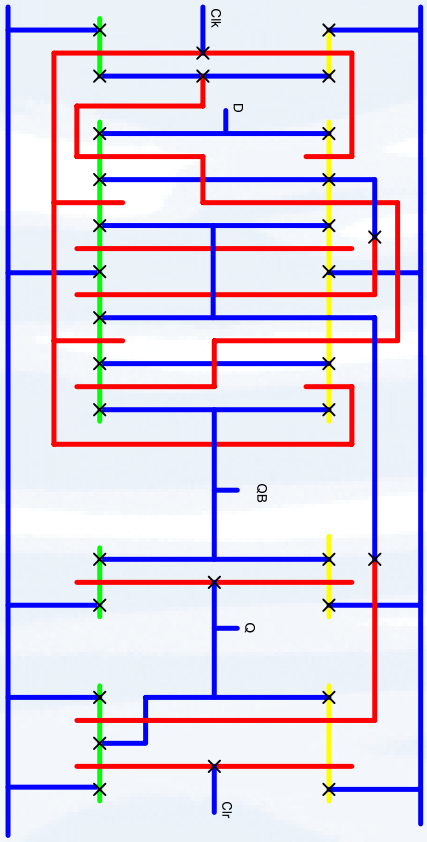
VLSI Scarlet Letters
— Final Design Report —

Report Due Date: Monday June 5th 2006



The Ohio State University

EE 683 - Senior Design (II)

VLSI Scarlet Letters Team Members:

-David W. Adams II -Steve Jocke

-Kristoffer Schacker -Adam Grether

Table of Contents

List of Figures

Executive Summary

Through design reuse and getting all Cadence users up to speed as quickly as possible, we will enable OSU to become better at Cadence. While all of us at OSU grown into VLSI, and our skills in Cadence become more evolved, that only leaves beginners further behind. In order to try and keep the newbie's on track, we must create tutorials and information that is helpful to all users. We also want to be able to reuse designs which other OSU students have created. Upgrading past designs and reusing designs will waste less time and benefit all users at our university. To this end, The VLSI Scarlet Letters have tried to create content which others can use, and content which aids all users of Cadence.

Purpose

The VLSI Scarlet Letters are tasked with creating digital cells to add to *The Ohio State University Digital Cell Library*. Our goal is to create a diverse set of digital cells to add to the current library. As of now the library is quite minimal, when we are done we plan to have a more robust library. Along with the cells that were created for the library in 582, the Scarlet Letters created a D Flip-Flop to replace the current version in the OSU library.

Problem Statement

One of our main tasks was to create a new D Flip-Flop for the OSU Digital Library. Currently the D Flip-Flop in the library contains Metal 2 and is an inefficient design. Our second main task is to generate tutorial in how to create clear and concise design content. These Tutorials include how to plot Cadence .out files in Matlab, Remote access to the Unix server so one can used Cadence anywhere, and the Importing and Exporting of libraries so we can reuse designs.

D Flip-Flop Design Approach

Since the old D Flip-Flop contained Metal 2, we knew we wanted to get rid of that layer of metal as a primary goal. Also, we wanted a Flip-Flop that has an asynchronous clear, and to minimize the footprint of the layout as much as possible. Using simulations we knew that we would be able to fully inspect the operation of the Flip-Flop.

D Flip Flop Design

Our initial design for the D Flip-Flop with asynchronous clear was done using standard logic gates. The design was logically correct using a combination of two and three input NAND gates as well as an inverter for the clock signal input to realize the functionality of the circuit. The fault of the design was the overall efficiency. To implement the initial design it required the use of 46 transistors, this would incorporate a large footprint to layout such a simple logic function.

After a few weeks in ECE721 we learned about pass transistors. Using that concept along with latches we were able to reduce the design to 20 transistors. This not only reduced the footprint of the layout but consumes less than half the power of the initial design.

The next stage of development was converting the transistor level schematic to a layout configuration. Seen in Figures 2 and 3, of Appendix A, is the original stick diagram for the Flip-Flop. While doing the layout in Cadence several changes were made to decrease the cell footprint along with compiling more detailed layout rules learned from the Design Rule Check (DRC). One important design change was the merging of a metal “H” connection. Seen in Section 1 of the stick diagram layout are two sections that were merged together, as noted on the layout. As the layout progressed small minor changes were made along the way to shrink the cell or to route poly lines.

Figure 5 is the final stick diagram representing the layout of the D Flip-Flop, and the physical Cadence layout can be seen in Figure 6.

