Test Generation for Clock-Domain Crossing Faults in Integrated Circuits*

Naghmeh Karimi[†], Krishnendu Chakrabarty[†], Pallav Gupta[‡] and Srinivas Patil[§]

[†]Electrical & Computer Engineering Department
Duke University
Durham, NC 27708

[‡]Test CAD Technology Group
Intel Corporation
Folsom, CA 95630

[§]SoC Enabling Group
Intel Corporation
Austin, TX 78704

Abstract—**Clock-domain crossing (CDC) faults are a serious concern for high-speed, multi-core integrated circuits. Even when robust design methods based on synchronizers and design verification techniques are used, process variations can introduce subtle timing problems that affect data transfer across clock-domain boundaries for fabricated chips. We present a test generation technique that leverages commercial ATPG tools, but introduces additional constraints, to detect CDC faults. We also present HSpice simulation data using a 45 nm technology to quantify the occurrence of CDC faults at clock-domain boundaries. Results are presented for synthesized IWLS05 benchmarks that include multiple clock domains. The results highlight the ineffectiveness of commercial transition-delay fault ATPG and the "coverage gap" resulting from the use of ATPG methods employed in industry today. While the proposed method can detect nearly all CDC faults, TDF ATPG is found to be severely deficient for screening CDC faults.**

I. Introduction

System-on-chip integrated circuits today offer diverse functionality and contain billions of transistors. However, high-speed communication between cores remains a major challenge. This problem is exacerbated when cores operate in separate clock domains and at different clock frequencies.

In multi-clock designs, a clock-domain crossing (CDC) occurs whenever data is transferred from one clock-domain to another. Depending on the relationship between the sender and receiver clocks, various types of problems may arise during data transfer. Propagation of metastability, data loss, and data incoherency are three fundamental problems of multi-clock design, all of which are caused by CDC faults. [1].

To reduce the probability of propagating metastability through the design, designers employ synchronizers at clock boundaries. Moreover, to avoid data loss and to ensure proper transmission and reception of data in multi-clock designs, designers also rely on appropriate CDC protocols. Data incoherency, which mainly occurs where CDC signals reconverge, is avoided by making designs tolerant of the variable delays that occur on reconvergent paths [2] [3]. Verification techniques and commercial verification tools enable designers to check designs for CDC-associated problems and verify the correctness of functional behavior [4]–[8].

If CDC errors are not addressed early in the design cycle, many chips are likely to exhibit functional errors during post-silicon validation. To address the metastability that occurs in multi-clock circuits and consequently to increase the mean-time between-failure, designers typically employ different types of synchronizers, among which the most commonly used is a pair of flip-flops residing on the clock boundaries.

As we move towards higher integration levels in VLSI technology and even smaller technology nodes, errors that occur due to process variations, design marginalities, and corner operating conditions are starting to play a more important role in multi-clock circuits. Consequently, circuits that were deemed to be fault free through CDC analysis during pre-silicon validation, may exhibit CDC errors after fabrication.

Therefore, the effect of process variations on correct operation of multi-clock circuits must be investigated, and there is a need for testing techniques for CDC faults. A test-pattern selection method for detecting CDC faults was recently proposed in [9]. A commercial ATPG tool and a commercial logic simulator were used to extract, from a pattern repository, a set of test patterns that detect CDC faults. However, repeated invocation of the commercial logic simulator leads to extremely long runtimes. Moreover, the tests derived in [9] do not target at-speed transfer of transition of data required between the clock domains, hence their effectiveness for high-speed circuits is questionable.

In this paper, we investigate the problem of testing for CDC faults, especially for at speed CDC, and develop an automatic test-pattern generation (ATPG) technique for CDC fault detection. The contributions of this paper are as follows:

- A fault model reflecting the various functional errors associated with CDCs;
- Detailed HSpice simulations to understand the impact of process variations on synchronizers;
- An ATPG method based on bounded time-frame expansion and logic constraints;
- A comprehensive set of test generation results to demonstrate the effectiveness of the proposed ATPG approach.

The rest of this paper is organized as follows. Section II discusses methods used to resolve metastability. Section III

*This research was supported in part by NSF under grant No. CCF-0903392, and by SRC under Contract No. 1992.

