LC-012

Dependable Clock Design for Level Sensitive Clock Signal

Yukiya Miura †

1. Introduction

Since recent VLSIs operate under the high speed and low voltage, noises generated by parasitic elements and signal changes come to influence circuit behaviors. Particularly, it is well known that a crosstalk noise induced by a parasitic capacitance between signal lines gives a large influence to a digital circuit [1]-[4]. Moreover, it is possible to cause incorrect behavior (i.e., malfunction) in many parts in the synchronous digital circuit when the crosstalk noise is generated on the clock signal line.

The crosstalk noise causes three fault types on the clock signal; (a) crosstalk-induced pulse, (b) crosstalk-induced speedup transition (advanced rising/falling transition), (c) crosstalk-induced slowdown transition (delayed rising/ falling transition) [2]. Besides, there are two clock types, the pulse clock signal (the edge trigger clock) and the level sensitive clock. The change of transition time results in a clock width change (i.e., duty cycle change), and as a result, it may cause the malfunction for the synchronous digital circuit. Therefore, we consider the fault causing the clock width change (i.e., duty cycle change) described the above (b) and (c) in this paper.

As measures against crosstalk faults in the digital circuit, a circuit layout/routing design to prevent the generation of the crosstalk noise is proposed [5]-[8]. However, it is very difficult that its incidence is zero because of design constraints. Methods for detecting malfunction caused by the crosstalk noise as a fault are proposed [9]-[12]; however, it is difficult to feedback to a design phase because we may not find its cause. Moreover, in order to detect the generation of all the crosstalk noises, it needs to spend a large effort. Particularly, in order to evaluate the influence of crosstalk faults caused in sequential circuits, the circuits must be tested under normal operation.

Metra et. al. proposed the on-line testing method to detect clock faults [13], [14]. Although the method can detect the fault of the clock signal, the change of the clock signal width, during system operation, it cannot guarantee the correct behavior of the system.

We consider that the introduction of a fault tolerant or dependable technique is effective measures against the crosstalk noise as follows. It is difficult to eliminate perfectly causes of crosstalk noises. Moreover, crosstalk noises are not always detected as faults, because their influence may cause behavior of intermittent faults and crosstalk noises happen accidentally by unintended causes. In this paper, we thus use aggressive approach for measures against crosstalk noises.

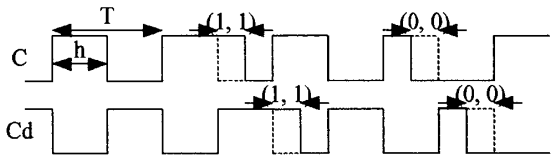
In this paper, we propose a method of self-correction for a clock signal, which can correct the change of the clock signal width during system operation. The proposed method does not require any reference signal for signal correction. We aim to realize the method as adapter circuits because it is applicable to the conventional sequential circuit and existing design properties.

This paper is organized as follows. Section 2 describes related works and our idea for the self-correction of the clock signal. In Section 3, we show an implementation example of the proposed method and simulation results. Section 4 concludes the paper.

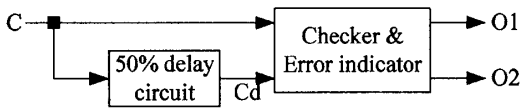
2. Self-correction for signal width change

2.1 Related works

Metra et. al. proposed the on-line testing to detect the change of the clock signal width [13], [14], which is the basis of this study. In the following, we consider the clock signal with a 50% duty cycle for simple discussion (h/T=50% in Fig. 1(a)).



(a) Clock signals



(b) Diagram

Fig. 1 Summary of on-line testing [13].

Their method is based on the one-out-of-two code (the two-rail code) for the signal of a 50% duty cycle. For this purpose, the original clock signal C and the properly delayed signal Cd that is delayed by 50% of the period of C are used (Fig. 1(a)). If the original clock is distributed normally, codewords (C, Cd)={(1, 0), (0, 1)} are generated. If a signal width is changed by a fault, non-codewords (C, Cd)={(1, 1), (0, 0)} are generated. These codewords are checked by a checker and then an error indicator produces error flags of (O1,O2)={(0, 0), (1, 1)} (Fig. 1(b)).

