CAD5

- Designing the first five cells in your library
  - Multiple cell views
- Liberate library characterizer
- Abstract generator
- Synopsys database generation
  - Using the cells in synthesis

CS/ECE 6710 Tool Suite

- Behavioral Verilog
- Synopsys Design Compiler
- Your Library
- Cadence Innovus
- Verilog sim
- Structural Verilog
- Circuit Layout
- VCAR AutoRouter
- Cadence Virtuoso Layout
- LVS
- Layout-XL
- Verilog sim
- Cadence Composer Schematic
In the CAD Book

- Chapter 8 on Cell Characterization
  - Section 8.1 describes .lib format
  - Section 8.2 describes ELC – that tool is gone…
    - Instead we’ll use Cadence Liberate
  - Section 8.3 describes characterization by hand with Spectre (don’t do it!)
  - Section 8.4 describes converting from .lib to .db format (used by Synopsys Design Compiler)
    - Use syn-lc instead of syn-dc…
    - (.lib is used also by other tools…)

Start with Cells
Use `cmos_sch` as the view type for your cell schematics.

This is to differentiate them from schematics that use the cells.
Symbol View

Behavioral View

module NAND2X1 (Y, A, B);
  output Y;
  input A;
  input B;
  
nand _i0(Y, A, B);
  
specify
    {A => Y} = {1,0, 1,0};
    {B => Y} = {1,0, 1,0};
 endspecify

endmodule
Start with six views of each cell

1. layout – make sure to use the template!
2. cmos_sch – use this view type for a schematic
3. symbol – Make them look nice
4. behavioral – Having this view makes simulation go much faster (if you use it)
5. extracted – generated from the extract processes
6. analog_extracted – after LVS

Characterize the cells

- Run a set of analog (Spectre) simulations that builds a table of delay values
  - Input drive vs. output load
  - For all outputs
  - Plus rise and fall times
  - And some power information
- Cadence Liberate
  - cad-lib from the bin directory
  - But, don’t call it directly… use additional scripts…
Single Schematic with All Cells
Create Netlist of that ss-test Cell

Call it foo.scs (for example)
scl2liberate foo.scs libcells.scs

Convert to Liberate format

e simulator long spectre
global vss vdd

subsection IMX1 R V
  M1 (V A vdd vdd) ami100F u=6v l=688n p=9e-12 ad=9e-12 ps=15.0u \ p=15.0u n=+1 region=set
  M2 (V A vss vss) ami100H u=3v l=688n n=1.5e-12 ad=1.5e-12 ps=9u \ p=9u n=+1 region=set
ends IMX1

subsection VAO2SL R 0 V
  M2 (V 0 vdd vdd) ami100F u=6v l=688n p=9e-12 ad=9e-12 ps=15.0u \ p=15.0u n=+1 region=set
  M3 (V 0 vdd vdd) ami100F u=6v l=688n p=9e-12 ad=9e-12 ps=15.0u \ p=15.0u n=+1 region=set
  M4 (V 0 vss vss) ami100H u=3v l=688n n=1.5e-12 ad=1.5e-12 \ p=15.0u p=15.0u n=+1 region=set
ends VAO2SL

Run Liberate

- Liberate Library Characterizer
  - Figures out what each cell is (logic)
  - Generates test inputs for Spectre
  - Runs Spectre
  - Checks output and extracts timings
  - Formats the output in .lib format

- I like to make a new VLSI/Liberate directory from which to run this tool...
  - cp –r /uusoc/facility/cad_common/local/class/6710/F17/cadence/Liberate .
Liberate setup…

- A **lib** directory
  - where the generated `<library>.lib` file will be generated.

- A **netlist** directory
  - put your `libcells.scs` file in this directory.

```
$ ls
models/ README tcl/ templates/ userdata/
netlist/ run.sh* TechHeader.lef UofU_Cells.tcl
```

Liberate setup

- A **templates** directory
  - edit the `UofU_CellDefs.tcl` file in this directory to include cell descriptions for each of the cells you want to characterize. The `UofU_Templates.tcl` file has definitions for the timing and power templates that will be used for characterization. You probably don’t need to modify these unless you are using a different technology than ON Semi C5N.

```
$ ls
models/ README tcl/ templates/ userdata/
netlist/ run.sh* TechHeader.lef UofU_Cells.tcl
```


Liberate setup

- A **UofU_Cells.tcl** file
  - edit this file to make a list of the cells that you want to characterize in this run. This could be a list of every cell described in `templates/UofU_Cell_Defs.tcl`, or it could be a subset if you just want to try a few.

- A **userdata** directory
  - edit the `userdata.lib` file to reflect the areas and footprints of each of the cells in your library.