978-3-9810801-8-6/DATE12/©2012EDAA

motivates the need for post-silicon CDC testing. In Section IV, we develop CDC fault models representing the faulty behavior in the presence of physical defects. The ATPG algorithm and implementation issues are described in Section V. Experimental results are presented in Section VI. Section VII concludes the paper.

II. Resolving Metastability

Synchronizers are used to mask the effect of metastability in multi-clock circuits. It is expected that in a design including synchronizers, the output of a flip-flop rarely becomes metastable, e.g., only once in every "mean-time between failure" (MTBF) years, typically three years for clock frequencies of 100 KHZ. However, for faster clocks, the probability of observing metastability at the outputs of flip-flops increases rapidly, e.g., the MTBF drops to one minute for a clock frequency of 1 GHz [10].

To prevent incorrect operation due to metastability, both asynchronous and synchronous handshaking mechanisms between different clock domains have been proposed in the literature. In the asynchronous handshaking mechanism, a request (req in Fig. 1) is first sent from the sender to receiver domain. After sending the request, the sender sends the data to the receiver. The receiver sends out acknowledgement (ack in Fig. 1) to the sender to indicate completion of data transfer. Upon receiving acknowledgement, the sender can send another request to the receiver. As shown in Fig. 1, to immunize the handshaking mechanism against the metastability of req and ack signals, synchronizer flip-flops are inserted in the circuit.

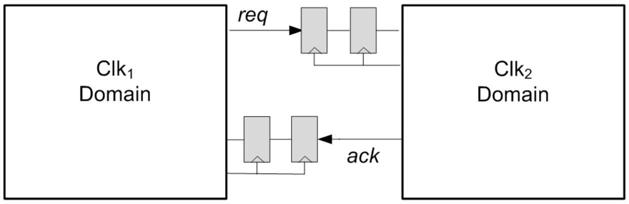


Fig. 1. A two-way control synchronizer [10].

Although an asynchronous handshaking method is immune against CDC faults, it suffers from uncertainly and indeterministic delay of data transfer between different domains. To resolve this bottleneck and to achieve higher performance, FIFOs (and particularly 2-clock FIFO synchronizers) are used in multi-clock circuits. However, the size of the FIFO buffers is a concern and "what size FIFO to use" can be a difficult design decision. The larger a FIFO is, the higher the cost.

Synchronizers without handshaking allow us to overcome the drawbacks of asynchronous handshaking in the transfer of data between different domains. The use of two flip-flop synchronizers is common in multi-clock circuits [10]. However, fast clocks, low supply voltages, and extremely low or high temperatures decrease MTBF and necessitate the use of additional synchronizer flip-flops [10].

Ideally, the flip-flops used as synchronizers are more robust to variations in process, temperature, and voltage. The setup- and hold-time of synchronizer flip-flops should be zero. However, it is costly to use synchronizer flip-flops with negligible setup- and hold-time. For example, a nearly-zero setup time flip-flop presented in [11] requires a 66% area overhead compared to a typical flip-flop.

In state-of-the-art SoCs, thousands of bits of data are transferred between different clock domains [12]. Due to the timing uncertainty of asynchronous handshaking as well as the high cost associated with the use of special synchronizer flip-flops with zero setup times, it is more practical for multi-clock SoCs to use typical synchronizer flip-flops to transfer data between clock domains.

III. Impact of Process Variation on CDC Faults

State-of-the-art SOC designs typically employ dozens of clock signals, many of which are asynchronous, i.e., the clock signals are fed either by different PLL sources, or by a common PLL source but with different phases and frequencies. Data transfer in such multi-clock circuits may result in metastability if the timing requirements of the flip-flops residing on the clock boundaries are not met. As discussed in Section II, at the design level, the effect of metastability is neutralized by using synchronizer flip-flops.

The motivation for our work lies in our observation that multi-clock circuits, even when equipped with synchronizers at clock boundaries, may exhibit incorrect behavior due to process variation-induced violation of setup- and hold-time at the boundary flip-flops.

