DFT for Testing High-Performance Pipelined Circuits with Slow-Speed Testers

Muhammad Nummer and Manoj Sachdev
University of Waterloo, Ontario, Canada
mnummer@vlsi.uwaterloo.ca, msachdev@ece.uwaterloo.ca

Abstract

This paper presents a methodology for testing high-performance pipelined circuits with slow-speed testers. The technique uses a clock timing circuit to control data transfer in the pipeline in test mode. A clock timing circuit capable of achieving a timing resolution of 50ps in 0.18μm CMOS technology is presented. The design provides the ability to test the clock timing circuit itself.

Keywords: Delay-fault testing, high-performance testing, design for testability, design for delay testability.

1. Introduction

The 2001 edition of the International Technology Roadmap for Semiconductors (ITRS) expects that the clock frequency of high-performance state-of-the-art CMOS VLSI circuits will exceed 6GHz by year 2007 [1]. According to the ITRS, potential manufacturing yield loss associated with the at-speed functional test methodology is related to the growing gap between automatic test equipment (ATE) performance and the ever increasing device I/O speed. In the last two decades, while the clock frequencies of VLSI circuits have improved at an average rate of 30% per year, the tester timing accuracy has improved only at a rate of 12%. For future high-performance VLSI circuits, it will be essential for the design to include design-for-testability/built-in-self-test (DFT/BIST) techniques to reduce the reliance on traditional, high-cost, full-feature testers. The requirements of ATEs designed to work on chips with DFT/BIST techniques are much simpler than those for the traditional testers.

The creation of a low frequency test mode in digital circuits was first introduced by Agrawal and Chakraborty [2] in 1995. They used a pulse-triggered flip-flop in which a dynamic latch is introduced inside a traditional master-slave flip-flop. In 1999, Shashaani and Sachdev proposed the controlled delay flip-flop (CDFF) [3] as an alternative to the pulse-triggered flip-flop. In this technique an additional test mode clock is used to control the delay of the flip-flop. The main advantages of the CDFF over the pulse-triggered flip-flop are the stable operation and improved performance in normal mode. In 2001, Nummer and Sachdev proposed an on-chip clock generation methodology that allows the test mode clock frequency of circuits using CDFFs to be reduced arbitrarily [4]

In this paper, we propose a methodology for testing high-performance pipelined circuits using slow speed testers. The technique depends on test mode clock shifting. It adds no extra hardware in the data path of the pipeline and therefore has virtually no performance penalty.

2. Testing Pipelines by Clock Shifting

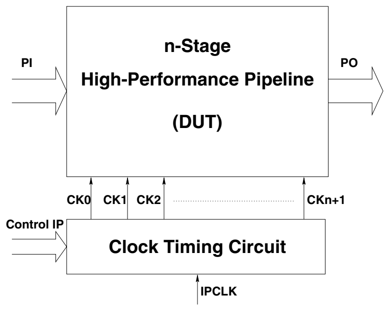
The basic idea in the proposed technique is demonstrated in Figure 1. In this study, we assume that the registers in the pipeline use a single phase common clock and are built using flip-flops as the storage element. A clock timing circuit is used to control clocks driving all registers in the pipeline. The idea is to use shifted version(s) of the input clock, IP-CLK, to control the timing of data flow through the pipeline in the test mode. In order to achieve that, each register in the pipeline has to have a separately routed clock, as shown in Figure 1(a). This results in higher complexity of clock generation and propagation. Special care should be given to keep skew between the different clock signals within acceptable limits.

In normal mode, a single phase high-frequency clock is used for all registers in the pipeline. As a result, the operation of the circuit depends on the period of this clock. The delay of stage i; t_{d_i}, can be expressed using the following equation.

