EE/CprE 465 -- Fall 2014

Lab 3: Standard Cell Characterization

In last lab, you have built the standard cell DFF with Cadence Virtuoso. In this lab, you will learn how to characterize the cell. In particular, the objective of this lab is to:

1. Get to know about standard cell library
2. Learn simulation in analog environment
3. Perform cell characterization and analysis

Standard cell characterization means creating a set of models that accurately and efficiently represents the behavior (e.g., delay, timing constraints, input capacitance, power) of a standard cell. These models enable the cell to be effectively useful at earlier design stages (e.g., functional design, logic design). Without the models, very time-consuming operations like timing simulation and power extraction have to be repeatedly performed. In this lab, we will perform characterization of the DFF designed in last lab.

Before we start, we will look at an example of a characterized standard cell library. The posted file, tcbn65gpluswc.lib, is the TSMC 65nm process standard cell library. It includes characterization of all kinds of standard cells that we need to perform synthesis in future labs. And the TCBN65GPLUS TSMC 65nm Core Library Data book posted at last lab is the PDF file version containing general information about all standard cells in the library.

Please find the DFF cell named DFQD1 in the TSMC 65nm process standard cell library. You will see all the characterizations of this cell specified in many lookup tables as follows:

```
/* -------------- *
 * Design : DFQD1 *
 * -------------- */
cell (DFQD1) {
  cell_leakage_power : 52.072 ;
  leakage_power() {
  when : "!CP !D" ;
   value : 57.523 ;
  }
  leakage_power() {
  when : "!CP D" ;
   value : 55.551 ;
  }
  leakage_power() {
  when : "CP !D" ;
   value : 50.134 ;
  }
  leakage_power() {
  when : "CP D" ;
   value : 45.080 ;
  }
  area : 6.84 ;
  cell_footprint : "dfqd1" ;
  ff (IQ,IQN) {
    next_state : "D";
    clocked_on : "CP";
```

```
}
pin(CP)  {
  direction : input;
  capacitance : 0.0008;
  rise_capacitance : 0.0008;
  fall_capacitance : 0.0008;
  clock : true;
  internal_power() {
    rise_power(passive_energy_template_7x1) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      values ("0.0021, 0.0021, 0.0020, 0.0020, 0.0020, 0.0021, 0.0024");
    }
    fall_power(passive_energy_template_7x1) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      values ("0.0042, 0.0042, 0.0042, 0.0041, 0.0041, 0.0043, 0.0046");
    }
  }
  min_pulse_width_high : 0.0571;
  min_pulse_width_low : 0.0778;
}
pin(D)  {
  direction : input;
  capacitance : 0.0011;
  rise_capacitance : 0.0011;
  fall_capacitance : 0.0011;
  nextstate_type : data;
  internal_power() {
