**Lab 4**

**Objective:**

Create a one-bit adder

Create a three-bit adder using one-bit adders as symbols (modular design)

**Design:**



Table - One-bit Adder Truth Table



Table - Three-bit Adder Test Values

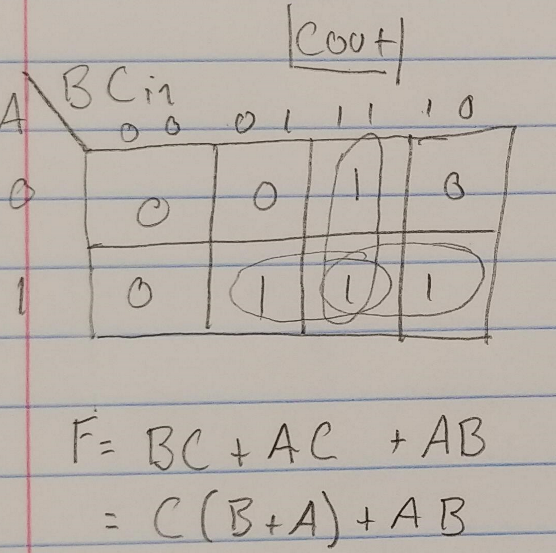
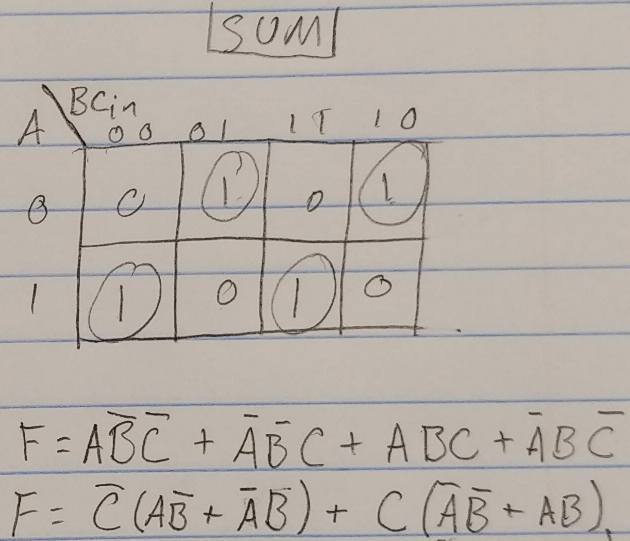
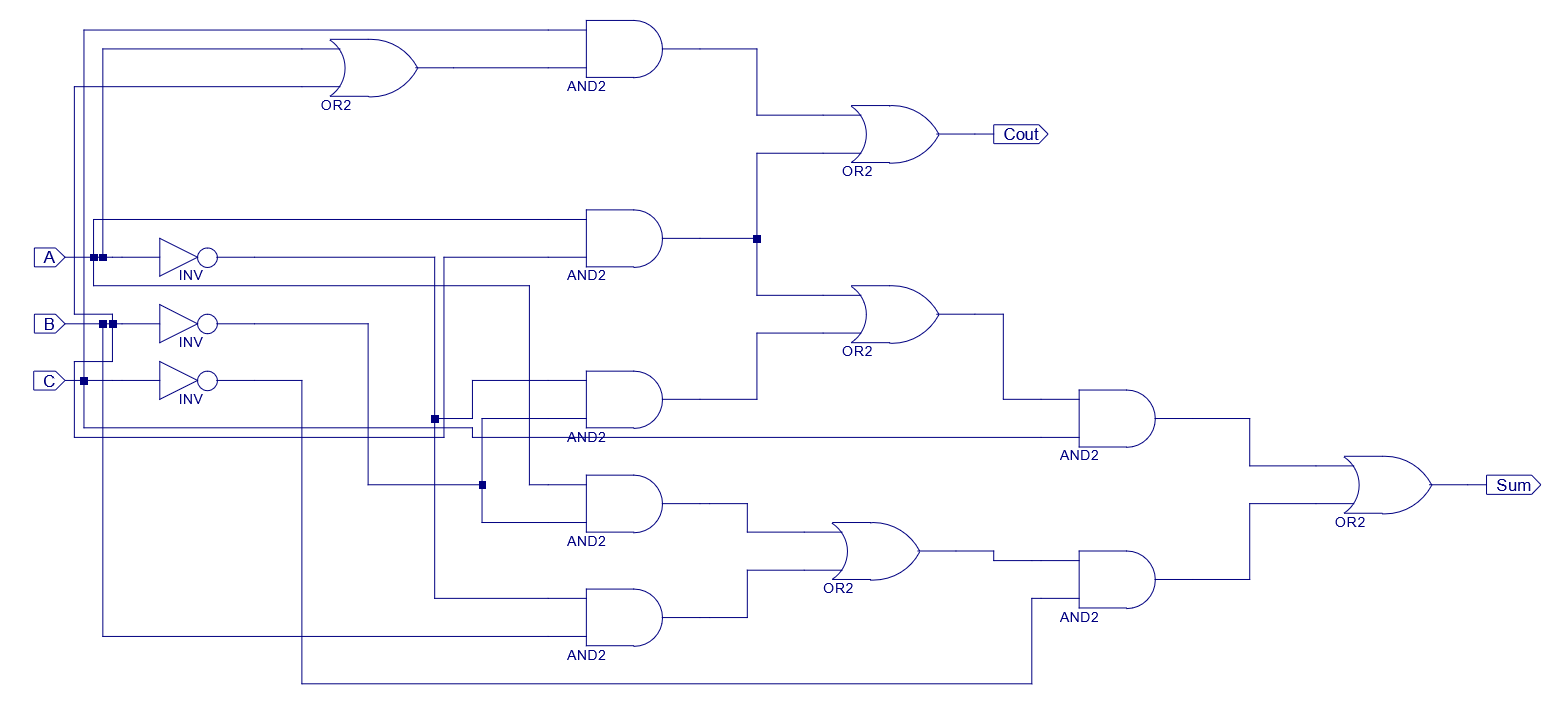
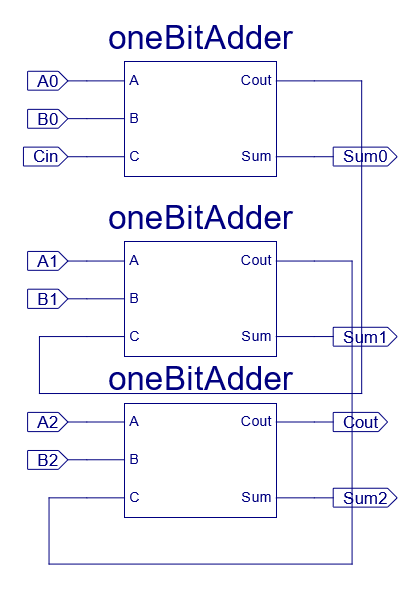


