Transceiver Block

•••

Sean Gordon and Bryan Friestad

The project

- The goal of this device is to more efficiently transmit serial data
- A clock line should not be sent
- Instead, we can recover the clock line from the data using a PLL
- This requires the data to be dense with transitions
- We have designed an encoding/decoding scheme to add additional transitions

Requirements

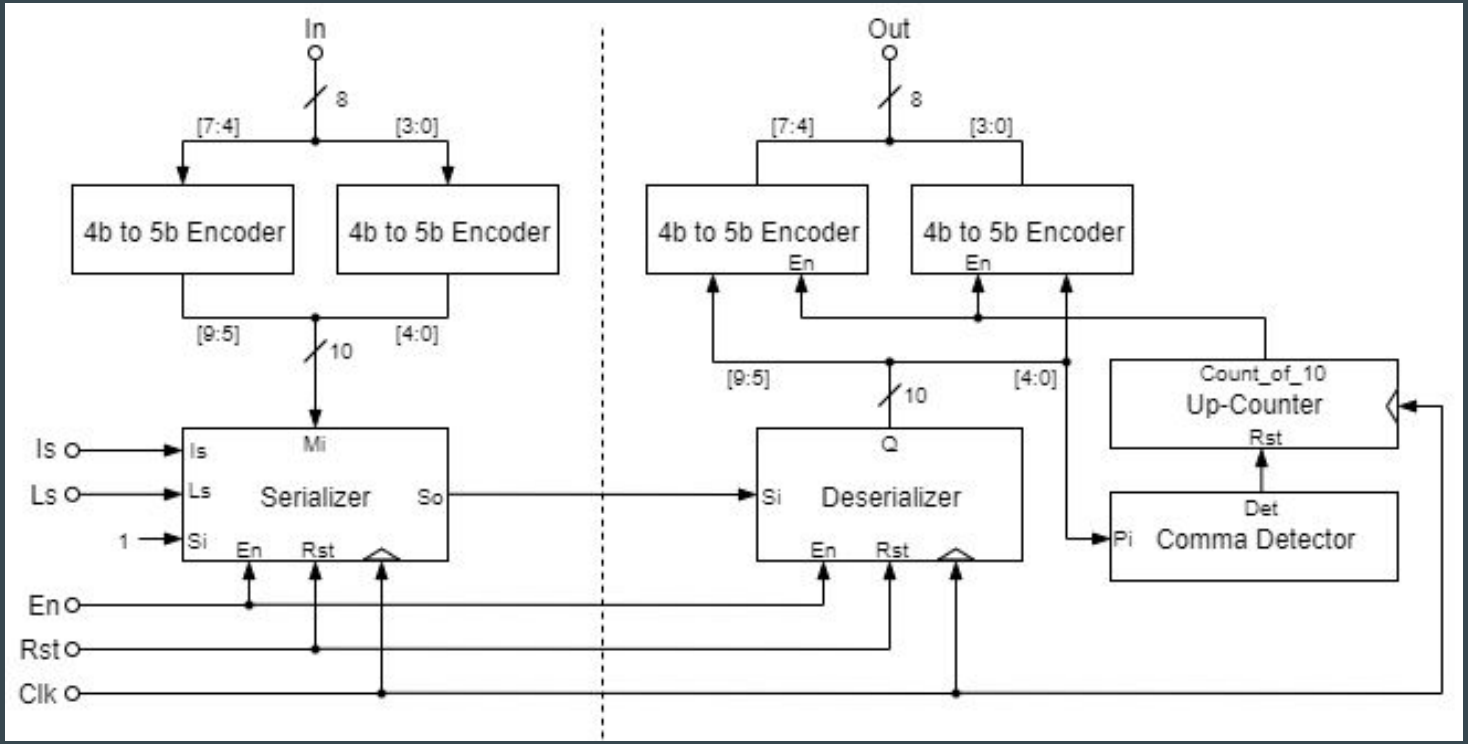
a. Devise a 4-bit to 5-bit coding scheme that will guarantee at most 4 consecutive 0s or 1s for any input data sequence.

b. Design a circuit that will take an 8-bit wide parallel data sequence at 10K bytes/sec and serialize it using the 4-bit to 5-bit coding scheme you devised in part (a). This assumes that a 10KHz clock is present that is synchronous with the input data.

c. Design a receiver that will take the serial data string, decode it, and recreate an 8-bit wide data sequence at the output.

d. Design a “comma detect” circuit that will allow for proper framing of the received data. The “comma” should be a 10-bit code that cannot represent any data sequence. The “comma” would be inserted in place of a byte in the transmitted data stream for synchronization and the receiver should frame the received data relative to the detected “comma” whenever a comma is detected. After the “comma” is detected, the received should be in synch with the input data sequence.

High-level diagram



Designing the Encoding/Decoding Scheme

- Needed to devise a 4-bit to 5-bit scheme that guaranteed at most 4 consecutive 0/1s
- Design drawback: Non-equal bit disparity