    rise_power(passive_energy_template_7x1) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      values ("0.0016, 0.0016, 0.0016, 0.0016, 0.0016, 0.0019, 0.0025");
    }
    fall_power(passive_energy_template_7x1) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      values ("0.0027, 0.0026, 0.0026, 0.0026, 0.0027, 0.0030, 0.0036");
    }
  }
  timing() {
    related_pin : "CP";
    timing_type : hold_rising;
    rise_constraint(hold_template_3x3) {
      index_1 ("0.0056, 0.0840, 0.7088");
      index_2 ("0.0056, 0.0840, 0.7088");
 values("-0.0012, -0.0181, -0.0476", \
        "0.0157, 0.0012, -0.0307", \
        "0.0537, 0.0416, 0.0024");
    }
    fall_constraint(hold_template_3x3) {
      index_1 ("0.0056, 0.0840, 0.7088");
      index_2 ("0.0056, 0.0840, 0.7088");
 values("0.0220, 0.0076, -0.0268", \
        "0.0437, 0.0281, -0.0099", \
        "0.1184, 0.1039, 0.0643");
    }
  }
  timing() {
    related_pin : "CP";
    timing_type : setup_rising;
```

```
    rise_constraint(setup_template_3x3) {
      index_1 ("0.0056, 0.0840, 0.7088");
      index_2 ("0.0056, 0.0840, 0.7088");
  values("0.0244, 0.0425, 0.1110", \
         "0.0026, 0.0220, 0.0868", \
         "-0.0366, -0.0221, 0.0350");
    }
    fall_constraint(setup_template_3x3) {
      index_1 ("0.0056, 0.0840, 0.7088");
      index_2 ("0.0056, 0.0840, 0.7088");
  values("0.0085, 0.0267, 0.1171", \
         "-0.0157, 0.0024, 0.0917", \
         "-0.0988, -0.0820, -0.0024");
    }
  }
}
pin(Q)  {
  direction : output;
  max_capacitance : 0.0842;
  function : "IQ";
  timing() {
    related_pin : "CP";
    timing_sense : non_unate;
    timing_type : rising_edge;
    cell_rise(delay_template_7x7) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
  values("0.1045, 0.1110, 0.1230, 0.1464, 0.1926, 0.2849, 0.4689", \
         "0.1077, 0.1142, 0.1262, 0.1496, 0.1959, 0.2882, 0.4723", \
         "0.1142, 0.1208, 0.1328, 0.1562, 0.2024, 0.2947, 0.4788", \
         "0.1272, 0.1337, 0.1458, 0.1692, 0.2153, 0.3077, 0.4918", \
         "0.1486, 0.1551, 0.1671, 0.1906, 0.2367, 0.3291, 0.5132", \
         "0.1781, 0.1846, 0.1966, 0.2200, 0.2662, 0.3585, 0.5425", \
         "0.2167, 0.2233, 0.2353, 0.2587, 0.3046, 0.3969, 0.5812");
    }
    rise_transition(delay_template_7x7) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
  values("0.0210, 0.0307, 0.0515, 0.0956, 0.1839, 0.3609, 0.7152", \
         "0.0210, 0.0306, 0.0515, 0.0955, 0.1841, 0.3612, 0.7152", \
         "0.0210, 0.0307, 0.0515, 0.0954, 0.1841, 0.3612, 0.7148", \
         "0.0210, 0.0306, 0.0514, 0.0956, 0.1835, 0.3614, 0.7150", \
         "0.0210, 0.0307, 0.0515, 0.0956, 0.1835, 0.3614, 0.7150", \
         "0.0211, 0.0307, 0.0516, 0.0955, 0.1838, 0.3608, 0.7151", \
         "0.0215, 0.0310, 0.0517, 0.0955, 0.1840, 0.3609, 0.7158");
