

Problem 5: IP Timing Characterization

Source: Global UniChip Corp.
December 29, 2000

1. Introduction

Timing characterization is very important for pre-designed hard intellectual property (IP) macros. IP providers must provide accurate timing and synthesis models for IP integrators such that they can synthesize the soft portion of the chip with the constraints from the hard IPs. This problem is to build a static timing analysis (STA) tool that is dedicated for extracting the timings of pre-designed hard IP macros.

The timing model of an IP block in general includes the timing and loading characteristics of macro's inputs and outputs. This problem concerns only the timing characteristics that are listed as follows.

- (1) The setup/hold timing check between input ports and the relative clock, e.g., the setup/hold time between I1 and CLK (the root of clk1) shown in Figure 1.
- (2) The maximum/minimum output delay from the relative clock to output ports, e.g., the maximum/minimum output delay from CLK (the root of clk2) to O1 shown in Figure 1.
- (3) The maximum/minimum path delays from input ports to output ports, e.g., the maximum/minimum path delay from I2 to O2 shown in Figure 1.
- (4). The recovery timing check between preset (clear) and the relative clock, e.g., the recovery time between Preset and CLK (the root of clk1 and clk2) shown in Figure 1.

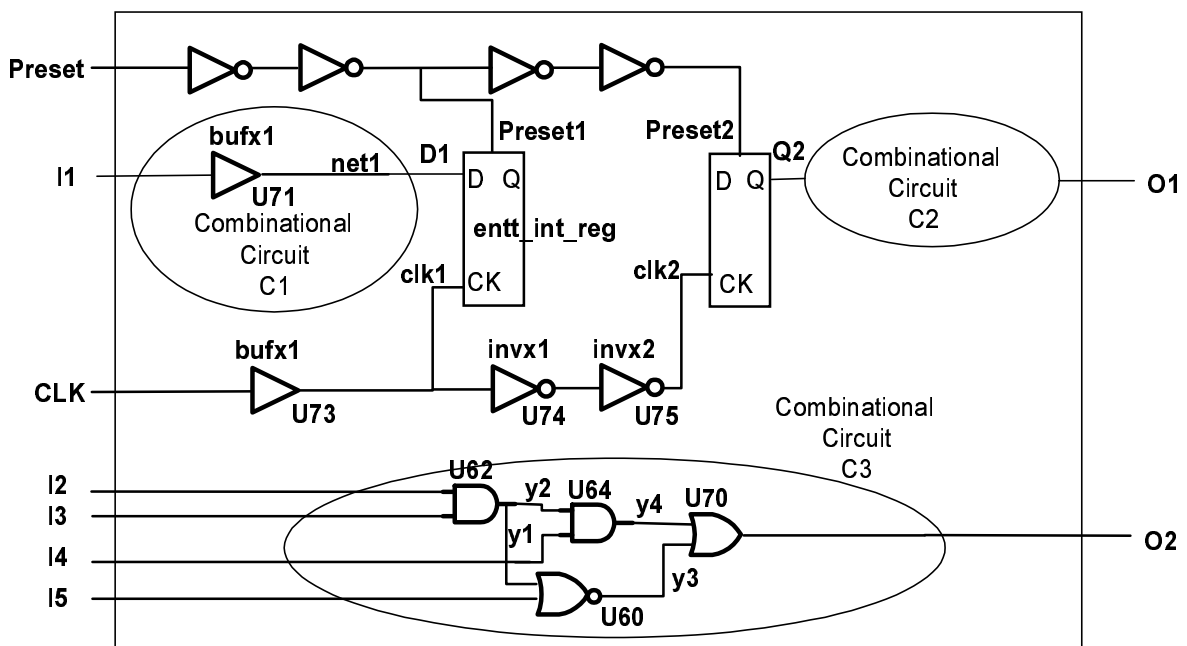
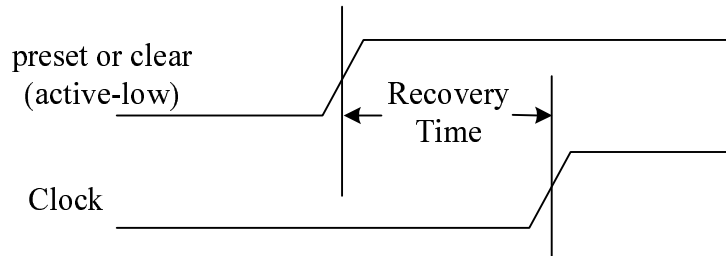


Figure 1: Input and Output Ports of an IP Macro for Timing Characterization

Recovery time is the minimum time that the active-low (active-high) preset or clear signal must remain high (low) before the active edge of the clock to ensure the cell working correctly.