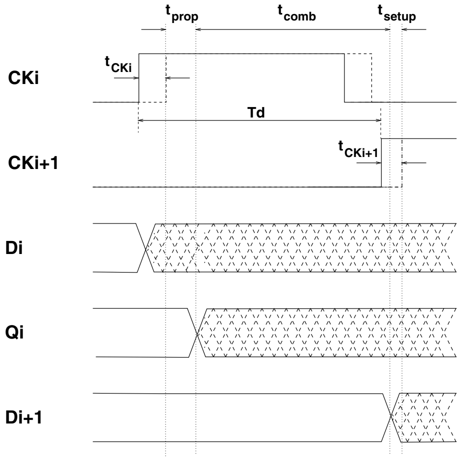
$$t_{d_i} = t_{prop_i} + t_{comb_i} + t_{setup_{i+1}} + (t_{CK_i} - t_{CK_{i+1}}) \quad (1)$$

where t_{prop_i} is the propagation delay of register i, t_{comb_i} is the delay of the i^{th} stage combinational block, $t_{setup_{i+1}}$ is the setup time of register $i+1$, and $(t_{CK_i} - t_{CK_{i+1}})$ is the difference between the delays through the clock driving networks of registers i and $i+1$. For the pipeline to function correctly, the normal mode clock period, T_{NM}, has to be at least equal to the largest stage delay, i.e.

$$T_{NM} \geq MAX_{i=0}^{n}(t_{d_i}) \quad (2)$$



(a)



(b)

Figure 1. (a) Block diagram of the DUT and the clock timing circuit. (b) Test mode operation.

In test mode, a delayed version of the clock is used to test the pipeline. This is illustrated in Figure 1(b). In order to test stage i, a delayed version of the input clock, IPCLK, with delay T_d is applied to register $i+1$, while the original clock is used for all other registers. The test mode clock period, T_{TM}, has to be at least equal to the normal mode clock period. Using a larger value of T_{TM} means that we can use a slow-speed tester in the test mode. Setting T_d to be equal to t_{d_i} allows the i^{th} stage to operate within its normal mode timing constraints while the whole circuit is running at a lower frequency. As a result, the slow-speed tester can be used to do performance binning and test the circuit for delay faults. This technique does not require any changes in the design of the registers or the combinational blocks and therefore has virtually no performance penalty.

3. Clock Timing Circuit

This section provides design details for the clock timing circuit. A number of issues have to be taken into account in the design of this circuit in order to ensure proper delay fault testability of the pipeline.

1. **Programmability and resolution:** The delay, T_d, should be programmable for two reasons: (i) to allow DUT performance binning , and (ii) to allow paths of different delays to be tested for delay faults. The resolution with which T_d is varied should be high enough to allow the detection of small delay faults.

2. **Timing accuracy:** Process, temperature, and supply voltage variations have a large effect on the delays of the circuit. The design should compensate for these effects such that the value of T_d will be as desired regardless of these variations.

3. **Tester edge placement accuracy (jitter):** A slow-speed tester is generally a low specification device. It has a low timing resolution and high edge placement inaccuracy, or jitter. The design should have the ability to suppress the tester's jitter in order to minimize its effect on the timing resolution and accuracy of the clock timing circuit.

4. **Testability:** Adding hardware to enhance testability begs the question: how can we insure the functionality of the extra hardware? is it testable? Answering these questions is very important and is a challenge to most design-for-testability techniques.

Figure 2 shows the clock timing circuit used in our design (multiplexer select inputs are not shown). The design assumes a 5 stage pipeline but can be expanded to larger and more complex circuits. This circuit is designed using CMOS 0.18μm technology provided through TSMC. It comprises a phase-locked-loop (which could be part of the clock generation circuit for the chip, as is the case for high-performance microprocessors, e.g.), three delay lines, a phase splitter & delay circuit, a DLL, and a number of multiplexers. The design allows T_d to vary between 250ps and 1000ps with 50ps increments (resolution). These are design variables and it is up to the designer to choose the values suiting a specific circuit or application. The design and operation of the different blocks used in the clock timing circuit is described below.