Figure – Cout K-Map and Function Figure 2 – SUM K-Map and Function



*Figure 3 – One-bit adder Circuit*



*Figure 4 – Three-bit adder Circuit*

**Procedure:**

**Using Xilinix ISE:**

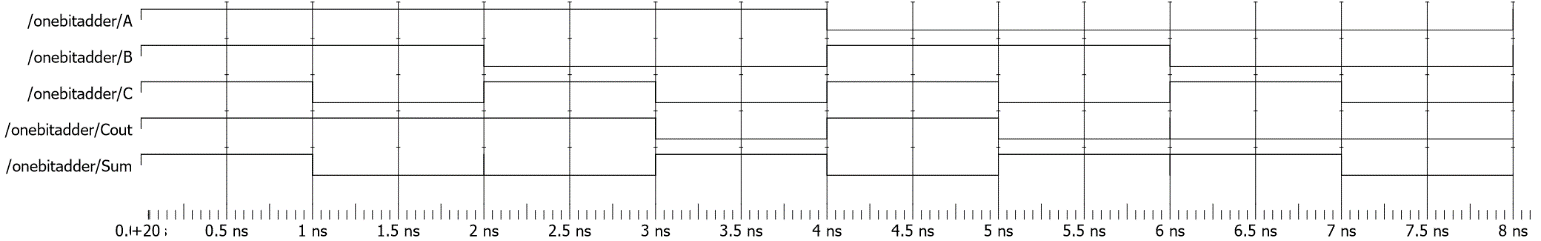
**One-bit Adder:**

* Construct one-bit adder schematic from prelab diagram (Fig. 3)
* Compile schematic
  + Troubleshoot if necessary
* Run behavioral simulation for all 7 inputs
  + Print results (Fig. 5)
* Run Post-Route Simulation (adds time delays)
  + Print results (Fig. 6)
* Download the design to the board, following previous lab’s steps (Lab 3, steps 17 through 19-XI)
  + Have outputs Cout and Sum each go to an LED
  + Have inputs A, B, and Cin accept input from physical switches
* Test the downloaded circuit against the prelab truth table (Table 1)

