Lab 9: Standard Cell Based ASIC Design Flow

Adam Sumner

Illinois Institute of Technology

ECE 429-01

Lab Date: November 9th, 2015 **Due Date:** November 18th, 2015

1 Introduction

The purpose of this lab is to introduce the student to the concept of standard cell based ASIC design flow. The student will follow a tutorial to create the layout of an accumulator and then verify its correctness using equivalence checking.

2 Theory/Pre-Lab

2.1 Theory

Standard cell based ASIC design flow automatically synthesizes a chip layout from a register transfer level (RTL) description of a chip. The design flow utilizes the standard cell library to synthesize a chip layout according to design constraints including cost, performance, power consumption, etc. While this may seem like a nice tool to completely replace manual chip design, in practice, the tool will seldom generate a satisfactory design in the beginning. It is necessary for chip designers to understand the various steps of the design flow to guide the tool through multiple iterations before an optimum solution is found. In general, the standard cell based ASIC design flow consists of two steps: Logic Synthesis and Physical Design.

2.1.1 Standard Cell Library

A standard cell is a logic gate with a layout designed for a specific fabrication process. Standard cells for different types of logic gates and different sizes of the same gate are generally grouped into a standard cell library. Cells from the same library are most likely with the same height. Since the layout of a standard cell is known, characteristics of the cell can easily be obtained via SPICE simulations.

2.1.2 Logic Synthesis

The purpose of logic synthesis is to transform RTL descriptions of chip functionality into a netlist consisting of standard cells. Currently, it is very common to have datapath or arithmetic operations in a RTL description. The logic synthesis tool should first implement these operations as boolean logic. After this synthesis, the tool performs generic logic optimizations on the design without any information form the standard cell library. After this, the netlist is generated with standard cells that have the same boolean functionality as the one generated by the tool in the beginning.

2.1.3 Physical Design

The purpose of the physical design is to create a physical implementation of the netlist consisting of standard cells. Due to the fact that every standard cell consumes silicon surface to form transistors, no two cell should overlap. Therefore, the most straightforward method is to place cells row by row, using metal layers to route wires.

2.2 Pre-Lab

The pre-lab involved familiarizing with Tutorial IV and the verilog code and testbench for the 8-bit accumulator. This was all done successfully.