As introduced here, this method can detect variations of a signal width by copying and delaying the original clock signal, and easily extends to other duty cycles. We extend this idea to a self-correction method for the change of the clock width.

As an alternative approach, methods for compensating the clock skew are proposed [15]-[17]. These methods

† Faculty of System Design, Tokyo Metropolitan University

compensate the time lag between distributed clock signals, and their approaches are for variations of the arrival times of clock signals between signal lines. In this sense, their objectives are different with that of our work. Besides, in order to realize the methods, they generally require a reference clock.

2.2 Principle of self-correction

We assume the following conditions in this paper.

- Circuit type: Synchronous sequential circuits composed of level sensitive clocked elements and flip-flops (FFs)
- Clock signal: Level sensitive clock type
- Waveform of the clock signal: Duty cycle is $d=h/T$ (see Fig. 1(a)), where we assume the signal width h to be $kh=T$, k is a positive integer. This assumption is the same as Refs. [13], [14]. Note that we consider the signal of a 50% duty cycle (d=50%, k=2) for simple discussion.
- Fault type: Change of a clock width (change of a h width)

Metra's method detects the fault that causes the change in the duty cycle of the clock signal by using the original signal C and the copied delay-signal Cd [13], [14]. Then, we can consider four conditions for the signal width change as shown in Figs. 1(a) and 2.

- If there is no fault, (C, Cd)=(1, 0), (0, 1).
- If there is a fault, (C, Cd)=(1, 1), (0, 0).

From these conditions, we can distribute the proper clock signal if we can correct the width change of the original clock signal C by referring the properly delayed signal Cd as follows.

(a) If (C, Cd)=(0, 0) at the odd number of times, modify the signal C to be logic 1 (e.g., parts marked by €).
(b) If (C, Cd)=(1, 1) at the odd number of times, modify the signal C to be logic 0 (e.g., parts marked by $).
(c) For the other cases, the clock signal C is directly distributed (e.g., (C, Cd)=(1, 0), (0, 1) and (C, Cd)=(1, 1), (0, 0) at the even number of times marked by ¥).

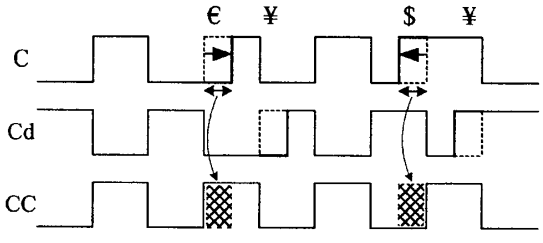


Fig. 2 Slowdown rising and speedup rising.

As shown here, comparing the original clock and the properly delayed clock, and counting the appearing times of non-codewords, we can correct the change of a clock width. Figure 3 shows the block diagram of the proposed self-correction method for the clock with a 50% duty cycle. The original clock C is compared with the delayed clock Cd, and the corrected clock CC is produced (hatched parts of Fig. 2). If there is no faults, the original clock is directly output. Note that the proposed method is applicable for clock signals with other duty cycles by modifying the dotted part of Fig. 3 whose original idea is proposed by Metra et. al. [13].

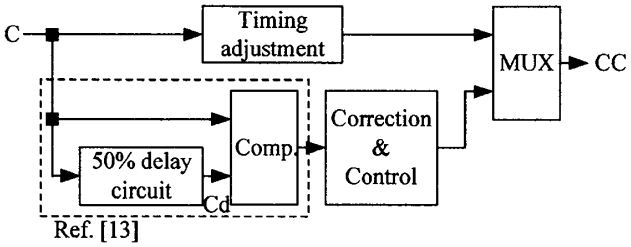


Fig. 3 Block diagram of the self-correction method.

3. Implementation and simulation results

3.1 Implementation

In order to realize the self-correction method, we design the circuit based on the following policies.

(1) The circuit structure is as simple as possible.
(2) The proposed method is applicable to existing design resources.

For this purpose, we design our method as an adapter circuit to satisfy both policies, because we can easy insert the proposed circuit into existing circuits if we realize it by the adapter circuit.

