An On-Chip Test Clock Control Scheme for Multi-Clock At-Speed Testing

1,2Xiao-Xin FAN, 1Yu HU, 3Laung-Terng (L.-T.) WANG

1Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences
2Graduate School of Chinese Academy of Sciences, Beijing, 100049
3SynTest Technologies, Inc., 505 S. Pastoria Ave., Suite 101, CA 94086, USA
{fanxiaoxin, huyu}@ict.ac.cn, wang@syntest.com

Abstract

To test timing-related faults between synchronous clocks, an at-speed test clock and an automatic test pattern generation scheme are needed. However, previous work on designing on-chip at-speed test clock controllers for multi-clock has quadratic increasing area overhead along with linearly increasing clocks. This paper presents a clock-chain based test clock control scheme using an internal phase-locked-loop (PLL) as the at-speed test clock generator, which supports at-speed testing for inter-clock domain and intra-clock domain logic. Experimental results demonstrate that the proposed design has low area overhead when increasing the number of clocks.

1. Introduction

Nowadays, very deep submicron processes are widely utilized to design and fabricate integrated circuits, which results in an increasing number of timing-related defects. Conventional test techniques, such as stuck-at fault testing supplemented with \( \frac{I_{DDQ}}{2} \) (Direct Drain Quiescent Current) testing, are inefficient to screen out the timing-related defects in small geometry size (e.g. 90nm) [1]. Therefore, at-speed testing of transition faults and path-delay faults is emerging as the necessary techniques for testing high performance circuits.

There are two ways to provide the at-speed test clock signal: from an external ATE (Automatic Test Equipment) or from an internal PLL. If using an external ATE to generate the high frequency test clock, then either the cost of ATE or the cost of circuit package is prohibitive, especially for circuits running at GHz. An alternative way is implementing a clock control design in the circuit so that the at-speed testing can be conducted by a low-speed ATE. The basic idea of the clock control is to use on-chip clock source, such as PLL or DLL, to provide at-speed test pulses, while the ATE provides shift pulses and test control signals at slow speed. On-chip test clock generation is economic thereafter is utilized in many industry designs [2-3].

Moreover, to improve the flexibility, designs with multiple clocks become more and more popular. Most system-on-chip (SoC) designs have multiple function components and various peripheral interfaces. The components and the interfaces following different standards may operate at different frequency. For example, the Intel® IXP425 network processor, which is widely used in communication systems, has a processor running at 533 MHz, three network processor engines running at 133 MHz, and variety of interfaces running at various frequencies [4]. This multi-clock trend makes a challenge for at-speed testing. Previous works mainly focus on the single clock domain, which is inefficient to test the timing-related faults between clocks. Ignoring these faults between clocks will reduce the test quality, which is critical to reliability, and become unacceptable. Therefore, it is very urgent to design a clock control scheme to support at-speed testing for detecting the faults in either the inter-clock logic or the intra-clock logic.

Many methods were proposed to address this issue. [5] and [6] respectively proposed an at-speed testing architecture for multi-clock designs based on logic built-in self-test (BIST). [7] presented a control scheme for inter-clock at-speed testing. This controller may efficiently test the timing-related faults between clocks but need additional logic to support intra-clock at-speed testing, thus the area overhead is increased. Besides, the scheme can generate only one type of test
clock pairs which means the test clock control scheme is not flexible to support efficient ATPG (Automatic Test Pattern Generation) techniques.

We implemented a clock-chain based clock control scheme in an industry design running at 1 GHz, and the results showed it was efficient to test the delay faults in the intra-clock domain [8]. Based on the previous work, we propose a new control scheme for multi-clock at-speed testing to generate various test clock sequences for both inter-clock and intra-clock at-speed testing while keep low area overhead when increasing the number of clocks. The rest of the paper is organized as follows: Section 2 introduces the background. In section 3, the new control scheme for multi-clock at-speed testing is presented in detail. Experimental results are shown in section 4. Finally, section 5 concludes the paper.

2. Background

2.1 At-speed testing methodology

The faults detected during at-speed testing are usually path-delay faults and transition faults. The path-delay fault model measures the cumulative effect of delay defects along a specific combinational path in the circuit. The transition fault model is used to detect large slow-to-rise or slow-to-fall defects at every site in the circuit.

Since a delay test launches a transition and propagates it across a certain path and captures the response at the end-point of the path, a pair of at-speed launch and capture pulses is needed to apply a delay test. There are two approaches to generate these pulses, one is launch-off-shift (LOS) and the other is launch-off-capture (LOC) [9], as shown in Fig.1.

