EECS 427
Lecture 13: Synthesis-Based Design Flow

Reading: Inserts E and F

Lecture Overview

- Synthesis introduction
  - Leads into Verilog, to be described in discussion today

Logic Synthesis: overview

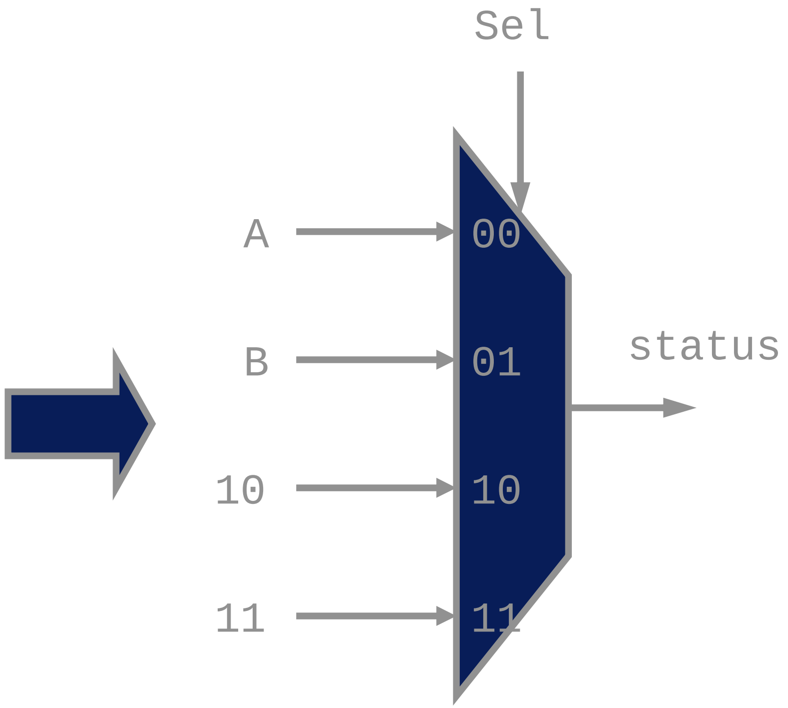
- Logic synthesis:
  - Process of transforming Hardware Description Language (HDL) code into a logic circuit
- HDL
  - VHDL
  - Verilog
- The circuitry:
  - Structural-level HDL netlist
  - Components from a technology specific library

Logic Synthesis

- Logic synthesis translates HDL code into a connected set a standard cells (called a netlist)
- Real cell properties must be accounted for in order to insure that the actual circuit will perform correctly:
  - Propagation delay through cells
  - Connection (wire) delay between cells
  - Load Capacitance
  - Drive Resistance
  - Slew rate (10%- 90%)
  - Area of Cells
  - Clock rates
  - Setup & hold times
  - Power Consumption

