TIMING VERIFICATION WITH THE NON-PERIODIC GATED CLOCKING

Hanbin Kim, Eui-Young Chung, Kyu-Myung Choi, Jeong-Taek Kong and Sang-Hoon Lee
CAE Team, R & D Center
Semiconductor Business, Samsung Electronics Co., Ltd., Korea
kycst@semigw.semi.samsung.co.kr

ABSTRACT

Digital circuits can be implemented based on highly complex non-periodic clocking schemes. However, the conventional timing verifiers do not guarantee the correctness of timing analysis because they cannot consider full timing behaviors of a gated clock. This paper describes a novel hybrid timing verification approach which handles circuits using non-periodic gated clocking schemes. For its non-periodic nature of the gated clock, timing constraints must be generated with considering full behaviors of the gated clock. Experimental results show that the proposed technique performs more complete and more reliable timing verification than conventional timing verifiers.

1. INTRODUCTION

Continuously shrinking geometry and prevalent use of automated tools such as logic synthesis and automatic layout make the timing verification more essential in VLSI design. For a decade, timing verifiers have been developed for detecting timing problems in switch-level [1][2] and gate-level circuits [3]. Because these algorithms use static timing analysis, all signal paths can be verified for timing problems regardless of input vectors. Therefore, we can expect complete timing verification without repetitive simulations. Although static timing analysis approach may produce pessimistic results due to false paths, several algorithms [4][5] have already been developed to eliminate such paths.

Static timing verifiers do not require input stimuli. Therefore, all signal paths can be examined completely for timing problems. This assumes that all clocked storage elements (CSEs), such as flipflops and latches, have a fully specified clock. When a primary clock goes through gated logic or CSEs for controlled clocking, a user must redefine clock waveforms to such internal clock sources. Though a static timing verifier needs no input patterns, it requires clock specification at each internal clock source of a circuit. Therefore, applying static timing verification to the circuit using highly complex clocking schemes requires hard work in clock specification which can be incorrect in delay estimation and incomplete in expressing the full behaviors. Thus, automation of capturing full behaviors and timing of internal clock sources are crucial for the complete timing verification.

Figure 1 shows typical forms of simple gated clocks. Use of gated clocks enables to control clocking and to reduce power consumption. Although it has many merits, timing verifiers and test-related CAD tools suffer from analyzing gated clocks. In the aspect of delay accuracy, because a static timing verifier can only handle a single path delay, it

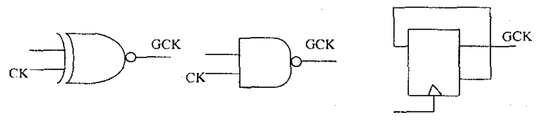


Figure 1: Simple form of gated clock

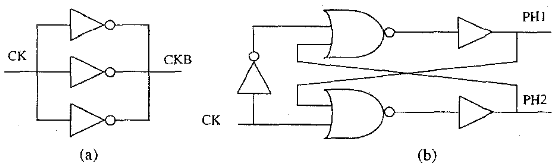


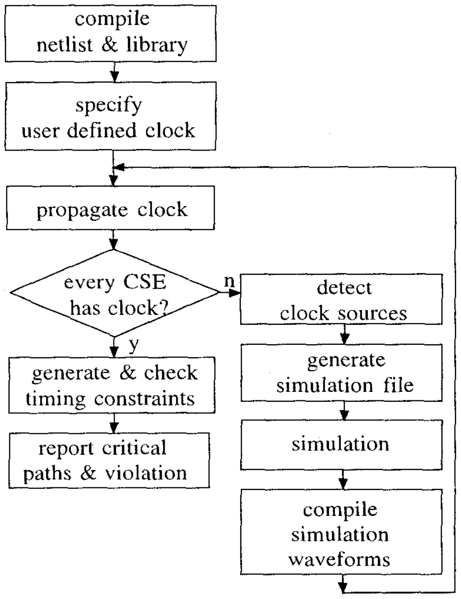
Figure 2: Multiple clock path

can be inaccurate for the circuits in Figure 2. The circuit in Figure 2(a) is used for increasing its driving capability and Figure 2(b) is used for generating non-overlapped two phase clocks. Those circuits have multiple paths such as reconvergent paths and feedback loops. They make it inaccurate to estimate the delay in clock path which plays a important role in timing verification.