![Fig. 1 At-speed testing approaches](image)

(a) Launch-off-shift

(b) Launch-off-capture

Fig. 1 At-speed testing approaches

The waveform of LOS method is shown in Fig.1 (a). During the shift phase, the scan enable signal (SE) keeps at active state, a test vector is shifted into the circuit by toggling scan chains at low frequency. After the test vector is launched by the last shift cycle, SE goes into inactive state immediately, and an at-speed pulse is applied to capture the response. Thus, SE is similar to a clock signal, which brings difficulty in physical design.

On the other hand, as shown in Fig.1 (b), the LOC approach uses a pair of at-speed pulses in functional mode when SE is inactive. Comparing with LOS, the timing constraint of SE in LOC is much slacker. Therefore, LOC is utilized in this work to ensure the test clock control logic is easy to implement.

2.2 Multiple clock at-speed testing

For the circuit designed with multi-clocks, there are communication data-paths between logic blocks of two clocks. For example, an AHB (Advanced High-performance Bus)-PCI (Peripheral Component Interconnect) bridge circuitry has two clock domains: one is the AHB clock domain and the other is the PCI clock domain. To reliably transfer data from one clock domain to the other clock domain, the two clocks are synchronous; otherwise, handshake signals are needed. The definitions of synchronous and asynchronous are as follows [10]:

- **Synchronous**: A clock and its inverted clock or its derived divided-by-two clocks are synchronous.
- **Asynchronous**: Clocks with no constant phase and time relationships are asynchronous.

Because timing in asynchronous circuits is not strict, multi-clock at-speed testing usually concentrates on the interaction region between synchronous clocks. Therefore, considering the location of the faults, the timing-related faults can be classified as inter-clock faults and intra-clock faults. We use the inter-clock logic block and intro-clock logic block concepts defined by [7]:

- **Intra-clock block**: The combination logic block exists between flip-flops driven by the same internal clock.
- **Inter-clock block**: The combination logic block exists between flip-flops driven by two synchronous internal clocks.

3. The Proposed Test Clock Control Scheme

3.1 Basic Concept

The basic idea of at-speed testing is to launch a transition at the start-point of a path and capture the response at the end-point. As mentioned above, we
adopt the LOC approach to avoid a timing-critical scan enable signal. Fig. 2 shows the basic concept of at-speed testing for inter-clock domain and intra-clock domain.

**Fig. 2 Multi-clock at-speed testing methodology**

In Fig. 2 (a), the flip-flops $FF0$ and $FF1$ are driven by the same clock. The test clock waveform underneath illustrates how at-speed testing is conducted to detect the timing-related faults in intra-clock domain between two flip-flops. The interval ($d_0$) between the launch pulse and capture pulse equals to the clock period in functional mode. Comparing with the intra-clock at-speed testing, the inter-clock at-speed testing clock waveform is more complex, as shown in Fig. 2 (b). The launch and capture interval depends on the direction of the data-path during communication. For instance, if the data is transferred from $FF0$ to $FF1$, we need to launch a transition at $FF0$ and capture the response at $FF1$. Similarly, if the data-flow is $FF1 \to FF0$, we may need a $TCK1$ launch pulse and a $TCK0$ capture pulse. Interval $d_0$ reflects the required time for transferring data from $FF1$ to $FF0$, while $d_1$ reflects the required time for transferring data from $FF0$ to $FF1$. Both $d_0$ and $d_1$ are often defined by the circuit designer according to the design specification.

For a multi-clock design, at-speed testing may need various types of launch-capture pairs. Table 1 gives an example of multi-clock at-speed testing waveforms in the case of two synchronous clocks with bidirectional data-paths. Follows are the explanation of the waveform types:

(a). Test clock is off, for testing other clock domains.
(b). Test $TCK0$ intra-clock logic block.
(c). Test $TCK1$ intra-clock logic block.
(d). Test inter-clock logic block, from $TCK0$ to $TCK1$.
(e). Test inter-clock logic block, from $TCK1$ to $TCK0$.

<table>
<thead>
<tr>
<th>Waveform Type</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>(b)</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>(c)</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>(d)</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>(e)</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

**Table 1: Example of multi-clock at-speed testing**

3.2 Detailed Scheme

Fig. 3 shows the general architecture of the proposed test clock control scheme.