In reality, the parameters of fabricated transistors do not always match design specifications due to process variations. These variations directly result in deviations in transistor parameters, such as threshold voltage, oxide thickness, and W/L ratios, and significantly impact the functionality of circuits in very-deep submicron technologies [13].

To evaluate the impact of process variations on the transfer of data between different clock domains, even when synchronizer flip-flops are employed at clock boundaries, we conducted a series of HSpice simulations under process variations for a generic CDC circuit shown in Fig. 2. In this circuit, flip-flops DFF_2 and DFF_4 reside in different clock domains and act as sender and receiver flip-flops, respectively. In addition, flip-flop DFF_3 is employed as a synchronizer.

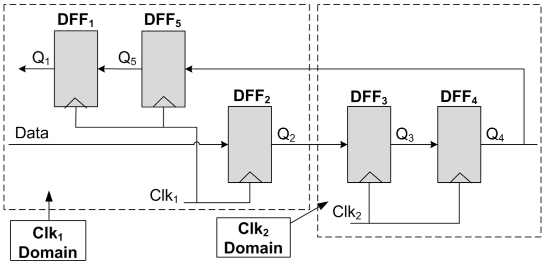


Fig. 2. A generic CDC circuit [14].

To determine the effect of process variation in the transfer of data between clock domains, we ran several HSpice Monte

TABLE I

NUMBER OF SETUP- AND HOLD- TIME VIOLATIONS FOR DIFFERENT NUMBERS OF MONTE CARLO (MC) SIMULATIONS.

Total no of MC runs	No. of runs with setup time violation (%)	No. of runs with hold time violation (%)
2000	975 (48.7%)	863 (43.1%)
4000	2012 (50.3%)	1723 (43%)
6000	3106 (51.8%)	2571 (42.8%)
8000	3985 (49.9%)	3324 (41.5%)
10000	4940 (49.4%)	4337 (43.3%)

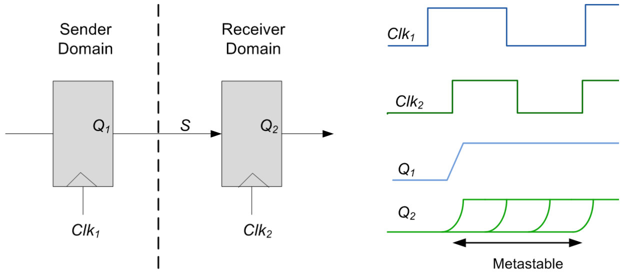
Carlo (MC) simulations on the circuit shown in Fig. 2 using the 45 nm predictive technology model [15]. Simulations were carried out using the following process-variation parameters for a Gaussian distribution: transistor gate length L: $3\sigma = 10\%$; threshold voltage V_{TH}: $3\sigma = 30\%$, and gate-oxide thickness t_{OX}: $3\sigma = 3\%$. The process variation data reflects a 45 nm process in commercial use today. To isolate the effect of process variation on data transfer between different clock domains of the circuit shown in Fig. 2, only the parameters for flip-flops DFF_2, DFF_3, and DFF_4 are assumed to have a Gaussian distribution, and the parameters for the other two flip-flops are assigned deterministic values.

We recorded the number of experiments in which the setup/hold time of the receiver flip-flop DFF_4 were violated under process variation. The results are shown in Table I. The experiments for evaluating the number of setup- and hold-time violations, were conducted with different sets of inputs. We found that in more than 50% of the experiments, process variations result in a setup time violation at the receiver flip-flop and consequently in incorrect circuit operation even when synchronizer flip-flops are employed. In addition, hold-time violations occur in 43% of the test cases considered. These results highlight the fact that due to the effect of process variations, design verification does not accurately predict silicon behavior for clock domain crossings and synchronizers do not prevent errors; therefore, manufacturing testing for CDC faults is necessary.

Transition delay fault (TDF) testing is widely used in industry to target timing-related defects. Despite their benefits, current transition ATPG tools are not adequate for detecting CDC faults because these tools do not model and target the interaction between logic residing at clock boundaries when test patterns are generated for TDFs. Path-delay test methods [16] [17] [18] suffer from the scalability problem for large designs, and the timing-critical paths that they target do not necessarily include clock-domain crossings. We show in this paper that TDF test patterns are not adequate for CDC faults and they lead to a "coverage gap". Therefore, fault models and ATPG methodologies need to be developed to specifically target CDC faults.