Flip-Flop Analysis & Evaluation

Testing of the transistor level schematic, seen in Figure 4, was done in Cadence using various test states to simulate the run conditions and check the effectiveness of the asynchronous clear. The table in appendix A, Figure 7, has a detailed setup of the test states to reproduce our simulation results. The first simulation results were incorrect half of the time. This was due to the pass transistor bodies being tied high for PMOS then low for NMOS, as done with standard transistor schematic design, and not being tied directly to the Vdd! / Gnd! nets. After linking the bodies to the proper nets the simulation results function as expected. Seen below in Figure 1 is a simulation of the D Flip-Flop at the transistor level. As expected the Flip-Flop loads the correct value at a rising clock edge

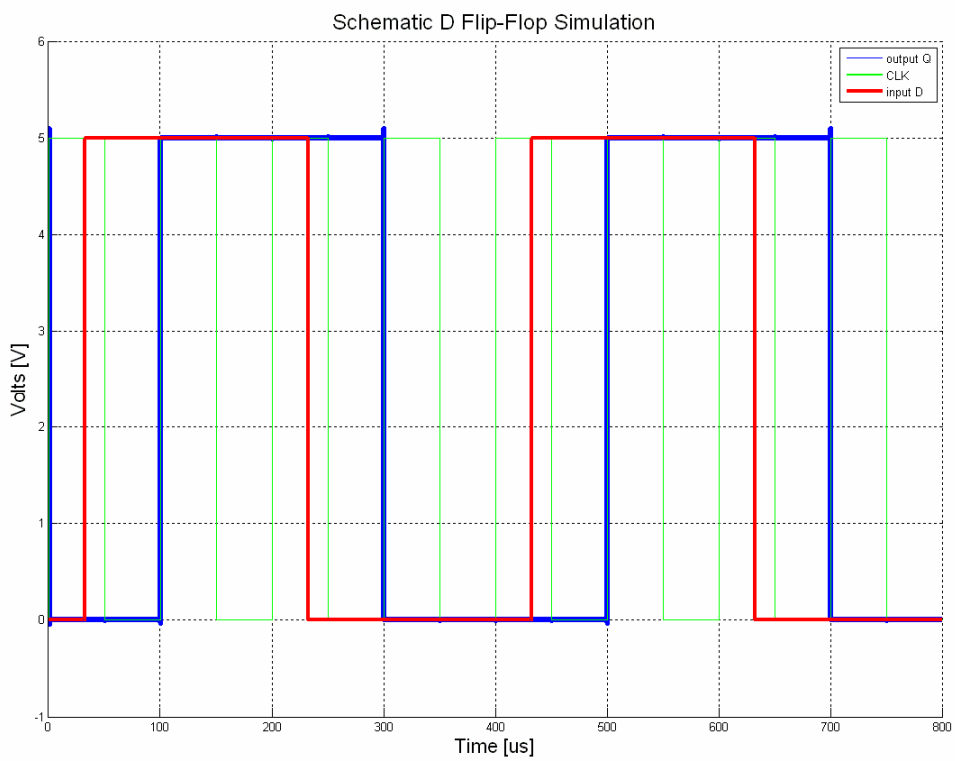


Figure 1 - D Flip-Flop Cadence Simulation

After completing the transistor schematic layout we proceeded to the layout design. The stick diagrams guided the D Flip-Flop layout along with the Design Rules Check (DRC). Having this tool allowed us to shrink the size of the D Flip-Flop without violating any of the MOSIS design rules. After completing the DRC we proceeded to the LVS, Layout Versus Schematic. In order to run the LVS the D Flip-Flop must be extracted to generate a netlist for the layout. While doing the extraction it will also calculate capacitances (based on layout choices), transistor sizes and numbers of other parameters. The LVS checks the layout by comparing the netlist generated by the layout and comparing it to that of the schematics'. This was the most difficult part of the design process. While DRC confirms that it can be manufactured correctly the LVS assures that it is connected properly for operation.

Correcting the 38 LVS errors was a slow process, and revealed multiple fine points of layout. Most of the errors were correctly once the Vdd! / Gnd! nets were properly accounted for. Our second issue was properly labeling pins and nets so that they would match that of the schematic. Our last, and most difficult, issue was that some nets were not matching properly. The first two were resolved by going by to the tutorials and make sure proper switches were activated. The last issue was an actual design error. While attempting to make the D Flip-Flop smaller, a critical piece of metal was removed. Without that piece of metal the pass transistors would not have been connected. Replacing that metal connection solved the remaining error and then gave us a successful LVS check.

Future Development

With the LVS completed the next stage of development is to use the Analog Environment to run simulations on the extracted layout. These simulations will give detailed timing information which will characterize the cell. Once the test simulations are completed the cell can be sent off to be fabricated. With the fabricated D Flip-

Flop testing can be done to see if actual performance reflects that of the simulated environment.

