

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

We will now begin the design by implementing the logic design of the 8-bit Pipelined adder. The following sections introduce you to the procedures to use Cadence for schematic capture and simulation which you will use to implement the required logic design.

1. Introduction to Cadence tool

- If you see “***/xlsfonts: not found”, disregard it. You can also see two pop-up windows telling you what’s new about this version of Cadence. Click ok. At the same time, a small rectangular window will pop up. This is called Command Interpreter Window (CIW). This is where virtuoso will display all the messages about your commands. Do not close this window (except when quitting!) and always keep this window visible. CIW displays all the error and warning messages, so please look into this whenever possible. A window called Library Manager will also pop up.

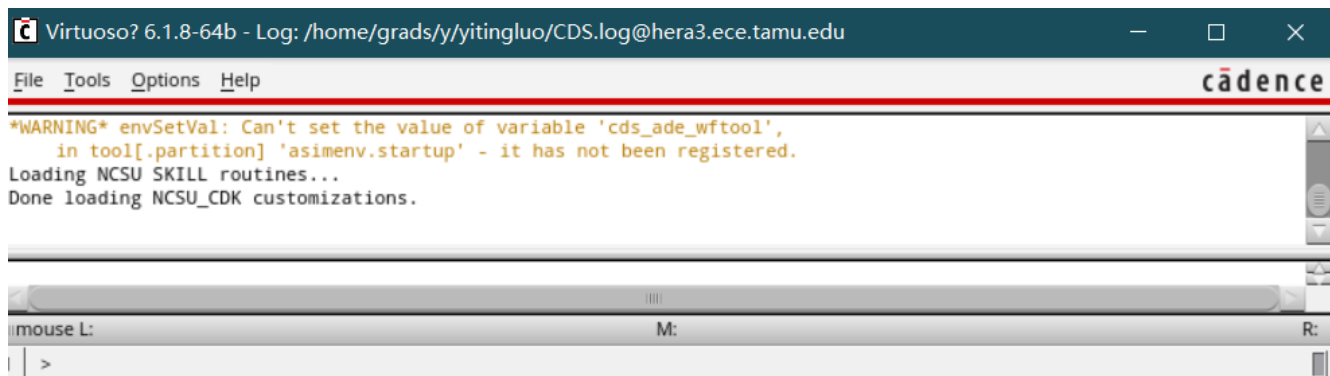


Figure 1. CIW Window

- The heading of the CIW is Virtuoso - Log: /homes/xxxx/CDS.log and has four menus File, Tools, Options and Help. The lower left corner has the virtuoso prompt which displays ">", when it awaits next action.
- We can now create designs in virtuoso. It stores all data in a library. A library can contain one or more designs. Designs in a library use a technology file, which describes the rules associated with the fabrication processes. So when we create a new library, the first thing which should be done is to attach a technology file to a new library. To create a new library, open Library Manager thru Tools > Library Manager in CIW, then choose File > New. You will see three items when you do “left click” on New: Library, Cellview, and Category. Library is the top concept in hierarchical structure. And below are cells with several different properties. For physical layout design, the Cellview name is layout. If it is a gate level design, the Cellview name is schematic. For transistor level design, we use cmos.sch. We have also “extracted”, and “symbol” which will be used in the following labs.
- Virtuoso uses Library Manager for browsing through all the libraries. The Library Manager has several libraries including any designs created by you. Several other predesigned libraries are also required before your own design. This is why file “cds.lib” was copied to your directory when you start cadence, make sure it exists under your current running directory. In the Library

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

Manager window, select Edit > Library Path, and you will see the Library Path Editor window pops up. In the window, select File > Open to find and select the “cds.lib” file just saved in your cadence directory, click it, and press OK. You can simply ignore warning messages about unlocked file. If the “cds.lib” file is well launched, you will see the Library path Editor window as below.

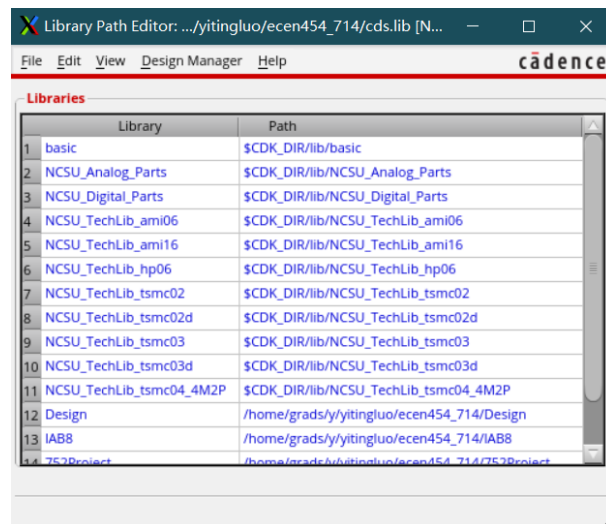


Fig 2. Library Management

- At the Library Manager window, click File > New > Library and type in name “Design” as shown below. Click OK.