To overcome the previously mentioned limitations, a hybrid approach[6] and STG (Signal Transition Graph) based approaches[7][8] have been developed. The hybrid approach combines simulation technique and static timing verification technique. This approach uses simulation results for setting proper values on the specified node (e.g., a clock source node). However, it is necessary to specify the path to be checked. Therefore, this approach is suitable only to be used locally.

The STG-based approach uses STG to determine sequential behaviors. Therefore, it cannot be used to gate-level designs due to an impractical run time or memory limitations.

Here a novel hybrid approach is proposed and implemented in the static timing verifier, called AMULET, which acquires simulation results for the clock source nodes needed. AMULET performs static timing verification automatically for the whole signal paths. Section 2 gives an overview of the hybrid approach in AMULET and Section 3 describes gated clock specification in previous timing verification environments and their incompleteness. Section 4 describes how to generate timing constraints from user specified clocks and simulation results. Finally, Section 5 shows experimental results of delay estimation for multiple and single clock paths and timing verification of complex



compile netlist & library
specify user defined clock
propagate clock
every CSE has clock?
n
y
detect clock sources
generate & check timing constraints
generate simulation file
report critical paths & violation
simulation
compile simulation waveforms

Figure 3: System configuration

gated-clocking circuits.

2. OVERVIEW OF THE HYBRID APPROACH IN AMULET

Clock waveforms are usually specified in a primary input. These waveforms are propagated to each CSE through a buffer in AMULET. However, for a multi-input element (i.e., an internal clock source), the output behavior of the element cannot be determined statically. Therefore, there must be CSEs that have no clock waveform in their clock pins. These elements are identified and traced back until the unspecified clock source is reached. Then, the waveforms at output nodes of these clock sources are determined using independent simulation and are used in deriving timing constraints for every CSE pair. Because the output waveform of an internal clock source can be non-periodic, every active edge of the clock signal is verified, whereas conventional timing verifiers [9][10] check only one condition for a CSE pair. Figure 3 shows an overall timing verification flow to handle complex gated clockings.

3. GATED CLOCK SPECIFICATION

This section discusses how to handle gated clocks in previous timing verification environments and their incompleteness. First, users should specify gated clock waveforms by inspection. Users have to understand fully the behavior of gated clocking and transform this non-periodic behavior into a periodic one. Thus, information may be lost during transformation and users are liable to err in estimating clock behaviors.

Figure 4 shows the common error and the incompleteness in such transformation. The first and second waveform show the correct form of non-periodic gated clocking for the preceding CSE and the succeeding CSE. In this case, setup time constraint relation is S_1 as in the figure. Therefore, the path delay should be less than (t_3 - t_1).

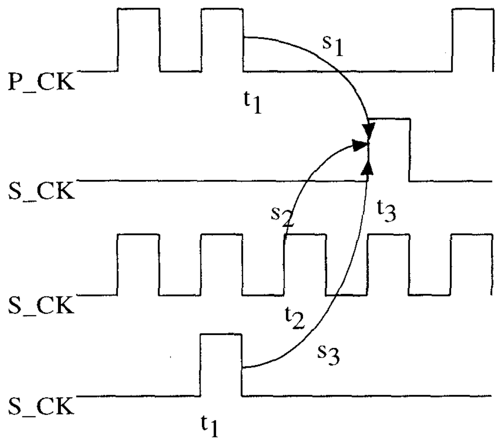


Figure 4: Gated clock waveform specification

When the first waveform is transformed into the third waveform for the clocking to be periodic, the setup time constraint becomes S_2. This is much tighter than the original one, which yields pessimistic results.

Alternatively, the first waveform can be simplified to the fourth waveform and the constraint remains the same as the original one. However, this simplification requires inspection of the original waveform throughout the whole simulation period.

Lastly, users can define the whole waveforms of a clock into one period. Also, it requires much work to model the non-periodic gated clock correctly.

4. TIMING CONSTRAINT GENERATION

Timing Constraints between a CSE pair, as shown in Figure 5(a), with the preceding and the succeeding clock waveforms as in Figure 5(b), can be generated as follows.