---

```bash
[lab@lab2-20 Liberate]$ ls
models/ README tcl/ templates/ userdata/
netlist/ run.sh* TechHeader.lef UofU_Cells.tcl
[lab@lab2-20 Liberate]$ 
```

---

Liberate setup

- A **tcl** directory
  - Edit the `UofU_Char.tcl` file in this directory to change the name of the library that the tool generates. If you don’t modify this, the tool will generate a file named `Lib6710_XX.lib` by default. The `settings.tcl` file in this directory has configuration commands for the Spectre simulator. You won’t need to modify this file at all.

```bash
[lab@lab2-20 Liberate]$ ls
models/ README tcl/ templates/ userdata/
netlist/ run.sh* TechHeader.lef UofU_Cells.tcl
[lab@lab2-20 Liberate]$ 
```
Liberate setup

- A **models** directory
  - Has the Spectre model files for the ami06N and ami06P transistors used by in your netlist.
- A **run.sh** shell script
  - This calls the cad-lib Liberate script with the appropriate input files, and makes a copy of the log information in a Liberate.log file.

```
$ models/ README.tcl/ templates/ userdata/
netlist/ run.sh* TechHeader.lef UofI_Cells.tcl
```

- Liberate tutorial on Canvas…
Changing library name in .lib
cell (INV1) {
  area : 129.6;
  cell_footprint : "INV1";
  cell_leakage_power : 0.0500074;
  pg_pin (vdd) {
    pg_type : primary_power;
    voltage_name : "vdd";
  }
  pg_pin (vss) {
    pg_type : primary_ground;
    voltage_name : "vss";
  }
  leakage_power () {
    value : 0.0500032;
    when : "(A = Y)";
    related_pg_pin : vdd;
  }
  leakage_power () {
    value : 0.0500137;
    when : "(!A = Y)";
    related_pg_pin : vdd;
  }
  leakage_power () {
    value : 0.0500074;
    related_pg_pin : vdd;
  }
  pin (Y) {
    direction : output;
    function : "A";
    nnn_capacitance : 0.01;
    power_down_function : "((vdd) + (vss))";
    related_ground_pin : vss;
  }
}

pin (Y) {
  direction : output;
  function : "A";
  nnn_capacitance : 0.01;
  power_down_function : "((vdd) + (vss))";
  related_ground_pin : vss;
  related_power_pin : vdd;
  nxx_capacitance : 0.6;
  timing () {
    related_pin : "A";
    timing_sense : negative_work;
    timing_type : combinational;
  
  cell_rise (delay_template_5x5 xl) {
    index_1 [0.06, 0.16, 0.42, 0.6, 1.2];
    index_2 [0.01, 0.05, 0.1, 0.3, 0.6];
    values (\
      0.113095, 0.226034, 0.329274, 0.514454, 1.74149, \
      0.184577, 0.301391, 0.42794, 0.67744, 1.81386, \
      0.364229, 0.50698, 0.59735, 1.10493, 1.96139, \
      0.495952, 0.549954, 0.719677, 1.25317, 2.07794, \
      0.634788, 0.847653, 1.04926, 1.65921, 2.46975, \n    );
  }
  rise_transition (delay_template_5x5 xl) {
    index_1 [0.06, 0.16, 0.42, 0.6, 1.2];
    index_2 [0.01, 0.05, 0.1, 0.3, 0.6];
    values (\n      0.087005, 0.160969, 0.394815, 0.844683, 1.6592, \
      0.0971017, 0.18262, 0.309307, 0.844673, 1.65416, \
      0.148417, 0.248046, 0.35952, 0.852047, 1.65428, \
      0.182127, 0.26049, 0.426854, 0.927542, 1.65481, \n      0.277962, 0.421202, 0.552645, 1.00554, 1.71487, \n    );
  }
}
Converting .lib to .db

Converting .lib to .db

Converting .lib to .db

Converting .lib to .db
Chapter 10 – Abstract Generation
- Abstract views are used by place and route
- Eventually they are captured in a .lef file that describes the place and route geometry
Generating Abstract Views

Imported library in cad-abstract
Green checks are good!