2. Problem Description

Given a Synopsys *.lib model*, a gate-level *Verilog netlist*, a *set-load* file, a *timing constraint* file, and a *configuration* file, the developed software can find the above four timing characteristics for all macro's inputs and outputs. The final results of this problem are to be compared with those from Synopsys *PrimeTime*. The input and output formats are similar to those for *PrimeTime*.

In general, to have a complete and accurate *non-linear* timing model, a hard macro is to be characterized for best, typical, and worst corners individually. This problem only concerns the timing characterization of *typical corner*. Your tool doesn't need to characterize each of the three corners individually and treats them as the same one.

3. Input

(1). Synopsys Model (.lib)

A non-linear Synopsys model of a simplified cell library will be given. The cell library includes a set of basic cells such as inverter (invx1, invx2), buffer (bufx1), 2-input nand gate (nand2x1), 2-input nor gate (nor2x1), and D flip-flop with set and reset (dff_sr_x1). Example 1 shows a sample of the given model. Please refer to [2] for the details of .lib format.

Example 1: A sample of Synopsys model.

```
library(test) {
    delay_model: table_model;
    time_unit : "1ns";
    capacitive_load_unit (1,pf);

    lu_table_template(setup_template_3x2) {
        variable_1 : constrained_pin_transition;
```

```

variable_2 : related_pin_transition;
index_1 ("1000, 1001, 1002");
index_2 ("1000, 1001");
}

lu_table_template(delay_template_3x3) {
variable_1 : input_net_transition;
variable_2 : total_output_net_capacitance;
index_1 ("1000, 1001, 1002");
index_2 ("1000, 1001, 1002");
}

cell (bufx1) {
  pin(A) {
    direction : input;
    capacitance : 0.01;
  }
  pin(Y) {
    direction : output;
    capacitance : 0.0;
    function : "A";
    timing() {
      related_pin : "A";
      cell_rise(delay_template_3x3) {
        index_1 ("0.5, 0.9, 2.0");
        index_2 ("0.006, 0.1, 0.5");
        values ( \
          "0.1060, 0.8043, 2.4750", \
          "0.1609, 0.8585, 2.5280", \
          "0.0829, 0.8050, 2.4720");
        }
      rise_transition(delay_template_3x3) {
        index_1 ("0.5, 0.9, 2.0");
        index_2 ("0.006, 0.1, 0.5");
        values ( \
          "0.2, 0.5, 2.0", \
          "0.22, 0.55, 2.2", \
          "0.25, 0.6, 2.5");
      }
    }
  }
}

```

```

    }
    cell_fall(delay_template_3x3) {
    ...
    }
    fall_transition(delay_template_3x3) {
    ...
    }
}
}
}
cell (dff_sr_x1) {
    pin(D) {
        direction : input;
        capacitance : 0.02;
        timing() {
            related_pin : "CK";
            timing_type : setup_rising;
            rise_constraint(setup_template_3x2) {
                index_1 ("0.5, 0.9, 2.0");
                index_2 ("0.5, 2.0");
                values ( \
                    "0.1094, 0.2187", \
                    "0.1484, 0.2344", \
                    "0.0859, 0.1875");
                }
            fall_constraint(setup_template_3x2) {
                index_1 ("0.5, 0.9, 2.0");
                index_2 ("0.5, 2.0");
                values ( \
                    "0.2500, 0.3359", \
                    "0.3984, 0.4766", \
                    "0.7734, 0.8359");
                }
            }
        }
    ...
}
cell (invx1) {
...

```

```

    }
    ...
}

```

The .lib file is used to define the *timing relationship* for each cell or flip-flop (FF). In the above example, you can find two templates and two (template) instances. The first template is to define the relationship for setup time between Din and Clock, while the second template is used to give the relationship for cell delay based on input transition time and output capacitance of the cell. In the template instance of bufx1, for example, the cell_rise table contains two indices: index1 and index2 where index1 (index2) represents input_net_transition times (total_output_net_capacitances). Based on both indices, the cell delay can be obtained by checking the look-up table, e.g., with input_net_transition = 0.9ns and total_output_net_capacitance = 0.5pf, the cell delay is 2.5280ns. Similarly, table for cell_fall can be defined. Note that if you cannot find the table entry for a specific (index1, index2) pair, then you must derive the cell delay using interpolation. Further, the table for rise_transition is used to calculate the output transition time for the cell. Again, note that the rise_transition time is used as the input_net_transition time for the next cell followed. For example, the rise_transition time for U73 (bufx1) in Fig. 1 is used as the input_net_transition times for U74 (invx1) and pin CK of entt_int_reg (D-FF). To calculate the rising delay from CLK to the output of U74 (invx1), you add the cell_rise value of U73 and the cell_fall value of U74.