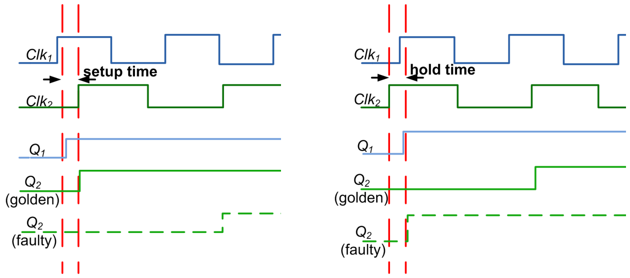
IV. CDC FAULT MODEL

To be able to screen CDC defects, the faulty behavior of these defects must be logically represented using a fault model. In this section, we present a CDC fault model to capture the erroneous behavior.



(a) CDC circuit. (b) Metastability on Q_2.

Fig. 3. An example of a CDC circuit and metastability.



(a) Setup-time violation. (b) Hold-time violation.

Fig. 4. Timing waveforms showing setup- and hold- time violations for the circuit in Fig. 3(a).

In a synchronous circuit, the proper operation of a flip-flop depends on the stability of its input signal for a certain period of time before (setup-time) and after (hold-time) its clock edge. If setup- and hold-times are violated, the flip-flop output may oscillate for an indefinite amount of time, and may or may not settle to a stable value before the next active clock edge. This unstable behavior is known as *metastability*. Fig. 3(a) shows an example of a multi-clock circuit in which signal S is launched by Clk_1, and needs to be captured properly by Clk_2. As shown in Fig. 3(b), if a transition on S happens very close to the active edge of Clk_2, a setup-time violation occurs, which may lead to metastability on Q_2.

CDC faults mainly occur due to setup- and hold-time violations on flip-flops residing at clock boundaries. If a flip-flop experiences a setup-time violation, it does not sample a change in value at its data input. In a hold-time violation, however, it may incorrectly capture a data change at its input. We next describe the fault model for each case.

A. Setup-Time Violation

Fig. 4 illustrates sample waveforms for the CDC circuit of Fig. 3(a). As shown in Fig. 4(a), if signal S experiences an unexpected delay and its value changes during the setup-time window of the receiver flip-flop, the receiver flip-flop may capture the value "0" even though the expected value is "1". Since the output of the sender flip-flop does not change in the subsequent clock cycle, Q_2 gets its expected value of "1" in the next clock cycle. In this case, the setup-time violation of the receiver flip-flop can be modeled as a slow-to-rise fault with a delay of one clock cycle. However, if the width of the transition on the output of the sender flip-flop is not long enough, the receiver flip-flop will not capture that transition, and remain unchanged. In this case, the setup-time violation of the receiver flip-flop can be modeled by a slow-to-rise fault with infinite delay.

In general, if a value change of a CDC signal S violates the setup-time of the receiver flip-flop, then the faulty behavior can be modeled as a transition (slow-to-rise or slow-to-fall) fault with a delay of k clock cycles, where $k = 1$ if the pulse observed in signal S is at least 1.5 times wider than the receiver clock period. Otherwise, $k = \infty$. In the rest of this paper, a CDC fault arising due to setup-time violations will be referred to as a *S-CDC* fault.

B. Hold-Time Violation

If a flip-flop experiences a hold-time violation, data changes on its input may be incorrectly sampled. Fig. 4(b) shows another sample waveform for the CDC circuit of Fig. 3(a). If signal S changes during the hold-time interval of the receiver flip-flop, an incorrect change on the output may be observed. The receiver flip-flop gets an output value of "1" one clock cycle earlier than expected. In this case, the hold-time violation at the receiver flip-flop can be modeled as a transient fault with a duration of one clock cycle. Similarly, if the output of the sender flip-flop changes before the next active edge of the receiver flip-flop, the receiver flip-flop captures the transition of signal S, and the hold-time violation of the receiver flip-flop can be modeled as a transient fault with a duration of one clock cycle. In this work, we focus on S-CDC faults and leave the treatment of hold-time violations for future work.