1. **Phase-locked-loop (PLL):** The PLL works as a frequency multiplier. It is used to generate a high-frequency clock from a low-frequency input clock in test mode. It is assumed that the normal mode input clock is a high-frequency clock, and hence the PLL is not active in that mode (the design could be such that the same PLL is used to generate the normal mode high-frequency clock from a slower input clock). Our design assumes that the input clock from the tester (IPCLK) has a frequency of 100MHz (this is a design variable and could be different). The PLL generates a high-frequency clock (HFCLK) with a frequency of 1GHz (10 times the input clock frequency). HFCLK is used only to calibrate the delay lines used to generate T_d. Using a PLL to generate the high-frequency clock used for calibration serves two purposes: (i) it helps suppress tester jitter and reduce its effect on the timing accuracy of the design, and (ii) it allows us to use smaller number of delay elements in the delay lines as will be shown later.

2. **Delay-locked-loop (DLL):** The main function of the

IEEE COMPUTER SOCIETY

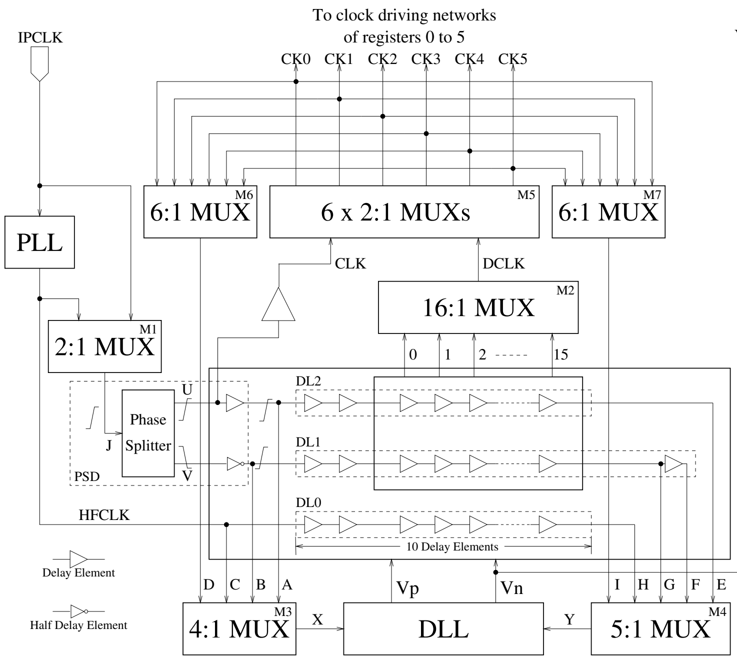


Figure 2. Clock Timing Circuit (multiplexer select inputs are not shown).

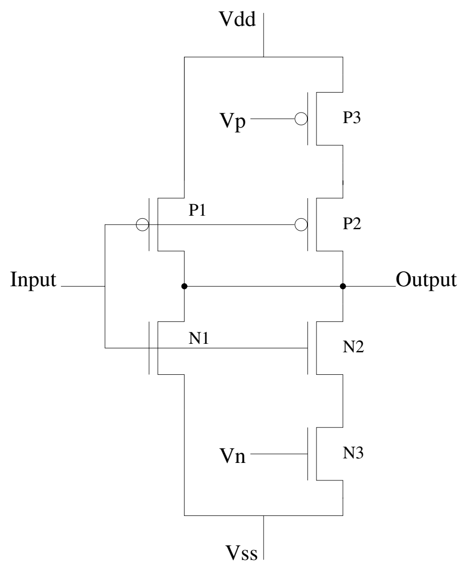


Figure 3. Schematic diagram of half the delay element [5].