The template instance for cell dff_sr_x1 shown in Example 1 gives the setup time for a D_FF. In Fig. 1, D is defined as a constrained pin while CK is defined as a related pin. The table for setup time can be read similarly to that for cell delay. In Fig. 1, if D1 has rise_transition time = 0.9ns, and clk1 has transition time = 2.0ns (assume positive_edge triggering), then the setup time is 0.2344ns by checking the table of rise_constraint in Example 1.

In combinational circuit C3 (Fig. 1), there are 4 inputs I2, I3, I4 and I5, and the circuit contains four cells. To characterize the rise (or fall) propagation times for each of I2-I5, all possible paths must be tried. For example, to characterize the timing arc between I2 and O2, denoted as I2_O2, both paths P1 = I2-y2-y4-O2 and P2 = I2-y1-y3-O2 must be evaluated. To characterize CELL_RISE of timing arc I2_O2 where CELL_RISE means O2 has a rising transition, P1 does not contain any inversion cells, so cell_rise and rise_transition tables are used. For P2, to active O2 with a rising transition, I2 has to have a falling transition. So, cell_fall and fall_transition must be used for U62 (while others using cell_rise and rise_transition). Note that side inputs of each path are assumed non-controlling value. For example, to characterize P1, y3 is assumed logic 0. After characterizing all paths for each timing arc, the longest path and the shortest path will be used as shown in Example 6 (model file).

(2). Gate-level Verilog Netlist

Example 2: An example of gate-level Verilog netlist

```
module test_top ( CLK, I1, O1, I2, O2, Preset );
    input  CLK, I1, I2, Preset;
    output O1, O2;
    wire mdt_4_, mdt_0_, addrt_0_7_,.....
    nor2x1 ctl_a_0_U58 ( .A(mdt_0_), .B(mdt_2_), .Y(ctl_a_0_n976) );
    bufx1 U71 ( .A(I1), .Y(net1) );
    bufx1 U73 ( .A(CLK), .Y(clk1) );
    nand2x1 ctl_a_0_U78 ( .A(ctl_a_0_n965), .B(ctl_a_0_n966), .Y(
        ctl_a_0_NextState_1_ ) );
    ...
    ...
    dff_sr_x1 entt_int_reg ( .D(net1), .CK(clk1), .SN(1'b1), .RN(Preset), .Q(entt) );
    dff_sr_x1 q2_int_reg ( .D(ctl_a_0_entt_int160), .CK(clk2), .SN(1'b1),
        .RN(Preset), .Q(Q2) );
endmodule
```

(3). Set Load File

The given set load file is a text file that sets the *capacitive loading* on all of the nets of the design. One can use the file for back-annotating the extracted post-layout capacitive information to the design. In this problem, you don't need to calculate the interconnection delays because we provided only set load files but no set resistance files. But, when you calculate the delays of a cell, you must take the capacitive loading of the driven net(s) into account.

Example 3: An example of set load file

```
CAP UNIT "1pf"
set_load 0.023620 [get_nets {ctl_a_0_n965}]
set_load 0.08 [get_nets {net1}]
...
...
set_load 0.46 [get_nets {clk1}]
set_load 0.014630 [get_nets {ctl_a_0_entt_int160}]
```

(4). Timing Constraint File

A timing constraint file that specifies the specification of clock pin will be given. By default, any timing constraint and delay characterizations are referred to the POSITIVE edge of specified clock.

Example 4: An example of timing constraint file

```
create_clock -name CLK -period 9.0 -waveform {0.0 4.5} { CLK }
```

(5). Configuration File

A configuration file, as shown below, that specifies time scale, the unit of capacitive load, and lookup table templates will be given. Its major purpose is to setup the characterization conditions for each circuit. In the templates of RISE_CONSTRAINT and FALL_CONSTRAINT (shown in Example 5), INDEX_1 and INDEX_2 specify two transition delays, 0.5ns and 2.0ns, for the constrained pin and related pin, respectively. As one characterizes a specific timing arc, e.g. from I1 to CLK shown in Figure 1, for finding RISE_CONSTRAINT and FALL_CONSTRAINT, the output model data file, that will be discussed below, must include these two templates with a 2x2 characterized timing values (i.e., INDEX_1=0.500, INDEX_2=0.500, ..., INDEX_1=2.000, INDEX_2= 2.000).