V. Proposed Test-Generation method

A TDF ATPG tool cannot be used to detect all S-CDC faults. It typically launches a transition at the fault site and propagates it to an observable output, i.e., either a scan flip-flop or a primary output. While these steps are also necessary to detect S-CDC faults, they are not sufficient. The detection of S-CDC faults requires fault excitation and propagation through paths from the sender domain. However, this requirement is not always met when TDF ATPG tools are used for test generation.

Launch-on-Shift (LoS) and Launch-on-Capture (LoC) are two widely used TDF testing methods [19]. Since LoC is easier to implement in practice [20], we only consider the LoC method for detecting S-CDC faults. The specific requirements and constraints for the LoS scheme are different and they are not considered here.

In this paper, we describe the proposed testing method to detect S-CDC faults using the simple multi-clock domain circuit shown in Fig. 5. In this circuit, for the sake of clarity, only the flip-flops at clock boundaries are shown. In this paper, we focus on CDC testing of the multi-clock circuits with synchronous handshaking mechanism between different clock domains.

Assume that we want to target the S-CDC fault modeled by a slow-to-rise fault at the output of the receiver flip-flop (Signal *B*) in the circuit shown in Fig. 5. To detect this fault, first a rising transition must be generated on signal *A* and then this transition must be propagated to signal *B* in the next active edge of Clk_2.

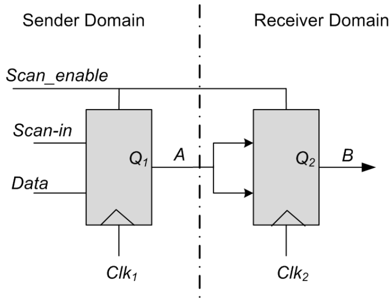
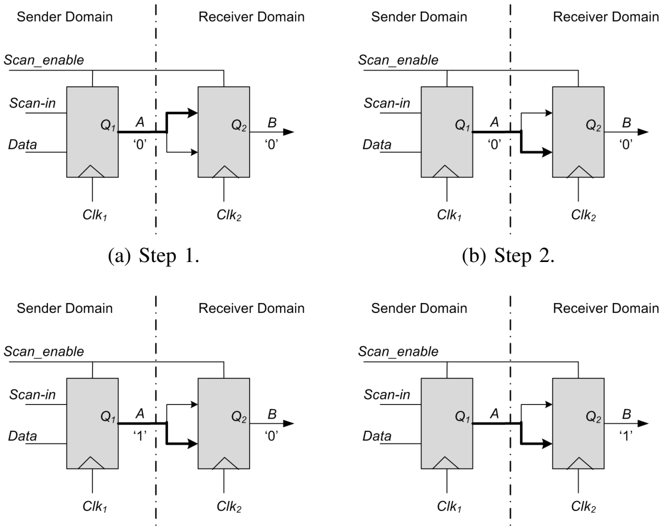


Fig. 5. A CDC example for illustrating the proposed ATPG method.



(a) Step 1.

(b) Step 2.

(c) Step 3.

(d) Step 4.

Fig. 6. Illustration of all steps to target slow-to-rise S-CDC fault on signal B (active path highlighted in bold).

To ensure an at-speed transition on Signal *A* with respect to Clk_1, and an at-speed transition on signal *B* with respect to Clk_2, we need to apply four test vectors instead of the two that are applied by the traditional LoC method. Fig. 6(a)- 6(d) show the active paths highlighted in bold for the four steps needed to detect the CDC fault. We refer to the proposed method as CDC-oriented Triple-Capture (CoTC).

The four steps in CoTC to target the S-CDC fault modeled by a slow-to-rise fault on signal *B* are as follows:

- **Step 1:** Shift vector V_1 to the circuit in scan mode such that A and B both get the value "0" in this step.
- **Step 2:** Switch to functional mode and generate vector V_2 such that A and B are both "0".
- **Step 3:** Operate in functional mode and generate vector V_3 such that in this step, the values on A and B are "1" and "0", respectively. This step ensures that a transition is launched at-speed across the CDC.
- **Step 4:** Operate in functional mode and generate vector V_4 such that B gets the value "1".

