2003 IC/CAD Contest

Problem 5: Clock Tree Optimization for Useful Skew

Source: Global UniChip Corp.

February 11, 2003, revised on March 21, 2003 **& April 22, 2003**

1. Introduction

There are two approaches to resolve the timing violation of a critical path. One is to apply logic optimization technique to the path for reducing its path delay, and the other is to adjust clock arrival times of the starting flip-flop and ending flip-flop of that path such that the starting flip-flop can launch data earlier or the ending flip-flop can latch data later to compensate the timing slack due to timing violation. For those critical paths whose logic levels are too high and are difficult to be optimized, the later technique would be much effective.

The clock latency of a flip-flop's clock pin is the path delay starting from clock root through the distribution cells, and ending at the clock pin (leaf pin). In other words, the arrival time of a flip-flop's clock pin is its clock latency. The clock latency of a clock tree is the maximum clock latency from clock root to any leaf pin. The clock skew of two flip-flops is the difference of their clock latencies, and the clock skew of a clock tree is the maximum difference among the clock latencies of the leaf pins. As an example shown in Fig. 1, the arrival times (clock latencies) of F1.CK, F2.CK, and F3.CK all are 2ns, and thus the clock latency of clock tree CLK is 2ns, where for example F1.CK stands for CK pin of flip-flop F1. The clock skew of F1.CK and F2.CK is 0ns, and the clock skew of clock tree CLK is 0ns as well. For another example shown in Fig. 3, the clock latencies of F1.CK, F2.CK, and F3.CK are 1.9ns, 2.4ns, and 2ns respectively, and the clock latency of clock tree CLK is 2.4ns. The clock skew of F1.CK and F2.CK is 0.5ns (2.4ns – 1.9ns) that is also the clock skew of CLK.

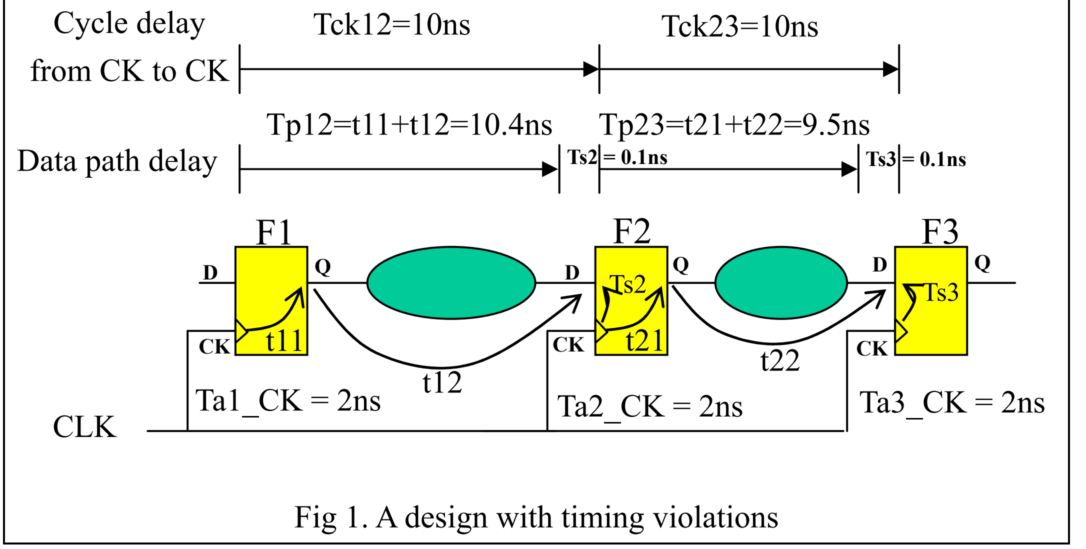


Fig 1. A design with timing violations

t11: cell delay from F1.CK to F1.Q

Tp12: path delay from F1.CK to F2.D (t11+t12)