The proposed self-correction method requires the comparison circuit between C and Cd together with the delay circuit, the counter for counting the number of non-codewords, and the multiplexer for switching the output of the original clock and the corrected clock. Figure 4 shows the example for implementing the proposed method. Primitive gates are used for comparators, and the one-bit counters are used for counters. The multiplexer to switching the output signal is realized by the combination of primitive gates because it does not need to adjust timing. The upper part of Fig. 4 is a correction circuit to be CC=0 and the lower is for CC=1. In the following, we describe the circuit function.

Figure 5 explains the function of the proposed method for the non-codeword of (C, Cd)=(0, 0) (i.e., case (a) in Section 2.2). This function is realized by the lower part of Fig. 4. For the appearance of the non-codeword, the OR gate (s4) produces logic 0, and the QB output of the falling edge T-FF (s3) produces logic 0 when (C, Cd)=(0, 0) appears at the odd number of times. If both outputs are logic 0 (i.e., the interval of t1 shown in Fig.5), C1 becomes logic 1, and as a result, the original clock signal is corrected to be logic 1. Thus, for the non-codewords, the proposed circuit distributes the corrected signal from its output terminal CC, and for the other cases, the original

clock signal C is directly transmitted from the CC terminal. Here, CC signal becomes logic 1 regardless of other signal values if C1=1. Thus, MUX of Fig. 4 is designed to realize this function.

On the other hand, the circuit distributes the corrected signal of logic 0 for the non-codeword (C, Cd)=(1, 1) (i.e., case (b) in Section 2.2) by using the AND gate and the rising edge T-FF (the upper part of Fig. 4). For the appearance of (C, Cd)=(1, 1) at the odd number of times, C0 becomes logic 0, and as a result, the original clock signal is corrected to be logic 0. Note that Δ in Fig. 4 denotes a delay circuit to adjust signal timing in the circuit.

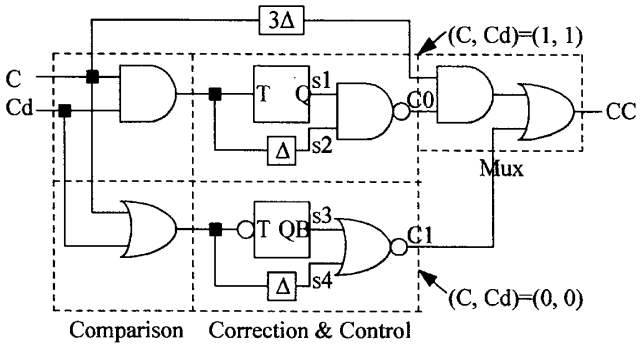


Fig. 4 Example circuit of the self-correction method.

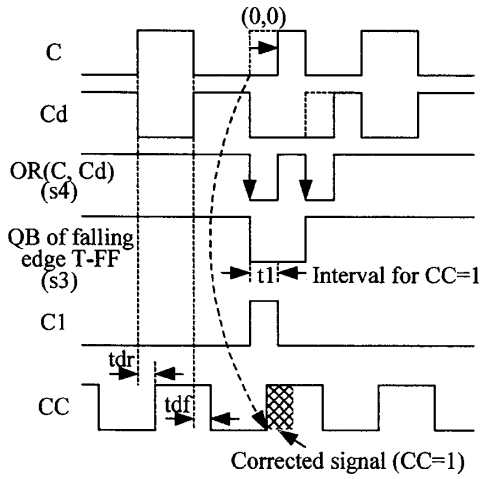


Fig. 5 Function of signal correction for (C, Cd)=(0, 0).

3.2 Simulation results and observations

We verify the circuit function by using circuit simulation and TSMC 0.18-μm parameters of 1.8-V VDD [18]. We also consider process variations in circuit simulation: ±10% process variations for *VDD* (power supply), *W/L* (channel width/length), *Vtn* (nMOS threshold), *Vtp* (pMOS threshold).

Figure 6 shows the example of simulation results, where the following conditions were used.

- Clock period: 10.0ns
- High level width: 5.0ns
- Clock width variation: 4.0ns (-1.0ns) and 6.0ns (+1.0ns)

The upper graph shows the original clock signal C (the solid line) and the properly delayed clock signal Cd (the dotted line). The second graph shows the corrected signal CC. The third graph shows correction signals for the non-codeword of (C, Cd)=(0, 0) (the solid line for s3 and the dotted line for s4 in Fig. 4). The lower graph shows correction signals for the non-codeword of (C, Cd)=(1, 1) (the solid line for s1 and the dotted line for s2). From these results, we find that the original clock signal is corrected in 15-16ns (the interval marked by €) and 34-35ns (the interval marked by $) and is output to the CC signal. Then, the regular clock signal CC is distributed. Therefore, we can say that the proposed method can correct the change of the clock width.