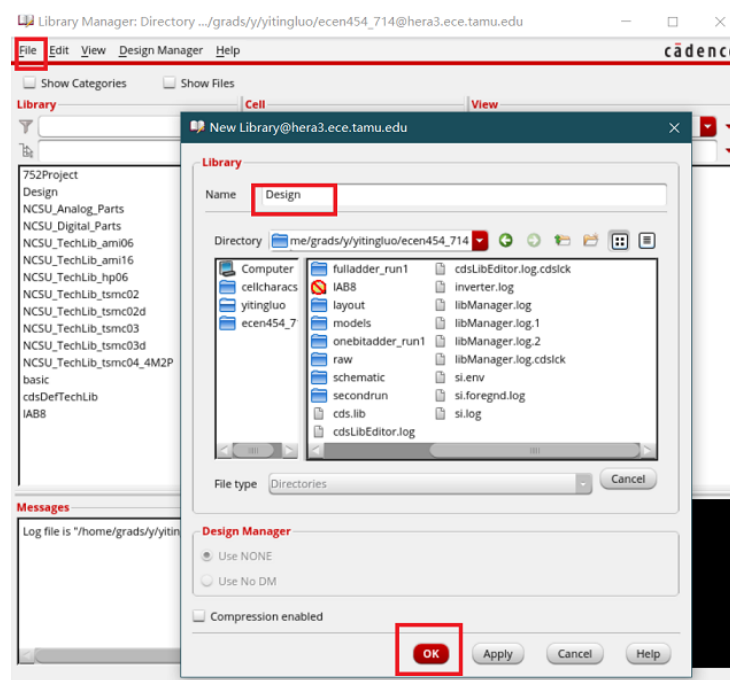


Figure 3. New library created.

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

- Next, you will find a menu for technology file. Choose “Attach to an existing technology library” and click OK. Then choose “NCSU_TechLib_tsmc02” as shown below. It means from now when you design a device, you will do it according to the rules related to this technology.

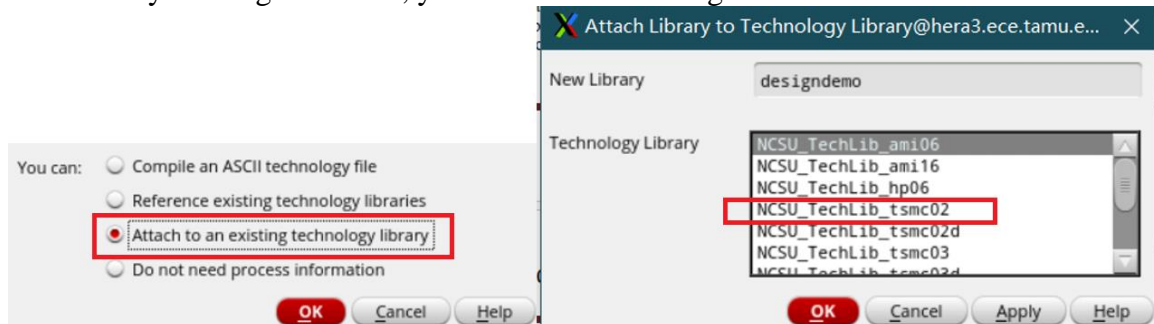


Figure 4. Attach tsmc02 library.

2. Schematic of the Full-adder Circuit.

The full adder is the basic building block of the 4-bit adder block that we need to design which we will eventually use to create the 8-bit Pipelined adder. The method is to create the design. (Schematic in this case) of the lowest building block (fulladder) and create a “symbol” of this design which can be reused in some other design (we would be using it in a higher-level design to create a 4-bit adder)

- In this lab we will design a one-bit adder. Let’s create a new cell called "fulladder". For this click on the “Design” library in the Library Manager window and choose File New > Cell view. Type cell name and view name.