If $k > 0$ synchronizers are placed between the sender and receiver flip-flops, then Step 2 involves one cycle of functional clock Clk_1 but $k+1$ cycles of functional clock Clk_2. Steps 3-4 remain unchanged.

The S-CDC fault modeled by a slow-to-rise fault on signal *B* can be detected by applying vectors V_1 to V_4 (as discussed above) in four consecutive clock cycles. During scan mode (Step 1), a common shift clock signal is applied to both sender and receiver domains but in Steps 2-4, the circuit operates in functional mode and we apply Clk_1 and Clk_2 to the first and second clock domains, respectively. No assumptions are made or restrictions are placed on the clocking scheme. The clock signals are fed either by different PLL sources, or by a common PLL source but with different phases and frequencies. The proposed ATPG method is also applicable when data arrives to the receiver domain within an upper limit of n clock cycles (instead of 1 clock cycle shown in Fig. 6). In this case, we can test for S-CDC faults by applying *n* functional clock cycles using Clk_2 and use a transition detector to record a transition on *B* within the window of *n* clock cycles.

To implement CoTC, we leveraged a commercial ATPG tool. First, full-scan insertion is performed. Next, pairs of flip-flops residing in clock boundaries (in different clock domains) are extracted. Finally, test generation is performed under the constraints discussed in Section V.

CoTC requires that the CDC flip-flops get specific values in four consequent clock cycles. However, commercial ATPG tools cannot be directly used to generate test patterns such that all of these requirements are met simultaneously. Therefore, to generate test patterns that satisfy the CoTC requirements for a S-CDC fault, we first expand the circuit in time and then use a commercial ATPG tool to generate test patterns targeting that fault in the time-expanded model of the circuit.

To implement CoTC with one launch and three capture cycles, we triplicate the combinational logic of the circuit under test and then use the triplicated version of the circuit for test generation. The values that should be considered for each pair of boundary flip-flops in four consecutive clock cycles in CoTC, provided as constraints for each time frame.

VI. Experimental Results And Analysis

First we provide details of the simulation setup used to evaluate the effectiveness of CoTC. Then we present results for the IWLS'05 benchmarks, and highlight some key observations.

A. Experimental Setup

We applied CoTC to five IWLS'05 benchmarks that contain multiple clock domains. They are the WISHBONE AC 97 Controller (ac97_ctrl), the WISHBONE Memory Controller (mem_ctrl), the USB function core (usb_funct), the Ethernet IP core (ethernet), and the WISHBONE rev.B2 compliant Enhanced VGA/LCD Controller (vga_lcd) [21]. Software to perform scan insertion, CDC-path extraction, replication, selection of the final test patterns, and evaluating the results were all implemented using Python. A commercial ATPG tool was used for test generation. As indicated in our results, the ATPG tool reported a number of S-CDC faults to be untestable (or redundant).

To generate a test-pattern set that detects TDFs as well as S-CDC faults, top-off ATPG was performed after applying CoTC to meet the fault coverage requirement for TDFs. The final pattern set for our procedure therefore includes the CoTC-generated patterns and the top-off ATPG patterns.

All experiments were performed on a dual-processor Xeon quad-core Intel server running at 2.53 GHz with 64 GB of memory. CPU time for CoTC was estimated by aggregating the times needed for the different steps. For the test cases in this paper, the runtimes per fault ranged from a few seconds to three minutes.

B. Experimental Results

1) Benchmark Statistics: Details of the IWLS'05 benchmark circuits used in this paper are shown in Table II. The benchmarks represent a wide range of application areas, including memory controllers and IP cores. The ethernet benchmark has three clock domains, and all other benchmarks have two clock domains each. Note that in our experiments, we only considered slow-to-rise S-CDC faults. We expect to get similar results for slow-to-fall S-CDC faults without any change in methodology.

TABLE II
BENCHMARKS STATISTICS.