    }
    cell_fall(delay_template_7x7) {
      index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
      index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
  values("0.1240, 0.1293, 0.1384, 0.1548, 0.1859, 0.2477, 0.3711", \
         "0.1271, 0.1325, 0.1416, 0.1579, 0.1891, 0.2510, 0.3743", \
         "0.1337, 0.1390, 0.1482, 0.1645, 0.1957, 0.2574, 0.3808", \
         "0.1468, 0.1521, 0.1613, 0.1776, 0.2088, 0.2706, 0.3939", \
         "0.1683, 0.1736, 0.1828, 0.1991, 0.2302, 0.2920, 0.4154", \
         "0.1978, 0.2031, 0.2122, 0.2285, 0.2598, 0.3215, 0.4451", \
         "0.2358, 0.2412, 0.2503, 0.2666, 0.2976, 0.3592, 0.4827");
    }
```

```
      fall_transition(delay_template_7x7) {
        index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
        index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
   values("0.0177, 0.0241, 0.0370, 0.0635, 0.1185, 0.2316, 0.4570", \
          "0.0177, 0.0241, 0.0369, 0.0635, 0.1186, 0.2312, 0.4570", \
          "0.0177, 0.0241, 0.0369, 0.0635, 0.1186, 0.2315, 0.4565", \
          "0.0176, 0.0241, 0.0369, 0.0634, 0.1187, 0.2313, 0.4570", \
          "0.0177, 0.0241, 0.0369, 0.0634, 0.1185, 0.2316, 0.4565", \
          "0.0176, 0.0241, 0.0369, 0.0635, 0.1187, 0.2317, 0.4566", \
          "0.0177, 0.0241, 0.0369, 0.0634, 0.1187, 0.2315, 0.4569");
      }
    }
    internal_power() {
      related_pin : "CP";
      rise_power(energy_template_7x7) {
        index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
        index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
   values("0.0028, 0.0028, 0.0029, 0.0029, 0.0029, 0.0030, 0.0030", \
          "0.0028, 0.0028, 0.0028, 0.0029, 0.0029, 0.0030, 0.0030", \
          "0.0029, 0.0029, 0.0029, 0.0030, 0.0030, 0.0030, 0.0031", \
          "0.0028, 0.0028, 0.0029, 0.0029, 0.0030, 0.0030, 0.0031", \
          "0.0028, 0.0028, 0.0029, 0.0029, 0.0030, 0.0030, 0.0030", \
          "0.0028, 0.0028, 0.0029, 0.0029, 0.0029, 0.0030, 0.0030", \
          "0.0028, 0.0029, 0.0029, 0.0029, 0.0030, 0.0030, 0.0031");
      }
      fall_power(energy_template_7x7) {
        index_1 ("0.0056, 0.0168, 0.0392, 0.0840, 0.1728, 0.3520, 0.7088");
        index_2 ("0.0009, 0.0023, 0.0049, 0.0102, 0.0208, 0.0419, 0.0842");
   values("0.0031, 0.0031, 0.0031, 0.0032, 0.0032, 0.0032, 0.0033", \
          "0.0031, 0.0031, 0.0031, 0.0032, 0.0032, 0.0032, 0.0033", \
          "0.0031, 0.0032, 0.0032, 0.0032, 0.0033, 0.0033, 0.0033", \
          "0.0031, 0.0031, 0.0032, 0.0032, 0.0032, 0.0033, 0.0033", \
          "0.0031, 0.0032, 0.0032, 0.0032, 0.0033, 0.0033, 0.0033", \
          "0.0032, 0.0032, 0.0032, 0.0033, 0.0033, 0.0033, 0.0034", \
          "0.0032, 0.0033, 0.0033, 0.0033, 0.0033, 0.0034, 0.0034");
      }
    }
  }
}
```