Tck12: the cycle delay from F1.CK to F2.CK

Tr2_D: the required time of F2.D

t12: path delay from F1.Q to F2.D

Ta1_CK: arrival time of F1.CK

Ts2: setup time of F2.D w.r.t. F2.CK

Ta2_D: the arrival time of F2.D

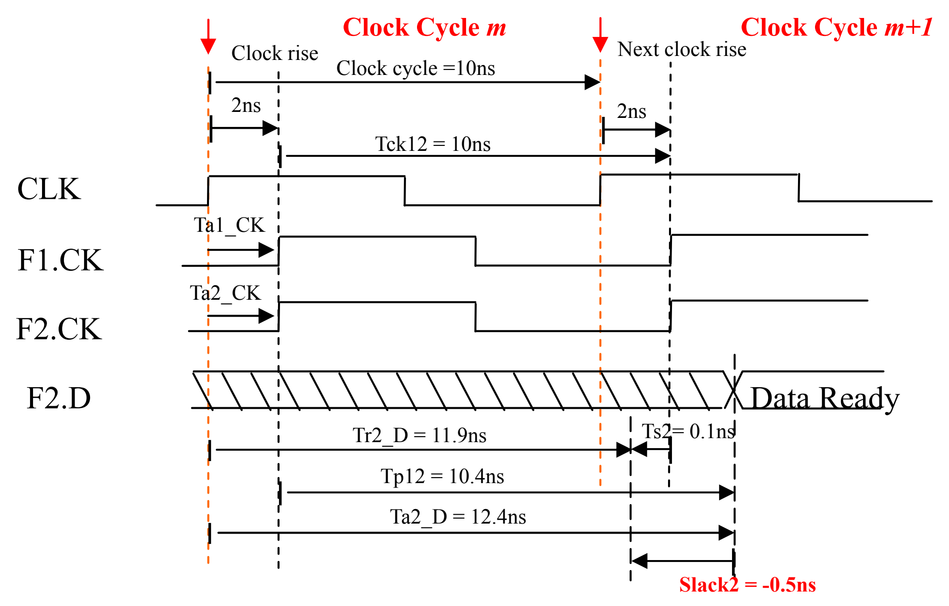


Fig 2. Timing diagram of Fig. 1

In Fig 1, the clock latency and clock skew of CLK are 2ns and 0ns, respectively. Assume that the clock cycle time is 10ns, both of Ts2 and Ts3 are 0.1ns, and the data path delays of Tp12 and Tp23 are 10.4ns and 9.5ns, respectively. By referring to the timing diagram shown in Fig. 2, the data launched at F1 (F2) at clock cycle *m* is to be captured at F2 (F3) at clock cycle *m+1*. Thus, the data required time of F2.D and F3.D are as follows.

Tr2_D = (Clock cycle time) + (F2.CK clock latency) – (F2.D setup time)
= 10ns + 2ns – 0.1ns = 11.9ns

Tr3_D = (Clock cycle time) + (F3.CK clock latency) – (F3.D setup time)
= 10ns + 2ns – 0.1ns = 11.9ns

And, the data arrival times are

Ta2_D = (F1.CK clock latency) + (F1.CK to F2.D path delay)
= 2ns + 10.4ns = 12.4ns

Ta3_D = (F2.CK clock latency) + (F2.CK to F3.D path delay)
= 2ns + 9.5ns = 11.5ns

Thus, we can find the timing slacks are

Slack2 = Tr2_D – Ta2_D = -0.5ns ←--------- Negative timing slack (setup time)

Slack3 = Tr3_D – Ta3_D = 0.4ns ←--------- Meet timing requirement.

There is 0.5ns negative timing slack in the critical path from F1.CK to F2.D.

