ECE 3270: Laboratory 2

LAB 2: OPENCL LIBRARIES

February 22, 2019

Chintan Patel
Clemson University
Department of Electrical and Computer Engineering
cdpatel@clemson.edu

Abstract

Lab 2 introduces the student to libraries for OpenCL. The student familiarized themselves with the concept of Libraries for OpenCL. The laboratory was divided into two parts. Part 1 involved designing a 16-bit ripple carry adder in VHDL and generating a test bench to test the functionality of the same. Part 2 involved creating a new entity and instantiating a component of the ripple adder within it and following the OpenCL compilation and usage instructions guide to run and test the circuit.

1 Introduction

Lab 2 introduces the student to OpenCL and ensures that each student can successfully use OpenCL and familiarize themselves with the concept of Libraries for OpenCL. The lab is divided in two parts. Part 1 involves designing a16-bit ripple carry adder in VHDL. A full adder will first be designed as a behavioral circuit and then used as a component to generate the ripple carry adder. Part 2 involves creating a new entity and instantiating a component of the ripple carry adder within it. The student will then follow OpenCL compilation and Usage instructions guide to run and test the circuit.

2 Design of the 16-bit Ripple Carry Adder

In the first part of the laboratory, a 16-bit ripple carry adder was designed. A full adder was first designed as a behavioral circuit and then used as a component for the 16-bit ripple adder. Figures 1,2,3 and 4 below (under Figures and Tables) show the full adder circuit, the full adder symbol, the full adder truth table and the 4-bit ripple carry adder circuit respectively. A karnaugh map was generated from the truth table in Figure 3 to help create SOP equations for the output bits C_{out} and *sum*. Figures 5 and 6 below show the karnaugh maps that were generated for the C_{out} and *sum* bits respectively and their associated SOP equations that were used in the design of the full adder component. The 16-bit ripple adder was then designed in VHDL as a new entity and a test bench was created to simulate the 16-bit ripple adder logic. Figure 8 below, under figures and tables, shows the output of the simulation test bench.

2.1 Simulation

The simulation involved setting a to "0000000000000001" and b to "0000000000000000" initially with C_{in} set to 1. As shown in Figure 7 below, the inputs a and b were then changed for various numbers to test the ripple carry adder. To test the C_{out}, the final test involved a being set to "1111111111111111", C_{in} set to '0' and b set to "0000000000000001". The output *sum* and C_{out} were as expected as seen from the simulation diagram in Figure 8 below.

2.2 Figures and Tables

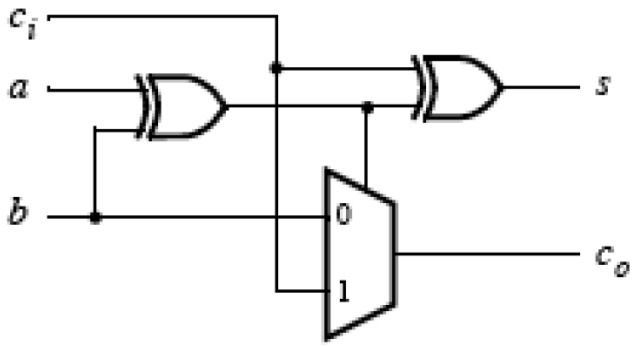

Figure 1: Full Adder Circuit

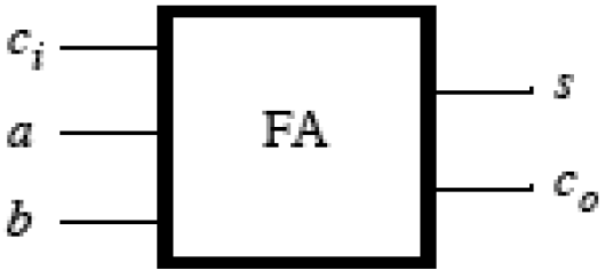

Figure 2: Full Adder Symbol

b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 3: Full Adder Truth Table

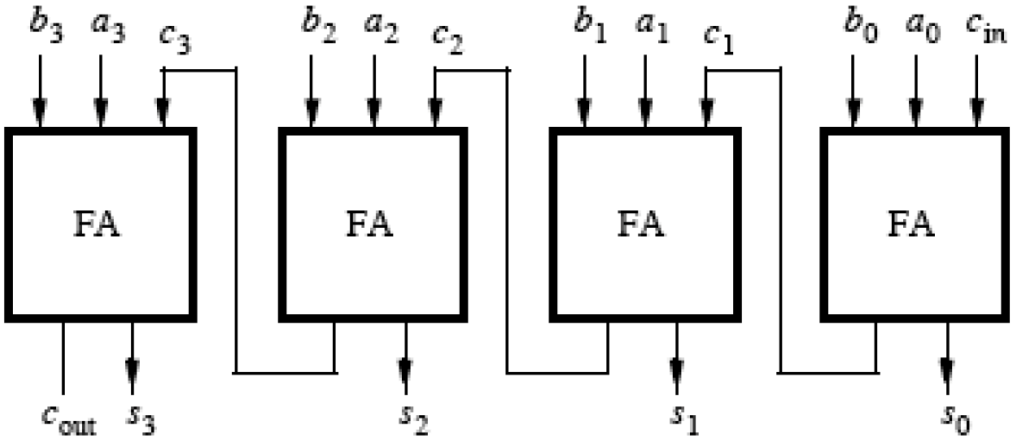

Figure 4: Four Bit Ripple Carry adder.

