

## Delay Prediction Homework

- Work in Groups of two or three
- All Spice problems in this homework are to be done for technologies
  - Take the last digit of each of your student ID's and form a sum.
  - If this sum is odd use Technology: tsmc025, Vdd=3.3V, default temp
  - If this sum is even use Technology: tsmc025, Vdd=2.5V, default temp
  - all input waveforms should have rise/fall times of 200 ps.
- Use either the new Linux boxes, or log in remotely to hoth0/hoth1/hoth2 which are compute servers.

BR 6/00

1

## Problem Statement

- Build 2D Transition, Propagation delay tables for 1X, 2X, 4X inverters, 2-input NAND gates (1X, 2X, 4X), 2-input NOR gates (1X, 2X, 4X)
- Build Tplh/Tphl delay tables, rising/falling output transition tables
  - Inverters require a total of 3 sizes \* 2 delays \* 2 tables/dly = 12 tables.
  - NAND gates require a total of 3 sizes \* 4 delays \* 2 tables/dly = 24 tables
  - NOR gates require a total of 3 sizes \* 4 delays \* 2 tables/dly = 24 tables
- Capacitive load points are measured in inverter equivalent loads. Table Cap load points should be: 1X, 3X, 6X, 12X, 25X inverter loads.
  - Use actual capacitance units for load
- Use three values for input transition time measured 30% to 70%.
  - Minimum input transition case: Largest inverter(4X) driving DUT
  - Maximum input transition case: 1X NOR driving DUT + off path load such that total load is 25X
  - Typical transition case (you pick).
  - Transition indexes are different for each DUT!
- Total number of points in table will be  $3 * 5 = 15$  data points.

BR 6/00

2

## Nand/Nor Gates

For 1X NAND gates, width of NMOS =  $2 * w_{min}$

Input pins are A,B with A input closest to output. Scale other gates linearly.

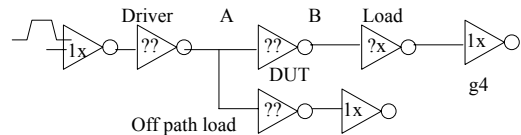
For 1X NOR gates, width of PMOS =  $1.5 * 2 * w_{min}$

Input pins are A,B, with A input closest to output. Scale other gates linearly.

BR 6/00

3

## Part #1 – Data Measurement

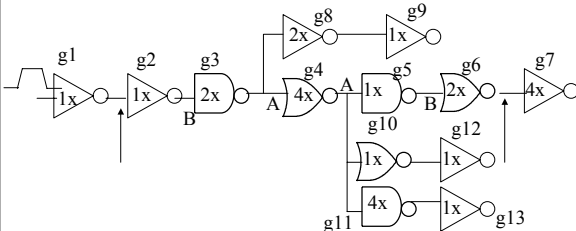


- Choose driver and offpath load for input transition case (minimum, maximum, typical).
- Measure TPLH, TPHL for the gate marked as DUT for output loads of 1X, 3X, 6X, 12X, 25X
- Measure Output Transition Rising, Output Transition falling same loads.
- Repeat a-c for all driver cases
- Repeat a-d for all DUT cases (1X, 2X, 3X).

BR 6/00

4

## Part #2: Delay Prediction



1. Measure TPLH, TPHL between the points indicated.
2. Compute the delay based upon your lookup tables using Gaussian elimination (see Synopsys Library Compiler Guide, Chap 2, pgs 2-25 thru 2-28).

BR 6/00

5

## Comparison

- Do delay computations by hand
  - Since starting rise/fall time = 200 ps, then the 30%-70% time for this would be  $0.4 * 200$  ps = 80 ps starting input transition time
- Plug data into Synopsys, compare your delays against what Synopsys predicts.
- Compare both against Spice measurements.
- Will talk more about Synopsys formats later
- For now, just gather data – Perl scripting could help!!!

BR 6/00

6

### 'C' Code for Gaussian Elimination

- I have provided some C code for solving the set of equations required for the 2-D interpolation of the look up tables
  - mygauss.c -- contains 'gaussj' procedure which does the solving, and a 'main' routine for driving it. The main routine uses the equations from the Synopsys documentation
  - To compile, just do 'make' (I have provided a Makefile for doing the compilation).
- When executed, 'mygauss' will print out the starting matrixes, then print out the solution 'B' matrix.
- Make use of this code in any way that you see fit to perform the hand calculation

### Synopsys .lib file

- I have provided a Synopsys .lib file called *lab1.lib*
  - Contains cell definitions for INVX1, INVX2, INVX4, NAND2X1, NAND2X2, NAND2X4, NOR2X1, NOR2X2, NOR2X4
  - All timing/capacitance data is dummy data -- replace with your own data
  - Represent all times in ps, all capacitances in fF.
- To compile this library do:
  - swsetup synopsys
  - dc\_shell -f compile\_lab1.script

### Using Synopsys to Compute Delays

- I have provided a Verilog file that defines the test circuit
  - Name of file is 'path.v'
- A Synopsys script file is provided that can be used to compute delay, and internal net transition times, net loading
  - Script file name is 'path.script'
- To run synopsys using this script do:
  - dc\_shell -f path.script

### Synopsys Docs

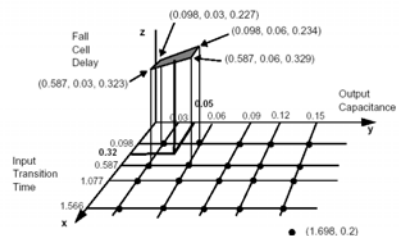
- Path to Synopsys Docs on 2-D interpolation
    - /opt/ecad/synopsys/default/doc/online/library/lcug2/lcug2\_2.pdf
- The PDF file is chapter 2.  
lcug2 = Library Compiler User Guide, Part 2.