You may find other DFFs named DFQD1, DFQD2, etc. They have the same function but have different driving strength. And so they have different characterizations in power, timing, area, etc.

In this lab, we will focus on the timing characterization of the cell. The part highlighted in red above describes part of the timing characterization of output pin Q. The line "`related_pin : "CP";`" means the action of the pin Q is caused by the action of the pin CP. For example, when pin CP rises from low to high, pin Q will rise or fall after a few nanoseconds delay. Let us use the red part as an example. In there, we have 2 lookup tables:

`cell_rise(delay_template_7x7)` and `rise_transition(delay_template_7x7)`.

They are 7x7 tables. In each table there are:

`index_1`, `index_2`, and `values`.

So what do they mean? In order to find out their meanings, we go back to the standard cell library file, tcbn65gpluswc.lib, and search for "`delay_template_7x7`". You should find:

```
  lu_table_template(delay_template_7x7) {
    variable_1 : input_net_transition;
    variable_2 : total_output_net_capacitance;
```

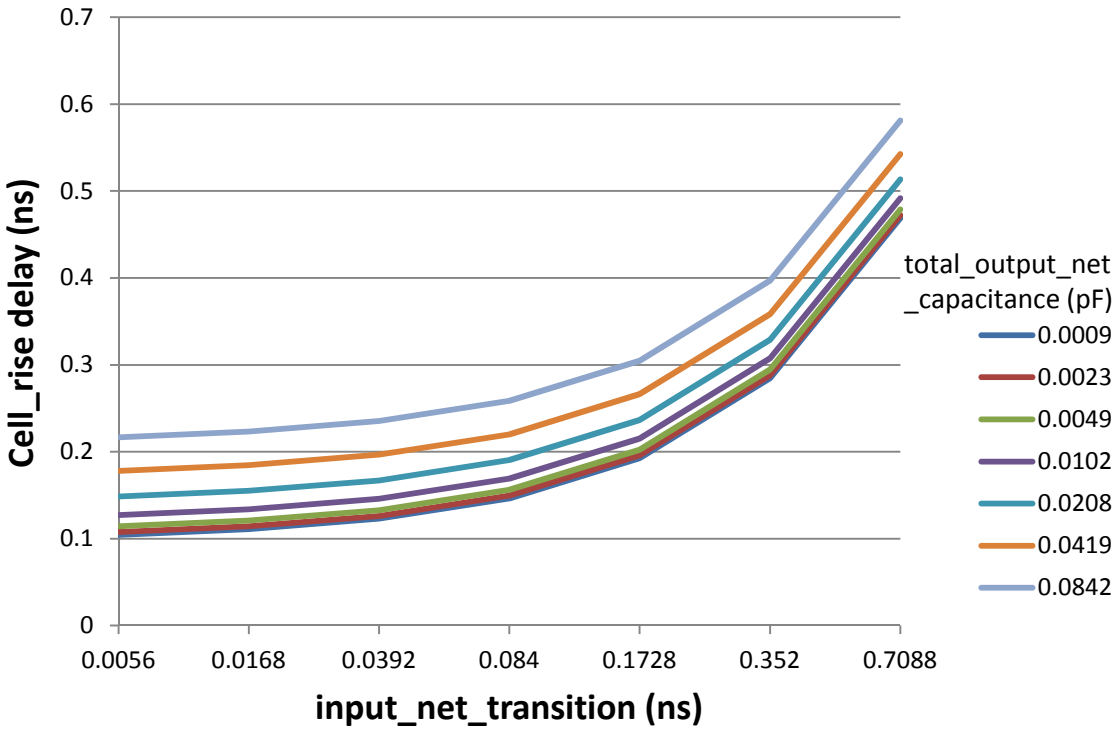
```
    index_1 ("1000.0, 1001.0, 1002.0, 1003.0, 1004.0, 1005.0, 1006.0");
    index_2 ("1000.0, 1001.0, 1002.0, 1003.0, 1004.0, 1005.0, 1006.0");
  }
```

This template tells you that for all tables using the `delay_template_7x7`, `variable_1` is the `input_net_transition` and `variable_2` is the `total_output_net_capacitance`. The values `"1000.0, …, 1006.0"` after `index_1` and `index_2` just placeholders indicating that the associated tables are 7x7. So the whole idea is, for different transitions of input signal CP, and for different load capacitance of the output signal Q, the output signal Q will have different timing characterizations as shown in the tables.

For the first table in red, `cell_rise`, it describes the rise propagation delay of output signal at pin Q according to different input transition time on CP and load capacitance at output pin Q itself. We can translate this table for convenience to read as below:

Rise propagation delay at output pin Q (ns)		input_net_transition at pin CP (ns)						
		0.0056	0.0168	0.0392	0.0840	0.1728	0.352	0.7088
total_output_net_capacitance at pin Q (pF)	0.0009	0.1045	0.1110	0.1230	0.1464	0.1926	0.2849	0.4689
	0.0023	0.1077	0.1142	0.1262	0.1496	0.1959	0.2882	0.4723
	0.0049	0.1142	0.1208	0.1328	0.1562	0.2024	0.2947	0.4788
	0.0102	0.1272	0.1337	0.1458	0.1692	0.2153	0.3077	0.4918
	0.0208	0.1486	0.1551	0.1671	0.1906	0.2367	0.3291	0.5132
	0.0419	0.1781	0.1846	0.1966	0.2200	0.2662	0.3585	0.5425
	0.0842	0.2167	0.2233	0.2353	0.2587	0.3046	0.3969	0.5812

If we show it in a trend chart, it will look like this:



Now let us talk about the second table, `rise_transition`, also highlighted in red above. It describes the rise transition time characterization (the time used from 30% to 70% of full voltage) of the output signal at pin Q, according to the different input transition time on CP and output load at pin Q itself.