DLL is to calibrate the delay lines used to generate the delayed version of the input low-frequency clock. This is particularly important in order to ensure correct timing regardless of any process, temperature, and/or supply voltage variations. Its outputs, V_p and V_n, are used to control the delays of the delay elements. The values of these control voltages depend on the phase difference between the two inputs of the DLL (X and Y). When locked, V_p and V_n should be such that the signal at Y is delayed by one complete cycle from the signal at X. The DLL serves another important purpose. It allows great flexibility for testing the clock timing circuit itself. As a result, most of the extra hardware in our design can be tested simply by comparing values of the DLL's outputs (V_p and V_n) when different signals are used to close the loop.

3. **Delay lines and phase splitter & delay circuit:** Three delay lines, DL0, DL1, and DL2, are used in the design. Each delay line comprises a number of delay elements. The schematic diagram of half the delay element is shown in Figure 3. It consists of a static inverter connected in parallel to a current starved inverter [5]. The circuit delay is controlled by the two bias voltages V_p and V_n. When transistors P3 and N3 are turned off, the delay is maximum and depends on the sizes of transistors P1 and N1. On the other hand, when V_p and V_n are equal to V_{ss} and V_{dd} respectively, maximum current is allowed to flow through the current starved inverter resulting in minimum delay.

Initially, our objective was to design a delay element with a delay of 50ps (the resolution of T_d). We were able to achieve that only for one edge of the input clock. The delay for the other edge was much higher. Using the DLL to calibrate the delay lines requires that the delay for any edge to be within a certain limit, otherwise the signal will be lost before reaching the end of the delay line. As this is not feasible using the technology at hand (0.18μm CMOS technology), we designed the delay elements to have a delay of 100ps and used two delay lines with a 50ps delay between them to get a resolution of 50ps. It is worth noting that sizing of the transistors in the delay element should be such that there are values of the control voltages that result in a delay of 100ps under worst case conditions (slow-PMOS and slow-NMOS transistor models, $T = 100^oC$, and V_{dd} is 10% less than its nominal value) as well as best case conditions (fast-PMOS and fast-NMOS models, room temperature, and V_{dd} is 10% more than its nominal value).

DL0 consists of 10 delay elements. It is used to calibrate the other two delay lines. This is achieved by closing the DLL loop using DL0 and applying HFCLK from the PLL to the input of this delay line. The period of HFCKL is 1ns for an input clock frequency of 100MHz. Once locked, the output voltages from the DLL should be such that the total delay across DL0 is equal to one clock period. As a result and since these voltages control all three delay lines, the delays of all delay elements in the circuit are adjusted to 100ps. If IPCLK were to be used to calibrate the delay lines, DL0 would have 100 delay elements rather than only 10. This explains the benefit of generating HFCLK using the PLL.

As shown in Figure 2, the phase splitter & delay (PSD) circuit accepts either IPCLK (low-frequency clock, in test mode) or HFCLK and generates two clocks A and B. PSD is designed such that the clock at A is delayed by half delay-element delay with reference to the clock at B. This means that when V_p and V_n are such that the delay of the delay element is 100ps, the delay from B to A is 50ps. These two clocks are fed to the other two delay lines, DL1 and DL2. DL1 and DL2 consist of 11 and 10 delay elements, respectively. The main function of these two delay lines is to generate DCLK with programmable delays with respect to CLK (Figure 2). It is worth noting that the input clock jitter has virtually no effect on the timing accuracy of DCLK and hence on high-performance delay fault testability. Any edge placement inaccuracy at the input propagates through DL1 and DL2. As a result, the same amount of jitter is added to both CLK and DCLK keeping the delay between these two clocks at the desired value of T_d.

4. **Multiplexers M1-M7:** The multiplexers shown in Figure 2 can be divided into two groups. The first group includes M2 and M5. Outputs from 16 delay elements in DL1 and DL2 are tapped and fed to M2. According to the select inputs of M2, one of the 16 inputs is selected to be DCLK. This allows the delay between CLK and DCLK to be varied between 250ps and 1000ps with 50ps increments (note that CLK is buffered to compensate for the delay through M2). M5 consists of six 2:1 multiplexers. It is used to control the clocks feeding all registers in the pipeline (CK0 to CK5). Depending on the mode of operation and the pipeline stage to be tested, M5 sets the clock of each register to either CLK or DCLK, according to its select inputs.

