Power-Constrained Test Scheduling for Multi-Clock Domain SoCs

Tomokazu Yoneda†, Kimihiko Masuda† and Hideo Fujiwara†
†Graduate School of Information Science, Nara Institute of Science and Technology
Kansai Science City, 630-0192, Japan
{yoneda, kimihi-m, fujiwara}@is.naist.jp

Abstract

This paper presents a wrapper and test access mechanism design for multi-clock domain SoCs that consists of cores with different clock frequencies during test. We also propose a test scheduling algorithm for multi-clock domain SoCs to minimize test time under power constraint. In the proposed method, we use TDM (Test Data Multiplexing) to solve the frequency gaps between cores and the ATE. We utilize the TDM to reduce power consumption of a core during test while maintaining the test time of the core. Experimental results show the effectiveness of our method not only for multi-clock domain SoCs, but also for single-clock domain SoCs with power constraints.

keywords: *multi-clock domain SoC, test scheduling, test access mechanism, power consumption, test data multiplexing*

1 Introduction

The systems-on-chip (SoC) design strategies help us to reduce the time-to-market and design cost for new products significantly. However, testing of SoC is a crucial and time consuming problem due to the increasing design complexity[1]. Therefore, the goal is to develop techniques for wrapper design, test access mechanism (TAM) design and test schedule that minimizes test application time under given constraints such as the number of test pins and power consumption. A number of approaches have addressed wrapper design [2, 3] which are IEEE P1500 [4] compliant. Similarly, several TAM architectures have been proposed such as *TestBus* [5, 6], *TESTRAIL* [7], *transparency based TAMs* [8, 9, 10]. However, wrapper and TAM co-optimization problem was shown to be NP-hard in [2]. Therefore, many heuristic approaches for this problem have been proposed [11, 12, 13, 14, 15, 16].

However, these previous approaches are applicable only to single-clock domain SoCs that consist of embedded cores working at the same clock frequency during test. Today's SoC designs in telecommunications, networking and digital signal processing applications consist of embedded cores working with different clock frequencies. The clock frequency of some embedded cores during test is limited by its scan chain frequencies, typically under 50 MHz. On the other hand, other cores may be testable at-speed in order to increase the coverage of non-modeled and performance-related defects. Consequently, we can consider that test frequency of a core is different from other cores in such multi-clock domain SoCs. Moreover, there also exists a frequency gap between each embedded core and ATE used to test the SoC. From this facts, we conclude that the previous approaches have the following two problems: 1) in the case when clock frequency of a core is higher than that of ATE, they cannot achieve at-speed test, and 2) in the case when clock frequency of a core is lower than that of ATE, testing of a core by lowering the frequency of ATE does not make use of ATE capability effectively. Therefore, it is necessary to develop techniques for such multi-clock domain SoCs.

Recently, a wrapper design for cores with multiple clock domains was proposed in [17] to achieve at-speed testing of the cores. In [19], *Virtual TAM* based on bandwidth matching [18] has been proposed to increase ATE capability when the clock frequency of a core is lower than that of ATE. However, the test scheduling problem for multi-clock domain SoCs was not addressed in these literatures.

To the best of our knowledge, this paper gives a first discussion and a formulation of the test scheduling problem for multi-clock domain SoCs. Moreover, we present a wrapper and TAM design for multi-clock domain SoCs and propose a test scheduling algorithm to minimize test time under power constraint. In the proposed method, we use TDM (Test Data Multiplexing) technique based on bandwidth matching [18] to solve a frequency gap between each core and a given ATE. We also present a technique to reduce power consumption of a core during test while controlling the test time by utilizing TDM technique. Therefore, our approach is applicable to multi-clock domain SoCs and is effective for power-constrained test scheduling. Experimental results show the effectiveness of our method not only for multi-clock domain SoCs, but also for single-clock domain SoCs with power constraints.

The rest of this paper is organized as follows. We discuss power consumption model and multi-clock domain SoCs in Section 2. Section 3 shows a power-conscious TDM technique. After formulating a test scheduling problem for multi-clock domain SoCs in Section 4, we present a power-constrained test scheduling algorithm in Section 5. Experimental results are discussed in Section 6. Finally, Section 7 concludes this paper.