Abstract Generation
Export Lib6710_00.lef file

Replace highlighted text with TechHeader.lef from Liberate directory
“technology lef”

Replace highlighted text with TechHeader.lef from class directory

Final CAD5 Files…

- **Nine views of every cell**
  - abstract, abstract.ext, abstract.pin,
    analog_extracted, behavioral, cmos_sch,
    extracted, layout, symbol
  - DRC and LVS-checked, and simulated

- **Three versions of the library description**
  - Lib6710_00.lib // timing information
  - Lib6710_00.db // design compiler target lib
  - Lib6710_00.lef // place and route info
All Nine Views…

Test with beh2str

- beh2str addsub.v addsub_dc.v Lib6710_00.db

- Results in addsub_dc.v and addsub_dc.v.rep

- NOTE! Your 5-cell library has no DFF, so you can’t synthesize any sequential circuits!
  - Combinational circuits only at this point…
module addsub ( a, b, addnsub, result );
  input [7:0] a;
  input [7:0] b;
  output [8:0] result;
  input addnsub;
  wire n26, n27, n28, n29, n30, n31, n32, n33, n34, n35, n36, n37, n38, n39,
            n40, n41, n42, n43, n44, n45, n46, n47, n48, n49, n50, n51, n52, n53,
            n54, n55, n56, n57, n58, n59, n60, n61, n62, n63, n64, n65, n66, n67,
            ...
  NAND2X1 U33 ( .A(n26), .B(n27), .Y(result[8]) );
  NAND2X1 U34 ( .A(b[7]), .B(n28), .Y(n27) );
  NAND2X1 U35 ( .A(n29), .B(n30), .Y(n28) );
  NOR2X1 U36 ( .A(n31), .B(n32), .Y(n29) );
  ...
  NOR2X1 U216 ( .A(n188), .B(a[0]), .Y(n175) );
  NAND2X1 U217 ( .A(a[0]), .B(n188), .Y(n202) );
  INVX1 U218 ( .A(b[0]), .Y(n188) );
endmodule

Operating Conditions: typical   Library: Lib6710_00
Wire Load Model Mode: top
Startpoint: b[1] (input port)
Endpoint: result[8] (output port)
Path Group: (none)
Path Type: max

<table>
<thead>
<tr>
<th>Point</th>
<th>Incr</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>input external delay</td>
<td>0.00</td>
<td>0.00 r</td>
</tr>
<tr>
<td>b[1] (in)</td>
<td>0.00</td>
<td>0.00 r</td>
</tr>
<tr>
<td>U198/Y (NOR2X1)</td>
<td>0.50</td>
<td>0.50 f</td>
</tr>
<tr>
<td>U194/Y (NOR2X1)</td>
<td>0.36</td>
<td>0.86 r</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U34/Y (NAND2X1)</td>
<td>0.25</td>
<td>10.16 f</td>
</tr>
<tr>
<td>U33/Y (NAND2X1)</td>
<td>0.24</td>
<td>10.40 r</td>
</tr>
<tr>
<td>result[8] (out)</td>
<td>0.00</td>
<td>10.40 r</td>
</tr>
<tr>
<td>data arrival time</td>
<td>10.40</td>
<td></td>
</tr>
</tbody>
</table>

(Path is unconstrained)
addsub_dc.v.rep

Library(s) Used:

Lib6710_00 (File: /home/elb/VLSI/cadence-f13/syn-f13/CAD5test/Lib6710_00.db)

- Number of ports: 26
- Number of nets: 203
- Number of cells: 186
- Number of combinational cells: 186
- Number of sequential cells: 0
- Number of macros: 0
- Number of buf/inv: 37
- Number of references: 3

Summary

- You now have a library that is fully functional
  - BUT – only on combinational circuits
  - No DFF yet!

- Every step of the way requires extreme care to get things exactly right
  - No trick to finding the right answer
  - The point is to practice working with the data & tools
Look ahead to Innovus

Modify the Script Files

# configuration file should be named <BASENAME>.globals.
set BASENAME "addsub"

# The following variables are used in fplan.tcl...
# These set the percent utilization target (how dense should
# the cells be placed), and the gap for routing between rows.
# These are good starting values for small macros. Larger or
# more complex macros will likely need a lowered uspect or
# larger rowgap or both.
# Note that rowgap and coregap should be divisible by the basic
# grid unit of 0.3 that our AMCSM/F process uses.
set uspect 0.78  # percent utilization in placing cells
set rowgap 15   # gap (microns) between pairs of std cell rows
set coregap 30.0 # gap (microns) between the core and the power rails
set aspect 0.66  # aspect ratio of overall cell (1 is square,
# >1 is landscape, >1 is portrait)

# The following variables are used in pplan.tcl...
# These numbers control the power and ground grid.
# Note that all these numbers should be divisible by 0.3 so
# that they fit on the lambda grid
set width 9.9    # power rail width
set space 1.8    # power rail space
set width 4.8    # power stripe width
set space 123    # power stripe spacing
set offset 128   # power stripe offset to first stripe
set power net vdd   # the name of the power net
set ground net gnd   # the name of the ground net
Modify the Script Files

Modify the Script Files

Modify the Script Files
Placed and Routed Result