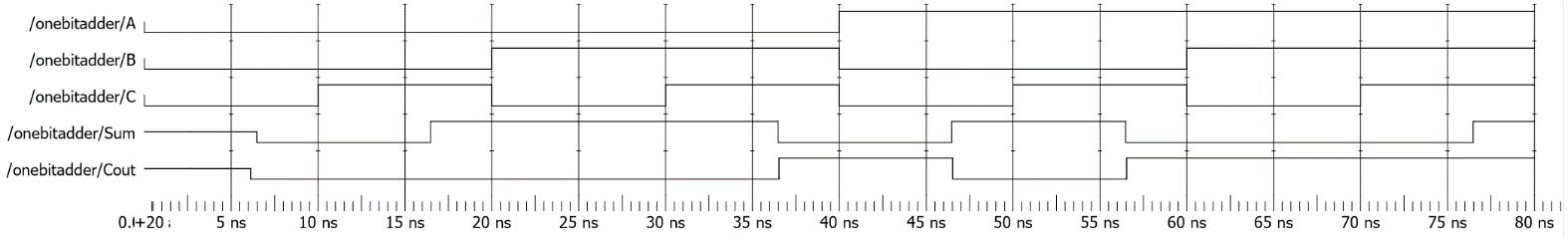
**Three-bit adder:**

* Create a schematic symbol from the one-bit adder schematic
  + Highlight the one-bit adder schematic file from the design window
  + Run “Create Schematic Symbol” in the process window
* Create a three-bit adder schematic using one-bit adder symbols
  + Create a new schematic file named “three-bit adder”
  + Select the “Symbol” tab
  + In the “Categories” section, select the project directory
  + Select the “one-bit adder” symbol
  + Place 3 “one-bit adder” symbols on the schematic
  + Connect the 3 Symbols appropriately
    - Connect the first symbol’s “Cout” to the second symbols “Cin”
    - Connect the second symbol’s “Cout” to the third symbols “Cin”
  + Add I/O markers
    - A0, B0 and Cin to the input of the first symbol
    - SUM0 to the output of the first symbol
    - A1 and B1 to the input of the second symbol
    - SUM1 to the output of the second symbol
    - A2 and B2 to the input of the third symbol
    - SUM2 and Cout to the output of the third symbol
  + Run behavioral simulation for given test inputs (Table 2)
    - Run each input for 5ns
    - Print results (Fig. 7)
  + Run Post-Route Simulation for first 4 test inputs (Table 2)
    - Run each input for 20ns
    - Print results (Fig. 8)
  + Download the schematic to the FPGA, similarly to the one-bit adder
    - Connect the 7 inputs to switches
    - Connect the 4 outputs to LEDs
  + Test the board with the test inputs (Table 2)