Benchmark	# Clock domains	# Flip-Flops	# Gates
ac97_ctrl	2	2,199	28,083
mem_ctrl	2	1,083	22,015
usb_funct	2	1,746	25,531
ethernet	3	10,544	153,948
vga_lcd	2	17,079	252,302

2) Detected S-CDC Faults: For each benchmark circuit, we first extracted all CDC paths of the circuit and then for each pair of the CDC flip-flops, we generated test patterns by applying CoTC to the time-expanded model of the circuit under test. The third column of Table III shows the number of testable S-CDC faults in each benchmark circuit. The fourth column of this table shows the number of slow-to-rise S-CDC faults detected by CoTC for each benchmark circuit.

To evaluate the number of S-CDC faults detected by the baseline LoC/TDF method, we used a commercial ATPG tool to generate test patterns detecting all slow-to-rise TDFs for that benchmark. Then, the subset of the generated patterns that satisfied the constraints of CoTC scheme was extracted, and the number of S-CDC faults detected by these vectors were reported (fifth column of Table III).

TABLE III
COMPARING COTC AND TRADITIONAL LOC SCHEMES IN TERMS OF S-CDC FAULT DETECTION.

Benchmark	# S-CDC faults	# Testable S-CDC faults	# Detected by CoTC	# Detected by LoC/TDF
ac97_ctrl	902	897	897	121
mem_ctrl	3,354	2,613	1,631	167
usb_funct	1,592	1,116	1,060	193
ethernet	4,862	643	529	391
vga_lcd	3,187	3,085	3,085	678

For the benchmark circuits considered in this paper, on average, the test patterns generated by CoTC can detect 88% of detectable S-CDC faults. We expect the fault coverage to be even higher since many faults that are aborted by the ATPG tool are most likely to be untestable. However, only 24% of the S-CDC faults can be detected using baseline LoC/TDF.

3) Detected slow-to-rise transition faults: We next compare the number of slow-to-rise TDFs detected by LoC/TDF to the corresponding number for CoTC with top-off ATPG. The results are shown in Table IV. The number of slow-to-rise TDFs detected by the traditional LoC method, is nearly equal to the number of transition faults detected by COTC and top-off ATPG. Therefore, the proposed method provides the same coverage for TDFs as the baseline LoC/TDF method, but with a significantly higher coverage of CDC faults.

TABLE IV
DETECTED SLOW-TO-RISE FAULTS.

Benchmark	# Slow-to-rise faults	# Detected by LoC/TDF	# Detected by CoTC + top-off ATPG
ac97_ctrl	40,916	37,154	37,140
mem_ctrl	38,086	17,266	17,482
usb_funct	40,108	34,718	34,850
ethernet	160,454	152,098	152,090
vga_lcd	382,927	317,092	317,074

4) Test Pattern Count: The fourth set of results compares the number of test patterns generated by LoC/TDF to the number of test patterns generated by CoTC with top-off ATPG. As shown in Table V, on average, for each circuit, the number of test patterns generated by CoTC with top-off ATPG is only 25% more than the patterns generated by using baseline LoC/TDF method. Therefore, higher test quality is attained with only a slight increase in test pattern count.

TABLE V
COMPARISON OF NUMBER OF TEST PATTERNS.

Benchmark	LoC/TDF	CoTC	Top-off ATPG	CoTC + top-off ATPG	% increase
ac97_ctrl	1,591	412	1,468	1,880	18
mem_ctrl	1,094	846	979	1,825	66
usb_funct	2,414	576	2,107	2,683	11
ethernet	10,095	291	9,715	10,006	-1
vga_lcd	11,335	3,083	11,549	14,632	29

VII. CONCLUSIONS

We have quantified the impact of process variations on CDC faults at clock-domain boundaries in multi-core SoCs. The results demonstrate that ATPG for post-silicon testing of CDC faults is necessary, even when synchronizers and design validation methods are used. We have presented fault models to represent the incorrect behavior in the presence of CDC faults, and based on these fault models, we have described a test generation method for detecting CDC faults. Experimental results for IWLS'05 benchmark circuits with multiple clock domains highlight the effectiveness of the proposed method for detecting CDC faults, while demonstrating the shortcomings of commercial transition-delay fault APTG tools.

VIII. ACKNOWLEDGMENT

The authors thank Andy Ni from Duke University for his help with experiments.