Importing D Flip-Flop for Continued Development

Included with this report are two files for importing the D Flip-Flop. The first is a GDSII file that contains the layout. Keep in mind this is only the layout and does not have the schematic that we performed the LVS with. The second file is the D Flip-Flop sub folder from the ECE683 library. Getting this file into a library is a bit more difficult but proves to be more useful if whomever continuing our work requires those files. Below lists a short procedure for pulling that file into a library. For the GDSII file simply follow the included tutorial for CIF files except for using a GDSII file.

1.) Copy the Dff_c folder into the ~/cadence/NCSU/your library name/

2.) Close the Library Manager

3.) Then reload Library Manager by going to “Tools” under ICFB and launching Library Manager. This will cause Library Manager to update its database of cells.

4.) If opening any one of the cell views causes an “unable to open for edit error” copy the entire cell to a new cell. The files provided are *not* locked and should not cause this error.

In order to pull cells out of a library just copy the entire folder from the library. Make sure that Cadence is not using those files because it will lock them. This is usually best done while Cadence is inactive to ensure the files remain unlocked.

Resources

Since this project did not require us to do any physical fabrication, we were able to design our Flip-Flop with very few resources. In developing our D Flip-Flop, we dealt mostly with software packages, university supplied computers / resources, and our

personnel. We also used the UV tutorials and Cadence libraries supplied by NCSU. Also, we used our faculty members for assistance when designing.

Personnel Tasks

Below is a listing of the specific tasks each member performed throughout the quarter in ECE 683.

- David W. Adams II – Mr. Adams researched the output files generated by Cadence and determined the feasibility for Matlab plotting routines. This task was competed and a tutorial was written. Dave has also been responsible for turning in group assignments and merging all documents produced by the VLSI Scarlet Letters. Dave was responsible for creating the Remote Unix Access Tutorial.

- Steve Jocke – Steve has worked mainly in the development and testing of the D Flip-Flop. He first started out by design the Flip-Flop at the transistor level and then testing it with multiple simulations. After the completion of simulations he developed the initial stick diagram layout of the Flip-Flop. With a completed stick layout he worked closely with Adam Grether in doing the layout in Cadence along with the DCR and LVS verification stages. Finally Steve researched on how to export data out of Cadence using the ICFB command line interface.

- Kristoffer Schacker – Kris has researched importing and exporting libraries in the Cadence environment. He has put together a tutorial and a report. His report and tutorial are step by step instructions on the process. Kris has also fostered development with other group members, and helped wherever needed.

- Adam Grether – Adam has researched layouts in Cadence by reviewing multiple layout techniques seen in other cells and tutorials. After his research he took the layout and entered it into Cadence. Working together with Steve Jocke they completed the LVS and DRC verification stages of the D Flip-Flop. Along with layout in Cadence Adam completed the original design the D Flip-Flop which was composed of logic gates.