For the setup time constraint:

$$\begin{aligned}\Delta_{D,max} \quad < \quad & [t_{NAB}(n_{setup}, t_0, S\_CSE) \\ & -T_{setup}(S\_CSE)] - t_0 - (\Delta_S - \Delta_P) \quad (1)\end{aligned}$$

For the hold time constraint:

$$\begin{aligned}\Delta_{D,min} \quad > \quad & [t_{NAB}(n_{setup}, t_0, S\_CSE) \\ & -T_{hold}(S\_CSE)] \\ & -t_{NAB}(n_{hold}, t_0, P\_CSE) - (\Delta_S - \Delta_P)(2)\end{aligned}$$

where Δ_D: path delay,
Δ_P:preceding clock delay,
Δ_S: succeeding clock delay,
n_{setup}: number of cycle for setup,
n_{hold}: number of cycle for hold,
$T_{setup}(CSE)$: setup time for CSE,
$T_{hold}(CSE)$: hold time for CSE,
t_0: reference clock edge
t_{NAB}(n, t, CSE) : n^{th} next active edge of the CSE after t.

In Equation 1 and 2, not only setup/hold time violation is checked but also multicycle operating conditions are considered. Figure 5(c) shows timing constraint relations

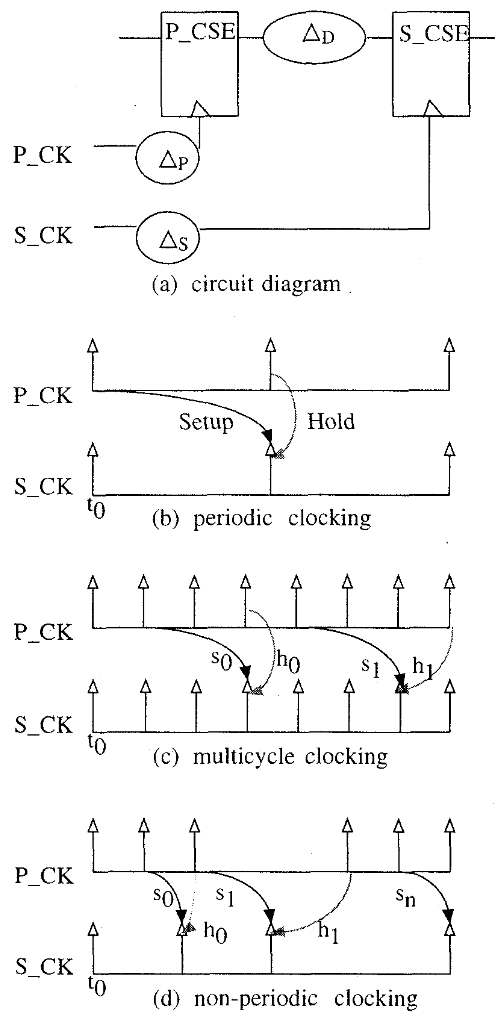


Figure 5: Timing constraints

for two multicycle conditions: $n_{setup} = 2/n_{hold} = 1$ and $n_{setup} = 1/n_{hold} = 2$. There are two types of clock waveforms in the proposed hybrid timing analysis. One is a user-defined clock waveform and the other is the non-periodic simulation-determined one.

We define the user-defined type and the simulation-determined type as U-type and S-type, respectively. There are following four cases for a CSE pair:

- User-defined, User-defined (U-U) type
- User-defined, Simulation-determined (U-S) type
- Simulation-determined, User-defined (S-U) type
- Simulation-determined, Simulation-determined (S-S) type

When a clock waveform pair is U-U type, the timing constraint can be derived using Equation 1 and 2. When S-S type is used in the pair, we need to check every active edge pair between a preceding and succeeding clock waveforms as in Figure 5(d). The worst case among all the extracted constraints is used for timing verification of the related path.