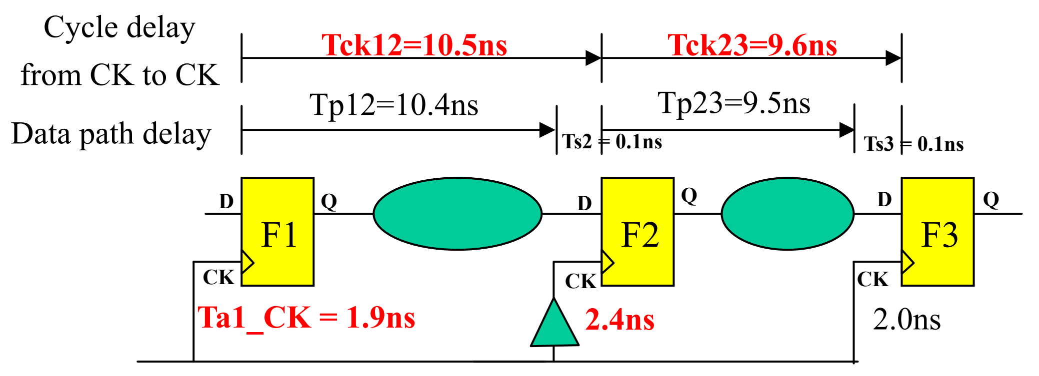


Fig 3. A design without timing violation

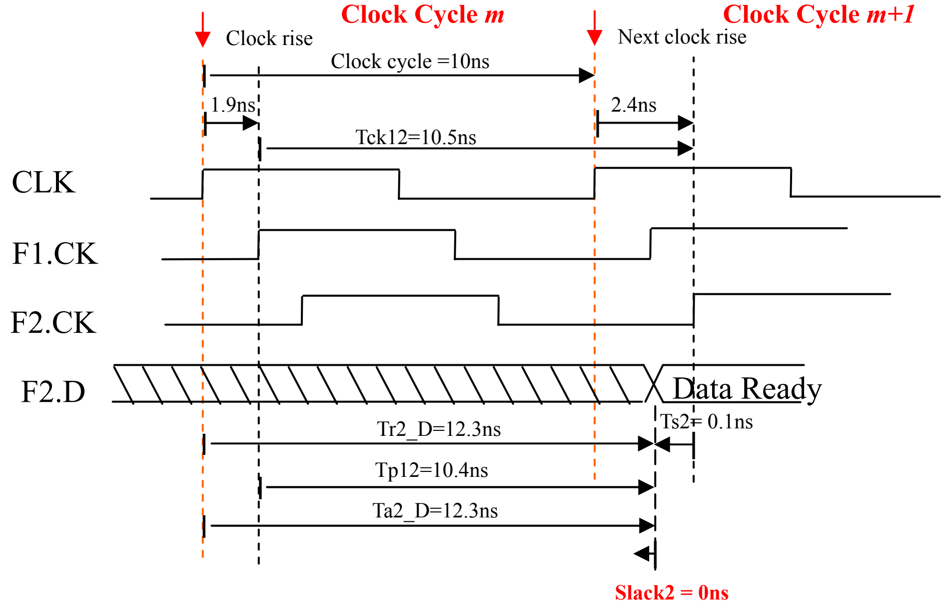


Fig 4. Timing diagram of Fig. 3

To resolve the timing violation above, one can adjust the clock latencies of F1 and F2, and let F1 launches data earlier and F2 captures data later to compensate the negative timing slack. Fig 3 demonstrates an example of fixing the timing violation, in which the data required times of F2.D and F3.D are

Tr2_D = (Clock cycle time) + (F2.CK clock latency) – (F2.D setup time)
= 10ns + 2.4ns – 0.1ns = 12.3ns

Tr3_D = (Clock cycle time) + (F3.CK clock latency) – (F3.D setup time)
= 10ns + 2ns – 0.1ns = 11.9ns

And, the data arrival times are

Ta2_D = (F1.CK clock latency) + (F1.CK to F2.D path delay)
= 1.9ns + 10.4ns = 12.3ns

Ta3_D = (F2.CK clock latency) + (F2.CK to F3.D path delay)
= 2.4ns + 9.5ns = 11.9ns

Now there is no more setup time violations as follows.

Slack2 = Tr2_D – Ta2_D = 0.0ns ←--------- Meet timing requirement.

