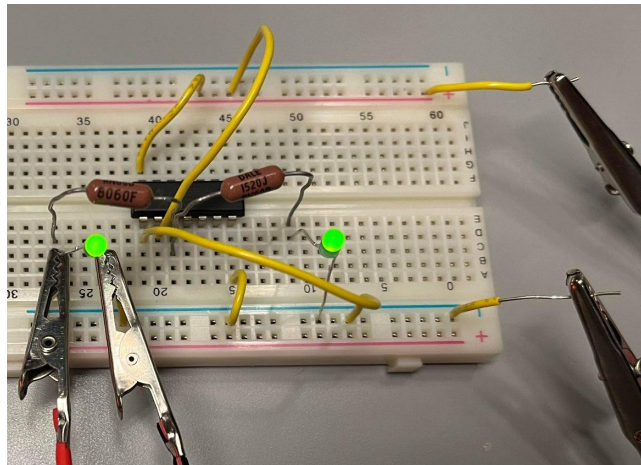


Figure 4: The OR circuit B constructed based on Figure 4.2 from the manual

The voltage measurements collected from these two circuits are shown in **Table 4** below.

Table 4: Voltage Measurements for OR gate

Item	Value for one R	Value for two R
V_1	2.91	2.79
V_2	1.96	1.94
V_t	$V_1 + V_2 = 4.87$	$V_1 + V_2 = 4.73$

Truth tables were constructed from the inputs and outputs of OR gate circuit A and B. This is recorded in **Table 5** and **Table 6** below.

Table 5: Truth Table for the OR gate circuit A

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	1

Table 6: Truth Table for the OR gate circuit B

Input A	Input B	Output LED A	Output LED B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Does it comply with an OR gate truth table shown here?

Yes, the truth tables for the OR gate for circuit A and B both comply with the OR gate truth table.

Why V_t is not the same for the two cases of one-resistor versus two-resistors? (Same answer as above)

Here, we can assume that the resistance of the LED is insignificant relative to the resistance of the resistor. By adding another resistor-LED pair in parallel, the equivalent resistance at the output decreases. With this decreased resistance, the gate tries to keep the output voltage at 5V, by increasing the output current. However, as the gate is non ideal, it has a small internal resistance at the output. As the current at the output is increased, and the value of the internal resistance remains constant, the voltage drop across the internal resistor increases relative to the one-resistor case, meaning that the actual voltage drop across each branch (V_t) decreases.

4.1.3 Four-bits adder

The VHDL code for the four-bits adder is shown in **Figure 5** below.

```
First_circuit.vhd
1  library ieee;                -- Declare that you want to use IEEE libraries
2  use ieee.std_logic_1164.all;  -- Library for standard logic circuits
3  use ieee.numeric_std.all;     -- Another useful library for UNSIGNED numbers
4
5  entity First_circuit is      -- entity definition
6  port( SW:    in    std_logic_vector(9 downto 0);    -- Toggle switches
7        LEDG:  out   std_logic_vector(4 downto 0);    -- Green LEDs
8        );
9  end entity First_circuit;
10
11 architecture main of First_circuit is
12  signal a, b:  unsigned(3 downto 0);
13  signal c:     unsigned(4 downto 0);  -- Naming inputs
14  begin
15  a <= unsigned(SW(3 downto 0));
16  b <= unsigned(SW(9 downto 6));
17  c <= ('0' & a) + ('0' & b);
18
19  LEDG(4 downto 0) <= std_logic_vector(c);
20  end architecture main;
```

Figure 5: VHDL code used to model a four-bits adder

Output from the simulated four-bits adder from Figure 5 is shown in **Figure 6** below.