The proposed timing constraint generation algorithm for the S-S type clock waveform is described in Figure 6. The

```
Constraints_Generation()
begin
1     S := MAX;
2     H := MIN;
2     for (every active clock signal t at P_CSE) do
3         $S_n := \Delta_{D,max} - [t_{NAB}(n_{setup}, t, S\_CSE)$
                 $-T_{setup}(S\_CSE)] - t - (\Delta_S - \Delta_P)$;
4         if ( $S_n$ < S ) then S := $S_n$;
5         $H_n := \Delta_{D,min} - [t_{NAB}(n_{setup}, t, S\_CSE)$
                 $-T_{hold}(S\_CSE)] - t_{NAB}(n_{hold}, t, P\_CSE)$
                 $-(\Delta_S - \Delta_P)$;
6         if ( $H_n$ > H ) then H := $H_n$;
7     return ( S, H );
end
```

Figure 6: Constraint generation algorithm

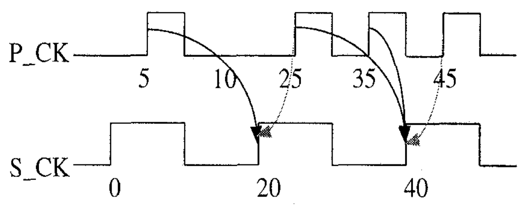


Figure 7: Example clock waveform

same algorithm is applied to the S-U and the U-S types except that the periodic waveform of a user-defined clock is expanded until the whole clock waveforms determined using simulation can be compared.

Figure 7 shows example waveforms of clocks applied to preceding and succeeding CSEs. To generate constraints, the first active edge of the P_CK signal at t = 0 is selected. The next active edge of the S_CK signal can be found at t = 20. These two edges form the first setup time constraint, 20 - 5 = 15, when we assume $T_{setup}(S\_CSE) = \Delta_S = \Delta_P = 0$. To get the hold time constraint, the next active edge of P_CK signal at t = 25 is searched from at t = 20. We can also get 20 - 25 = -5 as the hold time constraint. As we proceed this constraint generation process until the end of the active edge of the clock waveform, we can get setup/hold times as 15/-5, 5/-5, consecutively. Out of three constraints, 5/-5 is selected as the final constraint and is used for timing verification.

A careful handling of clock path delays is needed for correct timing verification. If there is a clock skew, a preceding and a succeeding clock path delays are different as in Figure 8, we should get the next active edge of clock separately from the clock delay. When there is no clock skew as in Figure 8(a), the setup time constraint is 25 - 5 = 20. When $\Delta_S - \Delta_P$ is 1 as in Figure 8(b), the setup time constraint is 24 - 5 = 19. But when $\Delta_S - \Delta_P$ is -1 as in Figure 8(b) and if we search the next active edge from the delay added clock waveform, the setup time constraint is 6 - 5 = 1 which is not correct. By getting the next active edge from the waveforms of clock sources and adding the clock delay later, we can get correct timing constraints.

5. EXPERIMENTAL RESULTS

To show the effect of multiple paths in delay estimation, the two circuits in Figure 2 were simulated using Hspice[11]. The circuit in Figure 2(a) is a template having three fanins

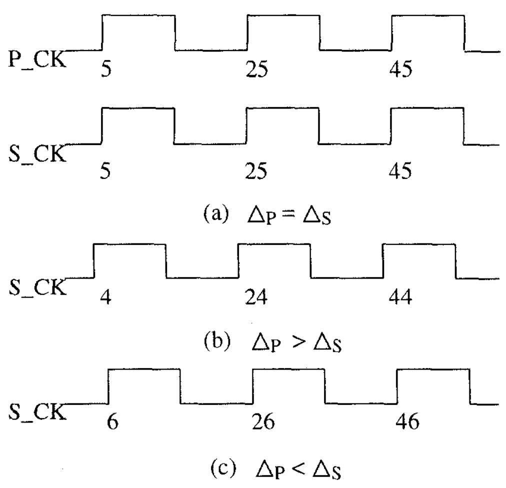

P_CK
5
25
45
S_CK
5
25
45
(a) $\Delta_P = \Delta_S$
S_CK
4
24
44
(b) $\Delta_P > \Delta_S$
S_CK
6
26
46
(c) $\Delta_P < \Delta_S$

Figure 8: Clock waveform with skew

Table 1: Simulation results for multiple fanin

# of fanin	# of stages		
	2	4	8
3	39.4	44.8	48.5
5	56.0	61.9	64.8
9	64.8	71.5	78.5

and one stage. This was expanded to have fanins and stages as in Table 1. This result was compared with the circuit having a single path and the same number of stages. The percentage error ranges from 39.4% to 78.5% and the average is 58.9%.