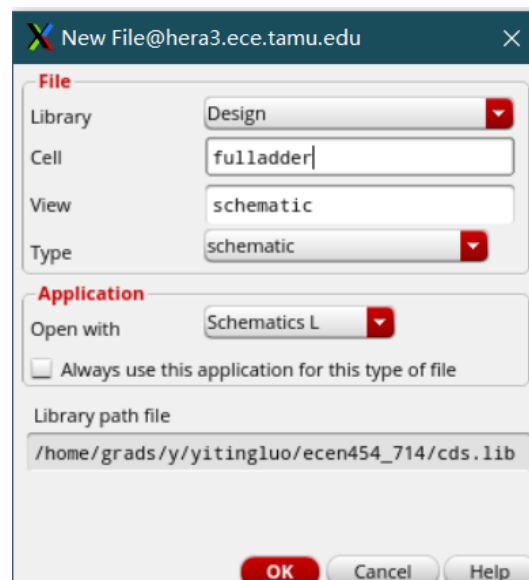


Figure 5. Creating a schematic for full-adder

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

- In the Cell Name field, type "fulladder". As mentioned, above there can be several designs in a library and cadence stores each design as a cell. Also, each cell can be represented in many ways for example as gate level or transistor level schematics, layouts, SPICE netlists or verilog etc. virtuoso calls these representations as cellviews.
- For the adder design we will create a gate level schematic. This is called a "schematic view". In the view name field, type "schematic". Click OK on the form and a schematic view is created.
- Click on the fulladder cell in the Library Manager and you can see the schematic cellview ("fulladder schematic"). **If there is an ADE license error message “ERROR (ELI-00111) Failed to check out license 'Virtuoso_Schematic_Editor_L.'”, select “always”.**
- We can now create the first design - a gate-level schematic of 1-bit full adder. The adder will take three inputs: A, B and C (carry-in). The output will be two bits: SUM and CARRY.
- From the truth table the SUM and the CARRY functions can be generated. This can be simplified to $CARRY = AB + (A \text{ XOR } B) C$ and $SUM = (A) \text{ XOR } (B) \text{ XOR } (C)$.
- To start the Cadence schematic editor, called Virtuoso, (if it is not already open), hold the middle mouse button on the schematic cellview of fulladder in the library Manager, scroll down and choose "Open". At most of time the editor will pop up automatically when you add the new cell. This is where you design your schematic. To create an instance of a gate, choose Create → Instance from the editor menu. Component Browser (CB) and Add Instance window pops up.
- Choose library “NCSU_Digital_Parts” in CB. You will see lots of items like And, Flipflops, etc. You can choose any gate you want from this list. If you want OR gate, click “OR” in CB. You will see OR2, OR3... OR6. Choose OR2 if you want 2 inputs ORgate. If you click OR2, the Add Instance window will be automatically filled. Move your mouse to Virtuoso window and click at the point where you want to place the OR gate. One more clicks you will have one more OR gate. Press ESC if you do not want OR gate anymore.

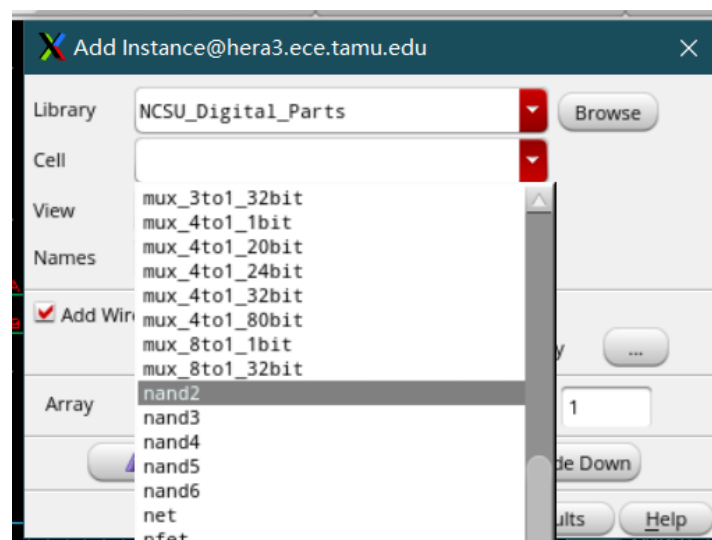


Figure 6. Add instance for full-adder schematic

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

NOTE: Use only XOR and NAND2 (you will have to figure out how to replace the AND and OR gates with only NAND2 gates) for the full adder. This is to maintain consistency among all students and it also means you have to implement only the XOR and NAND2 layouts. (Remember: step 2 in section 2 of Lab0 will ask you to implement layouts of all the logic gates you used in the schematic design. Not only will you have to design fewer layouts (NAND2 instead of AND and OR) but also the implementation of a NAND2 layout can be easier than that of an AND or OR layout as you will find in the following weeks). Also, try to minimize the number of gates used in your design. A full adder circuit can be composed of totally 5 gates of XOR and NAND2.

