Y.Y.R.Lee
SN: 19035879
ELEC0004
March 04, 2020

FPGA Workshop

Exercise 1

AND GATE

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

Figure 1: AND Gate Truth Table

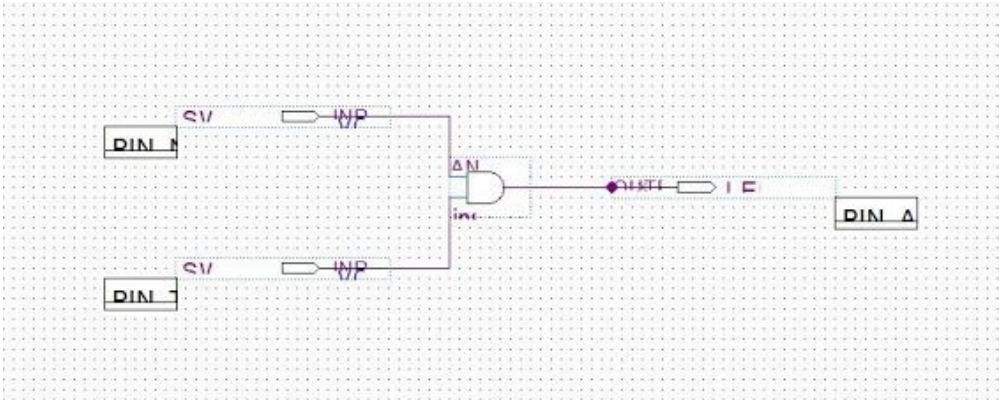

Figure 2: AND Gate Schematic

Prior to building the AND gate on Quartus, a truth table was plotted out according to Figure 1. Then, two inputs set to the first two switches on the FPGA were linked by an AND symbol to output a result to an LED (Figure 2). After compiling the program successfully and setting the inputs and the output to the correct ports on the board, the program was uploaded to the board.

The functionality was tested by running through the combinations of inputs according to the truth table. The outputs exactly matched the expected results and the LED turned on only when both switches were set to ‘on’.