Slack3 = Tr3_D – Ta3_D = 0.0ns ←--------- Meet timing requirement.

2. Problem Description

Given (1) a design that has been done cell placement and clock tree synthesis, (2) a static timing report containing all critical paths (maximum delay paths) that are between any two flip-flops, from primary inputs to flip-flops, and from flip-flops to primary outputs, (3) Timing model of clock buffers, the developed software has to apply clock tree optimization techniques such as buffer insertion, gate resizing, and buffer removal, and consider the placement of clock buffers to reduce the number of setup time violation paths and the total negative slack.

Basically, one has to take not only setup time problems but also hold time problems into account when applying clock tree optimization to resolve timing violations. Resolving setup time violations may result in new hold time violations, and vice versa. To simply this problem, the contestants need to take care setup time violations only and ignore all hold time violations.

3. Input

The default time, capacitance, and distance (coordinate) units are in nano-second (*ns*), pico-fara (*pf*), and micron-meter (*um*), respectively.

(1). Design file (design.def)

The format of the design file is as follows.

```
DIEAREA      (lower-left coordinate) (upper-right coordinate)
PINS
Pin_name1    Direction      X1-coordinate Y1-coordinate
Pin_name2    Direction      X2-coordinate Y2-coordinate
…
END PINS
COMPONENTS
Instance_Name1          Cell_Name1   X3-coordinate Y3-coordinate
Instance_Name2          Cell_Name1   X4-coordinate Y4-coordinate
…
END COMPONENTS
NET
```

```
Net_Name1    Type    Instance_Name1.pin1    Instance_Name2.pin1 ....
Net_Name2    Type    Instance_Name4.pin2    Instance_Name5.pin2 ....
....
END NET
```

lower-left coordinate: the coordinate at the lower-left corner of the die.

upper-right coordinate: the coordinate at the upper-right corner of the die

Pin_name: the pin name of input, output, and in/out pins

Direction: the direction of the corresponding pin. It can be *IN, OUT*, or *INOUT.*

X-coordinate: the coordinate in X-axis.

Y-coordinate: the coordinate in Y-axis.

Instance_Name: the instance name of a placed cell.

Cell_Name: the cell name of a placed cell.

Net_Name: the name of an interconnect.

Type: the type of the specified net. It can be *CLOCK* or *SIGNAL*.

Instance_Name.pin: the pin name of the cell connected to the net. For example, u1/u10/F1.CK represents that the instance name is u1/u10/F1 and CK pin is connected to the net. The slash "/" stands for hierarchy divider. The sequence of specified instances must be the driving cell first and followed by driven cells.

(2). Static timing report file (timing.inf)

Note that, in this problem, we only consider setup time constraints. The static timing report file contains the timing information of the critical paths (maximum delay paths) that are between any two flip-flops, from primary inputs to flip-flops, and from flip-flops to primary outputs. The format is defined as follows:

#start_point	*end_point*	*path_delay*	*setup*	*cap*	*s_clk*	*e_clk*	*slack*
data_in[0]	u0/rg_1	4.1	0.3	0.02	0.0	1.80	2.6
u1/u10/F1	**u1/u10/F2**	**10.4**	**0.1**	**0.03**	**2.0**	**2.0**	**-0.5**
u1/u10/F2	**u2/F3**	**9.5**	**0.1**	**0.05**	**2.0**	**2.0**	**0.4**
u1/rg_1	u1/u10/F2	9.60	0.1	0.03	1.85	2.0	0.45
u1/u10/F2	add_out[5]	2.7	0.0	0.0	2.0	0.0	0.6

The line begins with '#' is a comment.

start_point: a primary input port or a flip-flop instance name.

end_point: a flip-flop instance name or a primary output port.

path_delay: the maximum path delay from the *start_point* clock pin to the *end_point* data pin.