0000 → 00000

Input: 0000_0000
Conv: 00000_00000

0000 → 00100

Input: 0000_0000
Conv: 00100_00100

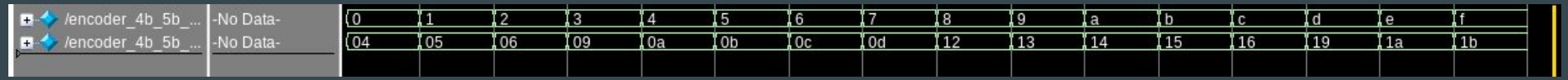
Encoding chart

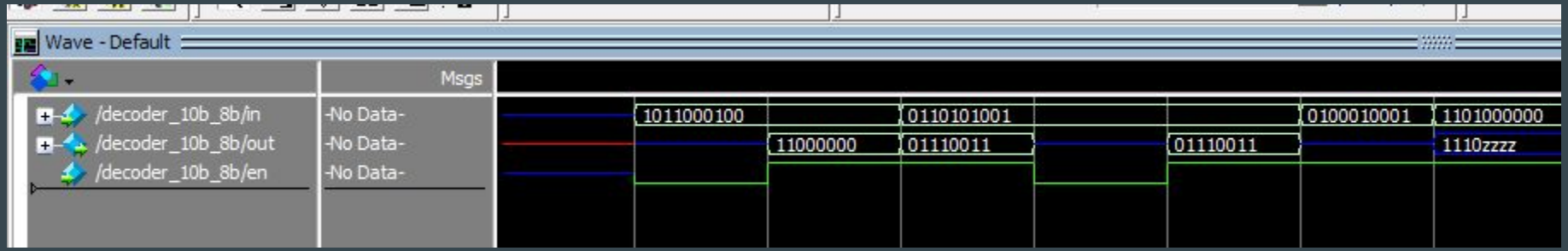
- From 5-bit strings, we selected the most transition heavy
- Each byte is split into a low and a high nibble
- Each nibble is equivalently encoded
- Decoding is just a reverse mapping
- All unmapped 5-bit words output Z

```
Transcript
VSIM 8> run 160ns
# 0000 -> 00100
# 0001 -> 00101
# 0010 -> 00110
# 0011 -> 01001
# 0100 -> 01010
# 0101 -> 01011
# 0110 -> 01100
# 0111 -> 01101
# 1000 -> 10010
# 1001 -> 10011
# 1010 -> 10100
# 1011 -> 10101
# 1100 -> 10110
# 1101 -> 11001
# 1110 -> 11010
# 1111 -> 11011
```

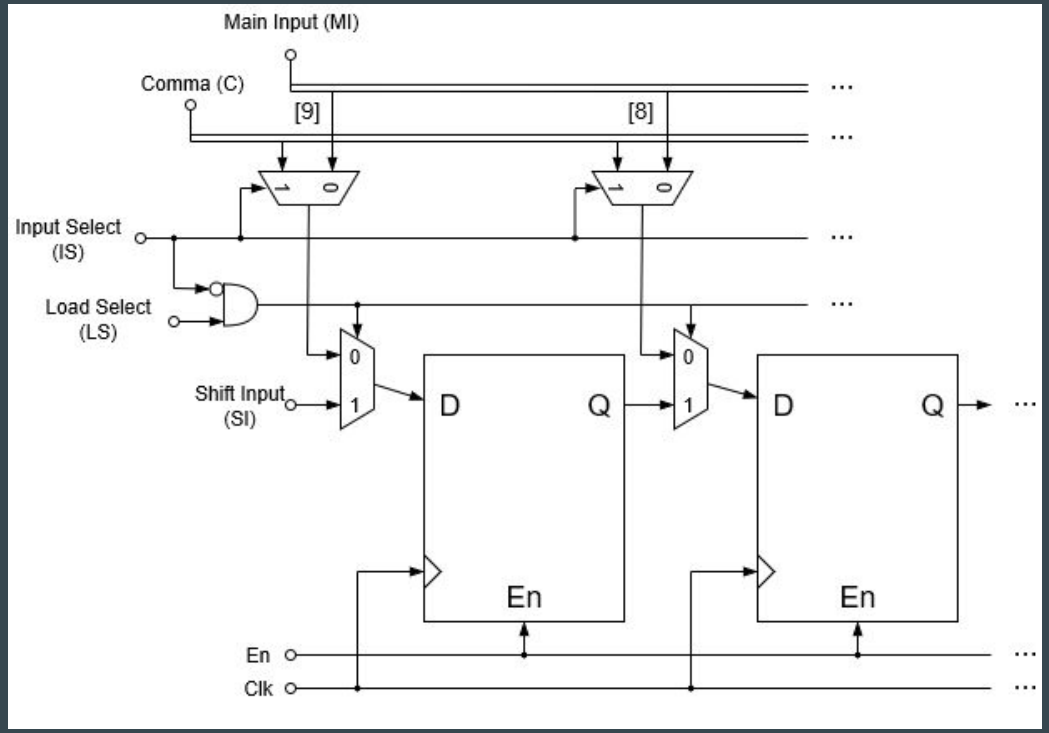
```
VSIM 17> restart -f
VSIM 18> run 320ns
# 00100 -> 0000
# 00101 -> 0001
# 00110 -> 0010
# 01001 -> 0011
# 01010 -> 0100
# 01011 -> 0101
# 01100 -> 0110
# 01101 -> 0111
# 10010 -> 1000
# 10011 -> 1001
# 10100 -> 1010
# 10101 -> 1011
# 10110 -> 1100
# 11001 -> 1101
# 11010 -> 1110
# 11011 -> 1111
# 11011 -> zzzz
# 11111 -> zzzz
```