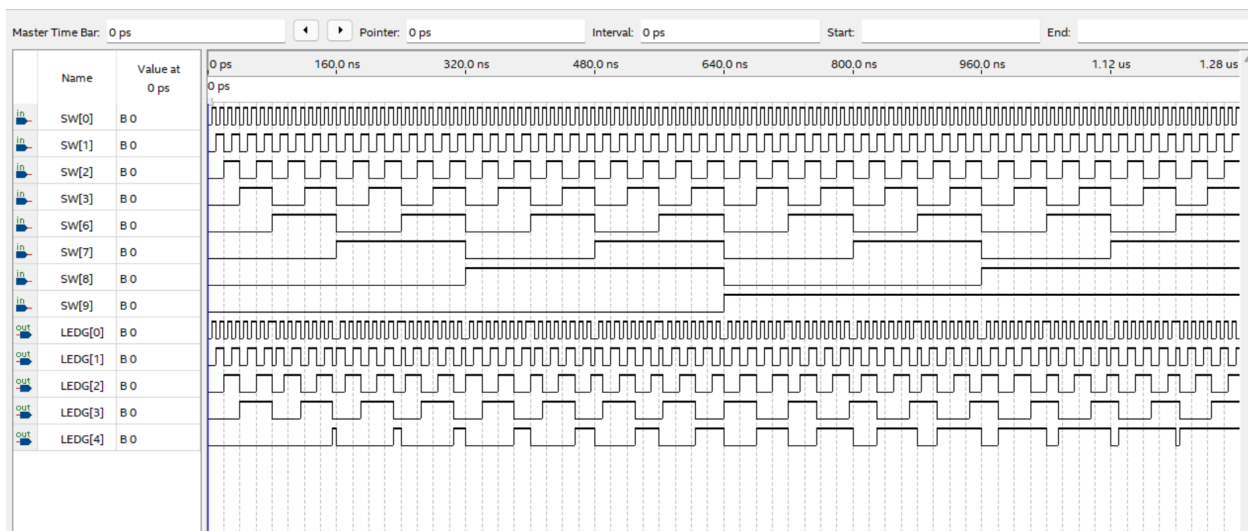


Figure 6: Input and output waveform for the simulation of the VHDL code in **Figure 5**

The simulated four-bits adder was tested using the various inputs from **Table 7** below (Table 4.2 from the lab manual), and the results were also included within the table.

Table 7: Partial truth table of the 4-bits adder, with inputs and outputs in hexadecimal (base 16)

Input a SW(3 downto 0)	Input b SW(9 downto 6)	Output c LEDG(4 downto 0)
0	0	0
2	5	7
8	3	B
A	5	F
7	9	10
6	D	13
B	7	12
E	C	1A
F	F	1E

4.1.4 Soldering components for the Project

Four resistors for the project were soldered onto the board. The results were approved by the TA. The result is shown in **Figure 7** below.