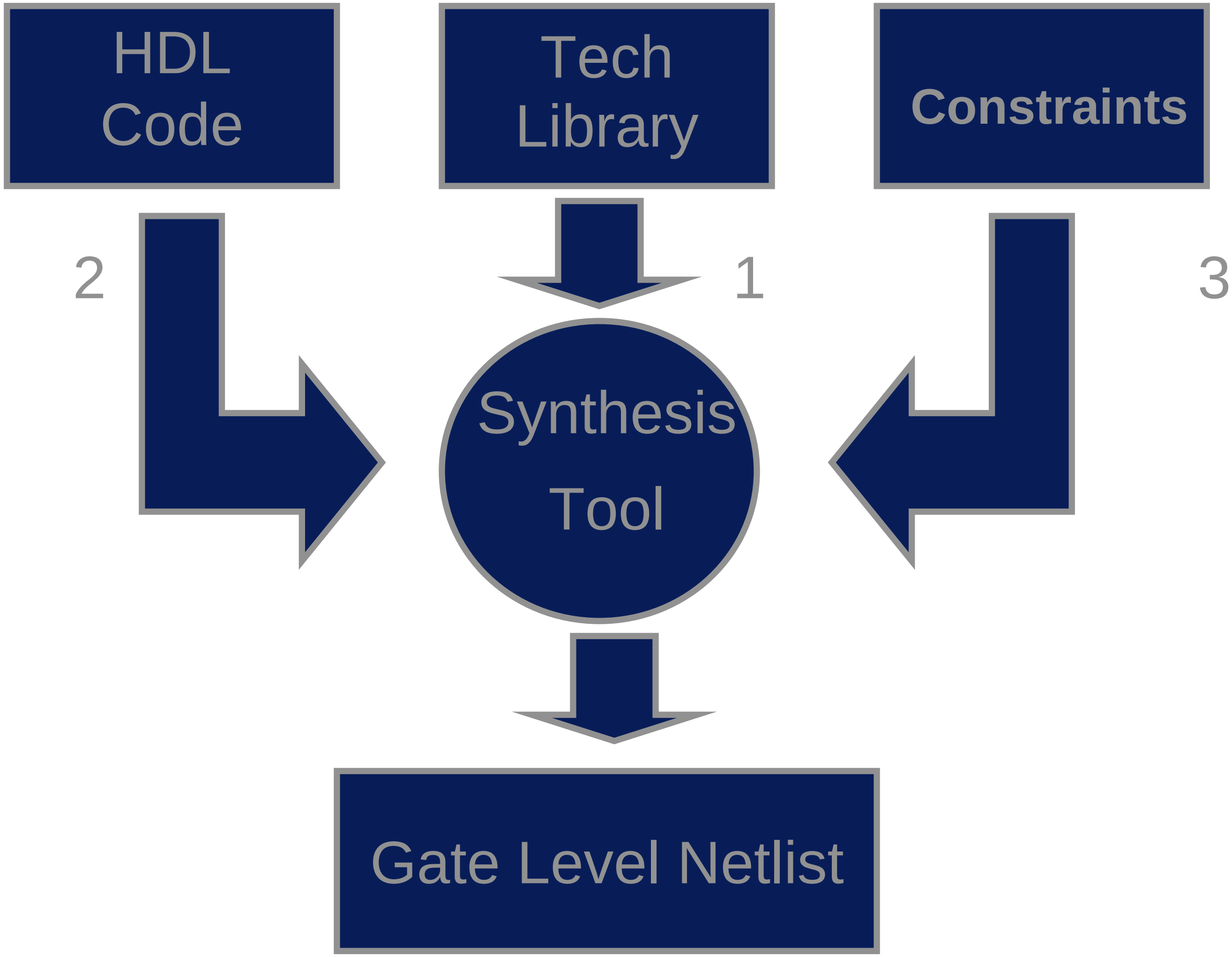
Ideal is converted into Real

```
case (Sel)
    2'b00   : status <= A;
    2'b01   : status <= B;
    2'b10   : status <= 2'b10;
    2'b11   : status <= 2'b11;
endcase
```




```
Mux4x1  1x Drive Strength
Area: 12microns
Max Transition 1.0 ns
Input Pin capacitance: .0015pf
Output Pin Max Load: .45 pf
Cell Delay (A -> Z) with .455pf load
   and .5ns slew is .3ns
```