The second group of multiplexers include M1, M3, M4, M6, and M7. These multiplexers are used only to ensure the functionality of the clock timing circuit. Details on the operation of these multiplexers are given in the next section.

The proposed clock timing circuit takes into account all the issues mentioned at the beginning of this section. Programmability is achieved through DL1, DL2, and the multiplexer M2. Using a DLL allows us to achieve the same timing accuracy regardless of process, temperature, and/or supply voltage variations. The DLL with a number of multiplexers help ensure the functionality of the clock timing circuit itself. With the help of the PLL, tester clock jitter is not allowed to propagate to DL0. It has no effect on the timing accuracy of signals generated form DL1 and DL2. The area overhead due to this design is estimated to be 100 gates per pipeline stage. This should be acceptable for medium to large pipelined circuits. It is important to note that matching between the different components in the design is essential to ensure correct timing even with small local process, temperature and/or supply voltage variations. This can be achieved through circuit layout techniques similar to those used for analog circuits.

4. Modes of Operation

In this section we put the different blocks of the clock timing circuit together and show how they function in the different modes of operation. The clock timing circuit operates in three modes. In *normal mode* the pipeline is used to perform the function it is designed for. In the *DUT test mode*, the clock timing circuit is used to verify the performance of the DUT and to test it for delay faults. In the *clock timing circuit test mode*, the clock timing circuit is tested to ensure its ability to give a correct image about the performance of the pipeline.

4.1 Normal Mode

In normal mode, the control inputs of multiplexers M1 and M5 are set such that IPCLK propagates to all registers in the pipeline. As a result, the whole pipeline runs using the same clock ensuring normal operation. The remaining part of the clock timing circuit serves no function in normal mode, and can be turned off. This is important to reduce the effect the clock timing circuit has on normal mode power consumption of the chip.

4.2 DUT Test Mode

This mode is used to verify the performance of the DUT and test it for delay faults. The pipeline is tested one stage at a time. For every target path, two vectors are used to test the circuit. The first vector initializes the DUT, while the second vector activates the target path of the stage to be tested. Using multiplexer M2, T_d is set according to the expected delay for the path tested by these vectors. Depending on the stage to be tested, multiplexer M5 is configured to set one of the register's clocks to DCLK and all other clocks to CLK. For example, to test stage 3 in the pipeline, clocks CK0, CK1, CK3, and CK5 are set to CLK while CK4 is set to DCLK. As a result, data flows between registers 3 and 4 within normal mode timing constraints. For all other stages, the low frequency clock allows operation under relaxed timing. This is important to ensure that delay faults in these stages do not affect the target path and hence the stage under test. This procedure is repeated for every path to be tested until the DUT is tested completely.

4.3 Clock Timing Circuit Test Mode

In this mode, the DLL, with the help of multiplexers M1, M3, M4, M6, and M7, is used to characterize and verify the performance of the clock timing circuit. Multiplexer M1, is set to propagate the high-frequency clock, HFCLK, through DL1 and DL2. The clock timing circuit is tested in three phases as shown in Figure 4. In the first phase, all three delay lines (excluding the last delay element in DL1) are tested. The second phase is used to test the PSD circuit. This is done by verifying that the delay between nodes A and B is equal to half delay-element delay. The third phase is used mainly to test multiplexer M5. This phase also covers multiplexers M6, M7, and a small part of M2. Each of the three phases cover parts of multiplexers M3 and M4, such that these two multiplexers are fully testable. In our design, only M1 and most of M2 are not testable. Compared to the rest of the clock timing circuit and the DUT, this is a very small area of the circuit.