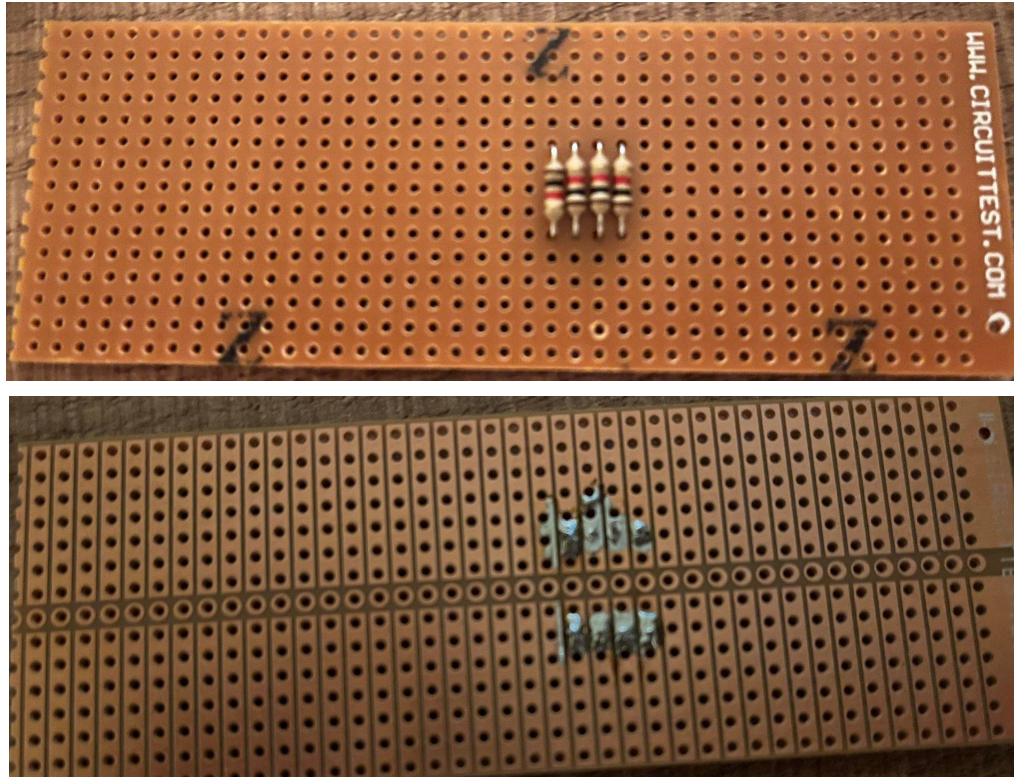
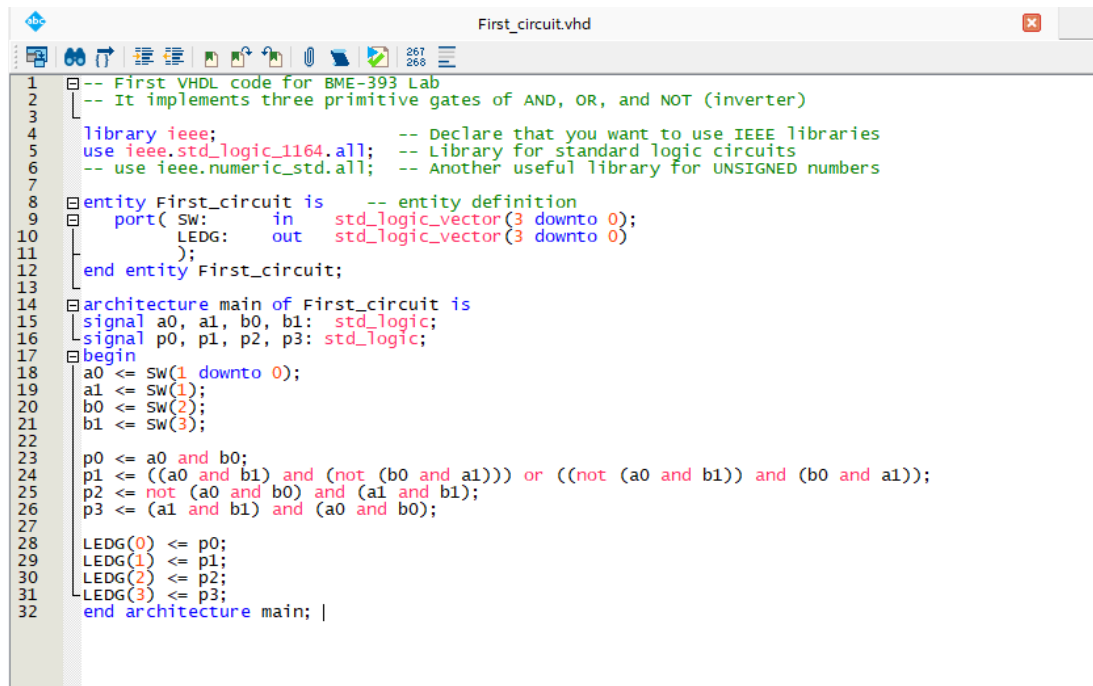


Figure 7: 4 resistors soldered onto breadboard for final project (Top Image - Front view, Bottom Image - Back view)