Synthesis Data Flow



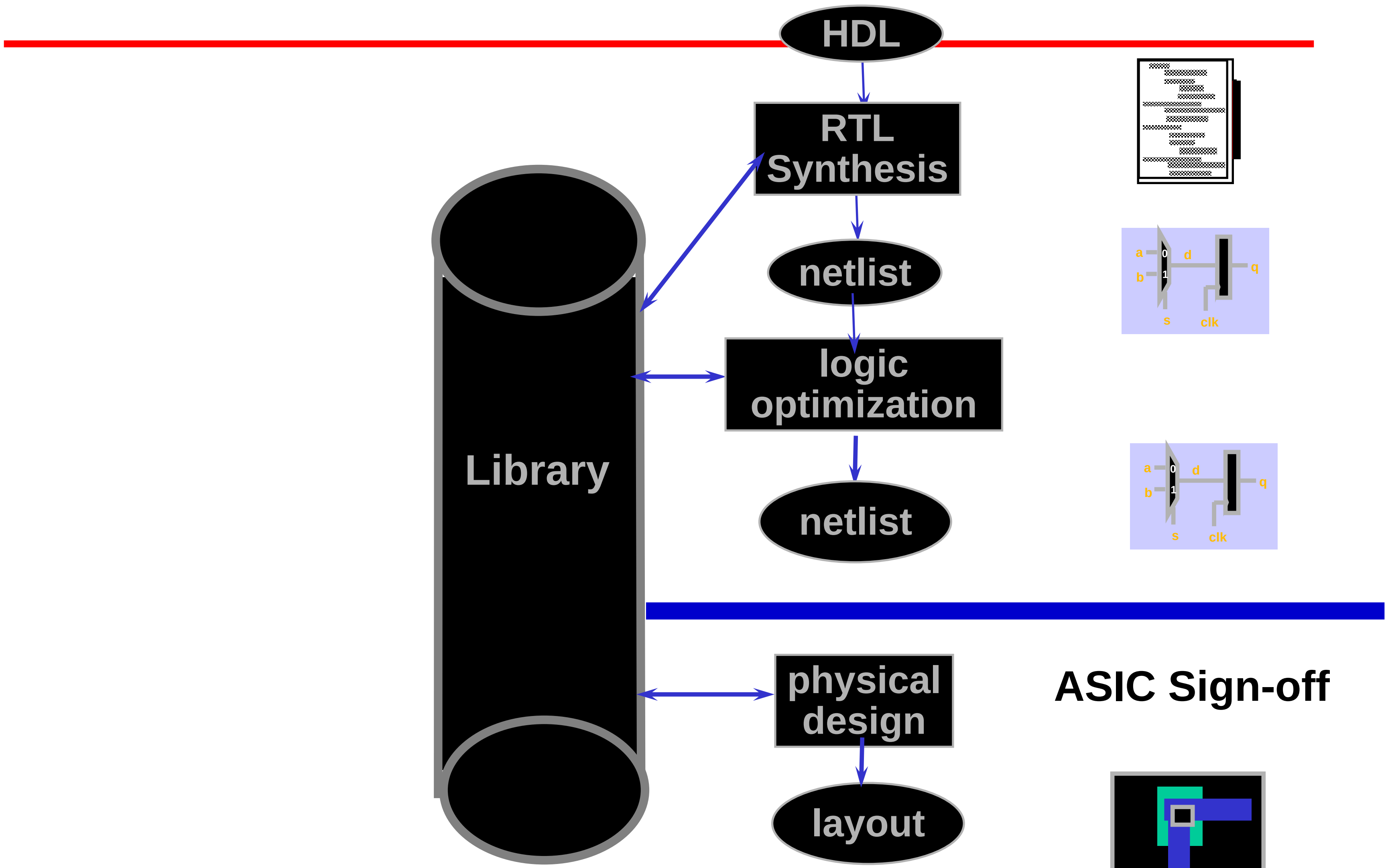
Overall Synthesis Flow

- Describe design with synthesizable Verilog HDL
  - Could be VHDL, another language
- Verify logic functionality (NCVerilog)
- Derive timing/area constraints
  - Increasingly common to add power constraints too
- Synthesize the design

Logic Synthesis

- Absolutely vital to the proliferation of complex digital ICs
  - Many large companies have their own tools
  - Synopsys Design Compiler (DC) dominates the commercial space
  - Traditional synthesis tools are being displaced by advanced Physical synthesis toole like Physical Compiler