- Place as many instances of this gate as required. When you finish press ESC key. If you have placed some extra instances, then go to Edit > Delete in the Composer menu and click on the instance to delete. To exit from the Delete mode press ESC. To save the design click on File > Check and Save. Always save your work when you finish some important operations.
- Place instances of all the gates required and make you familiar with Add Instance and Delete Instance functions. Make sure you have plenty of space left for wiring your schematic. You can Zoom In and out to fit a part or whole of your design using View → Zoom In and Zoom Out respectively.
- We now have to wire the design. Before that we need to place input and output pins. The pins are for the input output connection of this design and also help for the simulation. We need five pins, three input pins A, B, C and two output pins SUM and CARRY. Select Create → Pin in the Virtuoso menu. Add Pin form pops up. Type “A B C” in the Pin Names field. Make sure that the Direction of the pin is “input”. Go to the schematic window and place the three input pins wherever you feel appropriate. (You might have to zoom).



Figure 7. Add input pin for schematic.

- Note that the pins A, B and C are different from the A, B and C you see on the instances of the gates. These (the later) are for the gates themselves, one level below the hierarchy. In the same

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

way, create the output pins “SUM and CARRY” but remember to change the direction field in the Add Pin form to “**output**”.

- Before wiring let’s see some Composer functions. You can move and copy the instances using Edit > Move and Edit > Copy functions. Also, you can undo the changes made using Edit > Undo. The CIW and the lower corner of the Composer will guide you.
- Finally let’s wire the schematic. Choose “Create> Wire (narrow)” in the window. Click on the A pin, a wire follows you. Take it to the point you want it to go and notice that the editor does automatic routing for you. Place it at the output of the proper gate. Similarly wire the rest of the circuit. Note that the wires can overlap without making contacts unless you explicitly do so. To make sure you wire to the exact place you want, notice that there is a diamond appearing when you wire to some place.
- You can connect up to 3 wires at one branch of wire. Pins are to be connected to instances thru wires, and if you just put the Pins on the input/output pins of a gate instance, they are not actually connected. Always use wires to make connections between instances. For example, your input is B[3:0] for the 4-bit adder, you could use bus pin option to connect your input B[3],B[2],B[1],B[0] independently like below:

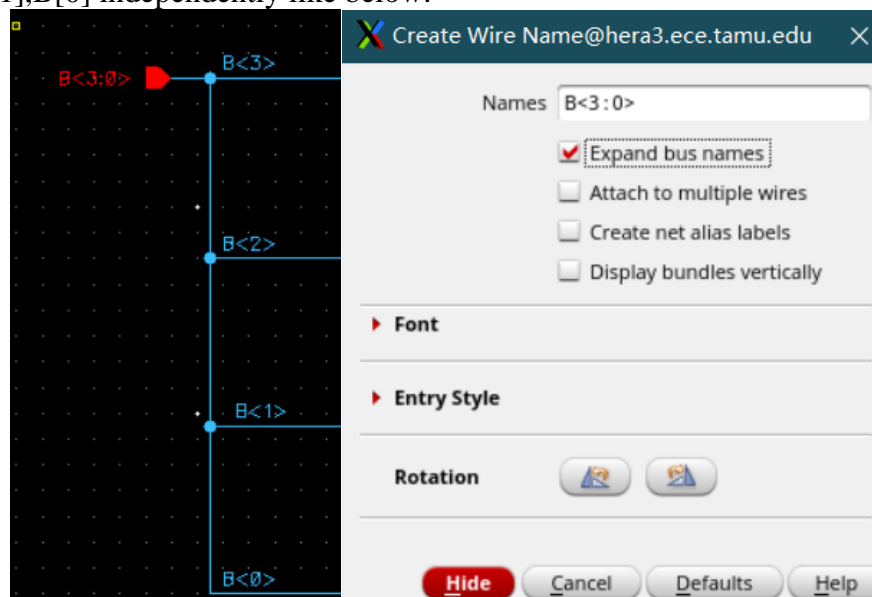


Figure 8. Add bus pin

Wire the rest of the circuit on your own. To save the design click on File > Check and Save. Look in the “CDS.log” Window for error messages. If there were errors, there would also be cross markers blinking on the schematic window. Try to correct the errors and ask your TA if you need help.