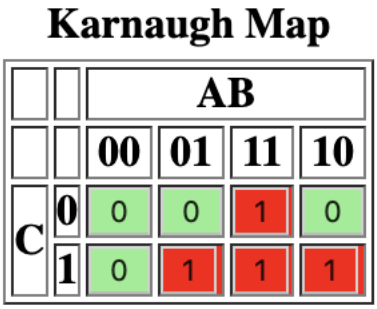

Figure 5: K-map for C_{out} [1]

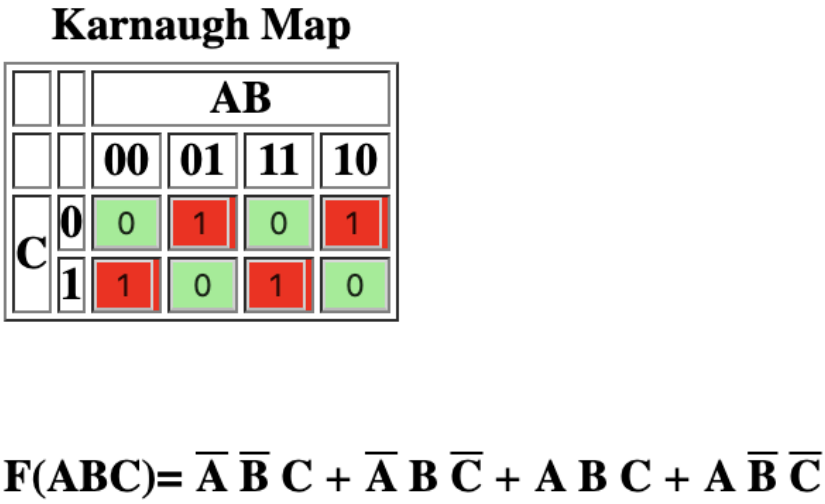

Figure 6: K-map for *sum* [1]

```
BEGIN
        -- code executes for every event on sensitivity list
        Cinr   <= '1';
        Ar     <= "0000000000000000";
        Br     <= "1000000000000000";

        wait for 100ns;

        Ar     <= "1000000000000000";
        Br     <= "0000000000000001";

        wait for 100ns;

        Ar     <= "1000000000000001";
        Br     <= "0000000000000001";

        wait for 100ns;

        Ar     <= "1111100000000000";
        Br     <= "0000000000000001";

        wait for 100ns;

        Ar     <= "0111111111111111";  --this will test is the intermediate caryy bits are working
        Br     <= "0000000000000001";  --output should be "1000000000000000"

        wait for 100ns;

        Cinr   <= '0';
        Ar     <= "1111111111111111";  --this will test for the carry out bit.
        Br     <= "0000000000000001";  --output should be "0000000000000000" and Coutr should be '1'

        wait for 100ns;

 WAIT;
END PROCESS always;
```

Figure 7: Code Snippet for 16-bit Ripple Adder Simulation

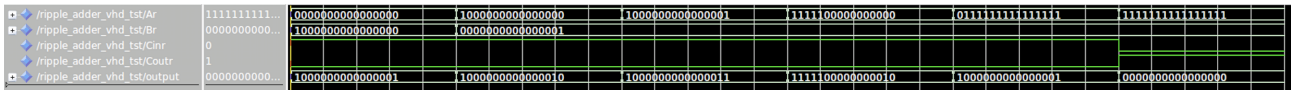

Figure 8: Simulation output for the 16-bit Ripple Adder

3 Design of the 16-bit ripple carry adder in OpenCL

Part II of the lab involved using the ripple adder designed in part I as a component of the OpenCL design. The 16-bit ripple adder was instantiated and port mapped with the carry bits ignored(i.e. C_{in} was set to '0' and C_{out} was set to OPEN). The required .vhd files for the OpenCL compilation and execution were then copied over into the device folder of the OpenCL project. Their respective file names were added to the .xml file inside that same folder and the project was compiled successfully. An SD card was required to store the executable so that it can be ran on the board.

4 Results

Overall the circuit design worked as expected both in simulation and on the board. The result in minicom was "Output Valid" signifying that the ripple adder worked as expected.

5 Conclusion

The second lab served as an introduction to programming the FPGA devices using OpenCL libraries. The lab teaches the student on how to set up and compile both the host code and the kernel code using the OpenCL libraries. It also teaches the student on how to use the imaged SD card to run the executable file on the FPGA device through the minicom terminal.

References

[1] C. Poly. Karnaugh map explorer 2.0, 2011.