Traditional Flow



Reading a Library

- Synopsys: read_db msi_10k.db
  - If the .db library file doesn't exist then it must be created from the .lib library file (vendor supplied)
  - .lib is readable by the user, .db is internal format
  - Use library compiler to create the .db
    [libcompile msi_10k.lib   -> msi_10k.db]
  - Library Compiler
    - Checks the .lib for errors
    - Translates to .db

Purpose of the Library

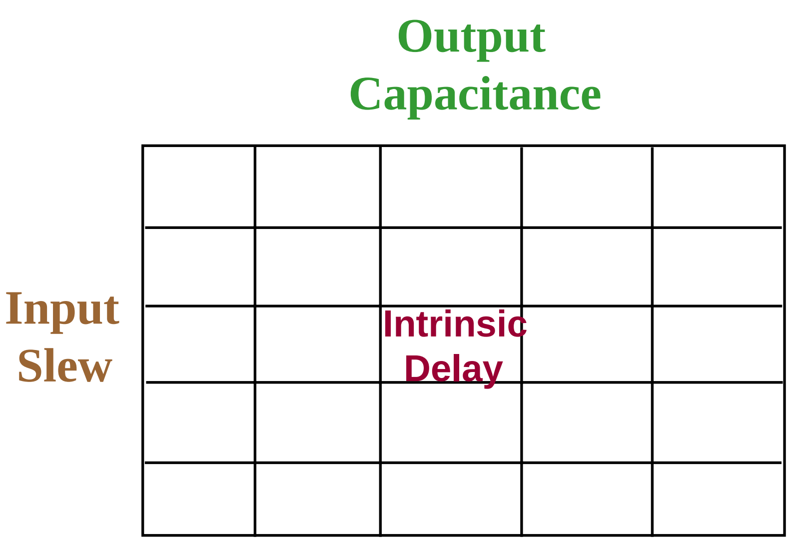
- The Library contains the cells of the technology (.25u)
  - Cells are “Building Blocks” for the circuit
  - Must use technology library to estimate physical properties
- Synthesis tools considers properties and function of cells
  - The key properties:
    - Cell delay
    - Rise/fall transitions
    - Capacitive load
    - Drive strength
    - Area and power
- $Delay_{Total} = Delay_{Cell} + Delay_{Wire}$

Contents of a Library

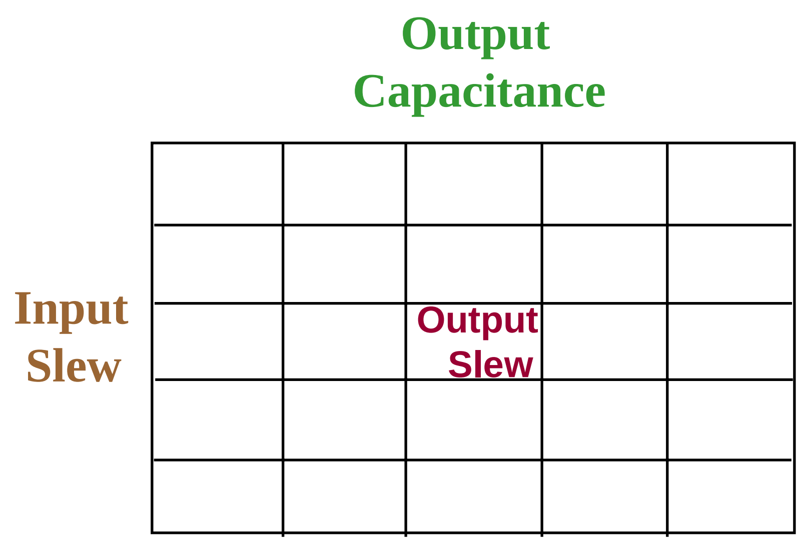
- Units (V, A, pW, KOhm, nS, etc)
- Default parameters
  - Max transition
  - Input pin cap
  - Wireload mode
  - Operating condition
  - Max fanout
- Nominal Parameters (PVT)
- Operating Conditions
  - Worst Case /Best Case
- Scaling factors
  - K Factors
- Wireload Models
  - Estimate for fan-in, fan-out
- Look-up table templates
- Cells: all properties & attributes, Delay Tables, Rise/Fall Transition Tables, Power Tables