In the templates of CELL_RISE, RISE_TRANSITION, CELL_FALL, and FALL_TRANSITION, INDEX_1 specifies two transition delays, 0.5ns and 2.0ns, for the input pin and INDEX_2 specifies two capacitive loading, 0.1pf and 0.5pf, for the output pin. Similarly, when characterizing a specific timing arc, e.g., from CLK to O1 shown in Figure 1, for finding the above timing delays, the characterization tool must account for four different combinations of input transition delays and output loading and characterize then individually.

Example 5: An example of configuration file

```
TIMESCALE "1ns";
```

```
CAPACITIVE_LOAD_UNIT (1.0,"pf");
```

```
RISE_CONSTRAINT (constraint_data) {  
  VARIABLE_1 : CONSTRAINED_PIN_TRANSITION  
  VARIABLE_2 : RELATED_PIN_TRANSITION  
  INDEX_1 (" 0.500, 2.000 ");  
  INDEX_2 (" 0.500, 2.000");  
}
```

```
FALL_CONSTRAINT (constraint_data) {
```

```
...
```

```

}
CELL_RISE (delay_data) {
VARIABLE_1 : INPUT_NET_TRANSITION
VARIABLE_2 : OUTPUT_NET_CAPACITANCE
INDEX_1 (" 0.500, 2.000 ");
INDEX_2 (" 0.100, 0.500");
}
RISE_TRANSITION (delay_data) {
...
}
CELL_FALL (delay_data) {
...
}
FALL_TRANSITION (delay_data) {
...
}

```

4. Output

For each test case, your tool must output a *model* file and a *model data* file similar to STAMP format used in Synopsys PrimeTime. The purpose of a model file is to give the IP terminals (e.g., I1 and CLK) whose timing will be characterized, while the purpose of a model data file is to give the timing values derived for each pair of IP terminals. Example 6 and Example 8 show the examples of model file and model data file, respectively. For the first test case of this problem, a template of model file and a template of model data file will be given later for your reference. These templates are not part of inputs of your tool and just used for your reference. They tell you what desired timing data must be included in your final outputs.

In the template of model data file, the timing values of each lookup table are not specified, e.g., the timing values of RISE_CONSTRAINT of I1_CLK shown in Example 7. Your tool must do timing characterization to find the missing delay values (value11, value12, value21, and value22) and then generate a complete model data file as that shown in Example 8. Please pay attention to that most information in the model data file are copied from those in configuration file. For example, the shadow parts in Example 7 and Example 8 are identical to that in the example of configuration file.

Example 6: Model file

```

MODEL
OUTPUT    O1;

```

```

OUTPUT    O2;
INPUT     I1;
INPUT     I2;
INPUT     I3;
INPUT     I4;
INPUT     I5;
INPUT     CLK;
INPUT     Preset;

```

```

/* The setup time constraint of I1 with respect to positive edge CLK. The timing data is
referred to model data file shown in Example 7. */

```

```

I1_CLK :

```

```

    SETUP (POSEDGE) I1  CLK ;

```

```

/* The hold time constraint of I1 with respect to positive edge CLK. The postfix “_min”
of label I1_CLK_min implies that the timing data was characterized by the shortest path
between I1 and CLK. */

```

```

I1_CLK_min :

```

```

    HOLD (POSEDGE) I1  CLK ;

```

```

/* The recovery time of Preset with respect to positive edge CLK. */

```

```

Preset_CLK :

```

```

    RECOVERY (POSEDGE) Preset  CLK ;

```

```

/* The timing data of O1 with respect to positive edge CLK. */

```

```

CLK_O1 :

```

```

    DELAY (POSEDGE) CLK  O1 ;

```

```

CLK_O1_min :

```

```

    DELAY (POSEDGE, MIN) CLK  O1 ;

```

```

/* The timing data of timing arc I2_O2. */

```

```

I2_O2 :

```

```

    DELAY (NONUNATE) I2  O2 ;

```

```

I2_O2_min:

```

```

    DELAY (NONUNATE, MIN) I2  O2 ;