2 Preliminaries

2.1 Power Consumption

Power consumption in CMOS circuits can be classified into two categories: static power and dynamic power. Static power dissipation is caused by leakage or other current drawn continuously from the power supply. On the other hand, Dynamic power dissipation is caused by output switching. For the current CMOS technology, dynamic power is the dominant source of power consumption. The power $P(k)$ consumed in the circuit on application of consecutive two test vectors (V_{k-1}, V_k) is as follows [21].

$$P(k) = 1/2 \cdot f \cdot V_{DD}^2 \cdot \sum C_i \cdot S_i(k) \tag{1}$$

Here, f is the clock frequency, V_{DD} is the power supply voltage, C_i is the output capacitance at node i and $S_i(k)$ is the number of switchings provoked by V_k at node i. By using this equation, the average power consumption P_{ave} and the peak power consumption P_{peak} during test can be represented as follows.

$$P_{ave} = \sum_{k=1}^{n} P(k)/n \tag{2}$$

$$P_{peak} = \max_k(P(k)) \tag{3}$$

Here, n is the number of test patterns. High average power consumption causes structural damage to the silicon, bonding wires or package. And if peak power consumption exceeds a certain limit, designers cannot guarantee that the entire circuit will function correctly. According to these equations, both the average and the peak power consumption are proportional to the clock frequency f. In the rest of this paper, we do not distinguish between average power and peak power since the proposed method can deal with both powers as a constraint.

2.2 Multi-Clock Domain SoCs

This section describes the formal notation we use to model the multi-clock domain SoC under test. An example of an SoC is shown in Figure 1 where each core is wrapped to ease test access. Test pattern source and test response sink are implemented off-chip as an ATE. The SoC can be modeled as a multi-clock domain SoC, $MCDS = (C, R, P_{max})$, where:

$C = \{c_1, c_2, ..., c_n\}$ is a set of cores;

Each core c_i is characterized by:

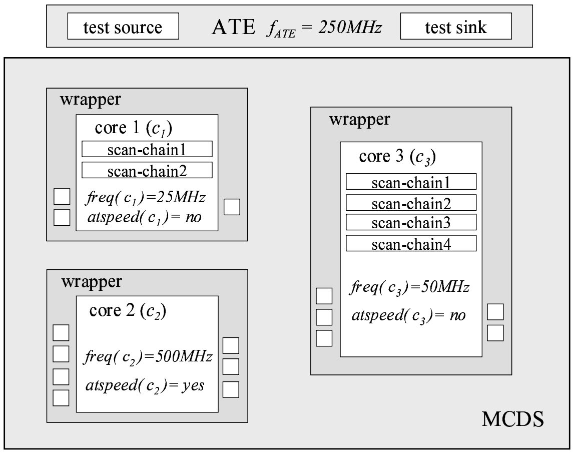

Figure 1. Multi-clock domain SoC.

- $freq(c_i)$: maximum test frequency of core $c_i \in C$;
- $power(c_i)$: power consumption of core $c_i \in C$ at test frequency $freq(c_i)$;
- $atspeed$: $C \rightarrow \{yes, no\}$: at-speed test requirement

$R = \{R_1, R_2, ..., R_n\}$ is a set of wrapper lists;

Each wrapper list R_i is characterized by:

- $R_i = \{r_{i1}, r_{i2}, ..., r_{ij}\}$ is a set of wrapper designs for core c_i;
- Each wrapper design r_{ij} is characterized by:
  - $pin(r_{ij})$: number of pins to test core c_i with j-th wrapper design ;
  - $cycle(r_{ij})$: number of clock cycles to test core c_i with j-th wrapper design;

P_{max}: maximum allowed power at any time;

We consider that an SoC consists of the maximum allowed power consumption and cores working at different test frequencies. However, we assume that each core has been designed with single-clock domain during test. For each core, a maximum test frequency and a power consumption at the given maximum frequency are given. Each core also has an information about the requirement of at-speed testing. $atspeed(c_i) = yes$ means that c_i must be tested at $freq(c_i)$ (i.e. we cannot change the test frequency of c_i for test scheduling). $atspeed(c_i) = no$ means that c_i can be tested at lower frequencies than $freq(c_i)$ (i.e. we can decrease the test frequency of c_i for test scheduling). Moreover, each core has a wrapper list that consists of possible wrapper designs for the core. Each wrapper design has a number of test pins and a number of clock cycles required to test the core with the wrapper design. The test time for c_i working at $freq(c_i)$ can be calculated as $cycle(c_i)$ / $freq(c_i)$.

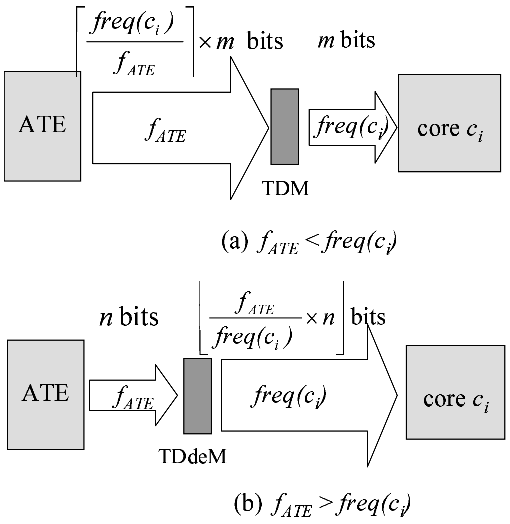

Figure 2. Test data multiplexing/de-multiplexing.