### Last Year's Solution

- I have included some files from last year's solution as examples, these will have to be modified and there are other ways to do this!
  - char\_mintran.sp, midtran.sp, maxtran.sp -- Spectre simulation files for characterization
  - There are no files for pin load characterization as last year we used inverter load units instead of absolute capacitance -- you will have to write these since this year want to use real capacitance values
  - parse\_logs.pl - Perl script used to parse log files from Spectre output and create tables needed by both the Synopsys .lib file and my header file
- Use these files just to give you *general* ideas on how to approach this. If you spend all of your time trying to figure what I did, then you will never figure out a correct solution.

### 2D Interpolation

Figure 2-7 Result of Delay Calculation

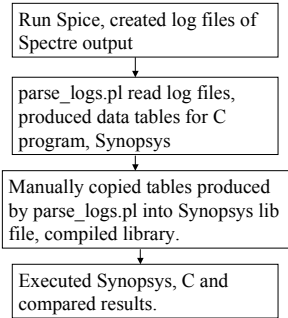


### 2D Interpolation (cont)

Table Entry      Transition time      Output Cap  
 $0.227 = A + B * 0.098 + C * 0.03 + D * 0.098 * 0.03$   
 $0.234 = A + B * 0.098 + C * 0.06 + D * 0.098 * 0.06$   
 $0.323 = A + B * 0.587 + C * 0.03 + D * 0.587 * 0.03$   
 $0.328 = A + B * 0.587 + C * 0.06 + D * 0.587 * 0.06$

Solve for A, B, C, D. Then plug in measured transition time and output load into equation to get new transition time or propagation delay.

### My Methodology



### My Methodology Details

Had Spice files called for mintran.sp, midtran.sp, maxtran.sp for three transition cases; the files ran all of the load cases. Probably could have collapsed these to one file.

The 11cells.sp defined a subcircuit called INVGEN that could generate a particular sized inverter.

parse\_logs.pl expected log files named mintran.log, midtran.log, maxtran.log. Read these logs, wrote results to 'synop\_table.txt', 'c\_info.txt'. Manually inserted synopsys data into .lib file, C tables into C header file contained c\_src directory.

Executed Synopsys, C program – compared the results.

### Report

- Must be in a file titled 'report.pdf'
- Have a table that compares
  - your calculations vs. Synopsys
  - Your calculations vs Spectre
  - Synopsys vs Spectre
  - Give percent error for all comparisons . If 'x' is the golden value, then % error is 'y-x'/x \* 100
  - If calculations/Synopsys deviate by more than 10% from Spectre, give me an explanation – stage by stage delay comparisons might help.
- Give your 'typical' drive choices and a rationale
- Explain the files in your submission and the methodology you used to produce your results. I want to be able to duplicate everything that you have done. This includes
  - Producing the raw data
  - Running Synopsys on your library file
  - Running any 'C' programs or equivalent that helped you with the delay calculations

### Submission Files

- I want all of your files except for Spectre simulation results (delete this directory before submitting your archive).
- Put all files in a directory called 'sim1'. Execute the submission script from the directory above 'sim1'
  - Do 'perl submit\_ee8273\_sim1.pl'
  - Will create a UU-encoded tar archive of your submission and email it to me
  - You can submit multiple times, I only look at the last one.

### My Solution

	Path TPLH	Path TPHL
Cadence	395	455
Synopsys	396	465
%diff (Cadence)	-0.3%	-2.2%
C_code	391	456
%diff (Synopsys)	1.01%	-0.22%

Very good agreement, better than what is usually expected.

## Sample Data

/\* Tables for invx2 \*/

```
cell_rise(t5x3){
  index_1("3.0, 9.28, 18.7, 37.7, 77.8");
  index_2("23.95, 38.46, 177.68");
  values("34.80, 45.31, 111.74",\
    "42.12, 54.08, 130.74",\
    "51.88, 65.91, 154.73",\
    "71.26, 87.26, 189.72",\
    "113.02, 129.77, 244.47");
}
```

For mid-transition,  
used nor2x2 with off  
path load of 3, 2, 1 for  
1x, 2x, 4x cases.

```
cell_fall(t5x3){
  index_1("3.0, 9.28, 18.7, 37.7, 77.8");
  index_2("22.56, 58.02, 273.63");
  values("31.07, 44.34, 113.51",\
    "38.40, 54.55, 132.81",\
    "48.35, 67.83, 159.37",\
    "66.96, 91.12, 205.22",\
    "105.78, 134.44, 270.58");
}
```

For max-transition  
used nor2x1 with off  
path load of 11x. This  
is probably too much  
load, transition was too  
slow.