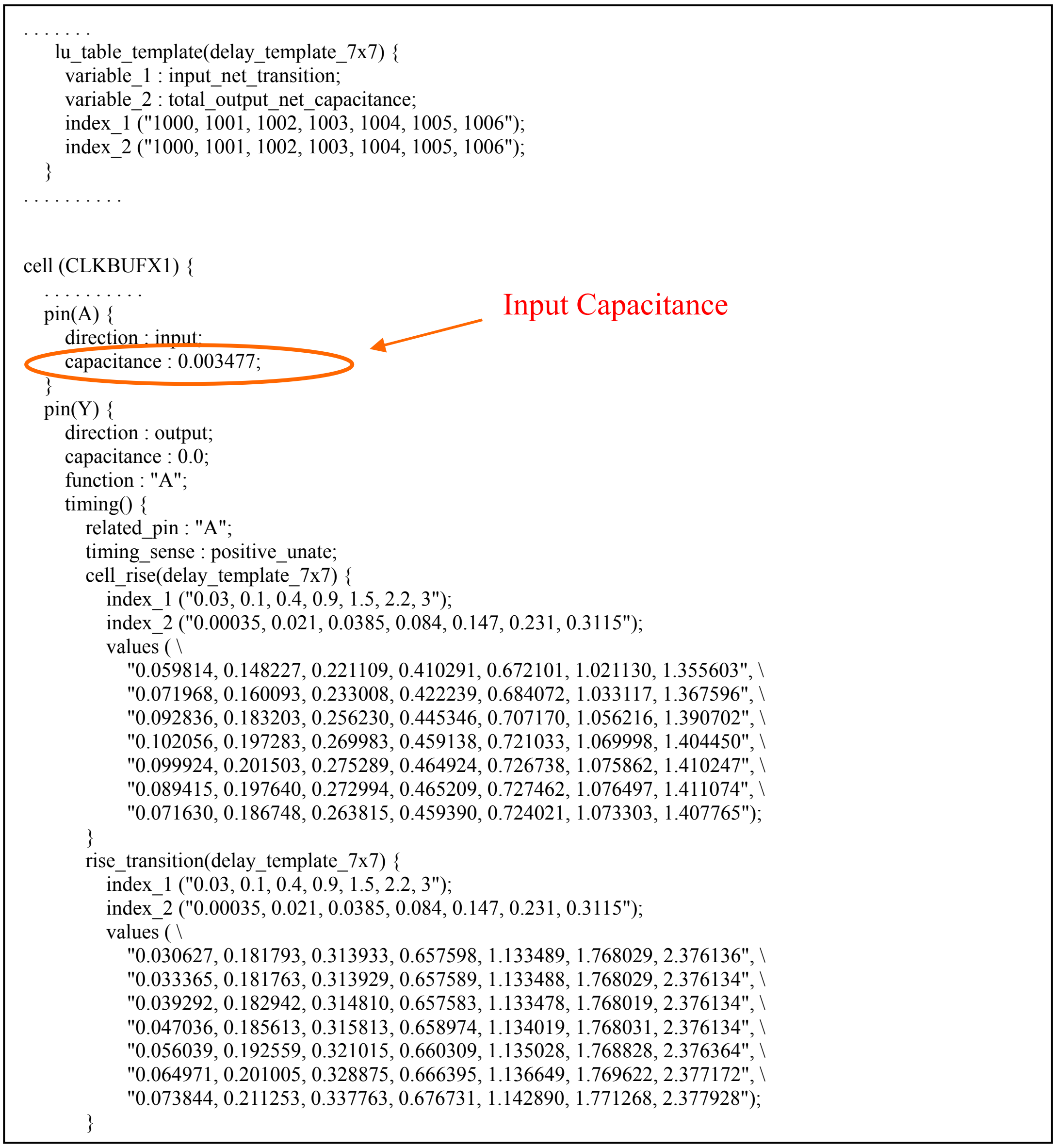
setup: the setup time requirement of the *end_point* flip-flop.

cap: the clock pin capacitance of the *end_point* if the *end_point* is a flip-flop.
s_clk: the clock latency of the *start_point* clock pin.
e_clk: the clock latency of the *end_point* clock pin.
The slack is the required time subtracts the arrival time of the *end_point*.