5. Lab report

1. It is a 2 input, 2-bits multiplier, where there are 2 total inputs, each of size 2 bits. The circuit multiplies the 2 inputs together to form one output of size 4 bits. The code is shown in **Figure 8** below.



```
1  -- First VHDL code for BME-393 Lab
2  -- It implements three primitive gates of AND, OR, and NOT (inverter)
3
4  library ieee;           -- Declare that you want to use IEEE libraries
5  use ieee.std_logic_1164.all; -- Library for standard logic circuits
6  -- use ieee.numeric_std.all; -- Another useful library for UNSIGNED numbers
7
8  entity First_circuit is  -- entity definition
9      port( SW: in std_logic_vector(3 downto 0);
10           LEDG: out std_logic_vector(3 downto 0)
11         );
12  end entity First_circuit;
13
14  architecture main of First_circuit is
15      signal a0, a1, b0, b1: std_logic;
16      signal p0, p1, p2, p3: std_logic;
17  begin
18      a0 <= SW(1 downto 0);
19      a1 <= SW(1);
20      b0 <= SW(2);
21      b1 <= SW(3);
22
23      p0 <= a0 and b0;
24      p1 <= ((a0 and b1) and (not (b0 and a1))) or ((not (a0 and b1)) and (b0 and a1));
25      p2 <= not (a0 and b0) and (a1 and b1);
26      p3 <= (a1 and b1) and (a0 and b0);
27
28      LEDG(0) <= p0;
29      LEDG(1) <= p1;
30      LEDG(2) <= p2;
31      LEDG(3) <= p3;
32  end architecture main;
```

Figure 8: VHDL code modelling the 2 input, 2-bits multiplier

2. The schematic entry for the reduced circuit is shown in **Figure 9** below. The total number of gates in this reduced circuit is **8 gates**.

The output of this circuit is shown in **Figure 10** below. The outputs were compared to the outputs of the simulation generated in the prelab for figure 3.1 from the lab manual. They were identical, thus, suggesting that the reduced circuit is correct.

The number of gates was reduced by sharing the outputs: $a_1 \cdot b_1$, $a_0 \cdot b_0$. The number of gates was reduced by also treating the outputs: $a_0 \cdot b_1$ and $a_1 \cdot b_0$ as individual variables and inputting them into an XOR gate, achieving the desired output for p1.