□ We next create a symbol view of the schematic. Choose Create > Cellview > From Cellview

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

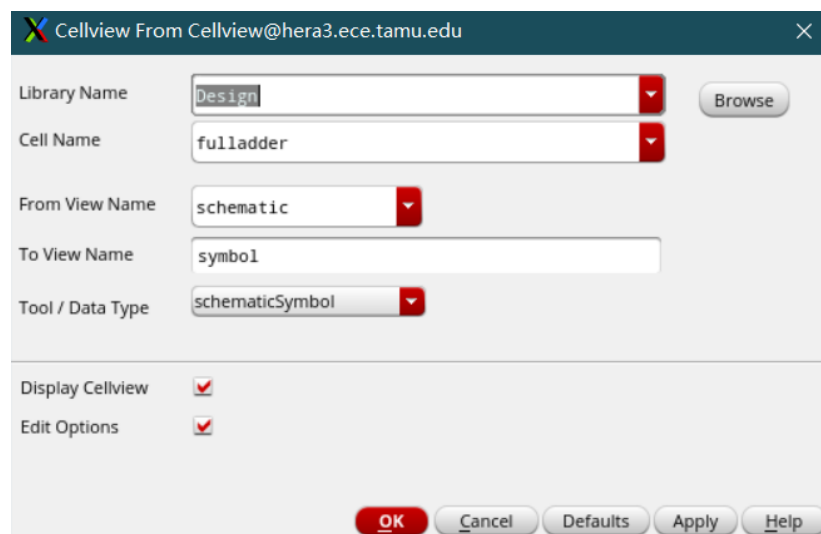


Figure 9. Creating a symbol for schematic view

This creates a symbol view of the fulladder in the Library. This can be observed in the Library manager. Observe Design > fulladder > symbol. This symbol consists of the fulladder block along with ports standing for the pins that were used in the design (in this case them being A, B, C as inputs and SUM, Carry as outputs). You could also edit it to make it look much better.

3. Simulation

- To make sure the schematic we just completed works properly, we will simulate it using NC-Verilog logic simulator. Note that this is just the logic simulation of schematic, and it does not give out accurate timing information (in fact, assuming zero gate delay model here).
- To invoke the NC-Verilog simulator select Launch> Plugin > Simulation > NC-Verilog in the schematic window. The Setup Environment window appears. Make sure the Run Directory (it will be fulladder_run1 or xxxx_run1 if you give different name of the cell), Library, View and Cell fields are filled correctly. First click Commands -> Initialize Design. If error happened, click Setup -> Simulation, change the path of “NCVerilog Executable” as: \

/opt/coe/cadence/INCISIVE152/tools/bin/ncxlmode

- Then click initialize again. Next, click Commands->Generate Netlist.

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

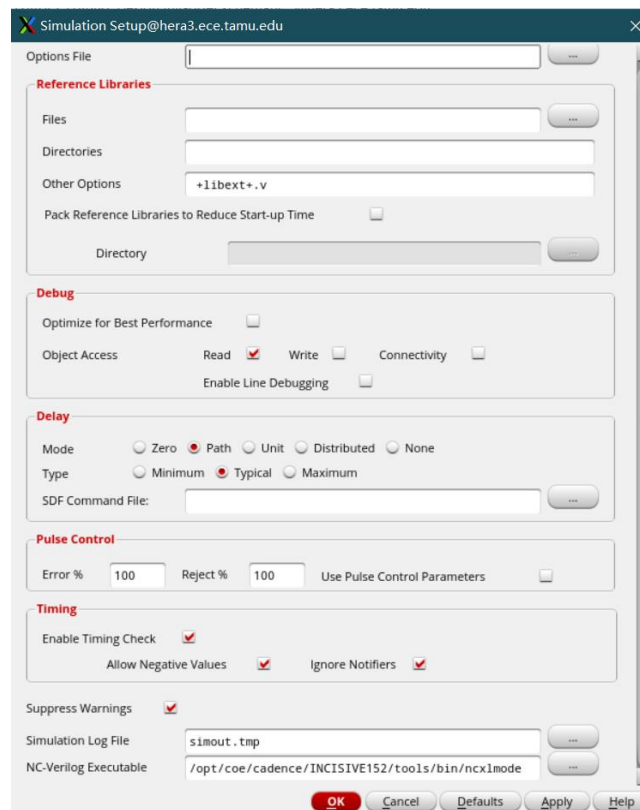


Figure 10. Simulation configuration

- We first need to edit a test fixture where we specify the test vectors, which NC-Verilog uses to simulate. We will use all the eight possible input combinations as test vectors to validate your design. Note that in real design; generally, we cannot afford the time and memory to do test on all input combinations. To edit test fixture, select Commands>Edit Test Fixture, You would find “testfixture.verilog” is the stimulus file. You can use any text editors (e.g., gedit, gvim, etc.) you prefer to edit it, its path should be > cadence > fulladder_run1 > testfixture.verilog