Encoder/Decoder Testbenches

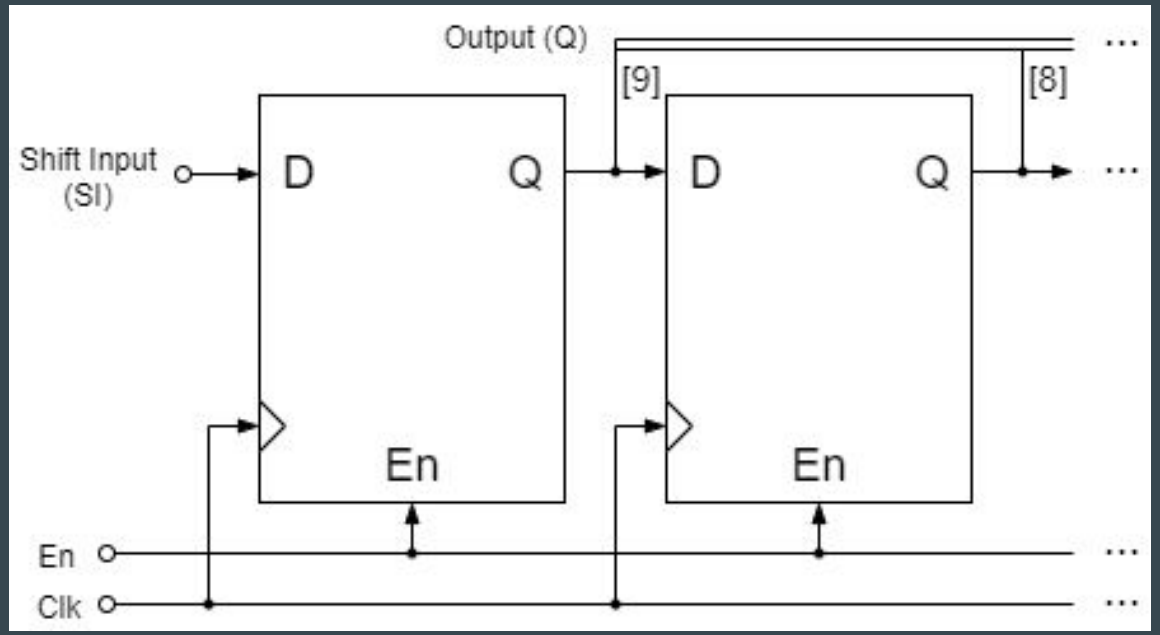




Serializer Block Diagram



Deserializer Block Diagram

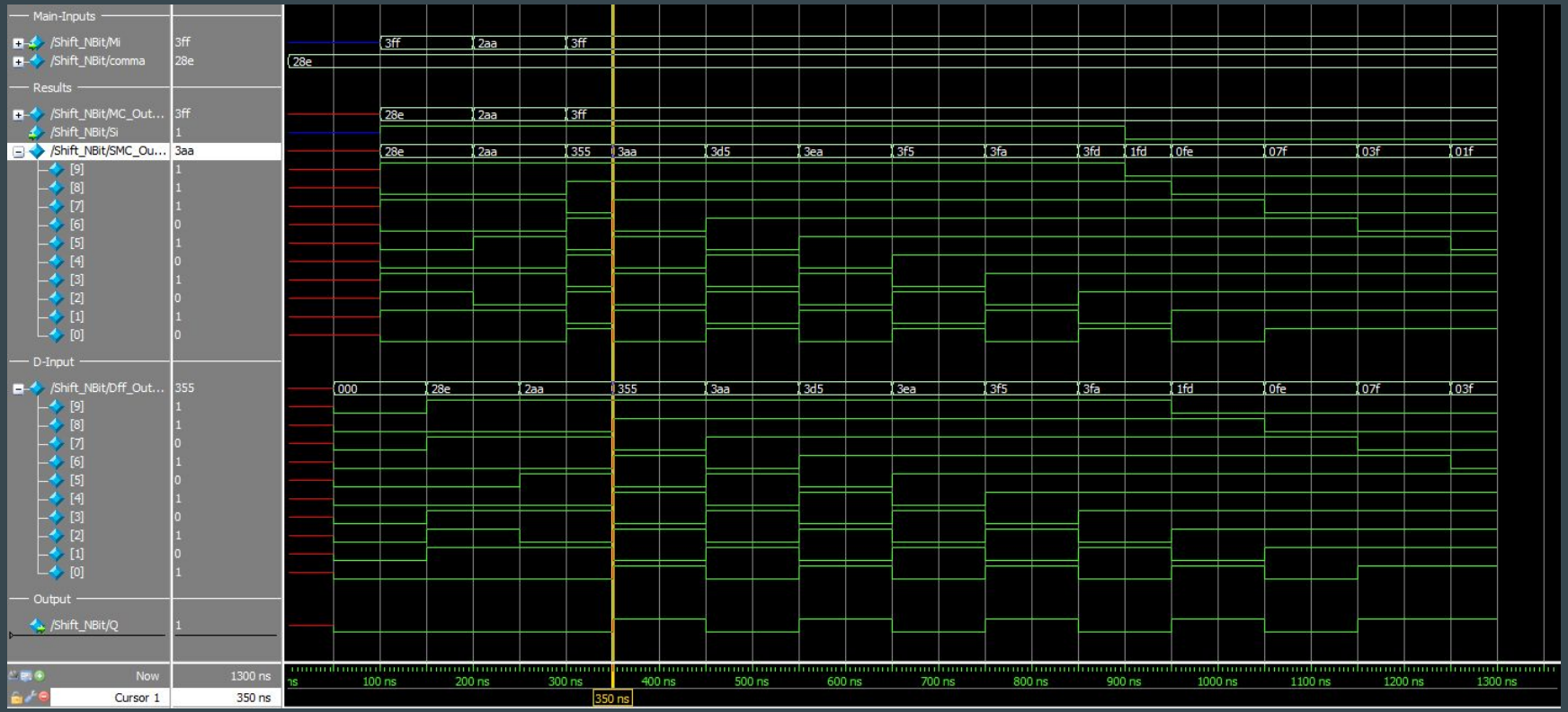


Serializer Testing

```
#Test Comma
force -deposit Mi     2#11111_111111 0
force -deposit Si    1 0
force -deposit Is    1 0
force -deposit Ls    0 0
run 100

#Test Main Input
force -deposit Mi     2#10101_01010 0
force -deposit Si    1 0
force -deposit Is    0 0
force -deposit Ls    0 0
run 100
```

```
#Test Shift Input
force -deposit Mi     2#11111_111111 0
force -deposit Si    1 0
force -deposit Is    0 0
force -deposit Ls    1 0
run 100


#Let shifter propagate out
run 500

force -deposit Si    0 0
run 400
```