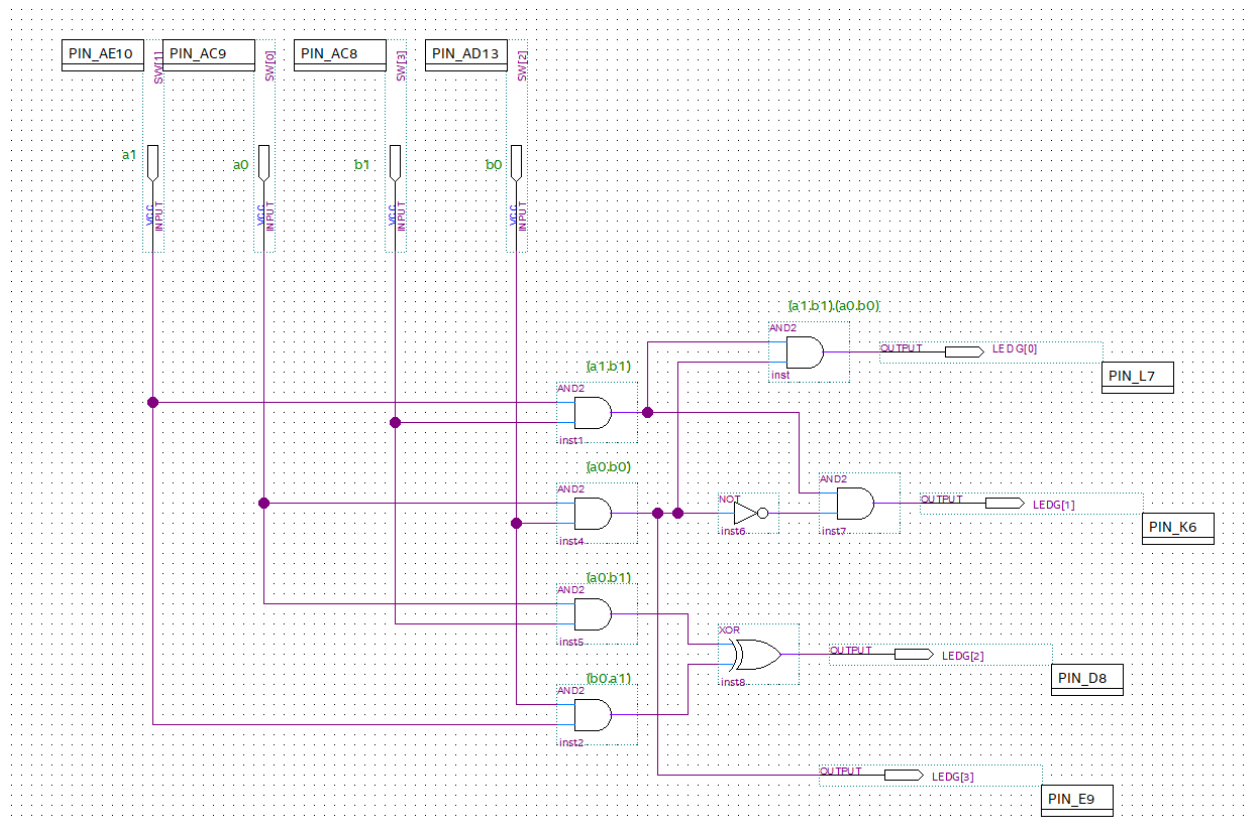


Figure 9: Schematic entry for the reduced digital circuit shown in Figure 3.1 in the lab manual

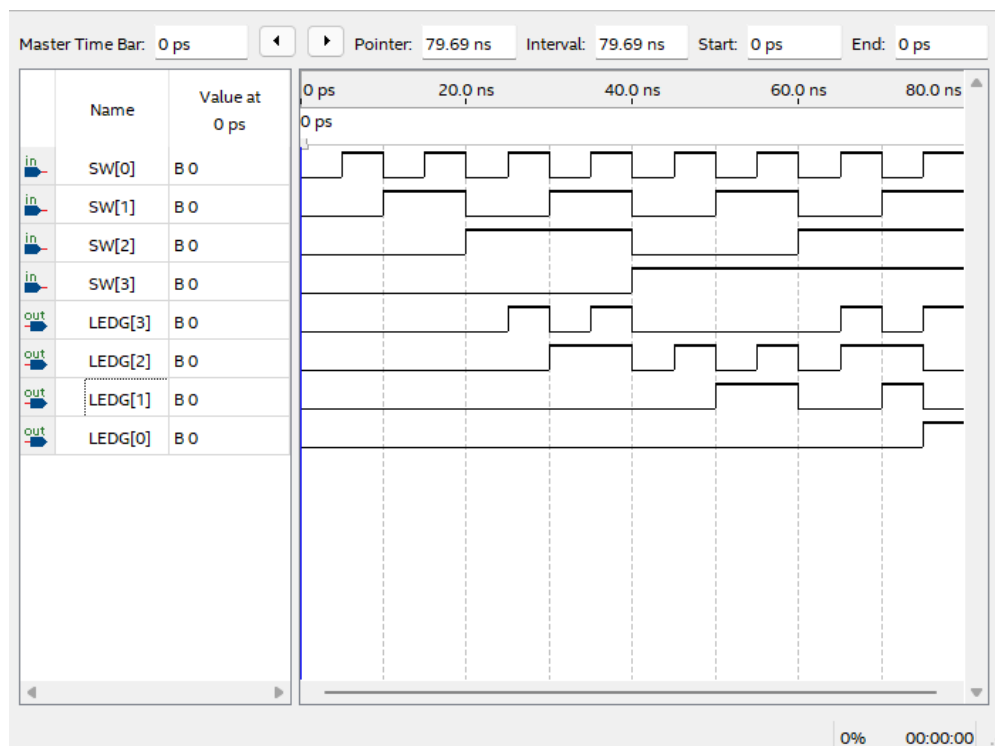


Figure 10: Simulation results with input and output waveform of the circuit shown in Figure 9

3. The circuit created in the prelab for figure 3.2 from the lab manual only had 7 gates. The circuit was made from 3 XOR, 3 AND, and 1 OR gate. It is already in the most simplified form, thus, it cannot be reduced further.
4. This was done in section 4.1.2 above.
5. The block diagram for the four-bit multiplier was constructed using three four-bit adders. The logic for determining the inputs, intermediates, and output of the four-bit multiplier is shown in **Figure 11** below.

