EE4301 Lab Report:

FULL ADDER/SUBTRACTOR IMPLEMENTATION

Authors: Yahya Alhinai

Date: June 26, 2018

TA: Shashank Hegde

Professor: Abolghassem Mahmoodi

INTRODUCTION

The purpose was to implement 8 bits full adder/subtractor circuit using FPGA. Using Verilog to describe logic gates circuit, we design, Synthesis, and Simulation a simple circuit through Vivado's environment. As well as conducting behavioral simulation of Verilog design circuit to verify its intended functionally. Afterwards, the Verilog described circuit had been converted to logic gates to be implemented on FPGA basys 3. Different implementation and timing analysis on the circuit had been tested within Vivado's environment. The optimization difference in the implementation had been noted. After making sure the circuit is working flawlessly on Vivado's simulation, a bin stream file had been generated to be pushed to the FPGA basys 3 and stored on the memory chip in the board. The switches on the board were coded to be inputs and the LEDs were treated as outputs. Also, there was a dedicated Button to switch between outputting the results on full adder or the full subtractor.

MAIN

We used Vivado to simulate the circuit with the following equation which represent a single full adder modular:

$$Sum(A, B, C_{in}) = A \oplus \mathrm{B} \oplus \mathrm{C_{in}}$$

$$Carry(A, B, C_{in}) = A * B + A * C_{in} + B * C_{in}$$

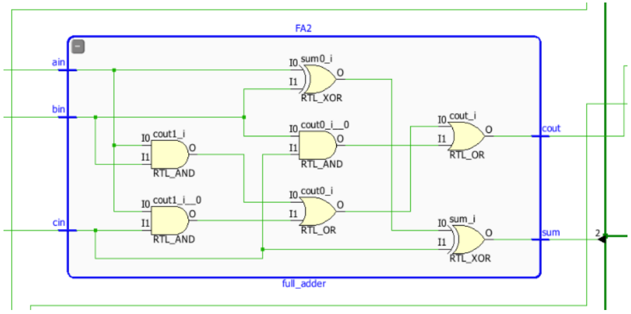
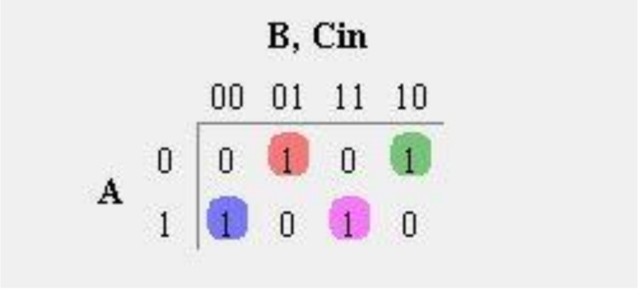


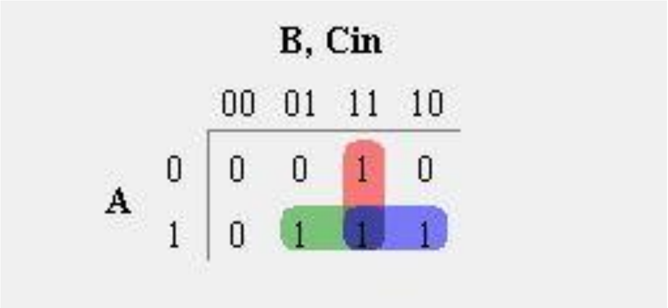
Figure 1: logic design for the building block of a single full adder modular

Full adder output equations were minimized using k-map:



A \ B, Cin	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Figure 2: equation to output the sum in a full adder modular



A \ B, Cin	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Figure 3: equation to output the carry in a full adder modular

Afterwards, the this modular had been instantiated to build an 8-bit full Adder. An 8-bit full adder using 8 had been module to compute the results of two 8-bits numbers. The synthesis of the our designed was successful. The 8-bit block diagram of the circuit generated by Verilog code is shown below (figure 4).

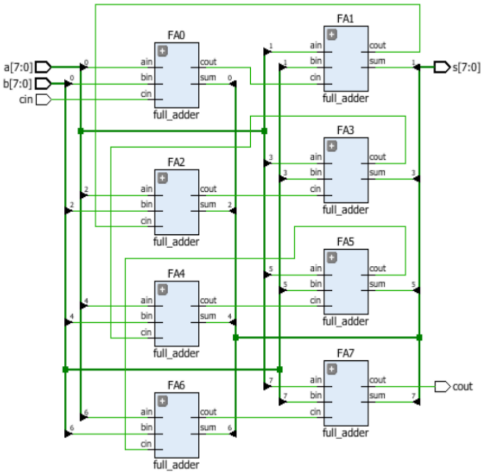


Figure 4: 8-bit full adder schematic generated by Verilog code

After making sure that the is working as it meant to be, behavioral simulation of Verilog designs has been made. Different test cases had been simulating to make sure the circuit is fully functioning including overflow triggering. All the cases have behaved as it was predicted previously.