Y.Y.R.Lee
SN: 19035879
ELEC0004
March 04, 2020

Exercise 2

1 BIT ADDER

A	B	C_{i-1}	S	C_i
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

Figure 3: 1 Bit Adder Truth Table

$$S=\overline{AB}C+\overline{AC}B+\overline{BC}A+ABC$$

Equation 1: Sum Output of 1 Bit Adder

$$C_i=\bar{A}BC+\bar{B}AC+\bar{C}AB+ABC$$

Equation 2: Carry Output of 1 Bit Adder

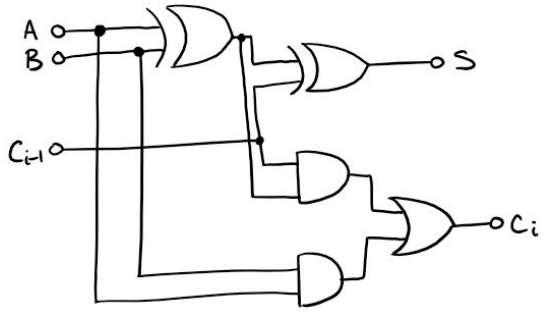

Figure 4: 1-Bit Adder Draft Schematic

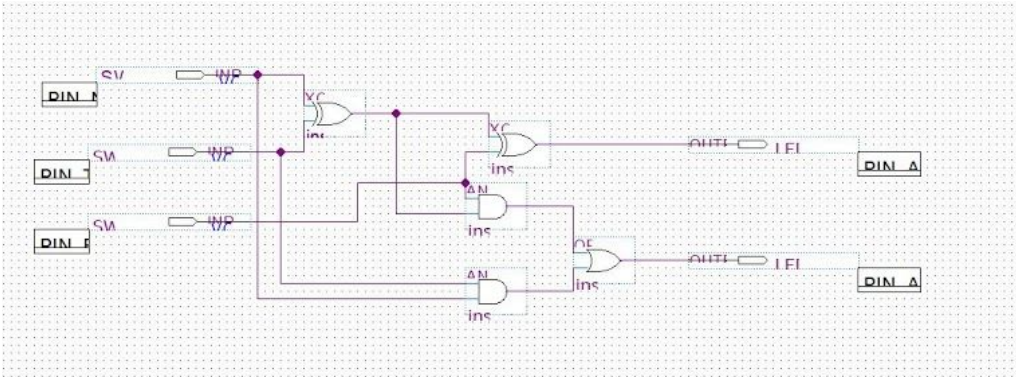

Figure 5: 1 Bit Adder Schematic

A one-bit adder was modelled on a truth table which was then converted into two boolean equations (Figure 3, Equation 1 and 2). According to the expected results, the draft schematic (Figure 4) was designed and checked back against the truth table to verify its logic. Then, this was plotted out in Quartus according to Figure 5 and assigned to three switches and two LEDs for the inputs and outputs respectively. The program was compiled successfully and uploaded onto the FPGA.

Similarly to the prior exercise, the program was tested by inputting the eight possible combinations of switches according to the truth table and observing the outputs on the LEDs. The results were as expected and the adder was successfully able to add up to three. The design file was then saved as a symbol file to be used in Exercise 3.

Exercise 3

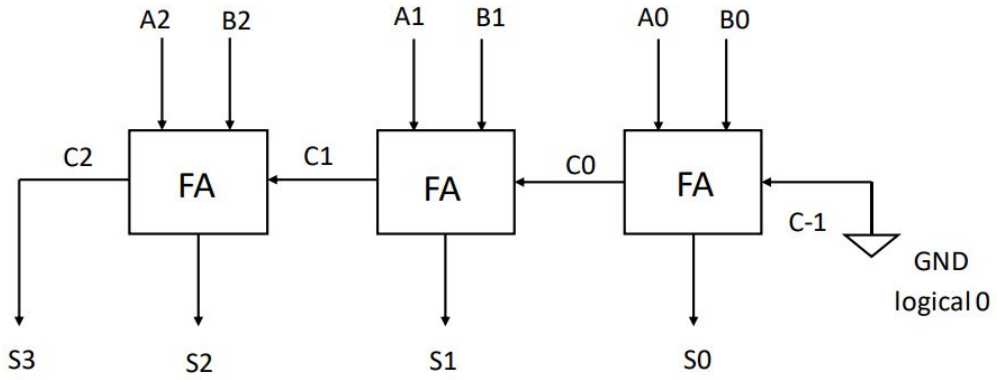

Figure 6: 3 Bit Adder Design

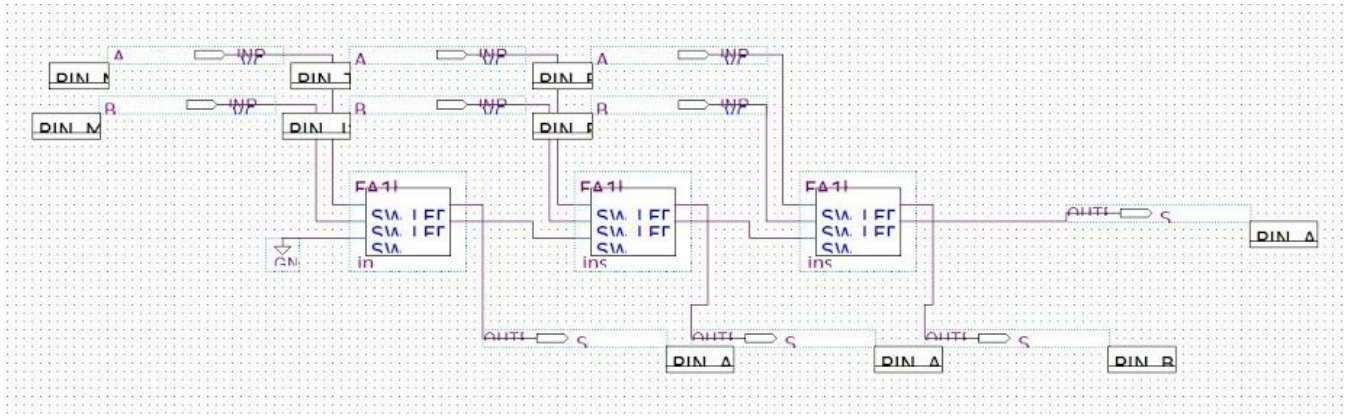

Figure 7: 3 Bit Adder Schematic

The 3 bit adder was created as an extension of exercise 2 using the symbol files created just prior. The schematic (Figure 6) was created according to Figure 5 given in the assignment document.