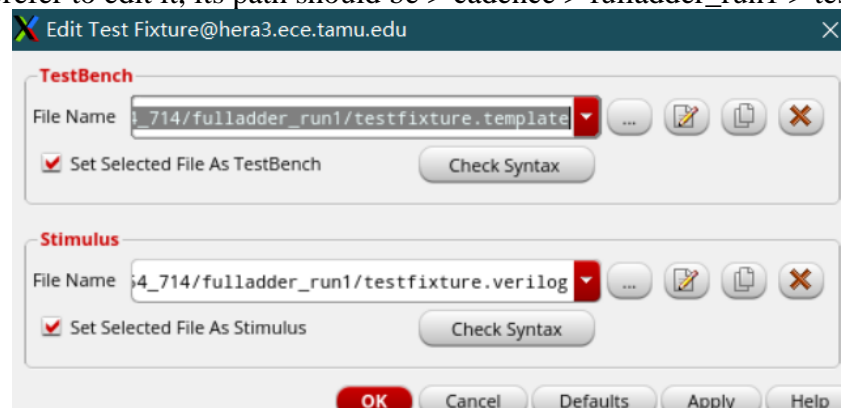


Figure 11. Edit test fixture.

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

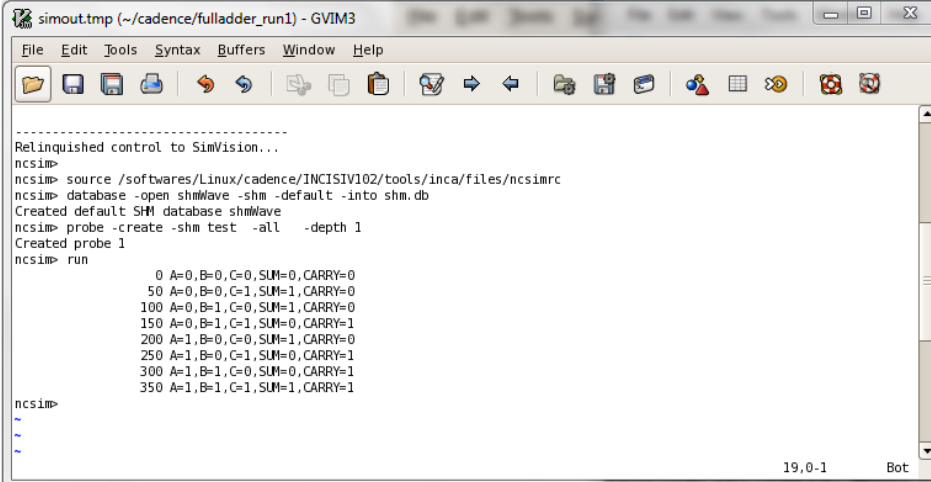
- The test fixture, which appears is the default test fixture in Verilog Hardware Description Language. A Hardware Description language (HDL) is used to describe a hardware circuit. At present we will just see very basics of Verilog HDL and study it in more detail in other labs.
- The default test fixture consists of some Verilog code and comments, which start with “//” (as in C++, in fact Verilog is similar to C). Anything following “//” is commented out. Next is an initial statement. This tells Verilog to perform the following task at start up. A begin and end tell Verilog to perform the following lines in the initial block.
- The three statements tell Verilog to set the three inputs to zero. The 1'b0 means “one bit, binary zero”. A 3-bit representation of decimal four would be 3'b100 or 3'd4 (d denotes decimal). Similarly hexadecimal can be specified as “h”. Note that since no timing information is associated with the assignment statements, all the three statements are executed in parallel. That is, all three bits are set to zero at time t = 0.
- We will add timing information right now. Here it works like a clock. Since we do not extract timing information from the circuit and this is still just logic simulation, we would feed different input signals at different time that can be seen as delay of input. The Verilog uses #time directive to add delay, which means “halt time nsec and then proceed”. The second test vector could then be #50 C = 'b1; //ABC = 001. Note that we are just changing the value of C, as A and B remain the same. Similarly add all the eight possible test vectors for the three inputs. You can add comments for each statement also to make it clear. When two statements are to be executed in parallel then use the same #time directive for two assignments. For example to produce vector 011 at 50 ns (if A = 0, B = 0 and C = 0 at 0ns) use #50 B = 1'b1; C = 1'b1.
- To observe the simulation results, add several lines at the end of your test fixture file as the following:

initial

\$monitor (\$time," A=%b, B=%b, C=%b, SUM=%b, CARRY=%b", A, B, C, SUM, CARRY);