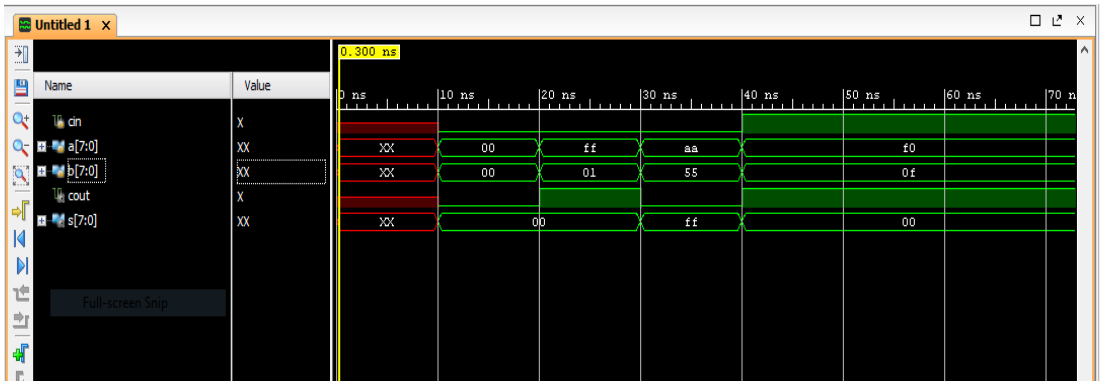


Figure 5: simulated results for the 8-bits full adder

Then a constrain file have been added and different implementation were conducted. The first implementation was Vivado’s default. The schematic design on FPGA basys 3 is shown below (figure 6).

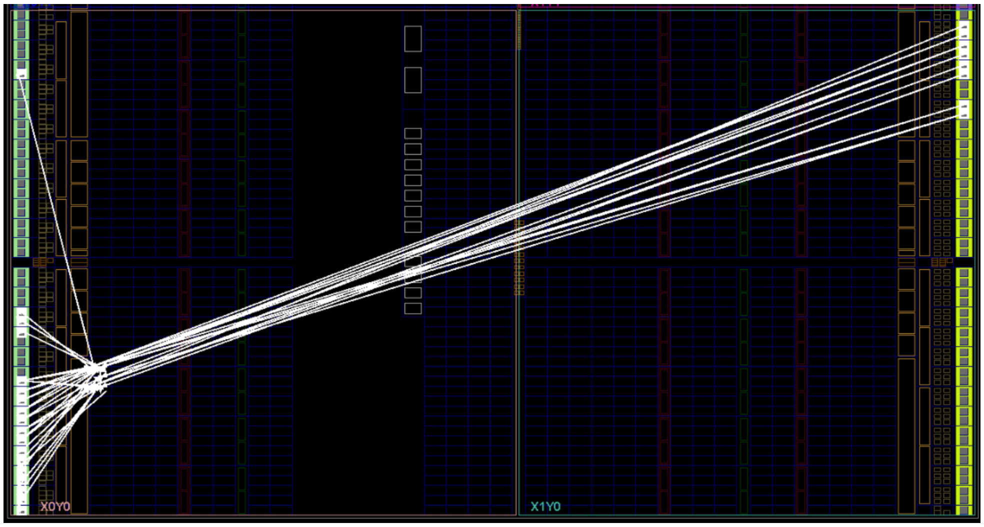
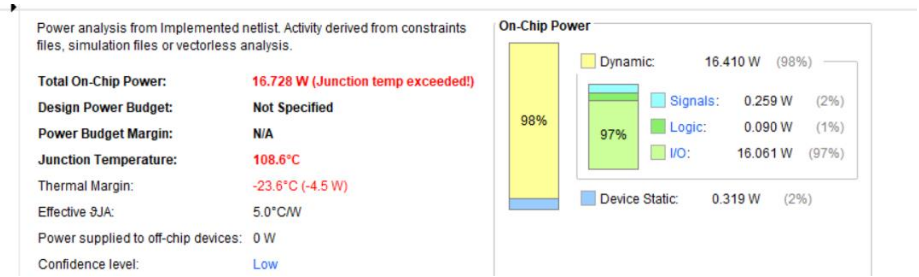


Figure 6: FPGA basys 3 schematic design

Afterwards, area explore implementation had been conducted to mark the difference it and Vivado’s default. The two implementations were exactly the same in terms of power consumption and the number of gates used to build up the circuits. This is might be because the circuit the was conducted was too simple to be optimized further; therefore, no noticeable difference has been spotted. The power estimation for Vivado's default implementation is shown (figure 7).