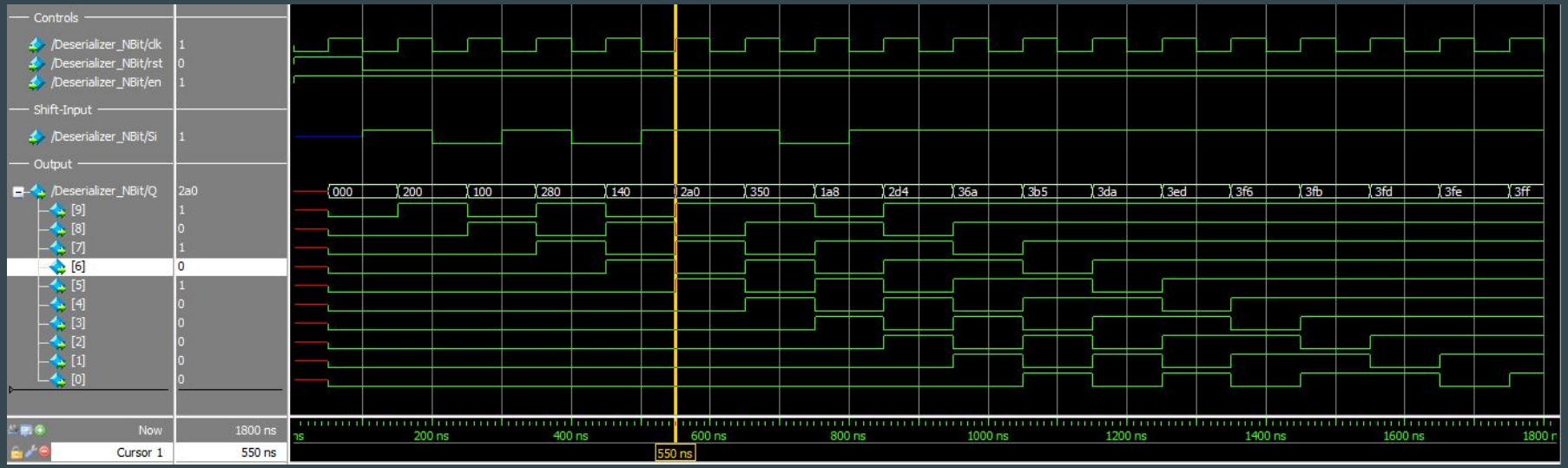
Serializer Waveform



Deserializer Testing

```
#### Clearing Things ####
force -deposit en     1 0
force -deposit rst    1 0
run 100

##### Commence Test #####
force -deposit rst    0 0

#Test Shift Input
force -deposit Si    1 0
run 100

force -deposit Si    0 0
run 100

force -deposit Si    1 0
run 100

force -deposit Si    0 0
run 100

force -deposit Si    1 0
run 100

force -deposit Si    1 0
run 100

force -deposit Si    0 0
run 100

force -deposit Si    1 0
run 100

#Allow deserializer to propagate
run 900
```

Deserializer Waveform



Receiver Synchronization

- In order to detect valid serial input, we need to align the data
- As data is shifted into the Deserializer, we want to be able to detect a start message
- We can devise a “Comma” which heads all serial messages
- When the comma is detected, the decoder is enabled.
- Additionally, a counter is reset to keep track of when a new 10-bit word has appeared

Selecting a Comma

- The comma must maintain that there are no more than 4 consecutive equal bits
- The comma must not appear in any other valid string of input to the Deserializer.
- We wrote a short program to assist in finding an optimal comma
- We selected 10_1000_1110