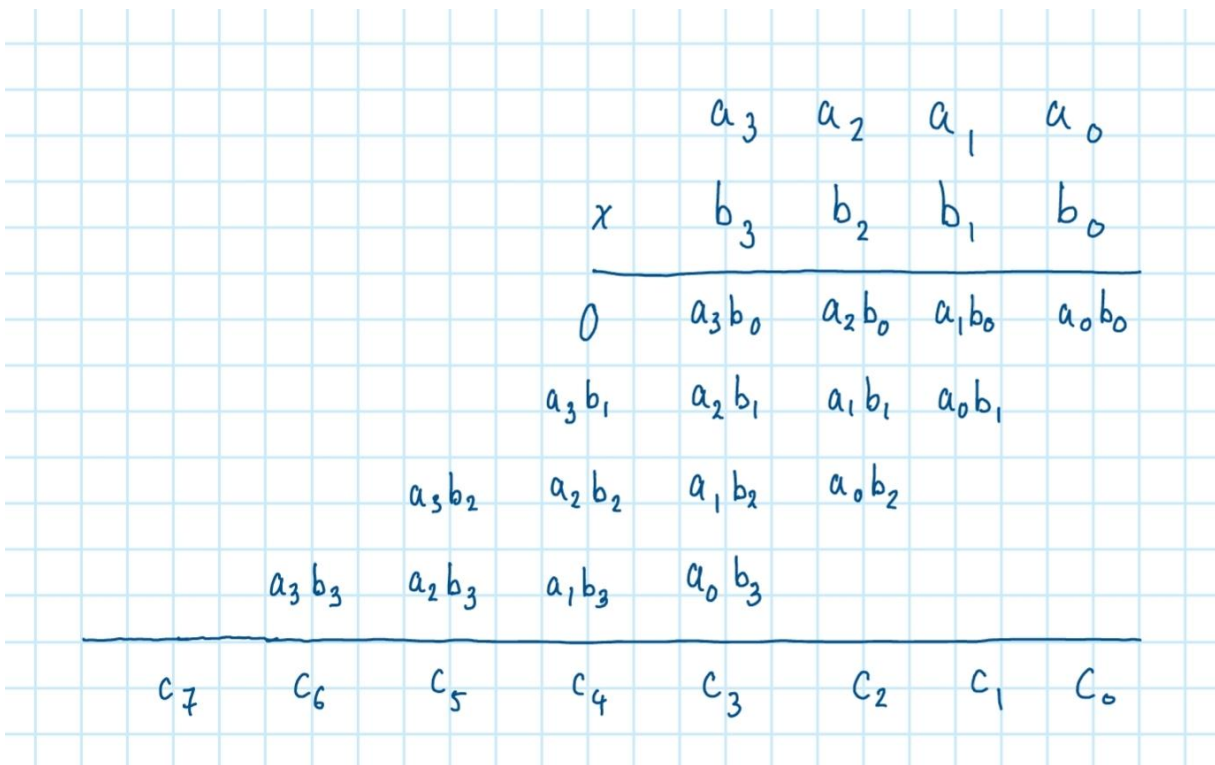


Figure 11: Inputs, intermediates, and outputs of the four-bit multiplier

The block diagram for the four-bit multiplier is shown in **Figure 12** below.