3 Power-Conscious Test Data Multiplexing

In multi-clock domain SoCs, there exists a clock frequency gap between each core and an ATE. The test frequencies of some embedded cores are limited by its scan chain frequencies, typically under 50 MHz. On the other hand, other cores may be able to be at-speed testable in order to increase the coverage of non-modeled and performance-related defects.

This frequency gaps between ATE and cores can be solved by using TDM (test data multiplexing) techniques based on bandwidth matching [18, 19]. When $freq(c_i)$ (clock frequency of core c_i during test) is higher than f_{ATE} (clock frequency of ATE) (Fig. 2(a)), we insert a TDM circuit between ATE outputs and the core inputs, and multiplex $\lceil freq(c_i)/f_{ATE} \rceil \cdot m$ bits test data at f_{ATE} into m bits test data at $freq(c_i)$. On the other hand, when $freq(c_i)$ is lower than f_{ATE} (Fig. 2(b)), we insert a TDdeM(test data de-multiplexing) circuit between ATE output and the core inputs, and de-multiplex n bits test data at f_{ATE} into $\lfloor n \cdot f_{ATE}/freq(c_i) \rfloor$ bits test data at $freq(c_i)$. To observe test responses, we need to insert TDM/TDdeM between the core output and ATE inputs in the similar fashion.

In this paper, we also utilize this TDM technique to reduce power consumption of a core while keeping test time of the core. From equation(1), we observe that the power consumption of a core during test can be reduced by lowering its test frequency. However, this causes test time increase which is proportional to the power reduction ratio. Here, we insert TDdeM circuit between the ATE outputs and the core inputs. Then, more test pins become available for the core. Therefore, test time can be reduce by replacing the wrapper design to another one while keeping the power reduction ratio because we can consider that the number of switchings is the same in both wrapper designs (i.e., all scan chains are active simultaneously in both designs).

Table 1. An example of power-conscious TDM for core7 in d695.

frequency(MHz)	# wrapper pins	test time(μs)	# cycles
50	10	264.86	13243
25	20	268.68	6717

For example, let c_i be a core and r_{ij} be a wrapper design to test the core, and let the $freq(c_i)$ be equal to f_{ATE} and c_i be tested at frequency $freq(c_i)$ (i.e., the test time of c_i is $cycle(r_{ij})/freq(c_i)$ sec.). If we decrease the test frequency of c_i from $freq(c_i)$ to $freq(c_i)/d$ (where d is an integer value), then, the power consumption is also decreased from $power(c_i)$ to $power(c_i)/d$ according to equation(1). However, the test time of c_i is increased from $cycle(r_{ij})/freq(c_i)$ to $d \times cycle(r_{ij})/freq(c_i)$. Here, by inserting TDdeM between the ATE output and the inputs of c_i, we can reduce the test time since $pin(r_{ij}) \times d$ pins become available through the TDdeM. The test time reduction ratio depends on the core c_i and the wrapper list R_i. Table 1 shows an example of this power-conscious TDM for core7 in d695 from ITC'02 SoC benchmarks [22]. In this example, we can achieve a 50% power reduction with an 1.4% test time overhead by decreasing the frequency and increasing wrapper pins through TDdeM. This technique can apply cores which are not required to test at-speed (i.e., $atspeed(c_i) = no$), and allows us to achieve a power-constrained test schedule more effectively.

4 Problem Formulation

In the previous section, we showed that the frequency gaps between cores and ATE can be solve by using TDM techniques. Design of TDM/TDdeM circuits for a core c_i can be uniquely determined when f_{ATE}, a wrapper design r_{ij} and a test frequency for c_i are given. However, depending on parameters for a given multi-clock domain SoC $MCDS$ and given constraints such as power consumption, test pins and at-speed test requirements, there exists a case where we cannot solve the frequency gap by using TDM. Therefore, we need to judge whether there is a test schedule for $MCDS$ under given constraints or not. If there exists such a solution, then, we determine a wrapper design and a test frequency for each core to minimize test time. We now formulate the power-constrained test scheduling problem for multi-clock domain SoCs P_{mcds} that we address in this paper as follows.

Definition 1 P_{mcds}: Given a multi-clock domain SoC $MCDS$, the number of available test pins W_{max} and the clock frequency of ATE f_{ATE}, is there a test schedule for $MCDS$ that satisfies all the following conditions?