Phase 1: Testing DL0, DL1, and DL2. In this phase, delay lines DL0, DL1, and DL2 are tested between nodes C and H, B and G, and A and E, respectively. Table 1 gives the number of delay element between the different nodes in the clock timing circuit. The first three entries in the table have the same number of delay elements. As a result, closing the DLL loop with these node pairs one at a time, should result in very close values of V_n (monitored off-chip) for all three configurations. As shown in Figure 4, the test in this phase is done in four steps:

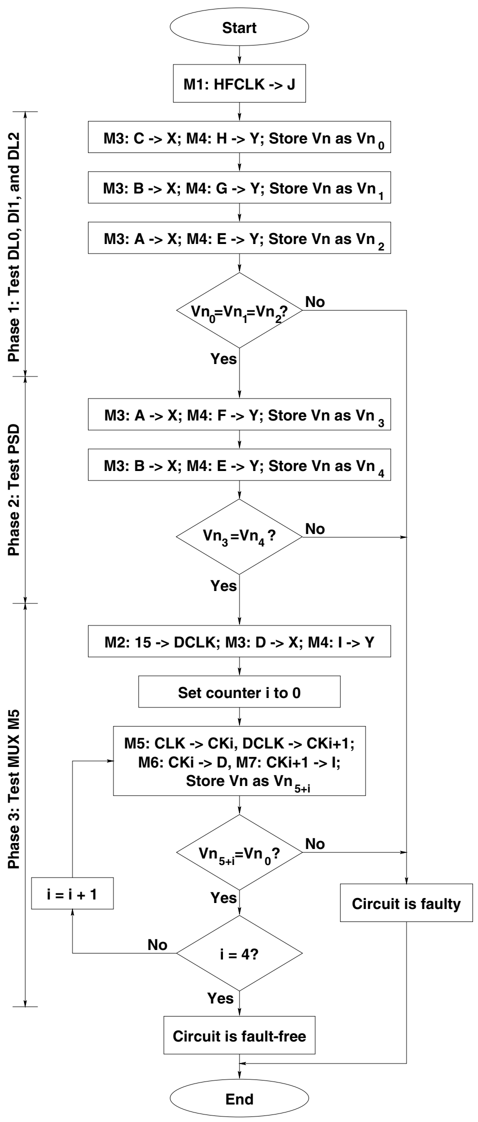
1. Close the DLL loop using C and H ($M3 : C \to X; M4 : H \to Y$). Measure V_n and record it as V_{n_0}.
2. Close the DLL loop using B and G ($M3 : B \to X; M4 : G \to Y$). Measure V_n and record it as V_{n_1}.
3. Close the DLL loop using A and E ($M3 : A \to X; M4 : E \to Y$). Measure V_n and record it as V_{n_2}.
4. Compare V_{n_0}, V_{n_1}, and V_{n_2}. If the differences between the three values are within acceptable limits, the three delay lines are considered to be free from delay faults.

Ideally, the three values should be equal. Mismatch between the delay elements and the different components in the DLL would result in some differences between them. Characterization is necessary to define how much difference due to mismatch and process variations is acceptable. In this context, we assume that only single delay fault exists in the circuit. It is worth noting that it is highly unlikely to have the exact same amount of delay fault in two of or all three delay lines. In addition to DL0, DL1, and DL2, the test in this phase covers paths through multiplexers M3 and M4 used to close the DLL loop (A-X, B-X, C-X, E-Y, G-Y, and H-Y).

Phase 2: Testing the Phase Splitter & Delay Circuit. Referring to Table 1, the delay between nodes A and F on one side and nodes B and E on the other are both equivalent to $10\frac{1}{2}$ delay elements. This is only true if the two outputs of the PSD circuit (A and B) are exactly half delay-element delay apart. These observations are used to test the PSD circuit for delay faults. As shown in Figure 4, the test is done first by closing the DLL loop using nodes A and F. When locked, DLL output V_n is recorded as V_{n_3}. Similarly, V_{n_4} is obtained by closing the DLL loop through nodes B and E. For fault-free PSD, the difference between V_{n_3} and V_{n_4} should be within acceptable limits (defined through characterization). The test in this phase covers the G-F-Y path as well.