3 Implementation

3.1 Schematics

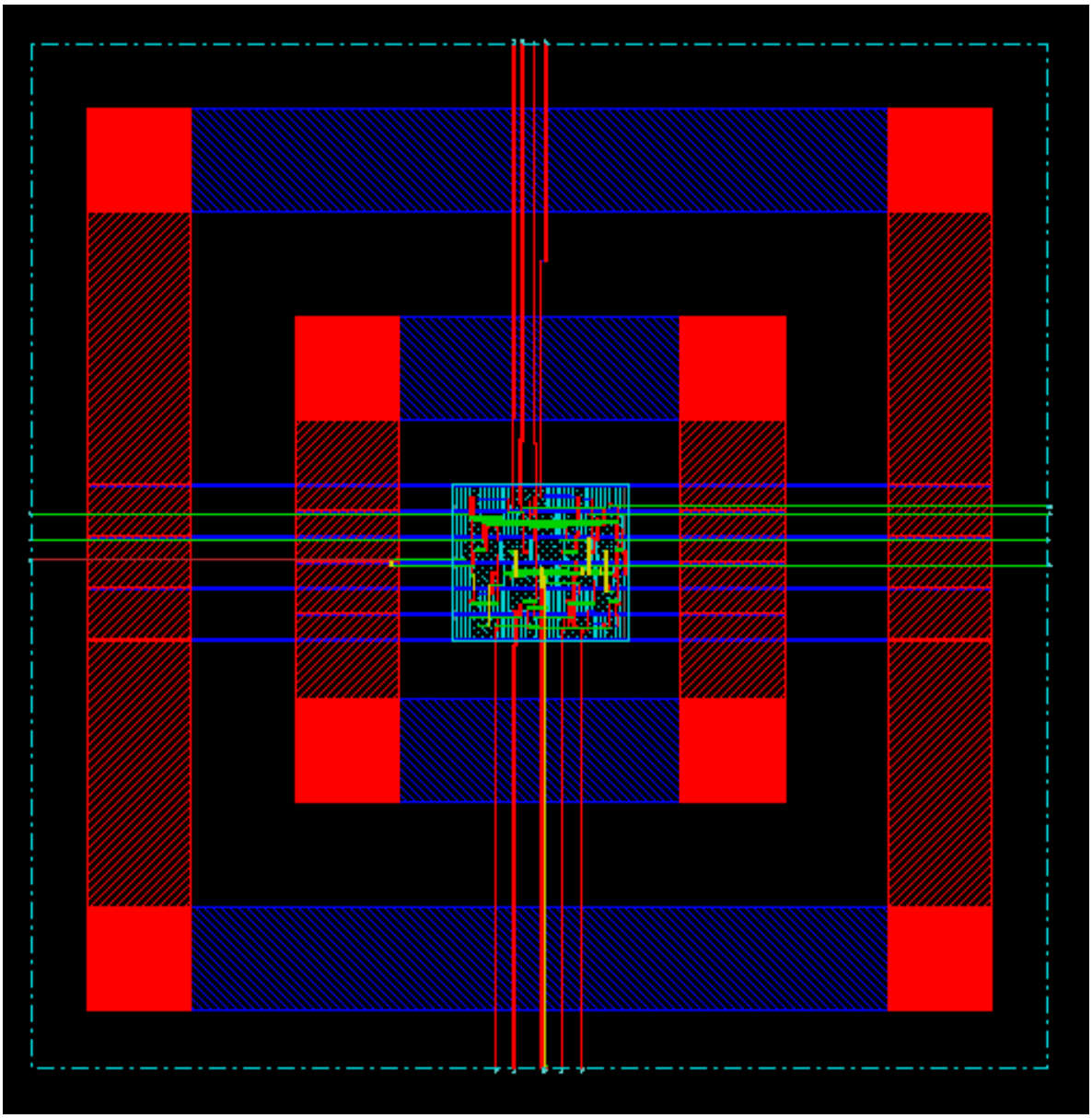

Figure 1: Generated Layout Overview

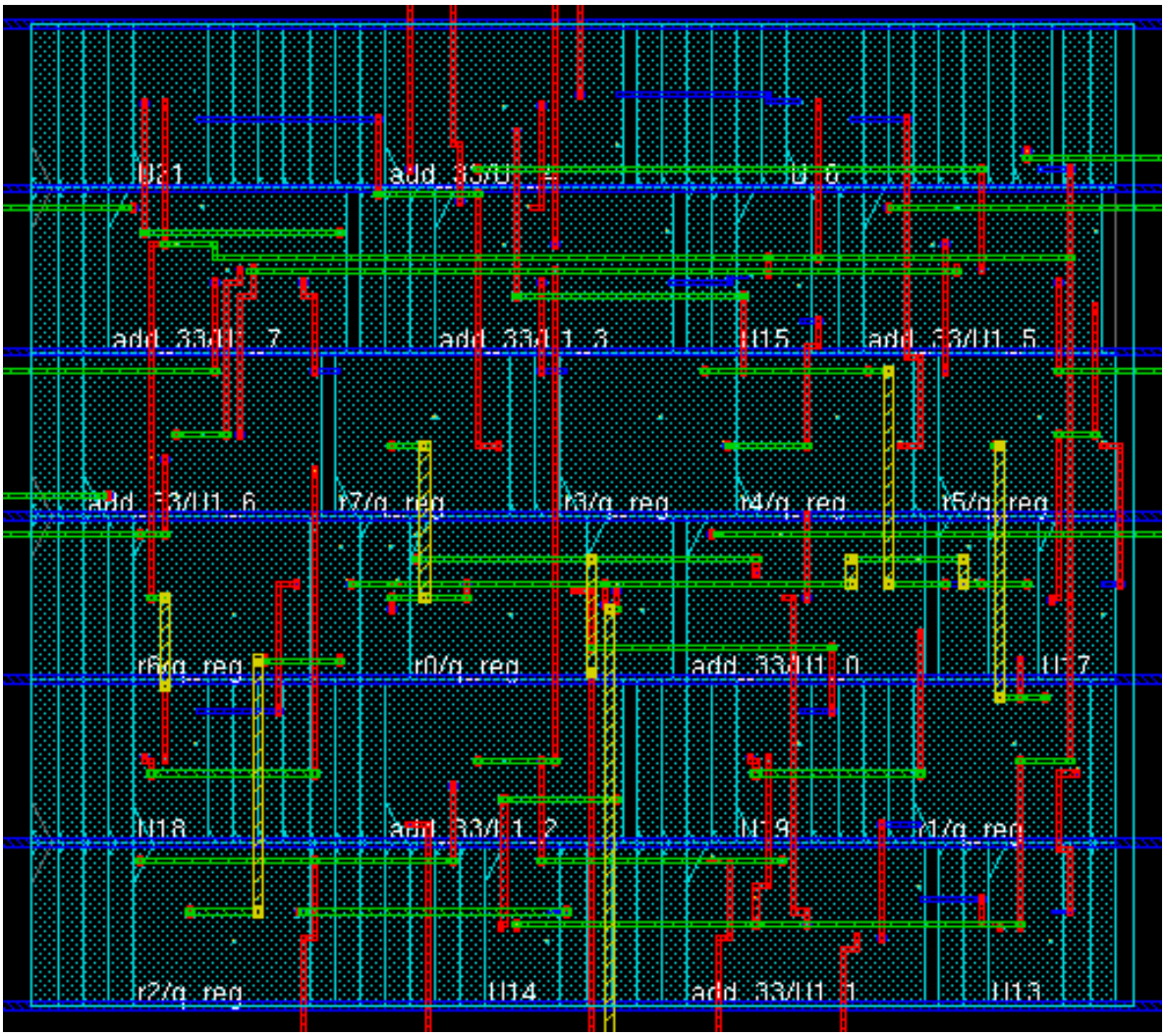

Figure 2: Generated Layout Closeup

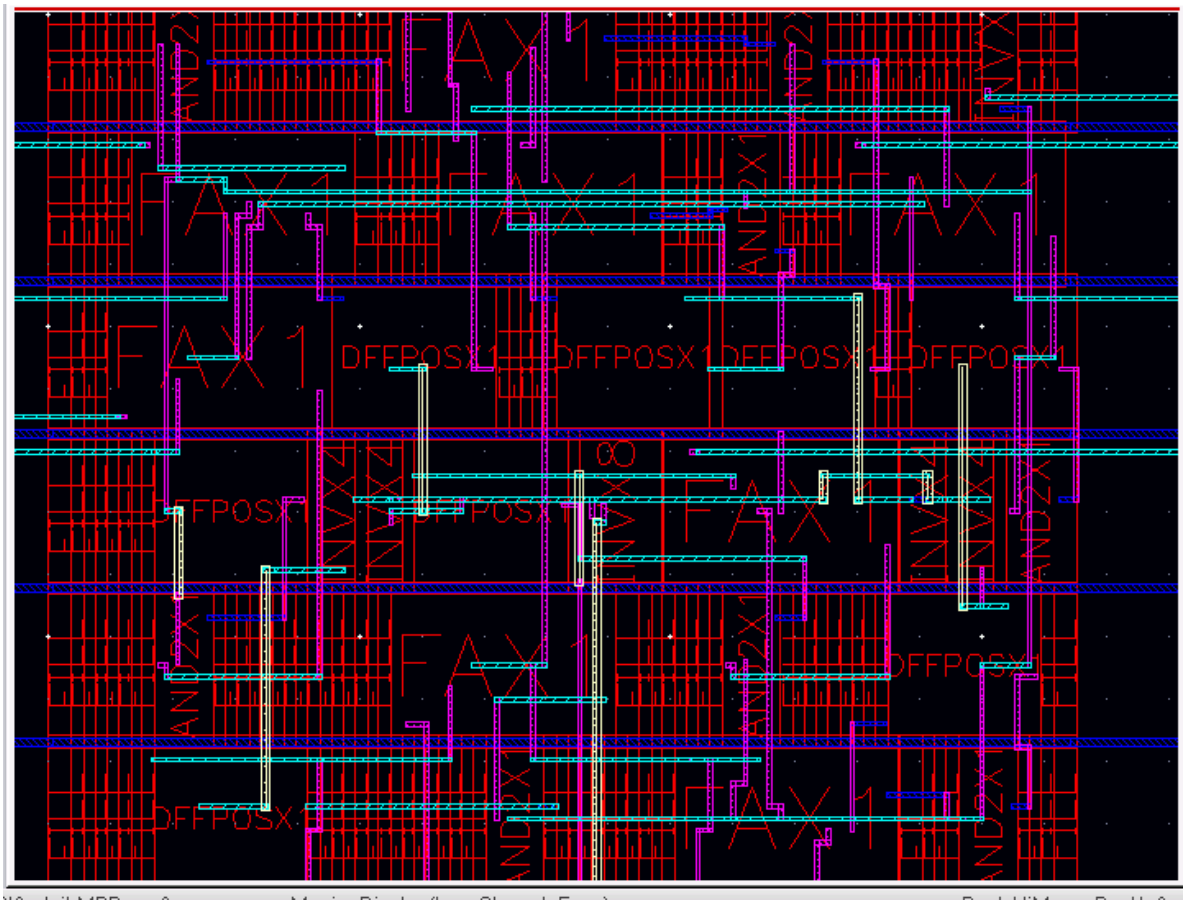

Figure 3: Generated Layout Closeup 2

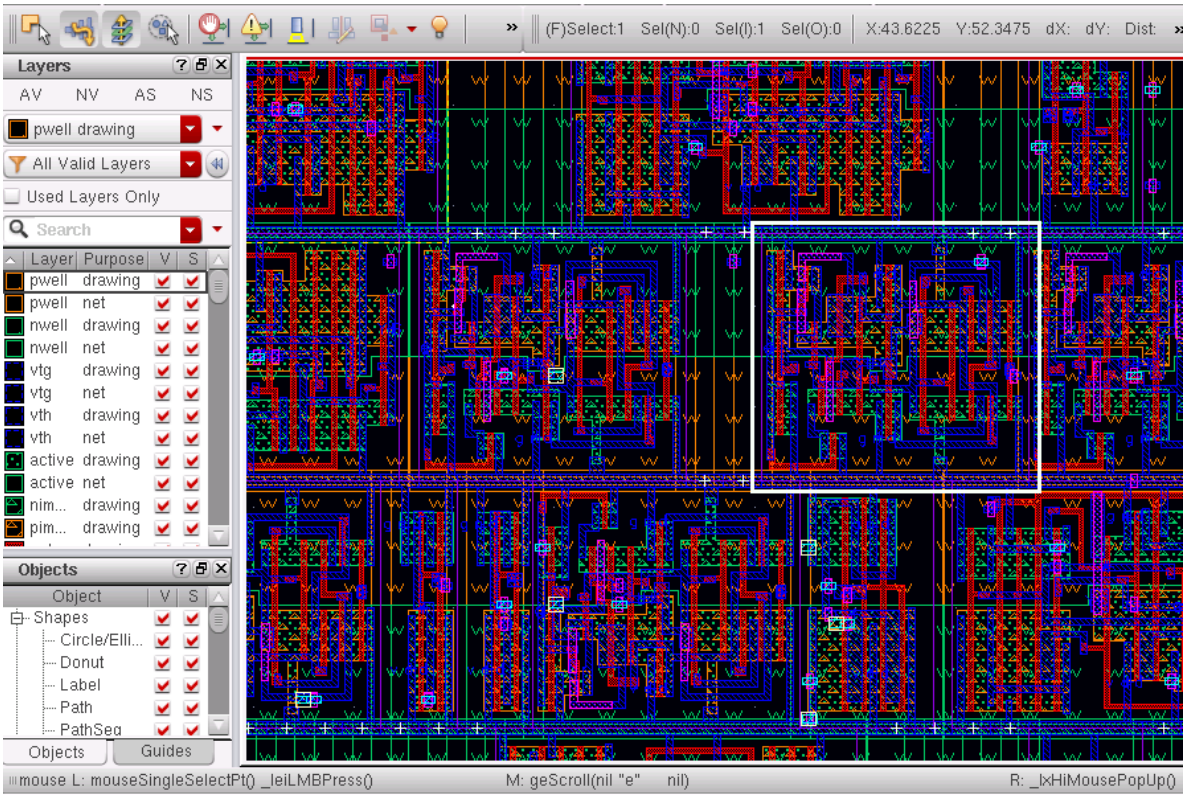

Figure 4: Detailed Layout

3.2 Procedure

This lab involved following the instructions in Tutorial IV. This