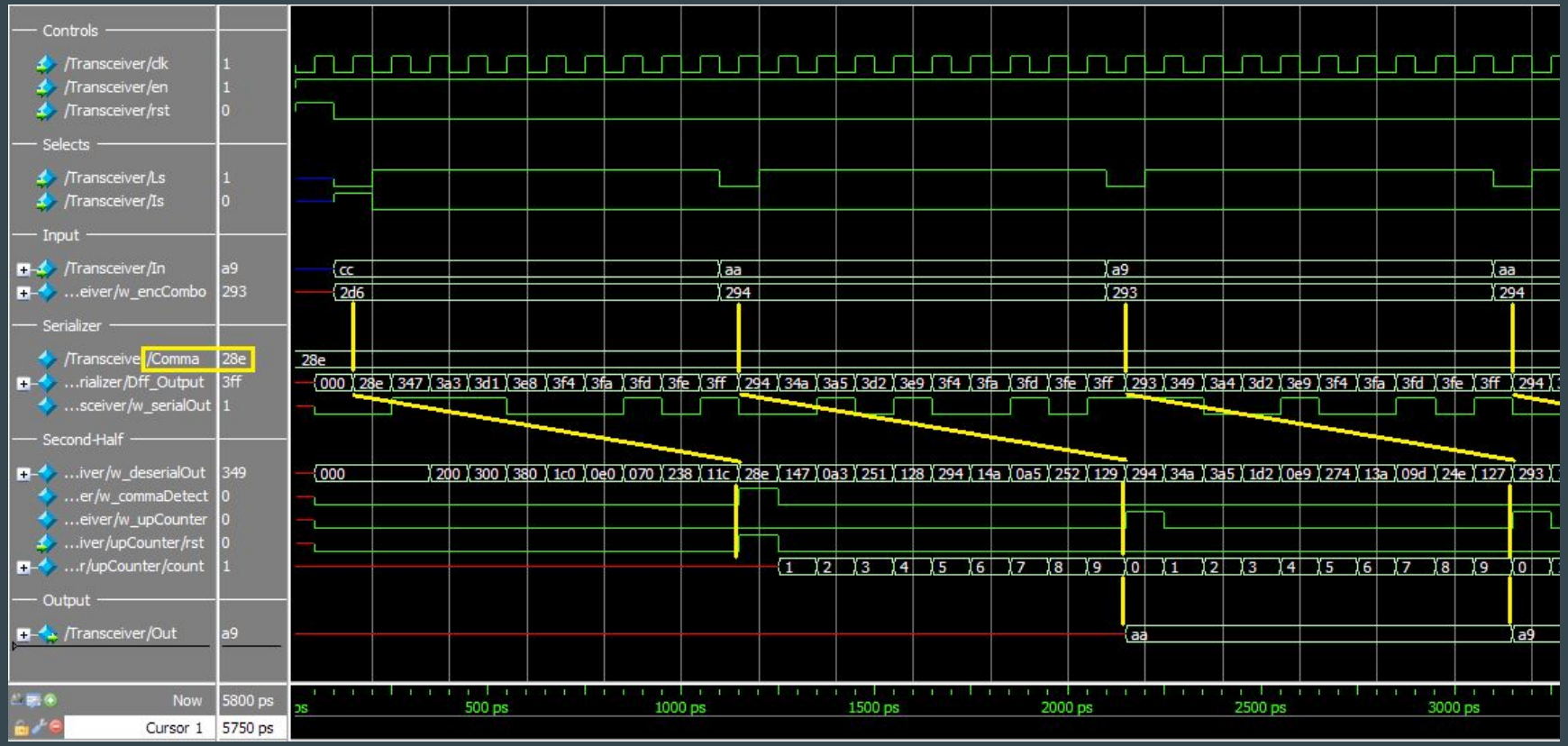
```java
public class FindComma {

	private static String[] five_bit_words = {"00100", "00101", "00110", "01001",
	                                          "01010", "01011", "01100", "01101",
	                                          "10010", "10011", "10100", "10101",
	                                          "10110", "11001", "11010", "11011"};
	private static ArrayList<String> ten_bit_words;
	private static String[] twenty_bit_words;

	public static void main(String[] args) {
		ten_bit_words = new ArrayList<String>();
		twenty_bit_words = new String[65536];

		for(int i = 0; i < 1024; i++){
			ten_bit_words.add(extendWord(Integer.toBinaryString(i), 10));
		}

		int n = 0;
		for(int i = 0; i < 16; i++){
			for(int j = 0; j < 16; j++){
				for(int k = 0; k < 16; k++){
					for(int l = 0; l < 16; l++){
						twenty_bit_words[n] = five_bit_words[i] + five_bit_words[j] + five_bit_words[k] + five_bit_words[l];
						//generates all possible, unique 20bit combinations of the 5bit words
						n++;
					}
				}
			}
		}

		for(int i = 0; i < 65536; i++){
			String[] frames = new String[10];
			for(int j = 0; j < 10; j++){
				frames[j] = twenty_bit_words[i].substring(j, j+10); //gets all possible 10bit frames from each 20bit word
			}

			//for each 10bit frame
			for(String s : frames){
				ten_bit_words.remove(s); //removes the 10bit frame from the list if it exists in the list
			}
		}

		for(String s : ten_bit_words){
			if(countTransitions(s) >= 5) //only display words with 5 or more transitions
				System.out.println(s + ": " + countTransitions(s)); //the only remaining 10bit words are valid for commas
		}

	}
```