Non-linear effects reflected in tables

- $D_G = f(C_L, S_{in})$ and $S_{out} = f(C_L, S_{in})$
  - Non-linear
- Interpolate between table entries
- Interpolation error is usually below 10% of SPICE



Delay at the gate output



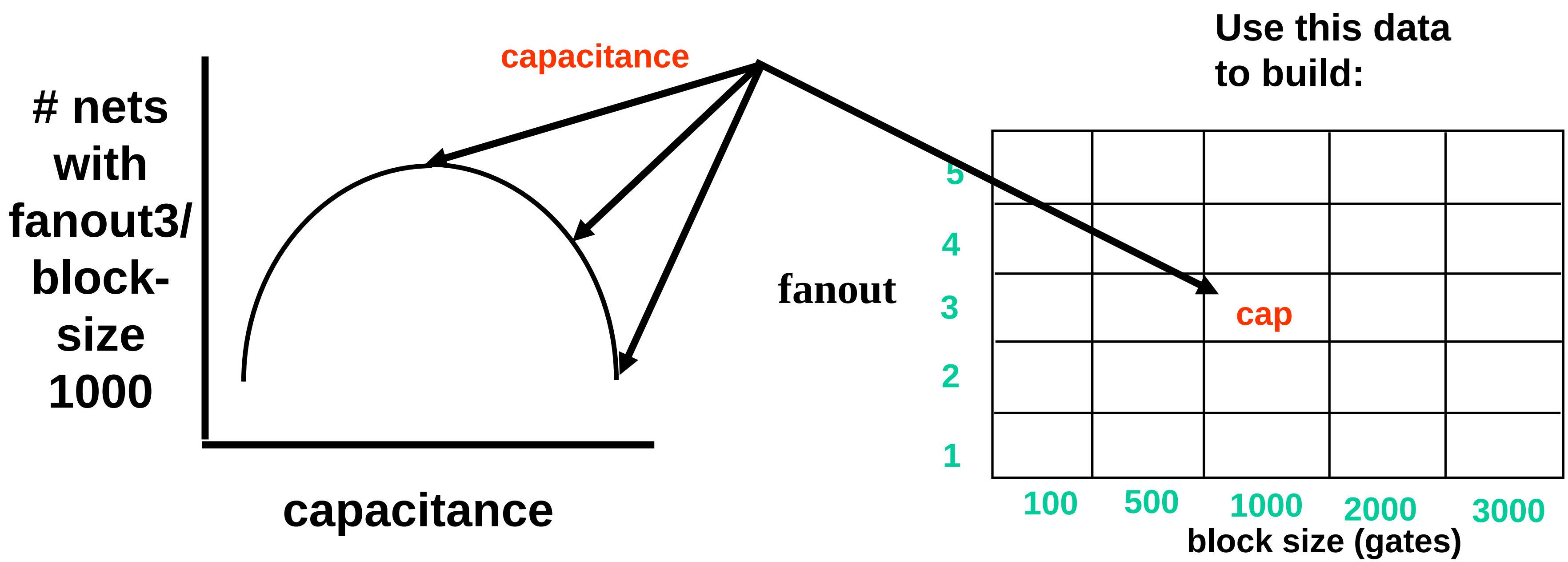
10-90% slew rate

Timing Library Example (.lib)

```
library(my_lib) {
  delay_model : table_lookup;
  library_features (report_delay_calculation);
  time_unit : "1ns";
  voltage_unit : "1V";
  current_unit : "1mA";
  leakage_power_unit : 1uW;
  capacitive_load_unit(1,pf);
  pulling_resistance_unit : "1kohm";
  default_fanout_load : 1.0;
  default_inout_pin_cap : 1.0;
  default_input_pin_cap : 1.0;
  default_output_pin_cap : 0.0;
  default_cell_leakage_power : 0.0;

  nom_voltage : 1.08;
  nom_temperature : 125.0;
  nom_process : 1.0;
  slew_derate_from_library : 0.500000;

operating_conditions("slow_125_1.08") {
  process     : 1.0 ;
  temperature : 125 ;
  voltage     : 1.08 ;
  tree_type   : "worst_case_tree" ;
}
default_operating_conditions : slow_125_1.08 ;

lu_table_template("load") {
  variable_1 : input_net_transition;
  variable_2 : total_output_net_capacitance;
  index_1( "1, 2, 3, 4" );
  index_2( "1, 2, 3, 4" );
}

cell("INV") {
pin(A) {
    max_transition : 1.500000;
    direction : input;
    rise_capacitance : 0.0739000;
    fall_capacitance : 0.0703340;
    capacitance : 0.07278646;
  }
  pin(Z) {
    direction : output;
    function : "!A";
    max_transition : 1.500000;
    max_capacitance : 5.1139;
    timing() {
      related_pin : "A";
      cell_rise(load) {
        index_1( "0.0375, 0.2329, 0.6904, 1.5008" );
        index_2( "0.0010, 0.9788, 2.2820, 5.1139" );
        values ( \
          "0.013211, 0.071051, 0.297500, 0.642340", \
          "0.028657, 0.110849, 0.362620, 0.707070", \