**Fig. 3 Architecture of the test clock control scheme**

The proposed test clock control logic is within the dashed line. It cooperates with the external ATE and the internal clock source (e.g. PLL) to provide at-speed testing clock cycles for the circuit-under-test. The ATE controls the test related signals, such as scan in ($SI$), scan enable ($SE$), shift clock ($SCK$) and test mode ($TM$). When $SE$ is asserted, the circuit-under-test is operating in shift mode, and the clock generator unit selects the $SCK$ to let test patterns be shifted with a low speed. In shift mode, both $TCK0$ and $TCK1$ are driven by $SCK$. Whenever $SE$ is deasserted, the circuit-under-test is operating in capture mode. In capture mode, the clock generator unit produces at-speed test clock which is derived from the PLL. The type of the test clock...
depends on the content of clock chain unit. In the following, details of every part in the test control logic will be described.

**Clock Chain Unit:** The clock chain unit in Fig. 3 consists of an n-bit shift registers \((SFF_n)\) driven by the scan clock. Fig. 4 shows the detailed implementation. The clock chain unit can be part of other regular scan chains to save scan ports.

The length of the clock chain is depending on the number of the synchronous clocks and types of the test clock. For example, if there are two synchronous clocks with bidirectional data-paths in the design, the at-speed testing has five types of test clock waveforms, e.g. the waveforms shown in Table. 1. In that case, the clock chain may need three shift registers to provide five types of test clock waveforms. The content stored in the clock chain indicates the type of test clock waveform. By filling the shift registers of clock chain, ATPG tools can decide which type of launch-capture pairs need to be generated.

**N-stage Delay Register:** As shown in Fig. 5, the n-stage delay register consists of n flip-flops which are driven by a clock derived from the PLL. Flip-flops \(FF_0-FF_n\) are used to provide long enough delay time for transition of \(SE\). The actual delay time is depending on the timing requirement of the circuit specification. In our experiments, since the function frequency is 500 MHz, we use a 10-stage delay line to provide 18 ns delay \((9 \times TCK0, \text{ where } TCK0 \text{ is the period of } CK0)\), thus making the \(SE\) no-timing-critical. Notice that the flip-flop \(FF_a\) is toggled by the negative edge of \(SCK\), which is used for preventing metastability.

**Clock Generator:** The clock generator is designed to create various types of test clock according to the content of the clock chain unit. Fig. 6 shows the detailed structure of the clock generator unit.

The counter in clock generator is an n-bit gray-code counter with high-speed and non-metastability features. The purpose of the counter is to count the number of fastest clock pulses. The delayed scan enable signal \((d_{SE})\) controls the counter, if \(d_{SE} = 0\), the counter starts to count, if \(d_{SE} = 1\), the counter stops. Also if the counter reaches its maximum value, it will stop counting. Consequently when the ATE finished shifting the test pattern, the counter will start after a given delay. The bits of the counter depend on the ratio between the fastest clock \((CK0)\) and the slowest clock \((CK1)\) in the design. For example, if the clock ratio is 4, which means the \(CK0\) is 4 times faster than the \(CK1\). Then if ATPG needs 2 successive \(CK1\) pulses, the counter must count to 8 to obtain 2 \(CK1\) cycles, thus the length of the counter is at least 3 bits.

The enable signals \(clk0_{en}\) and \(clk1_{en}\) are determined by the content of the clock chain and the current counter value. As shown in Table 1, for example, if the ATPG needs the type of (b) test waveform, then the clock chain will be filled with “001”. When the value of the counter reaches 2, the enable generator asserts the signal \(clk0_{en}\), and when 4, the enable generator cancels the \(clk0_{en}\) signal. As a result, the clock gating cell will let only two at-speed \(CK0\) pulses pass.

The test flow is described as follows and the corresponding waveform is shown in Fig. 7.

1) **Shift test pattern:** The ATE shifts test pattern through scan chains, including clock chain. In this phase, the counter is inactive.

2) **Delay scan enable:** Delay the slow scan enable signal to ensure it is efficiently fall down.

3) **Start Counter:** Start to count the \(CK0\) pulse.

4) **Generate clock enable signals:** Generate \(clk0_{en}\) and \(clk1_{en}\) signals according to the counter’s value and the content of the clock chain. After then the at-speed clocks are created to launch a transition and capture the response.

5) **Stop Counter:** The counter will stop when the
counter reaches its maximum value or the scan enable goes up.