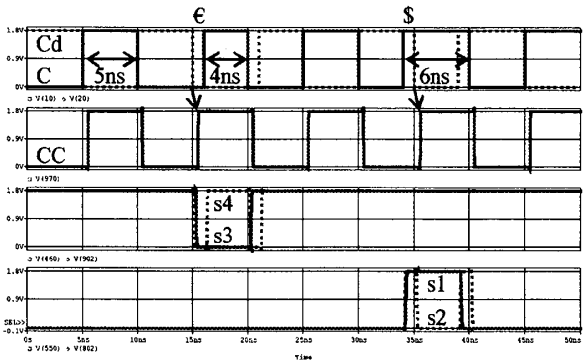


Fig. 6 Simulation results.

Table 1 summarizes the performance of signal correction of the proposed method. For variations of the clock signal width from ±4.0% to ±20.0% for typical conditions, the proposed method corrects them within the range from 0.44% to 1.65% by absolute values. Even if we give all combinations of process variations, the proposed circuit can correct the signal width and accuracy of the corrected signal is the range from -2.41% to +0.58%. Thus, we could achieve our initial objective.

Table 1 Summary of signal correction.

	C (orignal)		CC (corrected)					
			Typical		Minus variation		Plus variation	
"H" width (Fault-free)	5.000		4.968		4.965	(-0.06)	4.970	(+0.04)
"H" width (Faulty)	4.000	(-20.0)	4.990	(+0.44)	4.974	(+0.19)	4.999	(+0.58)
	6.000	(+20.0)	4.886	(-1.65)	4.845	(-2.41)	4.911	(-1.19)
	4.500	(-10.0)	4.991	(+0.46)	4.975	(+0.21)	4.999	(+0.58)
	5.500	(+10.0)	4.887	(-1.63)	4.846	(-2.39)	4.911	(-1.19)
	4.750	(-5.0)	4.991	(+0.46)	4.981	(+0.33)	4.998	(+0.56)
	5.250	(+5.0)	4.898	(-1.41)	4.873	(-1.84)	4.914	(-1.13)
	4.800	(-4.0)	4.991	(+0.46)	4.974	(+0.18)	4.998	(+0.56)
	5.200	(+4.0)	4.906	(-1.25)	4.874	(-1.83)	4.919	(-1.03)
	[ns]	[%]	[ns]	[%]	[ns]	[%]	[ns]	[%]

Table 2 Delay time of corrected signal.

	Typical	Minus variation	Plus variation
tdr	0.477	0.650	0.373
tdf	0.445	0.615	0.343
|tdr-tdf|	0.032	0.035	0.030

[ns]

The proposed method causes the delay times *tdr* and *tdf* for producing the correct clock signal CC as shown in Fig. 5 because the circuit of Fig. 4 is inserted into the clock signal line. Table 2 summarizes delay times of the implemented circuit. There is the difference between *tdr* and *tdf*, and as a result, the width of the corrected signal CC is difference with that of the original clock even if there is no fault. If all of clocked elements use the proposed method, there is no problem for the delay time caused by the correction circuit. On the other hand, if both the original clock signal and the corrected clock signal are used in one circuit, we must insert a timing adjustment element into the clock line for the original clock distribution.

The implemented circuit needs a processing time to correct the clock signal. From Table 1, it takes from 0.022ns (corresponding to 0.44% change) to 0.082ns (corresponding to 1.65% change) for signal correction. This processing time may influence behaviors of clocked elements because the output timing of the signal from the CC terminal varies depending on whether or not the original clock signal is corrected.

4. Conclusions

In this paper, we proposed the self-correction method that can distribute the level sensitive clock signal with the proper signal width without any external reference signal. The method can be realized as adapter circuits, and as a result, it is easy to build in the existing synchronous circuit. From simulation results, we find that the implemented circuit had the ability of the tolerance for the process variations. The proposed method is applicable to not only LSIs but also every synchronous circuit including board circuits. In addition, the proposed method is applicable to data signals periodically transmitted. Besides, it is useful for circuits demanding high reliability. Therefore, we believe that the proposed method is valuable for dependable design.