Final Test Conditions

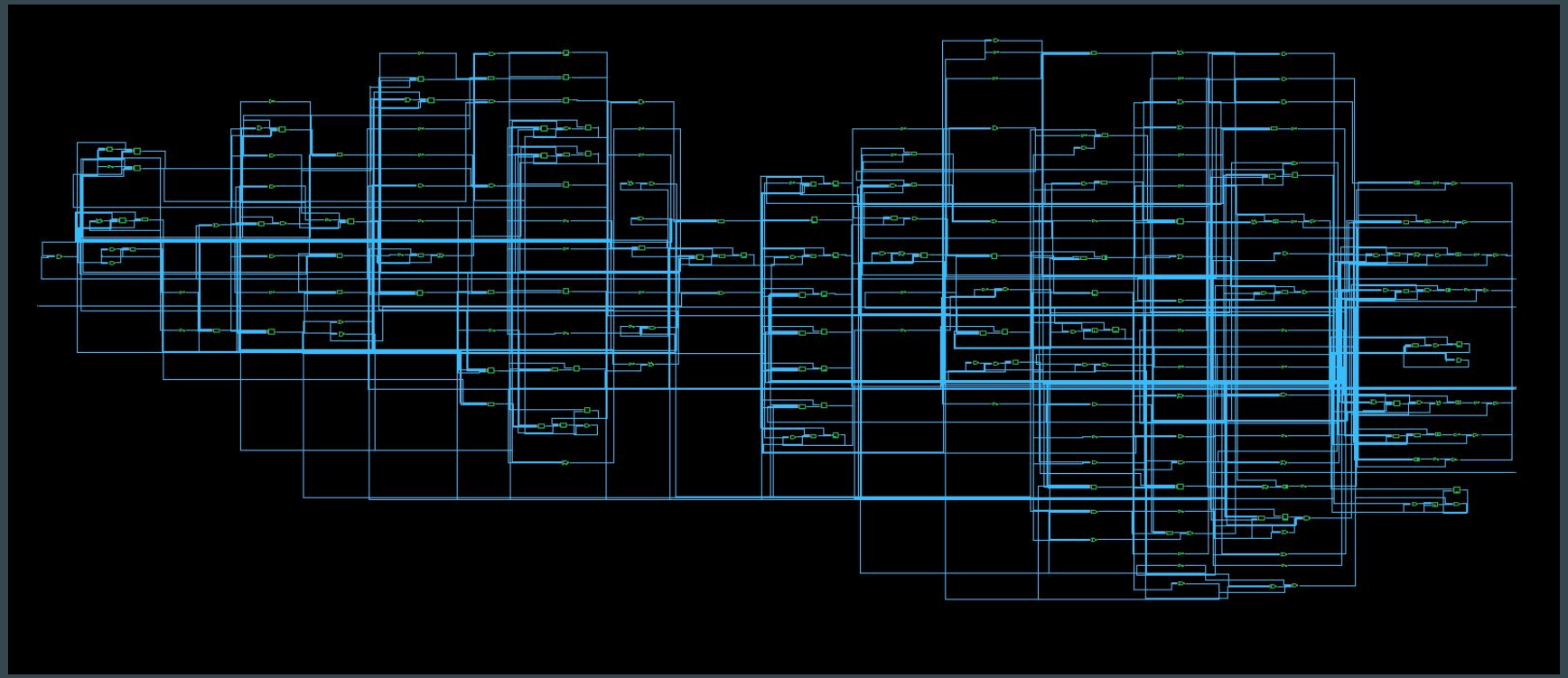
```
#### Clearing Things ####
force -deposit en     1 0
force -deposit rst     1 0
run 100

##### Commence Test #####
force -deposit rst     0 0

#Load Comma
#Setting In to 0xCC to point out that we're loading a 'comma'
force -deposit In     2#00110_01100 0
force -deposit Is    1 0
force -deposit Ls    0 0
run 100

#And send it over
force -deposit Is    0 0
force -deposit Ls    1 0
run 900
```

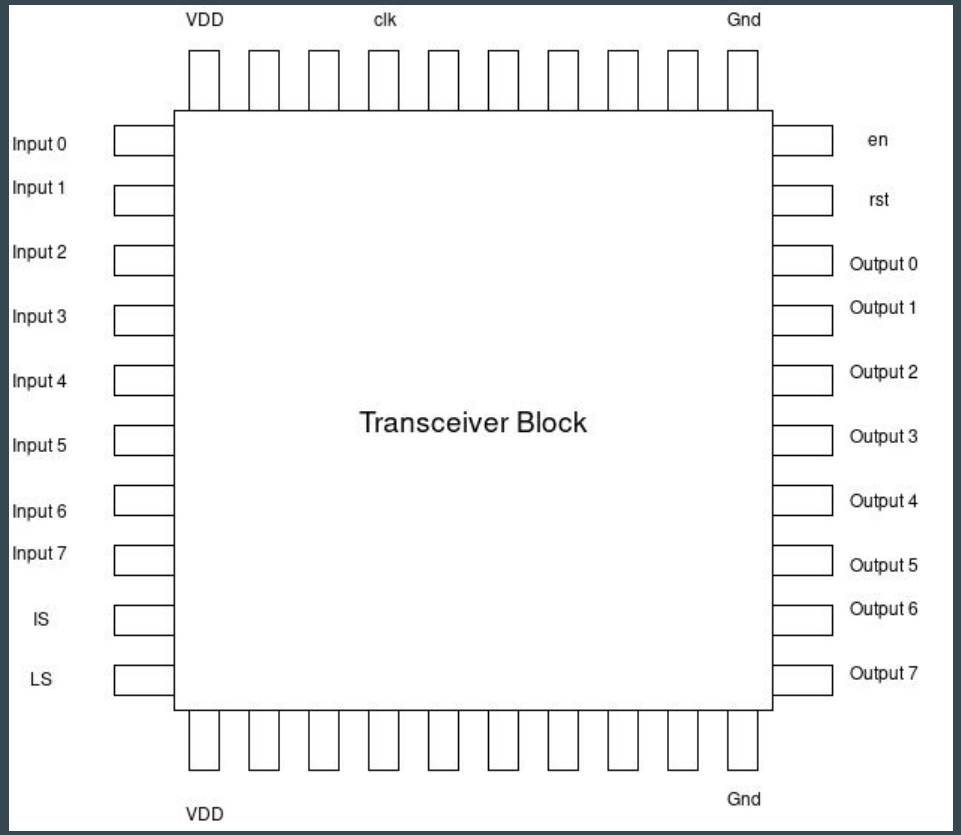
```
############## IMPORTANT #############
# Make sure when we load we do it on
# the 1000ns mark
# (run 900 -> load, run 100 -> repeat)

#Load Main Input
force -deposit In     2#10101_01010 0
force -deposit Is    0 0
force -deposit Ls    0 0
run 100

#And send it over
force -deposit Is    0 0
force -deposit Ls    1 0
run 900

#Load Main Input-1
force -deposit In     2#10101_01001 0
force -deposit Is    0 0
force -deposit Ls    0 0
run 100

#And send it over
force -deposit Is    0 0
force -deposit Ls    1 0
run 900
```