Appendix A – D Flip-Flop Content

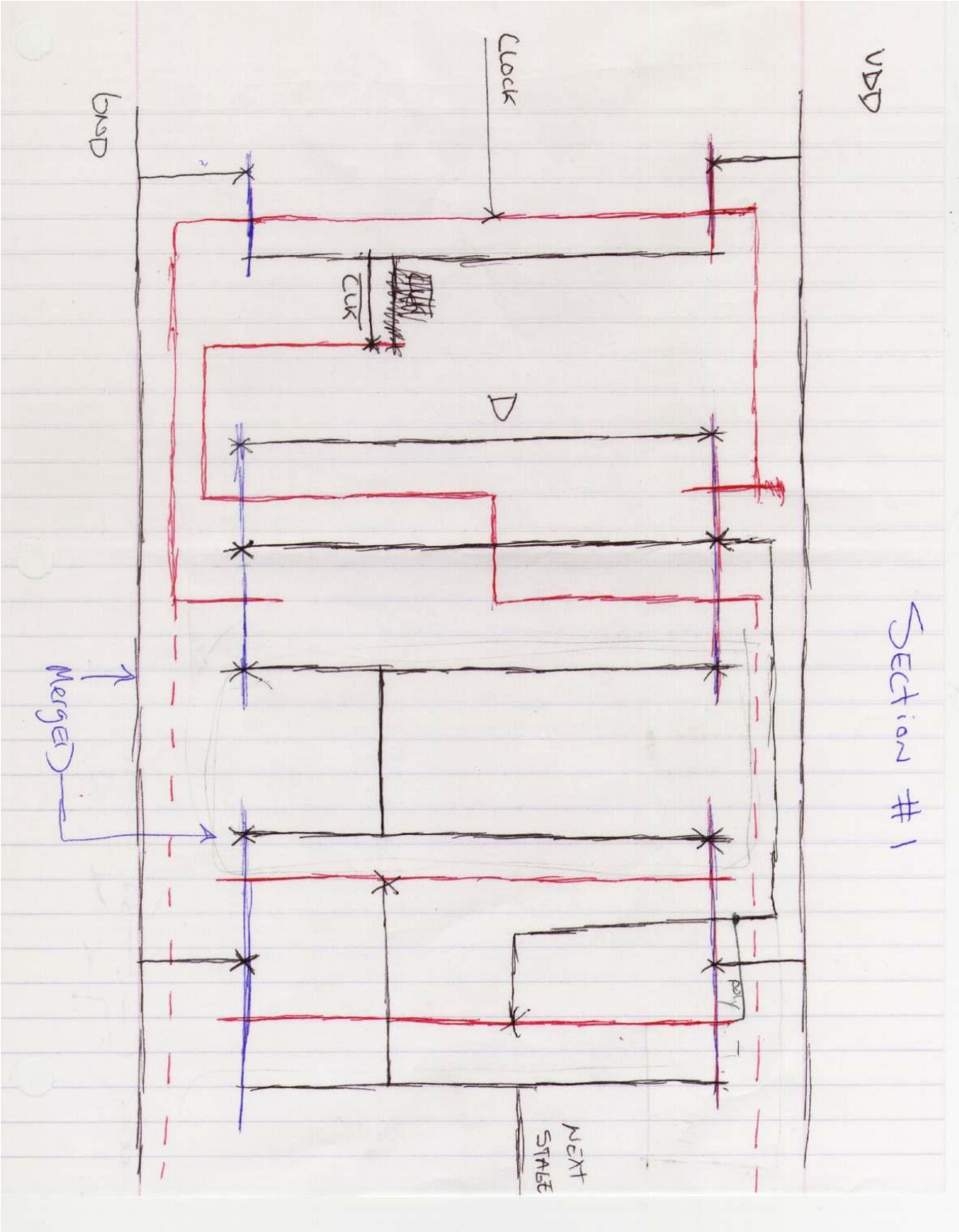


Figure 2 - Original Stick Diagram (1 of 2)