Nodes	# of delay elements
A and E	10
B and G	10
C and H	10
A and F	$10\frac{1}{2}$
B and E	$10\frac{1}{2}$

Table 1. Number of delay elements between different nodes in the delay lines.



Start
M1: HFCLK -> J
Phase 1: Test DL0, Dl1, and DL2
M3: C -> X; M4: H -> Y; Store Vn as Vn 0
M3: B -> X; M4: G -> Y; Store Vn as Vn 1
M3: A -> X; M4: E -> Y; Store Vn as Vn 2
Vn0=Vn1=Vn2?
No
Yes
Phase 2: Test PSD
M3: A -> X; M4: F -> Y; Store Vn as Vn 3
M3: B -> X; M4: E -> Y; Store Vn as Vn 4
Vn3=Vn4?
No
Yes
Phase 3: Test MUX M5
M2: 15 -> DCLK; M3: D -> X; M4: I -> Y
Set counter i to 0
M5: CLK -> CKi, DCLK -> CKi+1; M6: CKi -> D, M7: CKi+1 -> I; Store Vn as Vn5+i
Vn5+i=Vn0?
No
Yes
i = i + 1
Circuit is faulty
No
i = 4?
Yes
Circuit is fault-free
End

Figure 4. Procedure for testing the clock timing circuit.

Phase 3: Testing Multiplexer M5. Phase 3 of the clock timing circuit test procedure is used to test multiplexer M5. This is done with the help of multiplexers M6 and M7. The main idea is to set the delay between CLK and DCLK (T_d) to one cycle of HFCLK (1ns). The different paths in M5 are tested two at a time. As shown in Figure 4, to test the

IEEE COMPUTER SOCIETY

paths from CLK to CK_i and DCLK to CK_{i+1} (where i is any number between 0 and 4), M5 is set accordingly and M6 and M7 are used to close the DLL loop using nodes CK_i and CK_{i+1}. Under these conditions, if all signal paths are free of delay fault, when locked, the DLL output, V_n, should be equal to the values obtained in phase 1 of the test procedure (for fault free DL0, DL1, and DL2). The test in this phase covers M6, M7, the 15-DCKL path in M2, the D-X path in M3, the I-Y path in M4, and the buffer used for CLK.

5. Delay Fault Simulation Results

Delay fault simulations for the clock timing circuit are carried out in order to verify our ability to characterize the performance of the circuit. 15 delay faults are inserted in different locations of the circuit one at a time. These fault are generated using the same delay element used in the design. Using active elements to add extra delays ensures the existence of a pre-determined delay fault without degrading the quality (rise/fall times) of the delayed signals. The delay of the delay element is at its minimum of 60ps when $V_p = V_{ss}$ and $V_n = V_{dd}$. Hence the smallest delay fault used in the simulations is 60ps. Larger delays are achieved using larger values of V_p and smaller values of V_n. The test is done according to the procedure in Figure 4. The results are shown in Tables 2(a), (b), and (c), showing faults detected in each phase of the clock timing circuit test procedure.

A fault detectable in phase 1 or phase 3 of the test procedure causes only one value of V_n (V_{n_0}, V_{n_1}, or V_{n_2} for phase 1 or $V_{n_{5+i}}$ for phase 3) to deviate from the fault-free value. This is why for faults detected in phase 1 and phase 3, the last column in each table gives the difference in V_n compared to the fault free value. This is different from faults detectable in phase 2, where the fault causes both V_{n_3} and V_{n_4} to deviate in opposite directions from the fault free value. For faults detected in phase 2, $\triangle V_n$ in Table 2(b) is computed as the difference between V_{n_3} and V_{n_4}.