- **Make sure the signal name is exactly same as you used in the schematic.**
- Then remember to use Edit Test Fixture->Stimulus->Check Syntax to make sure your stimulus does not have any syntax error.
- After finish stimulus file, choose Commands->Simulate in the NC-Verilog Integration window and two SimVision window will pop out. On “Design Browser” window, click “Simulation->run”, which stimulate the actual simulation, then open file named simout.tmp under directory fulladder_run1, check whether logic is correct or not.

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation



The screenshot shows a terminal window titled 'simout.tmp (~/.cadence/fulladder_run1) - GVIM3'. The terminal output displays the following commands and results:

```
Relinquished control to SimVision...
ncsim>
ncsim> source /softwares/Linux/cadence/INCISIV102/tools/inca/files/ncsimrc
ncsim> database -open shmWave -shm -default -into shm.db
Created default SHM database shmWave
ncsim> probe -create -shm test -all -depth 1
Created probe 1
ncsim> run
      0 A=0,B=0,C=0,SUM=0,CARRY=0
     50 A=0,B=0,C=1,SUM=1,CARRY=0
    100 A=0,B=1,C=0,SUM=1,CARRY=0
    150 A=0,B=1,C=1,SUM=0,CARRY=1
    200 A=1,B=0,C=0,SUM=1,CARRY=0
    250 A=1,B=0,C=1,SUM=0,CARRY=1
    300 A=1,B=1,C=0,SUM=0,CARRY=1
    350 A=1,B=1,C=1,SUM=1,CARRY=1
ncsim>
```

Figure 12. Check simulation result

- If there is any error you will have to change the test fixture or the design and run the simulation again.

4. Complete logic Simulation

- The above sections contained instructions on how to capture a schematic and perform logic simulation. Now we have to follow these instructions to complete the entire design (keep figure 1 of “Lab 0” in mind...that is what we have to build and simulate to verify) of the pipelined adder. Make sure you name the schematics and other files you use appropriately so that you may look back anytime.
- The next step would be to design the 4-bit adder. For this, we use a new schematic window. Open a new schematic (preferably in the same library to maintain hierarchy) according to the procedures described in the above sections. We can now add instances of the “fulladder” by choosing the “Design” library in the component browser where we can select the “fulladder” as the component to be inserted. We can see that the symbol of the “fulladder” that was created previously appears as the component with 3 input pins (A,B,C) and 2 output pins(SUM, CARRY).
- Connect 4 such full adders in series to form the 4-bit adders. This forms the carry-ripple adder. After completion of wiring, generate a symbol for the schematic since it will further be used to obtain the 8-bit pipelined adder. Perform stimulation for this circuit to verify its operation. It helps to check each block individually since we will be using it in a larger design.
- Note: Although the adder can be implemented using various other styles such as the carry-look ahead adder, carry-select adder which can compute the results in a quicker fashion, we will still use the carry ripple adder to maintain simplicity in the basic design and consistency among students.
- After completing the 4-bit adder, we design another building block, the “4-bit register”. The register is basically a 4-bit memory element, and we would use 4-FF's to build it. We provide its inputs from 4 pins and tap its outputs through 4-other pins (be careful of the direction of signal when adding a pin). Apart from these we also provide a clock and _clock (_clock stands

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

for inverted clock signal) input to the Flip-Flops. Create symbol for the circuit after wiring and simulate the circuit to verify its operation.

- Now, after completing all the individual building blocks to design the 8-bit pipelined adder using these building blocks in the same manner. Use the 4-bit adder, 4-bit register that you have designed and FF's to complete the design as shown in Fig 1 from Lab0. Simulate the circuit by giving the appropriate inputs to verify the operation of the circuit (a subset of test cases is enough). (The output will appear at the immediate rising clock edge following the input).