```

```

    .....
```

```

    .....
```

```

ENDMODEL

```

Example 7: Template of model data file

```

MODELDATA
DESIGN "test_top";
TIMESCALE "1ns";
CAPACITIVE_LOAD_UNIT (1.0,"pf");
TIMINGDATA
/* The characterized setup time of timing arc I1_CLK. */
ARCDATA
I1_CLK :
/* I1: rising, CLK: rising (by default) */

```

```

    RISE_CONSTRAINT (constraint_data) {
        VARIABLE_1 : CONSTRAINED_PIN_TRANSITION
        VARIABLE_2 : RELATED_PIN_TRANSITION
        INDEX_1 (" 0.500, 2.000 ");
        INDEX_2 (" 0.500, 2.000");
        VALUES( "value11, value12",
                "value21, value22");
    }

```

value12 is the characterized timing constraint of timing arc I1_CLK under the conditions that the input transition delays of I1 and CLK are 2ns and 0.5ns, respectively.

```

/* I1: falling, CLK: rising (by default) */
    FALL_CONSTRAINT (constraint_data) {
        ...
    }
ENDARCDATA

```

```

/* The characterized hold time of timing arc I1_CLK. */
ARCDATA
I1_CLK_min :
    RISE_CONSTRAINT (constraint_data) {
        ...
    }
    FALL_CONSTRAINT (constraint_data) {
        ...
    }
ENDARCDATA
...
...
ARCDATA
CLK_O1 :

```

```

/* O1: rising, CLK: rising (by default) */
  CELL_RISE (delay_data) {
    VARIABLE_1 : INPUT_NET_TRANSITION
    VARIABLE_2 : OUTPUT_NET_CAPACITANCE
    INDEX_1 (" 0.500, 2.000 ");
    INDEX_2 (" 0.100, 0.500");
    VALUES( "value11, value12",
             "value21, value22");
  }
/* O1: rising, CLK: rising (by default) */
  RISE_TRANSITION (delay_data) {
    ...
  }
  CELL_FALL (delay_data) {
    ...
  }
  FALL_TRANSITION (delay_data) {
    ...
  }
ENDARCDATA

ARCDATA
I2_O2 :
/* O2: rising, I2: may be rising or falling */
  CELL_RISE (delay_data) {
    VARIABLE_1 : INPUT_NET_TRANSITION
    VARIABLE_2 : OUTPUT_NET_CAPACITANCE
    INDEX_1 (" 0.500, 2.000 ");
    INDEX_2 (" 0.100, 0.500");
    VALUES( "value11, value12",
             "value21, value22");
  }
....
ENDARCDATA

ENDTIMINGDATA
ENDMODELDATA

```

Example 8: Completed model data file

MODELDATA

DESIGN "test_top";

TIMESCALE "1ns";

CAPACITIVE_LOAD_UNIT (1.0,"pf");

TIMINGDATA

ARCDATA

I1_CLK :

```
RISE_CONSTRAINT (constraint_data) {  
    VARIABLE_1 : CONSTRAINED_PIN_TRANSITION  
    VARIABLE_2 : RELATED_PIN_TRANSITION  
    INDEX_1 (" 0.500, 2.000 ");  
    INDEX_2 (" 0.500, 2.000");  
    VALUES( "-1.452, -2.143",  
            "-1.023, -1.763");  
}
```

-1.452 is the characterized timing constraint of timing arc I1_CLK under the conditions that both of the input transition delays of I1 and CLK are 0.5ns.

```
FALL_CONSTRAINT (constraint_data) {  
    ...  
}
```

ENDARCDATA

...

ARCDATA

CLK_O1 :

```
CELL_RISE (delay_data) {  
    VARIABLE_1 : INPUT_NET_TRANSITION  
    VARIABLE_2 : OUTPUT_NET_CAPACITANCE  
    INDEX_1 (" 0.500, 2.000 ");  
    INDEX_2 (" 0.100, 0.500");  
    VALUES( "0.065, 1.901",  
            "0.194, 2.480");  
}  
RISE_TRANSITION (delay_data) {  
    ...  
}  
...
```

ENDARCDATA
...
ENDTIMINGDATA
ENDMODELDATA

5. An Example for Demonstrating Timing Characterization Procedure

In the following, the characterization procedure of the setup time between I1 (the constrained pin) and CLK (the related pin) shown in Figure 1 will be demonstrated. First, we consider the case where input signal I1 is rising. In the template of RISE_CONSTRAINT shown in Example 5, two transition delays are specified for both of constrained pin (I1) and related pin (CLK). Therefore, we need to characterize the setup time between I1 and CLK with four different conditions.