In the example above, the second (third) row demonstrates the timing information of the path between flip-flops F1 and F2 (F2 and F3) depicted in Fig. 1. The instance names of F1, F2, and F3 are u1/u10/F1, u1/u10/F2, and u2/F3, respectively. data_in[0] is a primary input port and add_out[5] is a preminary output port.

(3). Clock buffer's timing model (clkbuf.lib)

The given timing model is in Synopsys .lib format and contains timing information and input pin capacitance of clock buffers and inverters only. It is a two-dimension table-look-up model. The timing is related to the input transition time and the output load. Below is an example of the timing model:



```
. . . . . . .
     lu_table_template(delay_template_7x7) {
       variable_1 : input_net_transition;
       variable_2 : total_output_net_capacitance;
       index_1 ("1000, 1001, 1002, 1003, 1004, 1005, 1006");
       index_2 ("1000, 1001, 1002, 1003, 1004, 1005, 1006");
   }
. . . . . . . . . .


cell (CLKBUFX1) {
   . . . . . . . . . .
   pin(A) {
       direction : input;
       capacitance : 0.003477;
   }
   pin(Y) {
       direction : output;
       capacitance : 0.0;
       function : "A";
       timing() {
          related_pin : "A";
          timing_sense : positive_unate;
          cell_rise(delay_template_7x7) {
             index_1 ("0.03, 0.1, 0.4, 0.9, 1.5, 2.2, 3");
             index_2 ("0.00035, 0.021, 0.0385, 0.084, 0.147, 0.231, 0.3115");
             values ( \
                "0.059814, 0.148227, 0.221109, 0.410291, 0.672101, 1.021130, 1.355603", \
                "0.071968, 0.160093, 0.233008, 0.422239, 0.684072, 1.033117, 1.367596", \
                "0.092836, 0.183203, 0.256230, 0.445346, 0.707170, 1.056216, 1.390702", \
                "0.102056, 0.197283, 0.269983, 0.459138, 0.721033, 1.069998, 1.404450", \
                "0.099924, 0.201503, 0.275289, 0.464924, 0.726738, 1.075862, 1.410247", \
                "0.089415, 0.197640, 0.272994, 0.465209, 0.727462, 1.076497, 1.411074", \
                "0.071630, 0.186748, 0.263815, 0.459390, 0.724021, 1.073303, 1.407765");
          }
          rise_transition(delay_template_7x7) {
             index_1 ("0.03, 0.1, 0.4, 0.9, 1.5, 2.2, 3");
             index_2 ("0.00035, 0.021, 0.0385, 0.084, 0.147, 0.231, 0.3115");
             values ( \
                "0.030627, 0.181793, 0.313933, 0.657598, 1.133489, 1.768029, 2.376136", \
                "0.033365, 0.181763, 0.313929, 0.657589, 1.133488, 1.768029, 2.376134", \
                "0.039292, 0.182942, 0.314810, 0.657583, 1.133478, 1.768019, 2.376134", \
                "0.047036, 0.185613, 0.315813, 0.658974, 1.134019, 1.768031, 2.376134", \
                "0.056039, 0.192559, 0.321015, 0.660309, 1.135028, 1.768828, 2.376364", \
                "0.064971, 0.201005, 0.328875, 0.666395, 1.136649, 1.769622, 2.377172", \
                "0.073844, 0.211253, 0.337763, 0.676731, 1.142890, 1.771268, 2.377928");
          }
```