          "0.053289, 0.165930, 0.496550, 0.860400", \
          "0.091041, 0.234440, 0.661840, 1.091700" );
      }
      cell_fall(load) {
        index_1( "0.0326, 0.1614, 0.5432, 1.5017" );
        index_2( "0.0010, 0.4249, 3.6538, 8.1881" );
        values ( \
          "0.009472, 0.072284, 0.317370, 0.688390", \
          "0.009992, 0.095862, 0.360530,  0.731610", \
          "0.009994, 0.126620, 0.477260,  0.867670", \
          "0.009996, 0.144150, 0.644140,  1.127700" );
      }
fall_transition(load) {
        index_1( "0.0326, 0.1614, 0.4192, 1.5017" );
        index_2( "0.0010, 0.4249, 2.1491, 8.1881" );
        values ( \
          "0.011974, 0.071668, 0.317800, 1.189560", \
          "0.033212, 0.101182, 0.328540,  1.189562", \
          "0.059282, 0.155052, 0.389900, 1.202360", \
          "0.162830, 0.317380, 0.628160, 1.441260" );
      }
      rise_transition(load) {
        index_1( "0.0375, 0.1650, 0.5455, 1.5078" );
        index_2( "0.0010, 0.4449, 1.7753, 5.1139" );
        values ( \
          "0.016690, 0.115702, 0.418200, 1.189060", \
          "0.038256, 0.139336, 0.422960,  1.189081", \
          "0.076248, 0.213280, 0.491820, 1.203700", \
          "0.170992, 0.353120, 0.694740, 1.384760" );
      }
    }
```

Modeling Interconnect -Wire Load Models

- Synthesis and logic optimization rely on gross estimates of interconnect capacitance gathered from empirical data



Wireload model

```
wire_load("45Kto75K") {
        capacitance : 0.000070;
        resistance : 0.000042;
        area : 0.28;
        slope : 40.258665;
        fanout_length(1, 40.258865);
        fanout_length(2, 80.517750);
        fanout_length(3, 120.776600);
        fanout_length(4, 161.045450);
        fanout_length(5, 241.543200);
        fanout_length(6, 322.070900);
        fanout_length(7, 402.587600);
    }
```