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**16.728 W (Junction temp exceeded!)**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**108.6°C**
Thermal Margin:	-23.6°C (-4.5 W)
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

On-Chip Power

Dynamic:	16.410 W	(98%)
Signals:	0.259 W	(2%)
Logic:	0.090 W	(1%)
I/O:	16.061 W	(97%)
Device Static:	0.319 W	(2%)

Figure 7: power estimation for Vivado's default implementation

Unlike behavioral simulation which shows ideal transition between different states, timing analysis provide an observation of the delay when switching between two different states. The delay was estimated to be 1.8 ns for the results to be stable. Timing analysis is shown (figure 8). Unlike the prediction that have been made, there is no noticeable difference in timing analysis between the two conducted implementations.

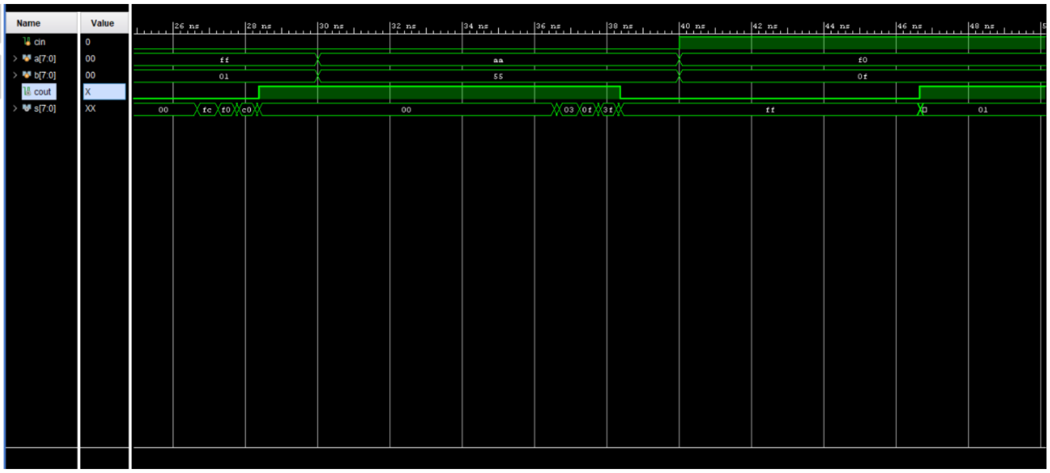


Figure 8: timing analysis for the conducted circuits

Subsequently, full subtractor modules had been added to the design. We used Vivado to simulate the circuit with the following equation which represent a full subtractor modular:

$$subtractor(A, B, C_{in}) = A \oplus \mathrm{B} \oplus \mathrm{C_{in}}$$

$$borrowed(A, B, C_{in}\ ) = {\sim}A * B + B * C_{in} + {\sim}B * C_{in}$$

The constrain file had been modified accordingly to accommodate the change in circuits as well as adding a button to switch between states. When the button is not pressed, the board will compute the given inputs as it activates the full adder modules. When the button is pressed, however, the two inputs will be subtracted from each other as the full subtractor modules are activated. The results are shown in the board using LEDs that are in the board. The circuit schematic that implements full adder and full subtractor is shown (figure 9).

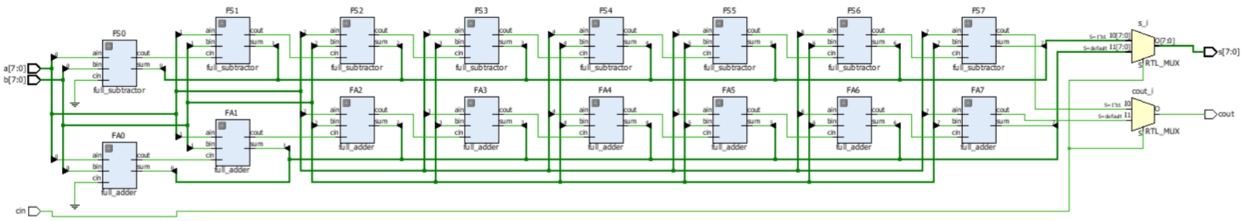


Figure 9: The circuit schematic that implements full adder and full subtractor

After the bin file was uploaded to the board, different inputs have been provided to make sure the full adder and the full subtractor are working as they meant to be.

SUMMARY

The goals for Labs 1,2 and 3 have been met. The implementation of 8 bits full adder/subtractor circuit have been successful. The generated circuit had been push for the FPGA basys 3 board after design, Synthesis, and Simulation the circuit in vivado and conducting behavioral simulation of Verilog design circuit to verify its intended functionally. The results on the board reflect the intended purpose of circuit as the switches on the board were coded to be inputs and LEDs were coded to be outputs. The results on the board has been verified.