- 1). I1 with 0.5ns transition delay, CLK with 0.5ns transition delay,
- 2). I1 with 0.5ns transition delay, CLK with 2ns transition delay,
- 3). I1 with 2ns transition delay, CLK with 0.5ns transition delay,
- 4). I1 with 2ns transition delay, CLK with 2ns transition delay.

In the first condition, both of the transition delays of I1 and CLK are 0.5ns. The cell delays and output transition delays of U71 and U73, referred to Figure 1 and Example 2, are calculated as follows.

(a). **Output loading of U71:** 20ff (the input pin load of **dff_sr_x1's D** specified in Synopsys model) + 80ff (the capacitive load of net1 specified in set load file)

Output loading of U73: 30ff (the input pin load of **dff_sr_x1's ck** specified in Synopsys model) + 10ff (the input pin load of **invx1's A** specified in Synopsys model) + 460ff (the capacitive load of **clk1** specified in set load file)

(b). **Cell rising delay of U71 (D_U71):** 0.8043ns (Because the output signal of U71 is rising, we looked up the cell_rise table of bufx1 shown in Example 1 with the conditions of input transition delay 0.5ns and output loading 100ff.)

Cell rising delay of U73 (D_U73): 2.475ns (Because the output signal of U73 is rising, we looked up the cell_rise table of bufx1 shown in Example 1 with the conditions of input transition delay 0.5ns and output loading 500ff.)

(c). **Output transition delay of U71 (T_U71):** 0.5ns (We looked up the rise_transition table of bufx1 shown in Example 1 with the conditions of input transition delay 0.5ns and output loading 100ff.)

Output transition delay of U73 (T_U73): 2.0ns (We looked up the rise_transition table of bufx1 shown in Example 1 with the conditions of input transition delay 0.5ns and output loading 500ff.)

Based on the output transition delays of U71 and U73, we can find the setup time between D and the relative clock ck of dff_sr_x1. We looked up the rise_constraint table of dff_sr_x1

shown in Example 1 with the conditions that the transition delays of the constrained pin (net1) and the related pin (clk1) are 0.5ns and 2.0ns, respectively. We found that the setup time between D and ck (**SETUP_entt_int_reg**) is 0.2187ns.

After obtaining the path delays of the path from I1 to entt_int_reg/D and the path from CLK to entt_int_reg/D, the setup time between I1 and CLK must satisfy the following equation.

$$\text{Arrival_time}(\text{CLK}) + \text{Path_delay}(\text{the longest path from CLK to entt_int_reg/ck}) \geq \text{Arrival_time}(\text{I1}) + \text{Path_delay}(\text{the longest path from I1 to entt_int_reg/D}) + \text{SETUP_entt_int_reg}$$

By re-arranging the above equation, we have

$$\text{Setup time between I1 and CLK} \geq D_U71 + \text{SETUP_entt_int_reg} - D_U73$$

So, we got the setup time between I1 and CLK as -1.452ns ($0.8043 + 0.2187 - 2.475$).

The setup time calculated above is reported in the timing arc I1_CLK section shown in Example 8. Notice that Example 8 only shows the timing constraint and delay data, the setup time of timing arc between I1 and CLK must be declared in Example 6 (model file) as follows.

I1_CLK :

SETUP (POSEDGE) I1 CLK ;

With the same manner, we can characterize setup time between I1 and CLK for other conditions.

6. Language/Platform

- Language: C or C++
- Platform: SUN OS/Solaris

7. Evaluation

The final results are to be compared with those generated by PrimeTime. For each test case, we will specify a transition delay for all inputs and a capacitive loading for all outputs. Following, we will find the timing delays from your model data file and compare them with those from PrimeTime. The cost function for the evaluation of accuracy is done by root-mean-square error. In addition, memory usage, CPU time, user interface are also considered for evaluation.

8. Questions

Please report any question regarding this problem to cad@cis.nctu.edu.tw with the email subject "CAD Contest: Problem 5." Your question(s) will be answered in two weeks, and the Q&A's will be posted at the contest web site

References

- [1] PrimeTime Modeling User Guide
- [2] Library Compiler Reference Manual
- [3] Design Compiler Reference Manual
- [4] H.-C. Chen, and D. Du, "Path Sensitization in Critical Path Problem," IEEE Trans. On CAD, Vol. 12, No. 2, pp.196-207, Feb 1993.
- [5] H. Yalcin, M. Mortazavi, R. Palermo, C. Bamji, K. Sakallah, "Funtional Timing Analysis for IP Characterization," IEEE Proc. Design Automation Conference, 1999.