Input Capacitance

```
        cell_fall(delay_template_7x7) {
          index_1 ("0.03, 0.1, 0.4, 0.9, 1.5, 2.2, 3");
          index_2 ("0.00035, 0.021, 0.0385, 0.084, 0.147, 0.231, 0.3115");
          values ( \
            "0.062397, 0.153531, 0.226701, 0.416510, 0.679066, 1.029075, 1.364481", \
            "0.079646, 0.170871, 0.244063, 0.433911, 0.696486, 1.046504, 1.381914", \
            "0.128636, 0.223286, 0.296756, 0.486452, 0.749012, 1.099029, 1.434439", \
            "0.186025, 0.285599, 0.359099, 0.549339, 0.811847, 1.161779, 1.497154", \
            "0.241396, 0.347741, 0.422915, 0.613536, 0.876430, 1.226407, 1.561703", \
            "0.297722, 0.411006, 0.488564, 0.682375, 0.945420, 1.295674, 1.631027", \
            "0.356177, 0.476429, 0.556420, 0.754852, 1.020361, 1.370541, 1.706061");
        }
        fall_transition(delay_template_7x7) {
          index_1 ("0.03, 0.1, 0.4, 0.9, 1.5, 2.2, 3");
          index_2 ("0.00035, 0.021, 0.0385, 0.084, 0.147, 0.231, 0.3115");
          values ( \
            "0.028478, 0.168009, 0.290407, 0.609373, 1.051023, 1.639923, 2.204286", \
            "0.029608, 0.168002, 0.290438, 0.609376, 1.051029, 1.639929, 2.204290", \
            "0.036506, 0.170717, 0.291427, 0.609361, 1.051037, 1.639929, 2.204290", \
            "0.045257, 0.174697, 0.293546, 0.611218, 1.051264, 1.639929, 2.204290", \
            "0.054797, 0.184138, 0.300151, 0.612955, 1.053136, 1.640645, 2.204290", \
            "0.063613, 0.195118, 0.311175, 0.619687, 1.054980, 1.642055, 2.205452", \
            "0.073599, 0.207052, 0.323270, 0.633024, 1.061348, 1.643938, 2.206660");
        }
      }
. . . . . . . .
  }
. . . . . . . . . . .
}
```

The lu_table_template defines the template of the look-up-table. This is a 2-dimension table, where the first index denotes the input net transition time and the second index denotes the total output net capacitance. The output net capacitance includes net load and the input pin load of driven cells. Assume that a CLKBUFX2 buffer drives three CLKBUFX1 and its net load is 10pf. Then, the total output net capacitance is 10.010431pf (10pf + 3 x 0.003477pf). Both indexes have 7 elements.

The cell_rise and cell_fall groups define cell's rising delay and falling delay of the output pin that related to an input pin. In the example, when input net transition is 0.1ns and the output net capacitance is 0.0385pf, the cell_rise delay is 0.233008ns.

The rise_transistion and fall_transistion groups define cell's output pin transition time. In the example, when input net transition is 0.1ns and the output net capacitance is 0.0385pf, the rise_transition time is 0.313929ns. The output pin transition is used as the input net transition for the delay calculation of next stage.

If the input net transition and/or the output net capacitance are not exactly exist in the indexes, the software should use interpolation to calculate the timing.

For the details of Synopsys timing model format, please refer to Synopsys Library Compiler Manual.

(4). Timing constraint file (timing.con)

Timing constraint file specifies the clock cycle time, and the input delay and output delay of primary IOs. The format is defined as follows:

```
Clock_cycle      clock_name       clock_cycle_time
Input_delay      input_pin_name   input_delay_time
Output_delay     output_pin_name  output_delay_time
```

Clock_cycle, *Input_delay*, and *Output_delay* are reserved keywords. *Input_delay* and *Output_delay* are the external delays of an input pin and an output pin respectively, whose definitions are identical to that defined in Synopsys PrimeTime.

As the example shown in Fig. 1 and the example of timing.inf shown in page 5, the timing constraint file is as follows.

```
Clock_cycle      CLK          10
Input_delay      data_in[0]   4.8
Output_delay     add_out[5]   4.7
```

4. Output

(1). Design file with optimized clock tree (design_opt.def)

The format of the output design file should be identical to that of the input design file. The output design file should contain entire design including optimized clock tree. All placed standard cells except clock buffers cannot be changed or moved. It is not allowed to place clock buffers outside the die area. In other words, the *lower-left coordinate* and the *upper-right coordinate* should be the same as those of the input design file. The new placed clock buffers are allowed to overlap the existing placed standard cells.