Final Waveform



Synthesized Schematic

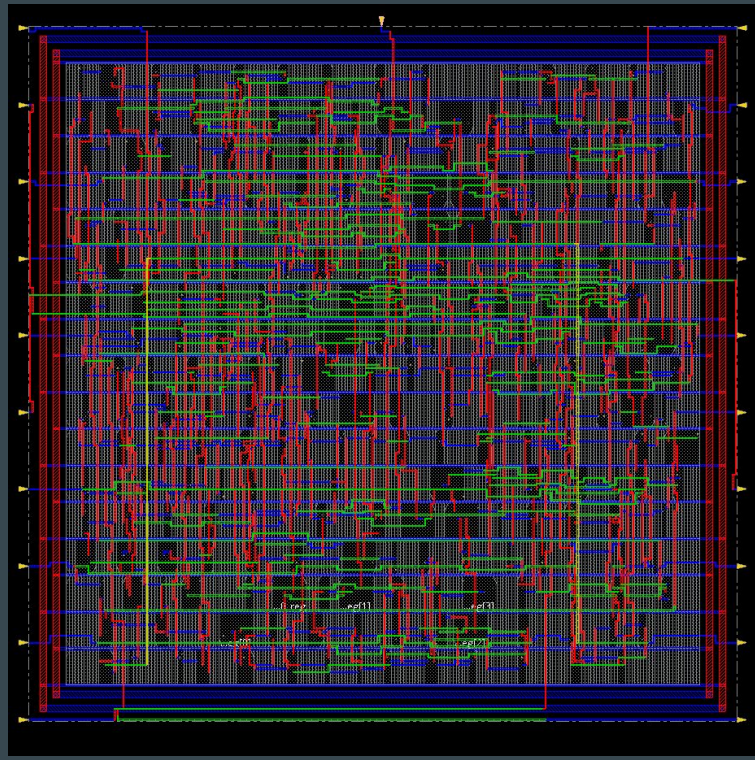




Our planned pinout on a 40-pin frame

Layout and Routing

- We wanted to maintain a roughly square area for the pad frame
- We had to set up the pins so that they were spread out
- The area was approximately 190 um by 192 um, or .0365 mm^2
- Our geometry checks and connectivity checks passed without error



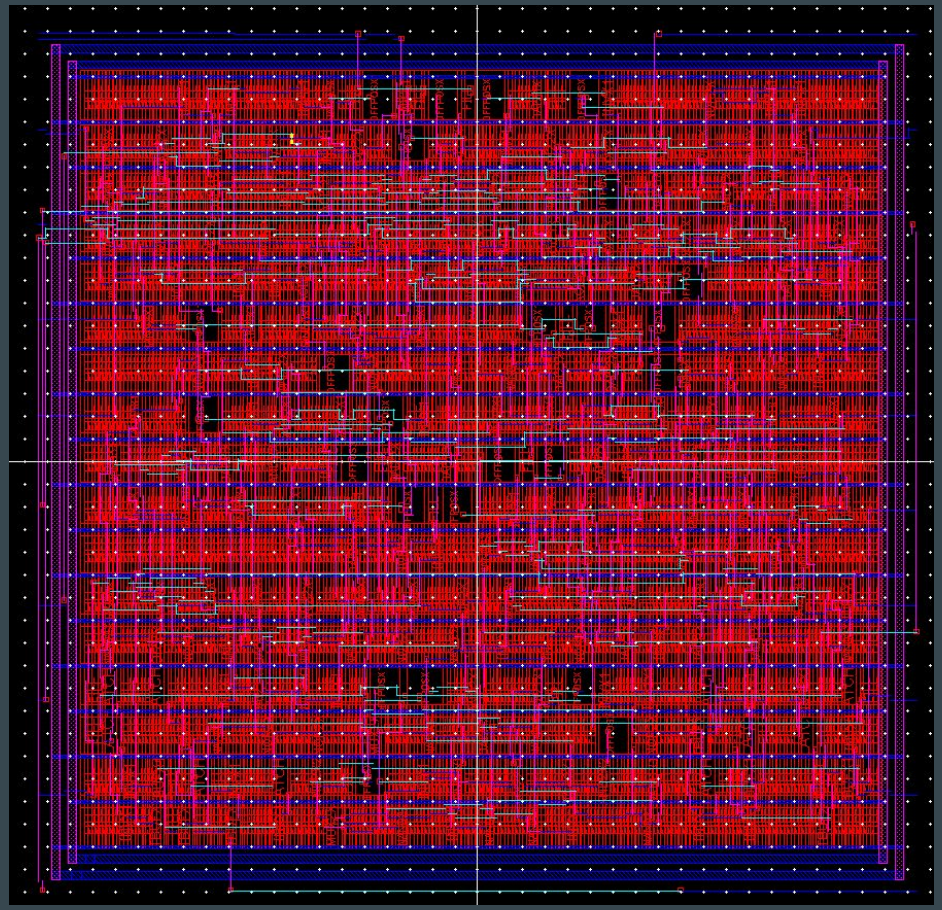
Terminal - bryanf@co2046-10:~/ee330

File Edit View Terminal Tabs Help

```
  VERIFY GEOMETRY ...... Wiring          :  0 Viols.
  VERIFY GEOMETRY ...... Antenna         :  0 Viols.
VG: elapsed time: 1.00
Begin Summary ...
  Cells       : 0
  SameNet     : 0
  Wiring      : 0
  Antenna     : 0
  Short       : 0
  Overlap     : 0
End Summary

  Verification Complete : 0 Viols.  0 Wrngs.

**********End: VERIFY GEOMETRY**********
 *** verify geometry (CPU: 0:00:00.1  MEM: 72.7M)

innovus 1> VERIFY_CONNECTIVITY use new engine.

******** Start: VERIFY CONNECTIVITY ********
Start Time: Mon Dec 16 10:49:04 2019

Design Name: Transceiver
Database Units: 1000
Design Boundary: (0.0000, 0.0000) (193.6000, 190.0000)
Error Limit = 1000; Warning Limit = 50
Check all nets

Begin Summary
  Found no problems or warnings.
End Summary

End Time: Mon Dec 16 10:49:04 2019
Time Elapsed: 0:00:00.0

******** End: VERIFY CONNECTIVITY ********
  Verification Complete : 0 Viols.  0 Wrngs.
  (CPU Time: 0:00:00.0  MEM: 0.000M)

innovus 1>
```