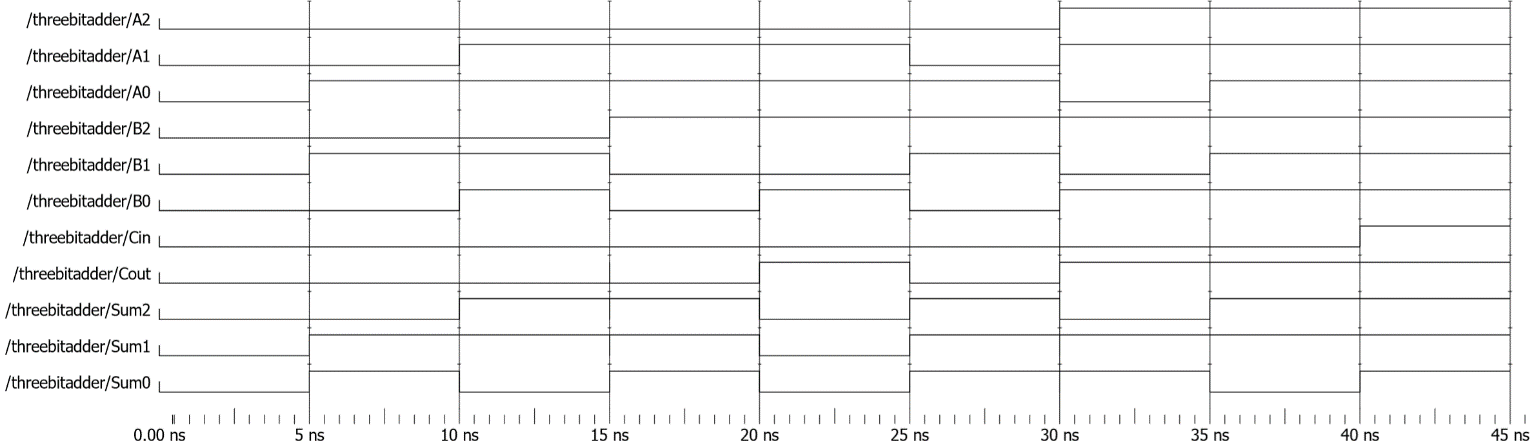
**Data:**



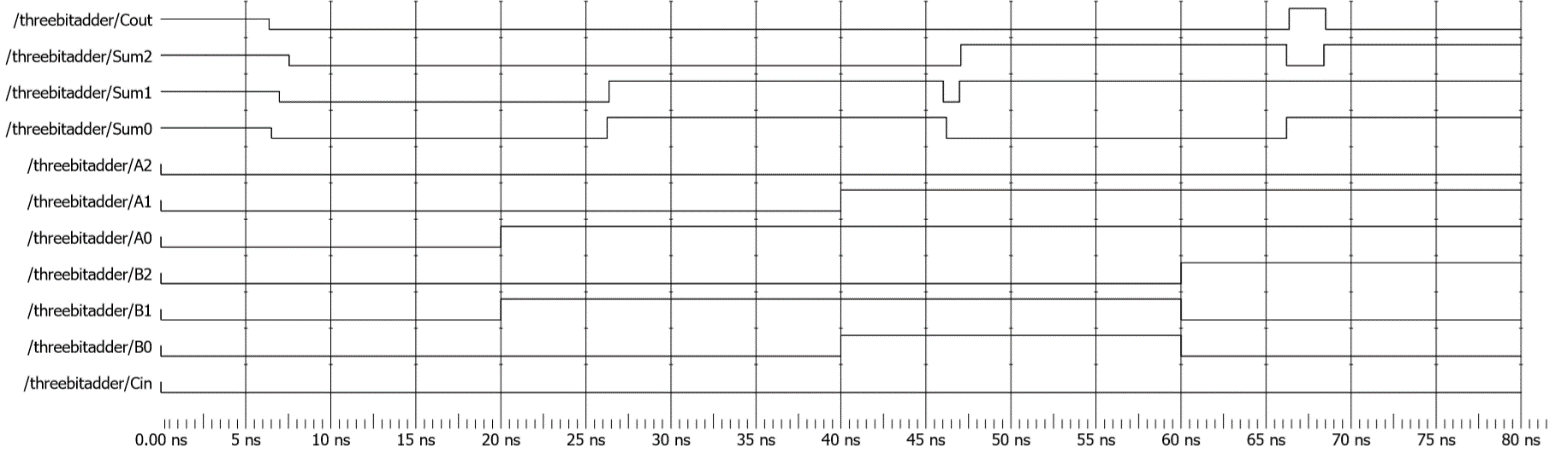
*Figure 5 – One-bit Adder Behavioral Simulation*



*Figure 6 – One-bit Adder Post-Route Simulation*

**

*Figure 7 – Three-bit Adder Behavioral Simulation*

**

*Figure 8 – Three-bit Adder Post-Route Simulation*

**Data Analysis:**

Each of the simulations (Figs. 5-8) and downloaded circuits matched the calculated truth tables (Tables 1 and 2). There was an issue where the three-bit adder was not matching the truth values expected. Troubleshooting showed that the Cout and SUM2 outputs were switched and correcting that gave the expected results.

**Conclusion:**

The goal of this experiment was to construct bit adders as well as observe the benefits of modular designs in increasingly complex circuits. Overall, the one-bit adder circuit seemed to become relatively complex for what seems to be a simple task, but it gets the job done. Constructing a three-bit adder from scratch in a similar fashion to the one-bit would begin to get prohibitively complex. By being able to break down the more complex end product (three-bit adder) into more manageable modules (one-bit adders) reduces the complexity significantly and makes the final design much faster to assemble.