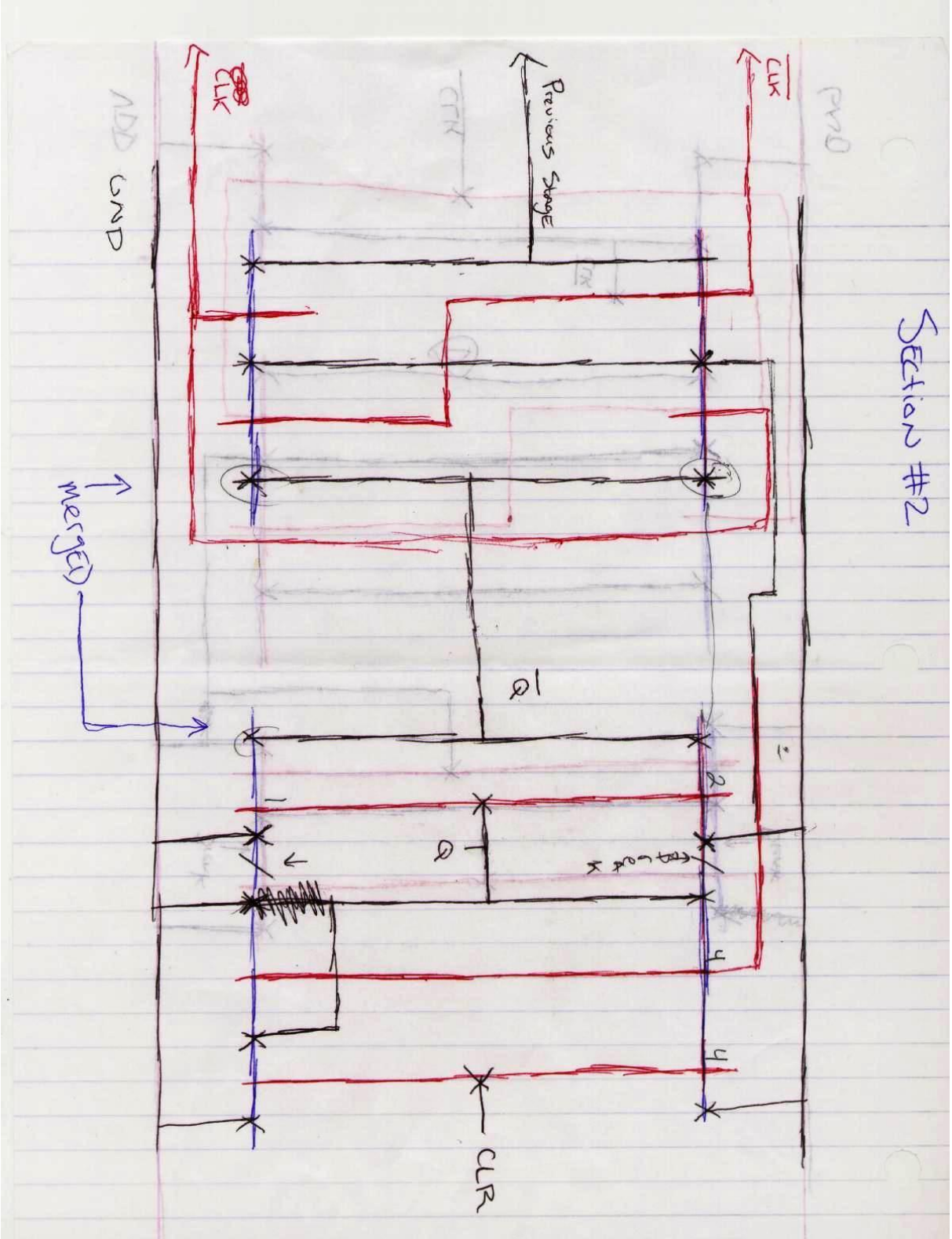


Figure 3 - Original Stick Diagram (2 of 2)