Importing, DRC & LVS

- We exported from Innovus and imported into Virtuoso (had to fix a bug)
- DRC and LVS both passed after some troubleshooting / bug fixing



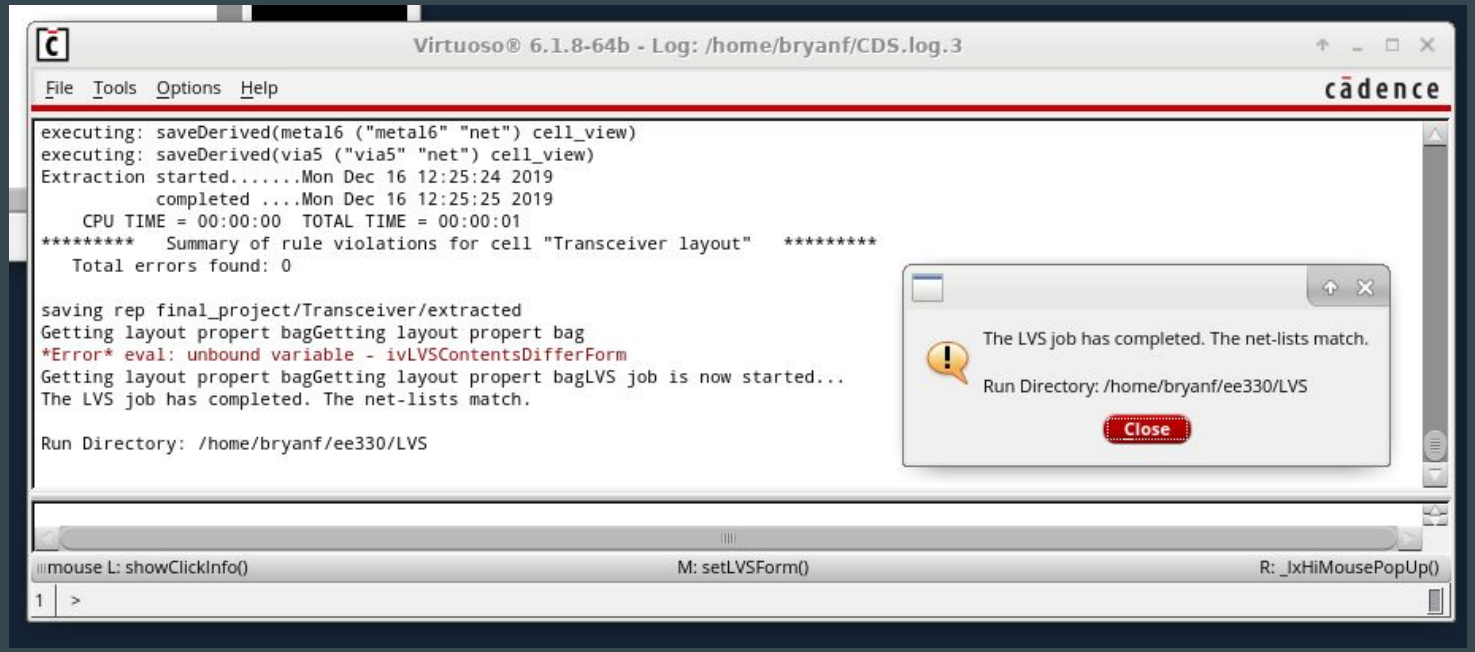
DRC Success

Virtuoso® 6.1.8-64b - Log: /home/bryanf/CDS.log.3

File Tools Options Help

```
            drc(metal6Edge (notch < (lambda * 5.0)) errMesg)
executing: drc(metal6Edge via5Edge (enc < (lambda * 1.0)) errMesg)
executing: saveDerived(geomAndNot(via5 metal6) errMesg)
DRC started.......Mon Dec 16 12:23:26 2019
    completed ....Mon Dec 16 12:23:27 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:01
*********   Summary of rule violations for cell "Transceiver layout"   *********
   Total errors found: 0
```

LVS Success


Virtuoso® 6.1.8-64b - Log: /home/bryanf/CDS.log.3
File Tools Options Help
cādence
executing: saveDerived(metal6 ("metal6" "net") cell_view)
executing: saveDerived(via5 ("via5" "net") cell_view)
Extraction started.....Mon Dec 16 12:25:24 2019
completed ....Mon Dec 16 12:25:25 2019
CPU TIME = 00:00:00 TOTAL TIME = 00:00:01
********* Summary of rule violations for cell "Transceiver layout" *********
Total errors found: 0
saving rep final_project/Transceiver/extracted
Getting layout propert bagGetting layout propert bag
Error eval: unbound variable - ivLVSContentsDifferForm
Getting layout propert bagGetting layout propert bagLVS job is now started...
The LVS job has completed. The net-lists match.
Run Directory: /home/bryanf/ee330/LVS
The LVS job has completed. The net-lists match.
Run Directory: /home/bryanf/ee330/LVS
Close
mouse L: showClickInfo()
M: setLVSForm()
R: _lxHiMousePopUp()
1
>

Proof of Concept for Pad Frame