But how are the thresholds for delay and transition defined? You will find these definitions in the library document as below:

```
slew_lower_threshold_pct_rise :  30.00
slew_upper_threshold_pct_rise :  70.00
slew_derate_from_library :  0.50
input_threshold_pct_fall :  50.00
output_threshold_pct_fall :  50.00
input_threshold_pct_rise :  50.00
output_threshold_pct_rise :  50.00
slew_lower_threshold_pct_fall :  30.00
slew_upper_threshold_pct_fall :  70.00
```

That means, for a signal rising from low voltage to high voltage, the transition time (i.e., slew) is calculated starting at the point when the signal passes 30% of the full voltage and stopping at the point when the signal passes 70% of the full voltage. The definitions are similar for a falling signal.

The delay for a rising signal is calculated starting at the point when the input signal passes 50% of the full voltage and stopping at the point when the output signal passes 50% of the full voltage. The definitions are similar for a falling delay.

But what are the units for time and load? And what are the working temperature and voltage condition? Please search for them in the library file. They are *ns* for time and *pF* for capacitance.

Lab Tasks:

1. Try your best to redraw your DFF layout to make it as small in area (calculated in layout environment) as possible. Finish LVS and DRC. Then please perform the following 16 simulation tasks with your DFF according to the table and the simulation settings below. Please note that one simulation can fill out two corresponding entries in the table. Do not waste time to simulate twice.

Cell area(um^2) :					
Output Rise delay value at pin Q (ns)		input_net_transition at pin CP (ns)			
		0.056	0.392	1.728	7.088
total_output_net_capacitance at pin Q (pF)	0.1				
	0.2				
	0.4				
	0.6				
Output Rise_transition at pin Q (ns)		input_net_transition at pin CP (ns)			
		0.056	0.392	1.728	7.088
total_output_net_capacitance at pin Q (pF)	0.1				
	0.2				
	0.4				
	0.6				

Simulation settings (the CMOS process we used may vary from year to year, so the simulation settings and the tables may change):

- Power supply: 1.3V
- Maximum metal layers to be used: 4

Note that to obtain more accurate timing for a design after layout, the parasitic should first be extracted and then post-layout simulation should be performed. Otherwise, the simulation would only be based on the schematic and it would always under-estimate the timing. A handout from EE330 explaining how to perform post-layout simulation is posted for your reference. However, to make this lab simpler, you may perform pre-layout simulation instead.

2. For this task, we do not only consider the area. Try your best to fine tune your DFF design to make it as small in area (calculated in layout environment) and as fast in timing as possible. For example, you may need to change the width to length ratio of your CMOS channels, and different CMOS transistors may have different ratios in your DFF design. The CMOS channel width and length partially decides the area of your DFF. Of course, your layout area also depends on how you draw your cell. You may notice that the timing and area you achieved is a trade-off. Sometimes, you have to use more area to make it switch faster. In this lab, you will experience this trade-off while tuning your DFF.

   We will set up a competition in the lab to see whose design has the smallest value of (cell_area*cell_rise_delay) for each of the 16 input transition and load combinations. We may have a similar competition for rise_transition if we have time. You should try to make the rise delay and fall delay roughly the same.

Cell area(um^2) :					
Output Rise delay value at pin Q (ns)		input_net_transition at pin CP (ns)			
		0.056	0.392	1.728	7.088
total_output_net_capacitance at pin Q (pF)	0.1				
	0.2				
	0.4				
	0.6				
Output Rise_transition at pin Q (ns)		input_net_transition at pin CP (ns)			
		0.056	0.392	1.728	7.088
total_output_net_capacitance at pin Q (pF)	0.1				
	0.2				
	0.4				
	0.6				

   Please make sure you have already finished LVS and DRC before you starting simulation every time after you modify your schematic and layout.

3. Please draw the trend charts for both rise delay and rise transition of both tables above.