The circuit in Figure 2(b) was evaluated under various loading capacitance values between 0.102pf and 1.36pf. The single clock path is from CK to PH1 and consists of a inverter, a NOR, and a buffer. In this case, the percentage error ranges from 58.5% to 89.7% and the average is 77.8%. By analyzing the two experiments, we can expect more accurate results in timing verification using the hybrid approach in AMULET.

Finally, AMULET was applied to a logic circuit which was designed for teletext processing. The circuit consists of about 83,000 transistors and is driven by 120 gated clock sources. For the simulation of gated clock source nodes, Verilog[12] was used and the run time was 125 minutes on Sun Sparc20. The runtime of timing verification with simulation results was 26 minutes.

Table 3 shows that different constraints can be derived from a single preceding and succeeding CSE pair due to the non-periodic change of the gated clocks signal. For the first case, 19170 / 32 events of active preceding / succeeding clock occur during the entire simulation period and form 63 setup and hold time constraints. When both preceding and succeeding clocks are fully periodic, those constraints must be the same. Otherwise, different constraints are generated, shown in the last column of Table 3.

From the extracted constraints, the worst timing requirement is compared to path delays and timing constraints for the path are verified. By exploring all the constraints generated by non-periodic clocking, AMULET has performed more complete and more reliable timing verification than conventional timing verifiers.

Table 2: Simulation results for a two phase clocks

output loading	0.102pf	0.608pf	1.36pf
error rate	58.5	85.3	89.7

Table 3: Timing constraints for non-periodic clocking

case	# of active preceding/ succeeding clock	# of const-raints	# of different const-raints
1	19170 / 32	63	4
2	18 /32	35	5
3	105/105	207	8

6. CONCLUSIONS

This paper has proposed a novel hybrid timing verification approach. This approach is useful in timing verification especially for highly complex gated clocking circuits. Using simulation values in understanding timing behaviors of such gated clocking nodes and generating timing constraints for static timing analysis, this approach achieves complete and reliable timing verification for the target circuit.

REFERENCES

[1] J. K. Ousterhout, "A switch-level timing verifier for digital MOS VLSI". *IEEE Transaction on Computer-Aided Design*, vol. CAD-4, pp. 336-349, July 1985.

[2] N. P. Jouppi, "Timing analysis and performance improvement of MOS VLSI designs". *IEEE Transaction on Computer-Aided Design*, vol. CAD-6, pp. 650-664, July 1987.

[3] R. B. Hitchcock, "Timing verification and the timing analysis program". *Proceedings of the* 19th *ACM/IEEE Design Automation Conference*, pp. 446-456, 1982.

[4] H. Chen and D. Du, "Path sensitization in critical path problem". *Digest of Technical Parers of the IEEE International Conference on Computer-Aided Design*, pp. 208-221, 1991.

[5] H. Chang and J. A. Abraham, "Viper: An efficient vigorously sensitizable path extractor". *Proceedings of the* 30th *ACM/IEEE Design Automation Conference*, pp. 112-117, 1989.

[6] Cadence Design Systems, *Veritime Reference Manual*, 1989.

[7] A. P. Gupta and D. P. Siewiorek, "Automated multi-cycle symbolic timing verification of microprocessor-based Designs". *Proceedings of the* 31th *ACM/IEEE Design Automation Conference*, pp. 113-119, 1995.

[8] M. Kawarabayashi, N. Shenoy, and A. Sangiovanni-Vincentelli, "A verification technique for gated clock". *Proceedings of the* 30th *ACM/IEEE Design Automation Conference*, pp. 123-127, 1989.

[9] Y. H. Kim, *Accurate timing verification for digital VLSI designs.*, Memorandum No. UCB/ERL M89/2, University of California, Berkeley, Jan. 1989.

[10] Synopsys Inc., *Designtime Reference Manual*, 1995.

[11] Meta-Software, *HSPICE User's Manual H95*, 1995.

[12] Cadence Design Systems, *Verilog Reference Manual*, 1989.