After assigning the inputs and outputs to the correct ports, the program was uploaded to the FPGA. Initially, it was thought that an error existed since two LEDs were lit up without any inputs turned on; however, the default settings for KEY0 and KEY1 are on rather than off which explains why two outputs are switched on. The program is meant to add two inputs at a time for each output, taking into account an additional third input in the form of the carry from the previous cell. The LED outputs shows the state of the sum outputs for each cell while the carry is carried over to the next cell; the fourth LED shows the state of the final carry.

The circuit was tested by adding inputs one by one and checking the logic of the output LED configurations. The adder performed the proper operation for each combination of inputs including when the switches and buttons were pressed in other configurations aside from adding from the least important bit inputs. Proof of logic was assessed by testing the individual adder results for each LED as individual cells and also in conjunction with and without a carry from the previous cell if applicable.

Exercise 4

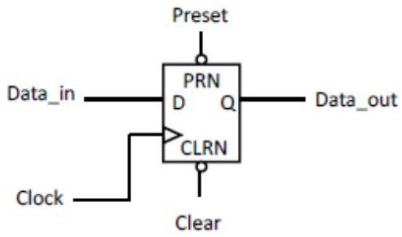

Figure 8: D Flip-Flop Symbol

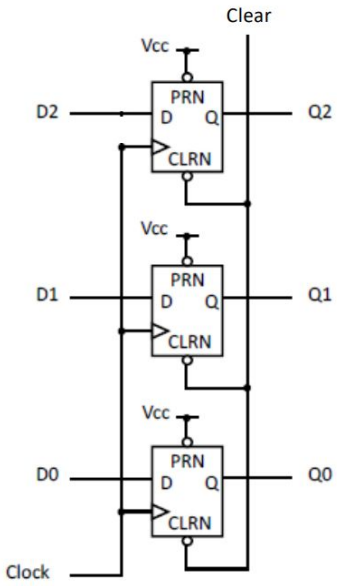

Figure 9: 3-Bit Register Design

Y.Y.R.Lee
SN: 19035879
ELEC0004
March 04, 2020

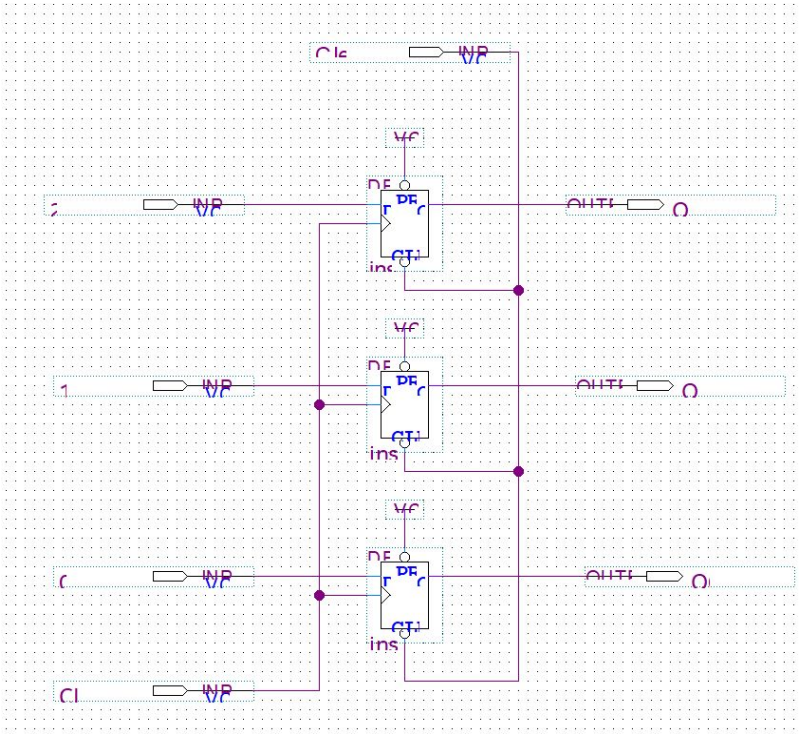

Figure 10: 3-Bit Register Schematic

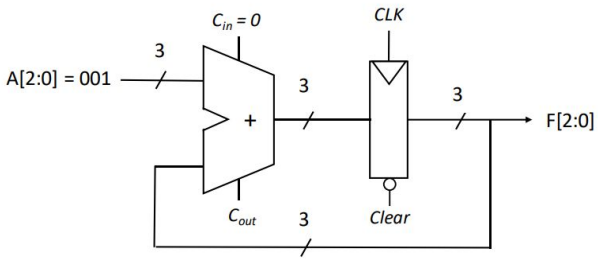

Figure 11: Modulo-8 Counter Design

Y.Y.R.Lee
SN: 19035879
ELEC0004
March 04, 2020

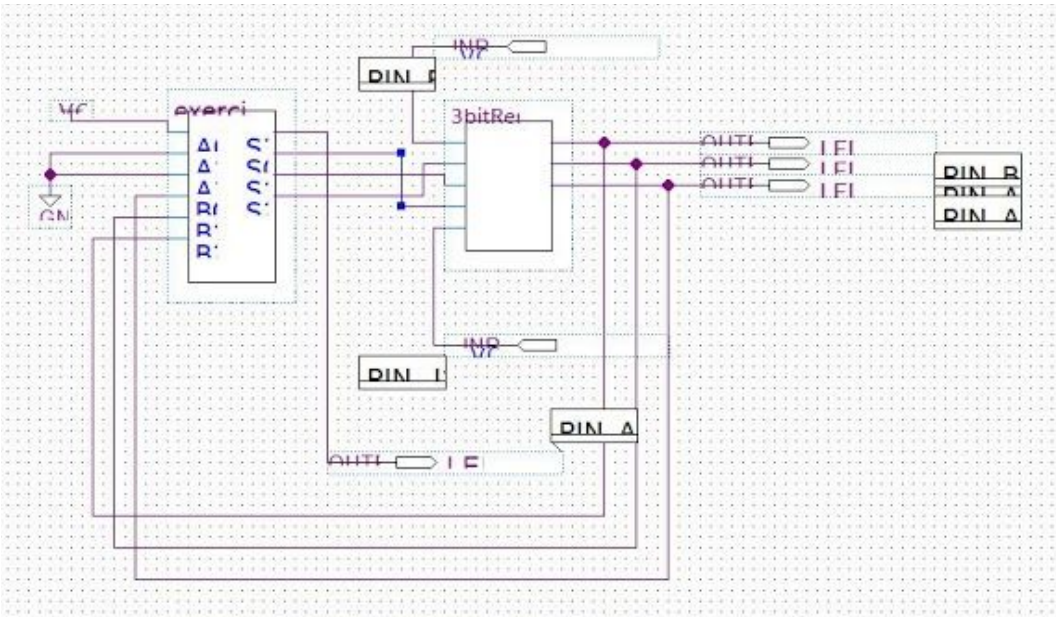

Figure 12: Modulo-8 Counter Schematic