(2). Clock tree timing report (clock.rpt)

The format of the clock tree timing report is as below:

```
#start_point   end_point    s_clk1  e_clk1  slack1   s_clk2   e_clk2   slack2
data_in[0]     u0/rg_1      0.0     1.80    2.6      0.0      1.80     2.6
u1/u10/F1      u1/u10/F2    2.0     2.0     -0.5     1.9      2.4      0.0
u1/u10/F2      u2/F3        2.0     2.0     0.4      2.4      2.0      0.0
u1/rg_1        u1/u10/F2    1.85    2.0     0.45     1.85     2.4      0.85
u1/u10/F2      add_out[5]   2.0     0.0     0.6      2.4      0.0      0.2
```

start_point: a primary input port or a flip-flop instance name.
end_point: a flip-flop instance name or a primary output port.

s_clk1: the original clock latency of the *start_point* clock pin.
e_clk1: the original clock latency of the *end_point* clock pin.
slack1: the original timing slack.
s_clk2: the optimized clock latency of the start_point clock pin.
e_clk2: the optimized clock latency of the end_point clock pin.
slack2: the optimized timing slack.

In the example above, the contents of the first two columns are identical to those given in input timing report file (timing.inf), and the following three columns are identical to the last three columns in timing.inf as well. The second (third) row demonstrates the timing information of the path between flip-flops F1 and F2 (F2 and F3) depicted in Fig. 3.

(3). Net load file (net_load.rpt)

The net load file contains capacitance of all clock nets. The capacitance unit is in *pf*.

```
# Net_Name   capacitance
CLK_L0_N1   0.002518
CLK_L0_N2   0.001450
CLK_L1_N1   0.002386
```

The program doesn't need to perform real routing but it must use the given formula below to estimate the net load for calculating the delay of clock buffers. The formula to estimate the net load C of a driver pin is

$$C = \sum_{all\_fanout} net\_length \times \phi ,$$

$$\phi = 0.00015$$

where the net_length is the net length from driving cell to the driven cell, its unit is in *um* and its formula is

$$net\_length = \left|X_{cell1} - X_{cell2}\right| + \left|Y_{cell1} - Y_{cell2}\right|$$

where the (X,Y) is the coordinates of a cell in X-axis and Y-axis, respectively.
ϕ is a capacitance factor and its value is 0.00015pf/um.

5. Delay Calculation

In this problem, we assume the resistance of interconnect is 0 ohm and thus the there is no interconnect delay for all nets. The effects of net load, pin capacitance of driven clock buffers, and pin capacitance of flip-flop's clock pin should be taken into account when calculating the cell delay of driving clock buffer. The input capacitances of clock buffers and flip-flops are specified in clock buffer's timing model and timing report file, respectively.

Assume that the input transition times of all primary inputs are 0ns and all primary output capacitance are 0pf.

6. Language/Platform

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

7. Evaluation

- The number of setup time violation paths
- **The maximum negative slack of setup time violations.**
- The total negative slack of setup time violations
- The worst clock latency
- Run time
- Memory usage
- The correctness of net load calculation

The maximum negative slack after optimization should be less than or equal to that before optimization. The total negative slack is the summation of all negative setup time slacks. The clock latency should be as less as possible. The final results will be compared with the report from Synopsys PrimeTime, in which the output design (design_opt.def) as well as clock net load (net_load.rpt) will be back-annotated to PrimaryTime to analyze the timing of clock buffers, the clock latencies of all flip-flop's clock pins, and the setup time of all critical paths. Again, all hold time violations can be ignored.

8. Reference

[1] J.P. Fishburn, "Clock Skew Optimization", IEEE Transaction on Computers, 39(7): 945—951, 1990.

[2] N. Masheshwari and S.S. Sapatnekar, "Timing Analysis and Optimization of Sequential Circuits", second edition, Kluwer Academic Publishers, 2001.