A positive $\triangle V_n$ indicates that the delay seen by the DLL is larger than it should be. As a result, the DLL causes the voltage to increase in to order to compensate for the extra delay. The opposite is true for negative $\triangle V_n$. For a certain delay fault, the larger the value of $\triangle V_n$, the easier it is to observe the error due to the fault. As shown in Table 2, the smallest value of $\triangle V_n$ is 17.3mV which can be easily measured off-chip. These results demonstrate our ability to test the clock timing circuit for delay faults. This is important to ensure its ability to give a true image about the operation and performance of the DUT.

Fault #	Fault location	Faulty Path	Delay fault (ps)	V_{n_0} mV	V_{n_1} mV	V_{n_2} mV	$\triangle V_n$ mV
F1	DL0	C-H	60	**633**	611.5	611.5	21.5
F2	DL1	B-G	100	611.5	**649**	611.5	37.5
F3	DL2	A-E	200	611.5	611.5	**691.7**	80.2
F4	M3	A-X	60	611.5	611.5	594.2	-17.3
F5	M3	B-X	100	611.5	**583.3**	611.5	-28.2
F6	M3	C-X	200	**554**	611.5	611.5	-57.5
F7	M4	E-Y	60	611.5	611.5	**634**	22.5
F8	M4	G-Y	100	611.5	**648.7**	611.5	37.2
F9	M4	H-Y	200	**697**	611.5	611.5	85.5

(a) Faults detected in phase 1 (fault-free V_n is 611.5mV, faulty values in bold)

Fault #	Fault location	Faulty Path	Delay fault (ps)	V_{n_3} mV	V_{n_4} mV	$\triangle V_n$ mV
F10	PSD	J-A	60	609.8	651.5	-41.7
F11	PSD	J-B	100	669	597.9	89.1
F12	M4	F-Y	200	720	628.6	91.4

(b) Faults detected in phase 2 (fault-free V_n is 628.6mV)

Fault #	Fault location	Faulty Path	Delay fault (ps)	V_n mV	$\triangle V_n$ mV
F13	M2	15-DCLK	60	633.2	21.7
F14	M5	DCLK-CK4	100	647	35.5
F15	Buffer	U-CLK	200	554.8	-56.7

(c) Faults detected in phase 3 (fault-free V_n is 611.5mV)

Table 2. Delay fault simulation results for the clock timing circuit.

6. Conclusions

We presented a methodology for testing high-performance pipelined circuits with slow-speed testers. In this technique, each pipeline stage is clocked using a separate clock generated from an on-chip clock timing circuit in test mode. The technique adds no extra hardware in the data path of the pipeline and therefore has no performance penalty.

A design for the clock timing circuit capable of achieving a timing resolution of 50ps in 0.18μm CMOS technology was presented. The design provides the ability to test the clock timing circuit itself. Simulations show that we are able to detect delay faults in the clock timing circuit as small as 60ps at an input clock frequency of 100MHz.

Future work includes testing the clock timing circuit with a pipelined test vehicle in order to demonstrate our ability to detect delay faults in the pipeline.

References

[1] Semiconductor Industry Association, "International Technology Roadmap for Semiconductor, 2001 Edition", 2001.

[2] V. D. Agrawal and T. J. Chakraborty, "High-Performance Circuit Testing with Slow-Speed Testers", *Proc. of International Test Conference*, pp. 302–310, 1995.

[3] M. Shashani and M. Sachdev, "A DFT Technique for High-Performance Circuit Testing", *Proc. of International Test Conference*, pp. 267–285, 1999.

[4] M. Nummer and M. Sachdev, "A Methodology for Testing High-Performance Circuits at Arbitrarily Low Test Frequency", *IEEE VLSI Test Symposium*, pp. 68–74, April 2001.

[5] Gary C. Moyer, *"The Vernier Technique for Precise Delay Generation and Other Applications"*, PhD thesis, The Department of Computer Engineering, North Carolina State University, 1996.

IEEE COMPUTER SOCIETY