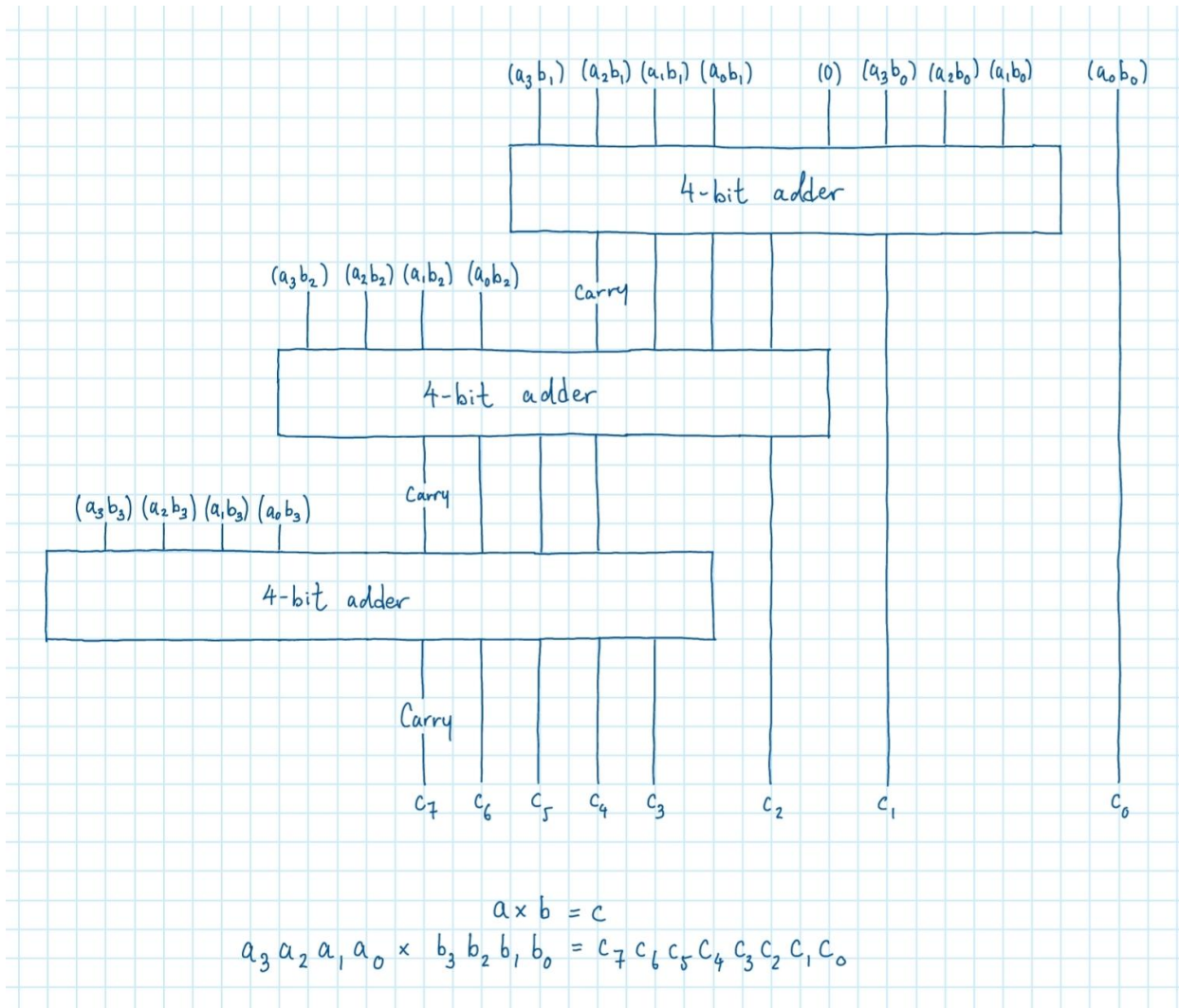


Figure 12: Block diagram for the 4-bit multiplier

The VHDL code for the 4-bit multiplier was implemented, it is shown in **Figure 13** below.

```

1  library ieee; -- Declare that you want to use IEEE libraries
2  use ieee.std_logic_1164.all; -- Library for standard logic circuits
3  use ieee.numeric_std.all; -- Another useful library for UNSIGNED numbers
4
5  entity First_circuit is -- entity definition
6  port( SW: in std_logic_vector(7 downto 0); -- Toggle switches
7        LEDG: out std_logic_vector(7 downto 0) -- Green LEDs
8  );
9  end entity First_circuit;
10
11 architecture main of First_circuit is
12 signal input_a: std_logic_vector(3 downto 0); -- Naming inputs
13 signal input_b: std_logic_vector(3 downto 0); -- Naming inputs
14 signal intermediate_0: unsigned(7 downto 0);
15 signal intermediate_1: unsigned(7 downto 0);
16 signal intermediate_2: unsigned(7 downto 0);
17 signal intermediate_3: unsigned(7 downto 0);
18 begin
19
20 input_a <= SW(3 downto 0);
21 input_b <= SW(7 downto 4);
22
23 intermediate_0(0) <= input_a(0) and input_b(0);
24 intermediate_0(1) <= input_a(1) and input_b(0);
25 intermediate_0(2) <= input_a(2) and input_b(0);
26 intermediate_0(3) <= input_a(3) and input_b(0);
27
28 intermediate_1(1) <= input_a(0) and input_b(1);
29 intermediate_1(2) <= input_a(1) and input_b(1);
30 intermediate_1(3) <= input_a(2) and input_b(1);
31 intermediate_1(4) <= input_a(3) and input_b(1);
32
33 intermediate_2(2) <= input_a(0) and input_b(2);
34 intermediate_2(3) <= input_a(1) and input_b(2);
35 intermediate_2(4) <= input_a(2) and input_b(2);
36 intermediate_2(5) <= input_a(3) and input_b(2);
37
38 intermediate_3(3) <= input_a(0) and input_b(3);
39 intermediate_3(4) <= input_a(1) and input_b(3);
40 intermediate_3(5) <= input_a(2) and input_b(3);
41 intermediate_3(6) <= input_a(3) and input_b(3);
42
43 LEDG <= std_logic_vector(intermediate_0 + intermediate_1 + intermediate_2 + intermediate_3);
44
45 end architecture main;