Constrain the Design

- Constraints define the parameters of the block which the synthesis tool must (try to) meet
- Constraints define the relationships between the block and the rest of the chip
- These items are defined:
  - Clocks
  - Arrival times
  - Loading on the output pins
  - Drive resistance on the input pins
  - Timing exceptions
  - Operating Conditions
  - Wireload model

Constraints: Commands

	Synopsys
Clocks	create_clock
Input Delay	set_input_delay
Output Delay	set_output_delay
Output Load	set_load
Input Drive Resistance	set_driving_cell
Operating Conditions	set_operating_conditions
Wireload Model	set_wire_load

Constraints: set performance targets

- Synthesized design:
  - Bounded by constraints
  - Input/Output, clk; all bound design
- The RTL defines the functionality of design
  - The library contains the building blocks for building the design
  - The constraints tell the synthesis tool the timing and electrical relationships between the block and the chip

Optimization

Optimization:

- The generic netlist
  - Mapped to the cells in the tech library
  - Timing is computed (with a clocked design)
- Iteratively
  - Design Compiler restructures design until timing is met
- Targets:
  - Power
  - Area
  - Timing (cycle time)

Command:

Synopsys: compile

Optimization issues

- Wire load model is pretty bogus.
  - Not that big a deal when wire capacitance is low
  - Wire capacitance is becoming much more important
  - Tools that break traditional boundries and consider placement along with logic (Physical Compiler) are taking over
- Finding the optimal circuit is a hard (provably) problem.
  - Simplifications (limited cell library, a few fixed sizes for those cells, simplistic model of clocks) make things simpler but limit design possibilities.
  - Tools are good at local optimizations (the trees) but not at big picture things (the forest).
  - e.g. Synthesis tools can exploit inversion property but not make a ripple carry adder into a lookahead adder
  - Synthesis and APR Tools are very bad at exploiting regularity so memories and regular structure (like a barrel shifter) are beyond particularly large when synthesized

Reports

- Timing reports:
  - Created in Synopsys script
  - Generates longest paths
  - Timing: set-up/hold-time violations
- Timing not met:
  - Reconstrain design
  - Error in netlist
  - Too long (depth) logic path
- Area Report
  - To make sure design is not too large
  - Shouldn’t overoptimize in DC since final placement will truly determine chip area

Example of a Timing Report

```
Operating Conditions: WCCOM   Library: lsi_10k
Wire Load Model Mode: top

  Startpoint: crcprv_reg[0]
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: err_reg (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  crcscram16bi       10x10                 lsi_10k

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  crcprv_reg[0]/CP (FD2)                   0.00       0.00 r
  crcprv_reg[0]/QN (FD2)                   3.36       3.36 f
  U56/Z (NR2P)                             1.41       4.77 r
  U45/Z (ENP)                              2.25       7.02 f
  U112/Z (EOP)                             2.07       9.09 f
  U52/Z (EOP)                              2.25      11.34 f
  U151/Z (MUX21LP)                         1.11      12.46 r
  U14/Z (EN3)                              3.67      16.13 f
  U86/Z (EOP)                              2.17      18.30 f
  U49/Z (EOP)                              2.07      20.38 f
  U27/Z (ND6)                              2.51      22.88 r
  err_reg/TE (FD2S)                        0.00      22.88 r
  data arrival time                                  22.88

  clock clk (rise edge)                   10.00      10.00
  clock network delay (ideal)              0.00      10.00
  err_reg/CP (FD2S)                        0.00      10.00 r
  library setup time                      -1.25       8.75
  data required time                                  8.75
  -----------------------------------------------------------
  data required time                                  8.75
  data arrival time                                 -22.88
  -----------------------------------------------------------
  slack (VIOLATED)                                  -14.13
```

Summary

- Logic synthesis takes in a Verilog representation of a circuit and performs logic optimization (technology independent) and mapping to a standard cell library (technology dependent)
- Verilog will be described next
- Auto-place and route will be Thursday