5. Lab Report

1. Take a plot of all the schematic designs and symbols (fulladder, 4bit adder, and 8bit adder). You may either screenshot your design or export the schematic.
2. Print out the respective simulation results for all designs. The simulation **must** display the inputs and the corresponding outputs along with the time at which these signals are monitored. Required input vector for each design is given below. Please initialize all inputs to zero and provide brief explanation why the results are correct.
 - Fulladder: all available input combinations.
 - 4bit adder
 - 1111 + 1111 + 0 (Carry In)
 - 1010 + 1010 + 1 (Carry In)
 - 0101 + 0101 + 1 (Carry In)
 - 8bit adder
 - 01111110 + 11100111 + 0 (Carry In)
 - 11111111 + 00000000 + 1 (Carry In)
 - 10101010 + 01010101 + 0 (Carry In)
 - 10101010 + 01010101 + 1 (Carry In)
 - 11001100 + 00110011 + 0 (Carry In)
 - 11001100 + 00110011 + 1 (Carry In)

Please be cautious to clearly see when the correct outputs are generated in accordance with clock input signal. Verify this in your lab report as well.

Lab reports should be in PDF format and submitted to Canvas. Due date is related to your registered lab session. Late penalty is 5% per day.

6. Appendix

Cadence tips:

1. DO NOT KILL cadence process if the machine hangs. This generates view locks, which won't allow you to open the views in edit mode.

ECEN 454/714 – Lab1: Introduction to Cadence Schematic Design & Simulation

How to remove the lock files? Follow these steps: 1) Exit Cadence (if you have it open) 2) Go to your design folder (for example lab4 folder) and delete any files in the cell folder with the .cdslck and .oacache format. 3) Now start cadence, you would be able to edit your file.

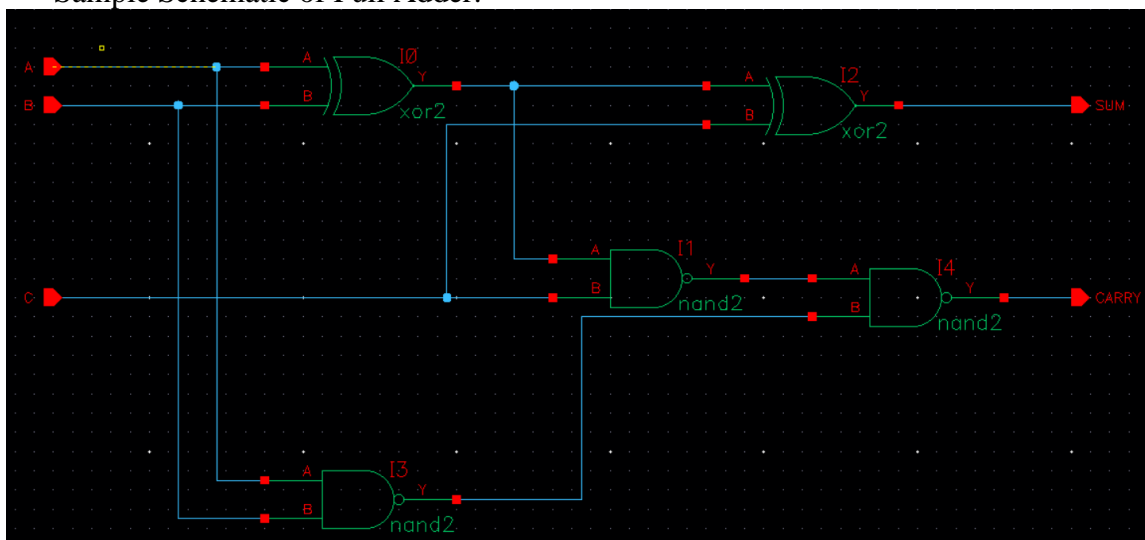
2. Always SAVE your work. Whenever working continuously, do not wait till end to save. If cadence crashes your work will be lost.

3. Please make sure you log off using “exit” when you leave since other students cannot access the machine if you do not log off and leave.

4. If you met license error, there should be a 'cdsenv' file in your home directory. You can change the license (to always) in that file.

- Go to home directory.
- type in 'gedit ~/.cdsenv'
- Change the code license *"never"* to *"always"*

Sample Schematic of Full Adder:



Grading Rubric of Lab 1 (for reference)

1. Schematic – full adder (1); 4-bit adder (1); 8-bit adder (1)
2. Symbol – full adder (0.5); 4-bit adder (0.5); 8-bit adder (0.5)
3. Simulation result - full adder (2); 4-bit adder (1.5); 8-bit adder (2)