REFERENCES

[1] Y. Feng, Z. Zhou, D. Tong, and X. Cheng, "Clock domain crossing fault model and coverage metric for validation of SoC design," in *Proc. Design Automation & Test in Europe Conf.*, 2007, pp. 1–6.

[2] R. Ginosar, "Fourteen ways to fool your synchronizer," in *Proc. Intl. Symp. Asynchronous Circuits and Systems*, 2003, pp. 89–96.

[3] M. Cole and D. Cohen, "Staying in sync," *Electronics*, vol. 5, no. 3, pp. 42–45, June-July 2007.

[4] "Clock domain crossing - Closing the loop on clock domain function implementation problems," Cadence Design Systems, Tech. Rep., 2004, "http://w2.cadence.com/whitepapers/cdc_wp.pdf" (last accessed 9 June, 2011).

[5] N. Hand, "The need for an automated clock-domain crossing verification solution," Mentor Graphics, Tech. Rep., May 2006, "http://www.mentor.com/products/fv/techpubs/emulation-systems-f1fc6a19-9e95-4fd0-8d84-d5e7cf0fc12a-dt?selid=28966" (last accessed 9 June, 2011).

[6] S. Sarwary and S. Verma, "Critical clock-domain-crossing bugs," *Electronics Design, Strategy, News*, pp. 55–60, Apr. 2008.

[7] C. Kwok, V. Gupta, and T. Ly, "Using assertion-based verification to verify clock domain crossing signals," in *Proc. Design and Verification Conf.*, 2003, pp. 654–659.

[8] T. Kapschitz and R. Ginosar, "Formal verification of synchronizers," in *Correct Hardware Design and Verification Methods*, ser. Lecture Notes in Computer Science, D. Borrione and W. Paul, Eds. Springer, 2005, vol. 3725, pp. 359–362.

[9] N. Karimi, Z. Kong, K. Chakrabarty, P. Gupta, and S. Patil, "Testing of clock-domain crossing faults in multi-core system-on-chip," in *Proc. Asian Test Symp.*, 2011, pp. 7–14.

[10] R. Ginosar, "Metastability and synchronizers: A tutorial," *IEEE Trans. on Design & Test of Computers*, vol. 28, no. 5, pp. 23–35, Sep. 2011.

[11] J. M. Bassam, "Zero setup time flip-flop," U.S. Patent 5867049, Feb. 2, 1999.

[12] K. K. Kwok, B. Li, T. A. Ly, and R. R. Sabbagh, "Formal verification of clock domain crossings," U.S. Patent 20100199244, August 5, 2010.

[13] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, "Digital integrated circuits, A design perspective (Second Edition)," Prentice Hall Publishers, 2003.

[14] B. Abramov, "Clock domain crossing," "http://www.abramovbenjamin.net/malas/l9.pdf" (last accessed 19 Aug, 2011).

[15] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm early design exploration," *IEEE Trans. on Electron Devices*, vol. 53, no. 11, pp. 2816–2823, Nov. 2006.

[16] A. Murakami et al., "Selection of potentially testable path delay faults for test generation," in *Proc. Intl. Test Conf.*, 2000, pp. 376–384.

[17] P. Bemardi et al., "On the Automatic Generation of Test Programs for Path-Delay Faults in Microprocessor Cores," in *Proc. European Test Symp.*, 2007, pp. 179–184.

[18] H. Hengster, R. Drechsler, and B. Becker, "On Local Transformations and Path Delay Fault Testability," *Journal of Electronic Testing: Theory and Applications*, vol.7, no. 3, pp. 173–191, Dec. 1995.

[19] Q. Xu and N. Nicolici, "Delay fault testing of core-based systems-on-a-Chip," in *Proc. Design Automation & Test in Europe Conf.*, 2003, pp. 744–749.

[20] F. Wu et al., "Analysis of power consumption and transition fault coverage for LOS and LOC testing schemes," *Proc. Design and Diagnostics of Electronic Circuits and Systems Symp.*, 2010, pp. 376–381.

[21] C. Albrecht, "IWLS 2005 benchmarks," in *Proc. Intl. Wksp. Logic Synthesis*, 2005.