```

Figure 11: VHDL code modelling a four-bit multiplier

The input and output waveform from the VHDL code simulation is shown in **Figure 12** below.

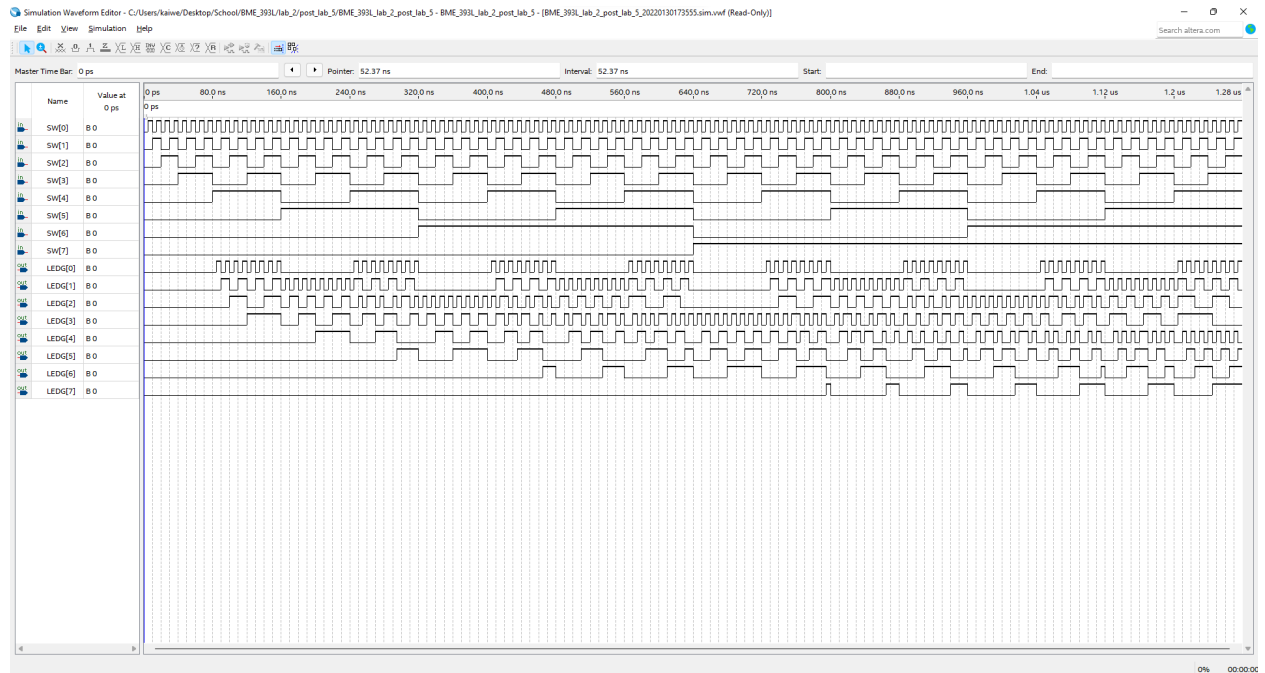


Figure 12: Input and output waveforms from the simulation of the VHDL code in **Figure 11**