Fig. 7 Timing diagram of the clock generator

For example, consider testing the inter-clock block faults in which the direction of data-path is from CK0 to CK1. Then a CK0 launch pulse followed by a CK1 capture pulse should be applied to conduct at-speed testing, as shown in Table 1 (d). Assuming the timing requirement for data transferring from CK0 to CK1 is the period of CK0. The clock chain is shifted in with ‘II’, thus means choosing the fourth type of testing clock. After the counter counting the first pulse of CK0, the enable generator provides the clock enable signals, thus the launch-capture pairs are created, as the Fig. 7 shows.

We will compare our work with the method proposed in [7]. Assuming a design contains \( N \) synchronous clocks, and each two clocks have a bidirectional data-path, then the number of test clock types is:

\[
\text{number} \_\text{clock} \_\text{types} = N + N(N-1) + 1 = N^2 + 1
\]

including \( N \) intra-clock domains, \( N(N-1) \) inter-clock domains and the clock off state. In [7]’s scheme, every inter-clock block needs an inter-clock enable generator, thus the number of flip-flops increases at the \( O(N^2) \) speed. While in this work, the number of flip-flops is determined by the clock chain length and the number of clock enable registers in the clock generator module, which means the number of the flip-flops will increase at a linear speed.

Fig. 8 gives a comparison on flip-flops consumption between the two schemes. The X axis presents the number of clock blocks, while Y axis shows the number of flip-flops used by the test clock controller. We can see that when the number of clock blocks reaches five, the number of flip-flops of the proposed scheme is ten times less than the scheme in [7].

4. Experimental Results

The proposed test control scheme is applied to three experimental circuits to validate the efficiency. These circuits consist of ISCAS’89 benchmark circuits. We integrate two S38417 in circuit 1, three in circuit 2 and four in circuit 3. Each S38417 operates with synchronous clocks in different frequency. And we manually connect some primary outputs of one S38417 to some primary inputs of another S38417 by inserting flip-flops between them to construct bidirectional data-path, and thus to build inter-clock logic blocks. These circuits are synthesized by a commercial synthesis tools in 0.18um process.

The statistics of the experimental circuits are shown in Table 2. The first row gives the circuit names. Row “Num. Clocks” shows the number of clocks in each experimental circuit. In the entry of “Inter & Intra-Logic Blocks”, the number of total logic blocks in each of these circuits is presented, which includes both the intra-logic blocks and the inter-logic blocks.

<table>
<thead>
<tr>
<th>Circuit Statistics</th>
<th>Circuit #1</th>
<th>Circuit #2</th>
<th>Circuit #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Clocks</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Inter &amp; Intra-Logic Blocks</td>
<td>4</td>
<td>9</td>
<td>16</td>
</tr>
</tbody>
</table>

Fig. 9 shows the comparison of the area overhead between the scheme proposed in [7] and our scheme. The area overhead is equivalent to 2-input NAND gates. We can see our design has lower area overhead than the scheme in [7]. Meanwhile, it is clear that Fig 9 is similar to Fig 8, which confirms that the area overhead is mainly determined by the number of flip-flops. In fact, the area overhead of the enable generator will also increase along with more flip-flops, but in a slower speed.

Moreover, besides the benefit of the lower area overhead, the new test control scheme can provide many types of waveform so that the number of test patterns may be reduced. For example, one pattern may simultaneously detect some faults in CK0 domain and CK0->CK1 domain. In that case, both CK0->CK0 and
CK0>CK1 launch-capture pairs are need to apply simultaneous at-speed test. In our future work, we will combine the proposed test clock control scheme with an ATPG tool [12] to verify at-speed test patterns generated under the proposed scheme can be reduced.

Fig. 9 Experimental comparison on area overhead

5. Conclusions

This paper proposes a new test control scheme to provide multi-clock at-speed testing. This scheme can generate various types of test clock waveforms for at-speed testing inter-clock blocks and intra-clock blocks. Theoretical analysis shows the proposed scheme has lower area overhead than that of previous work. Meanwhile, experimental results also demonstrated the advantage.

6. Acknowledgement

This paper was supported in part by National Basic Research Program of China under Grant No. 2005CB321604 and 2005CB321605, and in part by National Natural Science Foundation of China under Grant No. 60633060 and 60606008. Also the authors would like to thank Prof. Huawei Li and Prof. Xiaqing Wen for many helpful suggestions to the work. And the helps of SynTest colleagues Paul Hsu, Johnson Guo and Xiangfeng Li are gratefully appreciated.

Reference