entailed checking verilog files for equivalence, generating a layout of the design, and then checking the layout for equivalence using Formality.

3.3 Results

```
Compiling source file "accu_test.v"
Compiling source file "accu.v"
Highest level modules:
stimulus

At Time:                   5  Accumulator Output=  x
At Time:                  10  Accumulator Output=  0
At Time:                  15  Accumulator Output=  0
At Time:                  20  Accumulator Output=  1
At Time:                  25  Accumulator Output=  1
At Time:                  30  Accumulator Output=  2
At Time:                  35  Accumulator Output=  2
At Time:                  40  Accumulator Output=  3
At Time:                  45  Accumulator Output=  3
L23 "accu_test.v": $finish at simulation time 5000
0 simulation events (use +profile or +listcounts option to count)
CPU time: 0.0 secs to compile + 0.0 secs to link + 0.0 secs in simulation
End of Tool:    VERILOG-XL      08.20.001-p   Nov 15, 2015  03:52:19
```

Figure 5: Initial Verilog Verfication

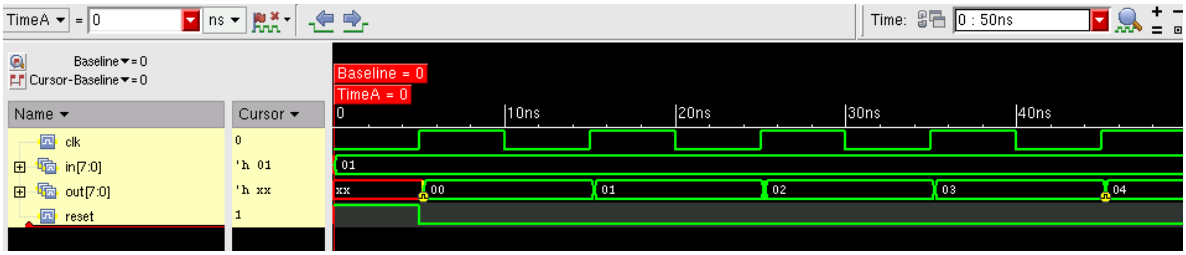

Figure 6: Initial Waveform Verification of Verilog

```
encounter 1> reportGateCount -limit 0
Gate area 2.8158 um^2
[0] accu Gates=56 Cells=30 Area=158.6 um^2
encounter 2>
```