A 3-bit register was created by connecting three d-flip flops (Figure 8) according to the design in Figure 9. This was saved as a symbol file (Figure 10); no pins had to be assigned since this block design would be used in the later schematic so it was only checked by compiling successfully. The register's purpose is to update the state of the three bits at the same time thanks to the clock which unifies the three D flip-flops.

The circuit design in Figure 11 was then created using the user-created symbols from Exercise 3 and the 3 bit register previously saved. According to the assignment, the schematic in figure 12 was created; the initial inputs A[0:2] were set to 001 using a ground node and a voltage input node. These three inputs lead into the 3-bit adder which outputted the bit-respective sums into the inputs of the 3-bit register while the carry is outputted directly into LED3. The 3-bit adder has additional clock and clear inputs assigned to KEY0 and KEY1 respectively which update or reset the circuit. The outputs of the 3-bit register are connected back into the second set of inputs B[0:2] of the 3-bit adder.

This circuit was tested by 'counting' up from 0 to 7. The LEDs output the respective binary values successfully until and including number 6; however, 7 is represented incorrectly as all four LEDs in the ON state which reads 9 instead. This is likely because of the 3-bit adder carry output to LED3. The results are as expected aside from the count 7, but after re-analyzing the circuit it was realized that the error was due to an extra carry output. An improvement for future designs is to troubleshoot the symbol created for the 3-bit register in particular as it can be seen that the inputs and outputs did not display. Additionally, all text was too large to be displayed fully and correctly, and this problem was not ameliorated after changing font presets. For easier legibility and therefore a smoother workflow, this should be displayed more clearly.