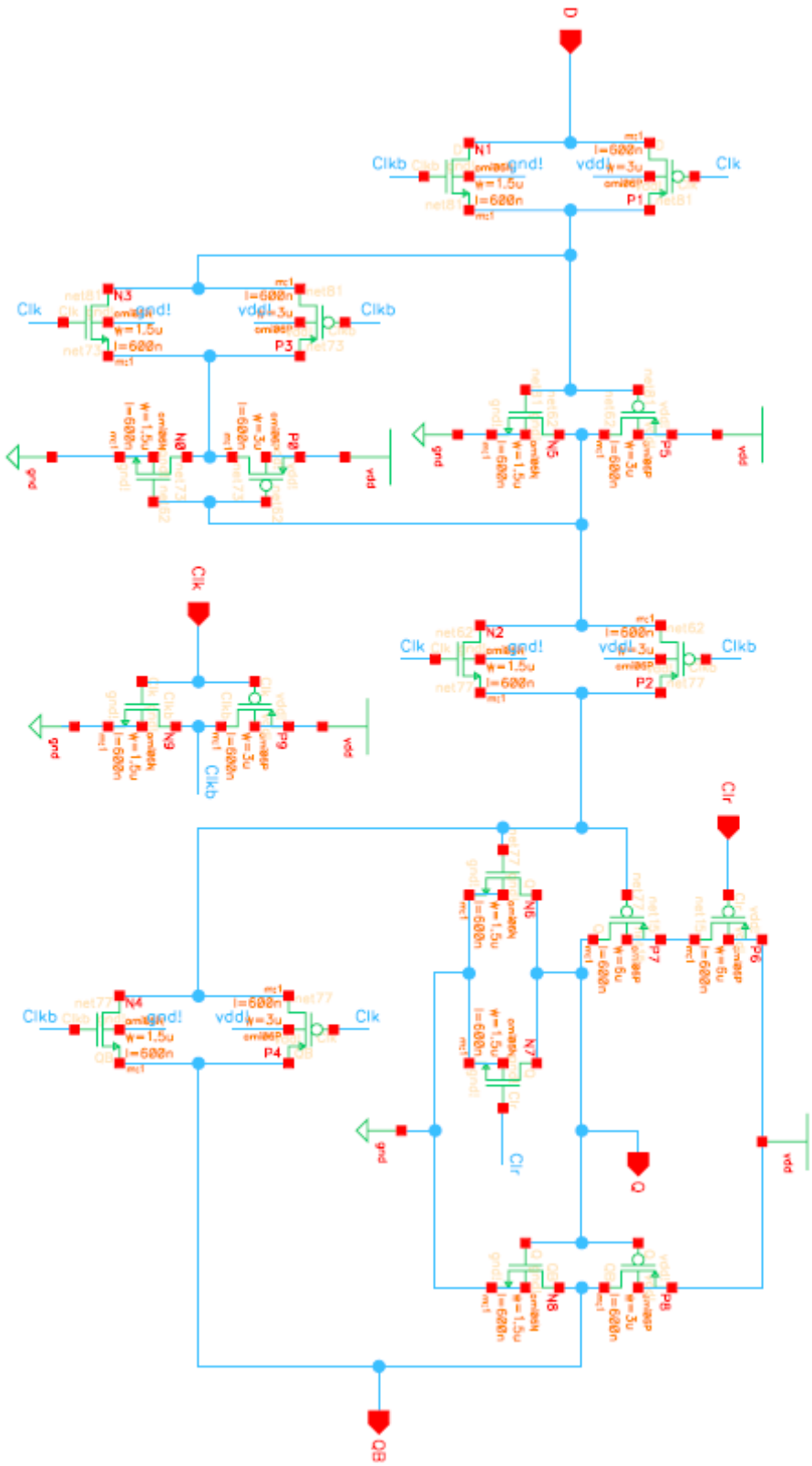


Figure 4 - D Flip-Flop Transistor Schematic

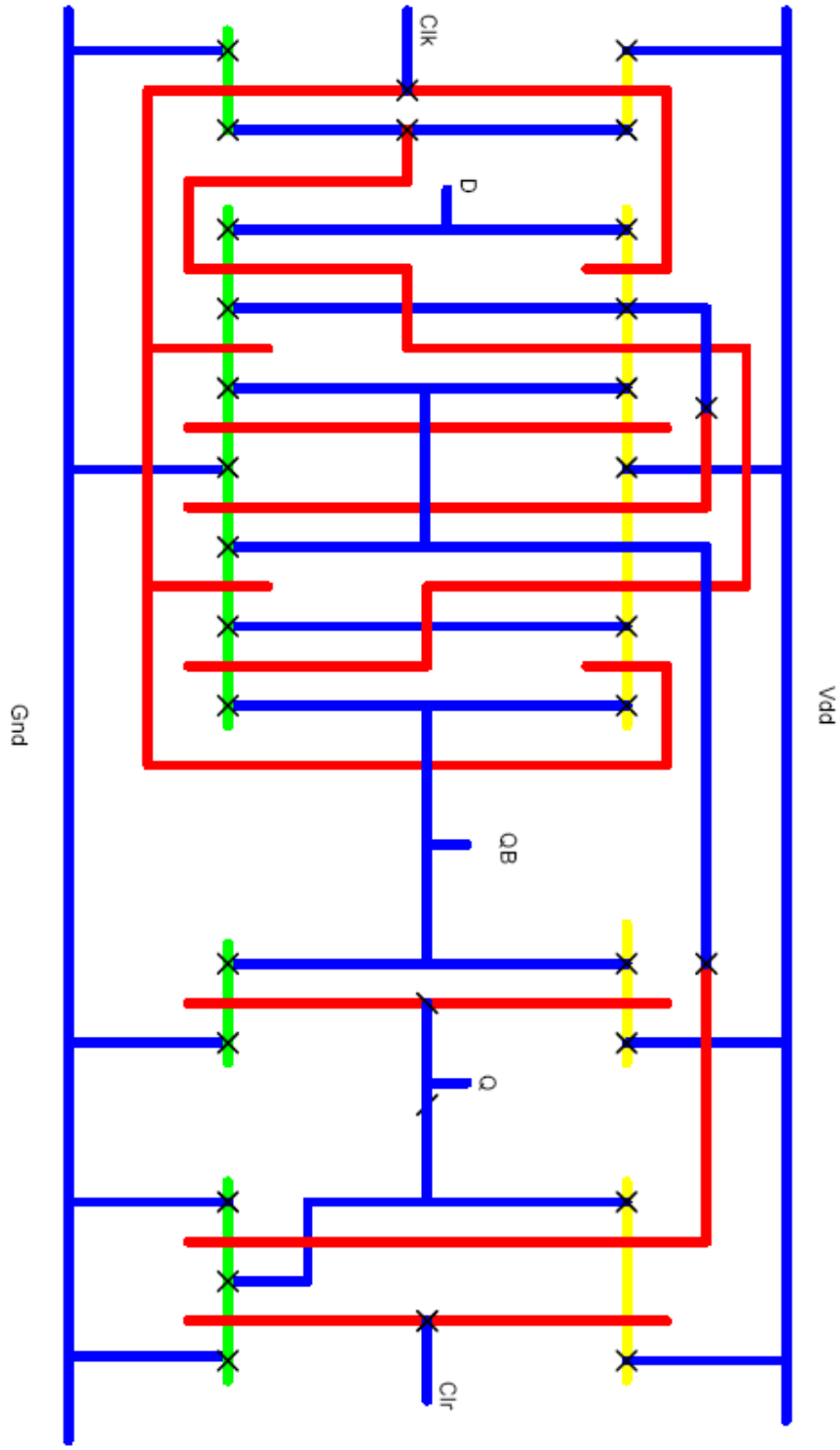


Figure 5 - D Flip-Flop Stick Diagram

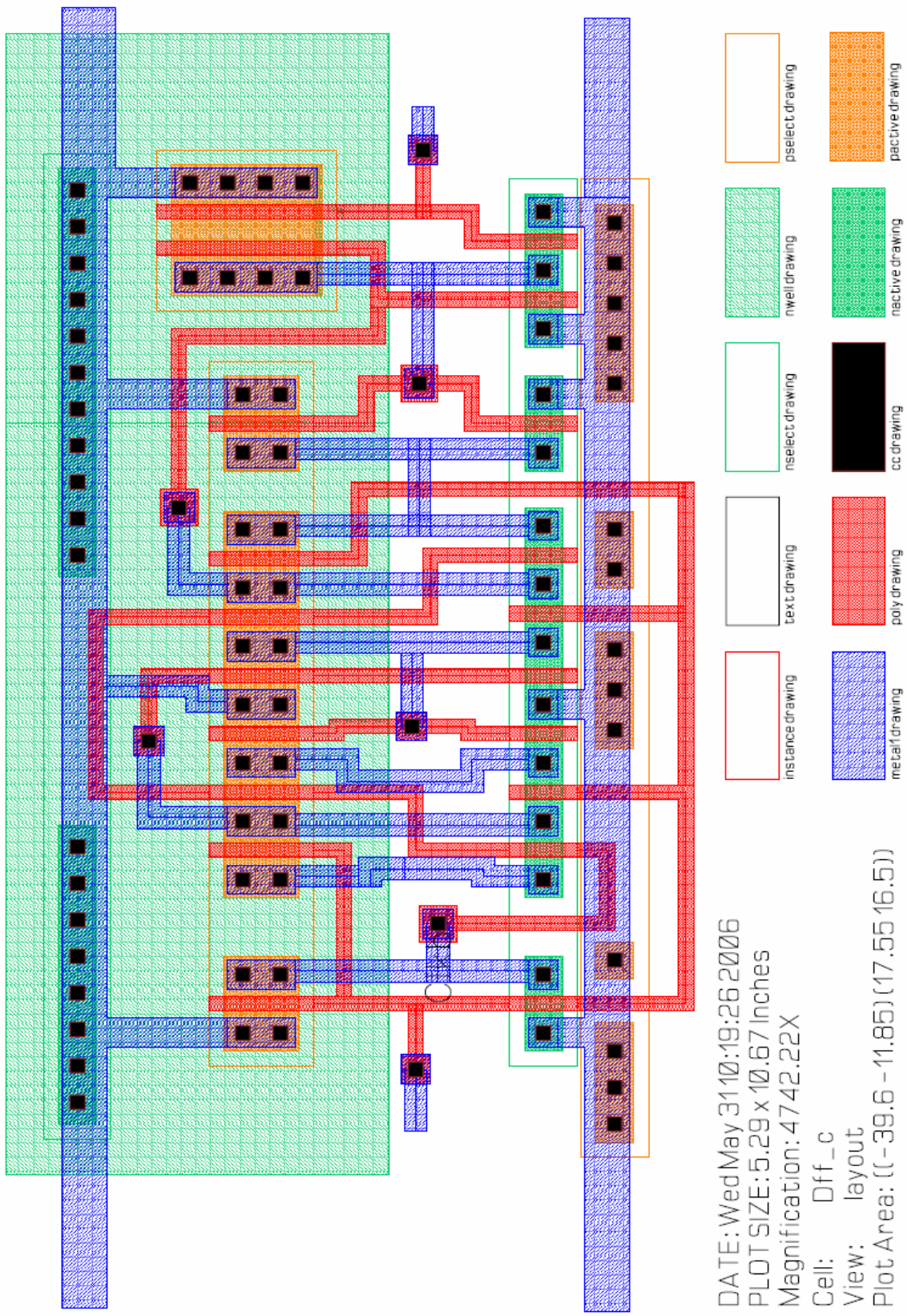


Figure 6 - D Flip-Flop Layout

State Name									
Plot									
Analysis Type	**Run Time**								
Transient	800 [uS]								
Inputs	**Function**	**Voltage 1**	**Voltage 2**	**DC Votlag**	**Delay Time**	**Rise Time**	**Fall Time**	**Pulse Wid**	**Pulse Period**
Clear	DC	NA	NA	0	NA	NA	NA	NA	NA
D	Pulse	0 [V]	5 [V]	NA	33 [uS]	.1 [nS]	.1 [nS]	200 [uS]	400 [uS]
clock	Pulse	0 [V]	5 [V]	NA	0	1 [nS]	1 [nS]	50 [uS]	100 [uS]

Global Sources
Vdd = 5 [V]