Figure 7: Area Report

```
Total Power
--------------------------------------------------------------------------------
Total Internal Power:      0.2402      57.88%
Total Switching Power:     0.1738      41.89%
Total Leakage Power:    0.000954     0.2299%
Total Power:       0.415
--------------------------------------------------------------------------------

Group                    Internal  Switching  Leakage  Total  Percentage
                         Power     Power      Power    Power  (%)
--------------------------------------------------------------------------------
Sequential                0.1531     0.0291  0.0004398  0.1827       44.02
Macro                          0          0          0       0           0
IO                             0          0          0       0           0
Combinational             0.05951    0.03121  0.0004403  0.09116      21.97
Clock (Combinational)     0.02752     0.1135  7.386e-05   0.1411      34.01
Clock (Sequential)             0          0          0       0           0
--------------------------------------------------------------------------------
Total                     0.2402     0.1738   0.000954   0.415         100
--------------------------------------------------------------------------------

Rail                Voltage  Internal  Switching  Leakage  Total  Percentage
                             Power     Power      Power    Power  (%)
--------------------------------------------------------------------------------
vdd                     1.1   0.2402     0.1738   0.000954   0.415         100

Clock                    Internal  Switching  Leakage  Total  Percentage
                         Power     Power      Power    Power  (%)
--------------------------------------------------------------------------------
clk                       0.02752     0.1135  7.386e-05   0.1411      34.01
--------------------------------------------------------------------------------
Total                     0.02752     0.1135  7.386e-05   0.1411      34.01
--------------------------------------------------------------------------------


--------------------------------------------------------------------------------
*       Power Distribution Summary:
*              Highest Average Power:              clk__L1_I0 (INVX8):          0.04083
*              Highest Leakage Power:                r1/q_reg (DFFPOSX1):     5.498e-05
*              Total Cap:      4.01408e-13 F
*              Total instances in design:    30
*              Total instances in design with no power:     0
*               Total instances in design with no activty:     0

*              Total Fillers and Decap:     0
--------------------------------------------------------------------------------
```

Figure 8: Power Report

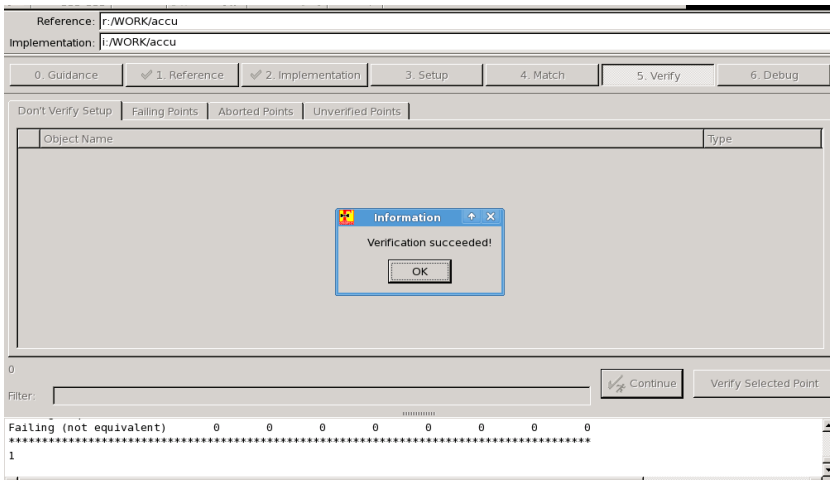

Figure 9: Formality Verification

3.4 Discussion

Figures 1-4 show the generated layout design from the accumulator verilog file. From the results gathered in Figures 5-9, it is clear that the layout was successfully generated and has equivalent functionality to the verilog file.

3.4.1 Questions

What is a standard cell?
It's a group of transistors and interconnects that provide boolean logic functions or storage functions. They are used in conjunction with one another to produce complex circuits such as adders or registers.

What are the differences among accu.v, accu.vh, and final.v ?
`accu.v` is the original verilog file, `accu.vh` is the generated logic synthesis netlist, and `final.v` is the generated verilog file from the layout created in Encounter.

How does the area of your design change after place & route?
This information is shown in Figure 7. It lists the gate area, how many gates there are, how many cells there are, and the total area. The area is optimized to be as small as possible.

How does the timing of your design change after place & route?
Because the area of the design was optimized to take up the minimum amount of space, the timing of the design is also optimized to be as fast as possible.

Why do we need to use Virtuoso to generate the final layout for fabrication? What information is available from final.gds2?
`final.gds2` only contains the placement of the cells and metal interconnects. It doesn't contain any layout details inside the cells, for example, wells and poly. This is why it's necessary to use Virtuoso to generate the final layout.

4 Conclusions

Overall this lab was a success. Tutorial IV was successfully followed, the layout of an accumulator was generated, and its functionality was checked for accuracy. Standard Cell Based ASIC Design Flow was successfully explored.