In order to show practicality, we further examine circuit performance including area overhead and other variations of manufacturing, other simple and useful implementation methods, testability of implemented circuits, and design limitation of the proposed method.

Acknowledgement

This research was supported in part by Japan Society for the Promotion of Science under Grant-in-Aid for Scientific Research (C) (No. 18500040).

References

[1] J.M. Rabaey, Digital Integrated Circuits: A Design Perspective, Prentice-Hall, NJ, 1996.

[2] W. Chen, S.K. Gupta, and M.A. Breuer, "Analytic Models for Crosstalk Delay and Pulse Analysis Under Non-Ideal Inputs," Proc. Int. Test Conf., pp.809-818, 1997.

[3] X. Aragonès, J.L. González, F. Moll, and A. Rubio, "Noise Generation and Coupling Mechanisms in Deep-Submicron ICs," IEEE Design & Test Computers, vol.19, no.5, pp.27-35, September-October 2002.

[4] C. Metra, S.D. Francescantonio, and T.M. Mak, "Implications of Clock Distribution Faults and Issues with Screening Them during Manufacturing Testing," IEEE Trans. Comput., vol.53, no.5, pp.531-546, May 2004.

[5] A. Vittal and M. Marek-Sadowska, "Crosstalk Reduction for VLSI," IEEE Trans. Computer-Aided Design, vol.16, no.3, pp.290-298, March 1997.

[6] M.A. Elgamel and M.A. Bayoumi, "Interconnect Noise Analysis and Optimization in Deep Submicron Technology," IEEE Circuits and Systems Magazine, vol.3, no.4, pp.6-17, 2003.

[7] T. Yamada, A. Sakai, Y. Matsushita, and H. Yasuura, "Routing Methodology for Minimizing Crosstalk in SoC," IEICE Trans. Fundamentals, vol.E86-A, no.9, pp.2347-2356, September 2003.

[8] J. Lou and W. Chen, "Crosstalk-Aware Placement," IEEE Design & Test Computers, vol.21, no.1, pp.24-32, March-April 2004.

[9] W.-Y. Chen, S.K. Gupta, and M.A. Breuer, "Test Generation for Crosstalk-Induced Faults: Framework and Computational Results," Proc. Asian Test Symp., pp.305-310, 2000.

[10] L.-C. Chen, T.M. Mak, M.A. Breuer, and S.K. Gupta, "Crosstalk Test Generation on Pseudo Industrial Circuits: A Case Study," Proc. Int. Test Conf., pp.548-557, 2001.

[11] B.C. Paul and K. Roy, "Testing Crosstalk Induced Delay Faults in Static CMOS Circuits Through Dynamic Timing Analysis," Proc. Int. Test Conf., pp.384-390, 2002.

[12] M.S. Wu and C.L. Lee, "Using a Periodic Square Wave Test Signal to Detect Crosstalk Faults," IEEE Design & Test Computers, vol.22, no.2, pp.160-169, March-April 2005.

[13] C. Metra, M. Favalli, and B. Riccò, "On-Line Testing Scheme for Clock's Faults," Proc. Int. Test Conf., pp.587-596, 1997.

[14] C. Metra, M. Favalli, S.D. Francescantonio, and B. Riccò, "On-Chip Clock Faults' Detector," J. Electronic Testing: Theory and Application, vol.18, no.4, pp.555-564, August 2002.

[15] C. Metra, F. Ciovanelli, M. Soma, and B. Riccò, "Self-Checking Scheme for Very Fast Clocks' Skew Correction," Proc. Int. Test Conf., pp.652-661, 1999.

[16] C.-Y. Yang and S.-I. Lin, "A One-Wire Approach for Skew-Compensating Clock Distribution Based on Bidirectional Techniques," IEEE J. Solid-State Circuits, vol.36, no.2, pp.266-272, February 2001.

[17] M. Omaña, D. Rossi, and C. Metra, "Low Cost Scheme for On-Line Clock Skew Compensation," Proc. VLSI Test Symp., pp.90-95, 2005.

[18] The MOSIS Service, http://www.mosis.org/.