VLSI Assignment - 3 Report

Ahish Deshpande

October 2, 2019

Contents

1 Aim

The aim of this assignment is to design ASIC(from Verilog code till GDSII File) using Cadence and perform full FPGA Flow(from Verilog to Dumping code into FPGAs) for the following Digital Blocks:

1. **1-Bit Full Adder**
   - Design using Logic gates
2. **4-Bit Ripple Carry Adder**
   - Design using 1-bit full adders
3. **16-Bit Ripple Carry Adder**
   - Design using 4-bit full adders
4. **4-Bit Carry Look-Ahead**
   - Design using logic gates
5. **16-Bit Carry Look-Ahead**
   - Design using 4-bit CLA's

2 Introduction

For this assignment, **Xilinx ISE** and **Cadence** will be used to do the following:

1. **Xilinx ISE**
   (a) Create and submit **Verilog Code**, **Testbench** & **Implementation Constraints File**
   (b) Take screenshots of **RTL Schematics** & **Timing Diagram** depicting all logic gates present in the design and all possible combinations of test inputs, respectively
   (c) Report **Area**, **Power** & **Delay** of the design
2. **Cadence**
   (a) **Genus**
      i. **Area**, **Power** & **Timing** Reports
      ii. **Netlist** generated
      iii. Screenshot of **RTL Schematic**
   (b) **NCLaunch**
      i. Screenshot of **Timing Diagram**
   (c) **Innovus**
      i. **Area** & **Power** report
      ii. **Timing**(pre-place, post-place, post-rout, & sign-off) report
      iii. Screenshot of **Physical Design**

3 Procedure

The following procedure will be followed:

1. Write the required Verilog code and testbench using **Xilinx ISE**
2. Using the above, generate the RTL Schematics and Timing Diagram in Xilinx
3. The Area, Power & Delay reports can also be generated now
4. Using the Verilog code already written, use **Cadence NCLaunch** to generate a Timing Diagram
5. Next, use **Cadence Genus** to generate Area, Power and Timing reports of the gates and combinational circuits generated. The netlist will also be generated here
6. Use the generated netlist with **Cadence Innovus** to optimize the design. The Timing reports for pre-placement, post-placement, pre-rout, post-rout & sign-off are then generated along with the Power and Area reports. The optimized netlist is also generated.
7. **Cadence Innovus** will also generate the physical design

4 Result & Discussion

The attached screenshots show the values for the function for all values of the input.
Tabulating the values observed in the reports from Genus(without optimization) and Innovus(with optimization):

Design	**Genus**		**Innovus**		
	Area*(units)*	**Power***(nW)*	**Area***(units)*	**Power***(nW)*	**Delay***(ns)*
FullAdder1Bit	20	691.602	19.679	393.2	0.368
RippleAdder4Bit	79	3145.34	78.716	2168	1.146
RippleAdder16Bit	315	14330.95	314.87	10680	4.22
LookAheadAdder4Bit	86	3590.6	86.28	2505	0.896
LookAheadAdder16Bit	516	21890.59	516.2	18450	3.872

1. We observe that the **power** dissipated by the design is proportional to the **area/size** of the design.
2. We can also observe that the power consumed by the design **can be significantly reduced** by optimizing the design.
3. We can also see how using the Look Ahead Carry adder reduces the delay in the output significantly as it does not have to wait for the carry from the previous bit.
4. We observe that in trying to reduce the delay, the power and area of the circuit increases.

5 Conclusion

We can conclude that:

- The required designs were successfully implemented using **Xilinx ISE** and **Cadence**
- We can construct more complicated circuits using self-made building blocks
- The power dissipated by the circuits is **proportional** to the size of the design
- The power consumption of the design can be significantly reduced by **optimizing the design** appropriately
- The Carry Look Ahead Adder reduces the delay time of the output, compared to the Ripple Carry Adder.

6 References

1. *www.asic-world.com*
2. *www.xilinx.com*
3. *www.cadence.com*
4. *electronics.stackexchange.com*
5. *Tutorials and Lab Sessions*