State Name									
State1									
Analysis Type	**Run Time**								
Transient	150 [uS]								
Inputs	**Function**	**Voltage 1**	**Voltage 2**	**DC Votlag**	**Delay Time**	**Rise Time**	**Fall Time**	**Pulse Wid**	**Pulse Period**
Clear	DC	NA	NA	0	NA	NA	NA	NA	NA
D	Pulse	0 [V]	5 [V]	NA	0 [uS]	.1 [nS]	.1 [nS]	70 [uS]	140 [uS]
clock	Pulse	0 [V]	5 [V]	NA	0	1 [nS]	1 [nS]	20 [uS]	40 [uS]

State Name									
State2									
Analysis Type	**Run Time**								
Transient	5 [mS]								
Inputs	**Function**	**Voltage 1**	**Voltage 2**	**DC Votlag**	**Delay Time**	**Rise Time**	**Fall Time**	**Pulse Wid**	**Pulse Period**
Clear	DC	NA	NA	0	NA	NA	NA	NA	NA
D	Pulse	0 [V]	5 [V]	NA	33 [uS]	.1 [nS]	.1 [nS]	45 [uS]	90 [uS]
clock	Pulse	0 [V]	5 [V]	NA	0	1 [nS]	1 [nS]	50 [uS]	100 [uS]

State Name									
Clrchk									
Analysis Type	**Run Time**								
Transient	5 [mS]								
Inputs	**Function**	**Voltage 1**	**Voltage 2**	**DC Votlag**	**Delay Time**	**Rise Time**	**Fall Time**	**Pulse Wid**	**Pulse Period**
Clear	DC	NA	NA	5	NA	NA	NA	NA	NA
D	Pulse	0 [V]	5 [V]	NA	33 [uS]	.1 [nS]	.1 [nS]	45 [uS]	90 [uS]
clock	Pulse	0 [V]	5 [V]	NA	0	1 [nS]	1 [nS]	50 [uS]	100 [uS]

Figure 7 - Test States

Appendix B – Matlab Plotting Tutorial

```
%Dave Adams
%EE683 - Bibyk SP06
%Matlab Plotting of Cadence .out files

%_____________________________________________________________________________
%_____________________________________________________________________________
%this code enables users to create configurable plots
%which are much easier to read than the plots which cadence
%creates in the UNIX environment.
%_____________________________________________________________________________
%_____________________________________________________________________________

%Shown below is the command to output data from a simulation to a data file.
%This command is to be typed into ICFB once a simulation has been executed.
%Keep in mind that the commands are case sensitive.

%ocnPrint(VT("/Q") ?output "./myoutfile.txt" ?numberNotation 'none)

%VT("/Q"):  This specifies the type of analysis and what variable
%or net name should be outputted.  Once again keep in mind that everything
%is case sensitive.  Seen below is a list of the possible analysis outputs
%that can be use.

%VT:  Nodal Voltage (Transient Analysis)
%IT:  Terminal Current (Transient Analysis)
%VF:  Nodal Voltage (AC Analysis)
%IF:  Terminal Current (AC Analysis)
%VS:  Nodal Voltage (DC Sweep)
%IS:  Terminal Current (DC Sweep)
%VDC:  Nodal Voltage (Quiescent Value)
%IDC: Terminal Current (Quiescent Value)

%"./myoutfile.txt":   This is simply the file name of the data output file.

%First load the text file output from cadence, ensure that the file
%is in the Matlab "Current Directory", (i.e. same as the .m file
%which is being ran).  Be sure that the .txt file contains NO header
%if there are headers Matlab will create an error and not be able to load
%the file.
load('D_ff_output_edit.txt')


%view the text file to see what each column data corresponds with.
%here we see that column 2 is the output Q and column 1 is time.
%lets extract theses from the matrix we loaded in the previous line of code.
Q=D_ff_output_edit(:,2);
t=D_ff_output_edit(:,1);

%open a figure to create the plot.
figure(1)

%turn hold in if we are going to ploy more than one output vs. time
```

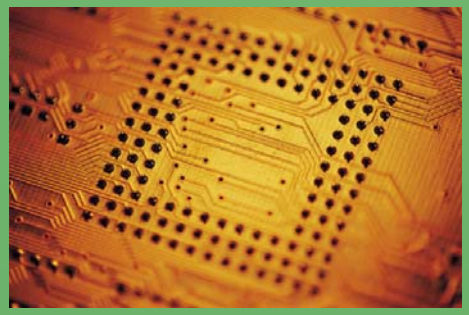
```
hold on

%turn the grid on to make the graph easier to read.
grid on

%create the firs plot to appear on figure 1, this is the output Q plotted
%vs time, it will show up as a blue line by default.  Colors and line types
%can be changed for other plots we may want to present.
plot(t,Q)

%label the Axies
xlabel('Time [s]')
ylabel('Volts [V]')
```

Appendix C – Remote Unix Access Tutorial



Remote Unix Access - Setup Documentation