1. the total number of test pins used at any moment does not exceed W_{max},
2. the total power consumption used at any moment does not exceed P_{max},
3. each core satisfies at-speed test requirement (i.e., if $atspeed(c_i) = yes$, c_i must be tested at $freq(c_i)$. Otherwise, c_i can be tested at frequencies lower than $freq(c_i)$),
4. the overall SoC test time is minimized

If there is such a test schedule, determine a wrapper design and test frequency of each core for the test schedule.

5 Scheduling Algorithm

This section presents a heuristic algorithm for P_{mcds} that consists of the following three stages: 1) testability analysis, 2) test scheduling at time 0 for cores with large amount of test data, and 3) test scheduling based on Best Fit Decreasing (BFD) heuristic for remaining cores. In the first stage, it decides whether there exists a solution for a given problem by considering the worst case scheduling where each core is tested one by one sequentially. Stage 2 determines cores which start their tests at time 0 in the descending order based on the test data amount. The shaded cores in Figure 3 are examples scheduled in this stage. In this stage, we design a wrapper r_{ij} for a core c_i such that $pin(r_{ij})$ is maximized and test time of c_i at frequency $freq(c_i)$ does not exceed the lower bound T_{LB} on the SoC test time defined in Section 5.2. Moreover, we determine a test frequency $f_{c_i}^{test}$ for c_i such that $f_{c_i}^{test}$ is minimized and test time of c_i at frequency $f_{c_i}^{test}$ does not exceed T_{LB} with the wrapper r_{ij}. By lowering the test frequencies of cores which start their test at time 0, we consider that test concurrency under power constraint can be increased and test time can be reduced. In the third stage, it determines a test schedule for each remaining (un-scheduled) core based on BFD heuristic. The unshaded cores in Figure 3 are examples scheduled in this stage. We pick a core in the descending order based on its test data amount. Then, we find the best start time, a wrapper design and a test frequency for the core such that the total test time of the given SoC is minimized. The following subsections describe the details of each stage.

5.1 Testability Analysis (Stage 1)

If $MCDS$ cannot satisfy the following two conditions for the given parameters: f_{ATE} and W_{max}, then there is no solution for P_{mcds}.

For each $c_i \in C$ such that $atspeed(c_i) = yes$,
[power limitation]

$$P_{max} \geq power(c_i) \quad (4)$$

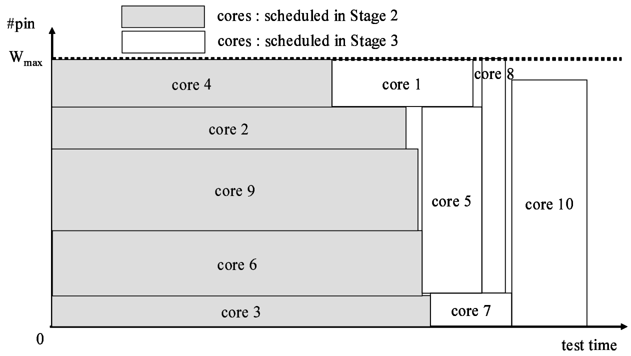

Figure 3. Test schedule generated by our proposed method.

[pin limitation]

$$W_{max} \geq \min_j(pin(r_{ij})) \cdot \lceil freq(c_i)/f_{ATE} \rceil \quad (5)$$

For a core c_i such that $atspeed(c_i) = yes$, we cannot change the test frequency $freq(c_i)$ and power consumption $power(c_i)$ during test. Therefore, the core that cannot satisfy equation(4) exceeds a given power limitation even if it is tested alone. Moreover, as explained before, TDM/TDdeM circuits can be uniquely determined when f_{ATE}, a wrapper design r_{ij} and a test frequency for c_i are given. Therefore, the core that doesn't satisfy equation(5) cannot be assigned enough wrapper pins to achieve at-speed test at $freq(c_i)$.

5.2 Test scheduling at time 0 with minimum test frequency (Stage 2)

This stage consists of the following three steps.

Step 1: *determine a wrapper design and test frequency for each core*

Main idea in this step is to increase test concurrency for power-constrained test scheduling by lowering the test frequencies of cores which do not require at-speed test. For each core c_i, we determine a wrapper design r_i^{test} and a multiplicity m_{c_i} such that

1. $T_{LB} \geq cycle(r_i^{test})/(freq(c_i)/m_{c_i})$,
2. $pin(r_i^{test}) \leq W_{max} \cdot f_{ATE}/(freq(c_i)/m_{c_i})$,
3. $pin(r_i^{test})$ is maximized, and
4. m_{c_i} is maximized.

Here, lower bound T_{LB} on the SoC test time is defined as follows.

$$T_{LB} = \max\{\max_i(T_{LB}^{c_i}), TotalData/(W_{max} \cdot f_{ATE})\} \quad (6)$$

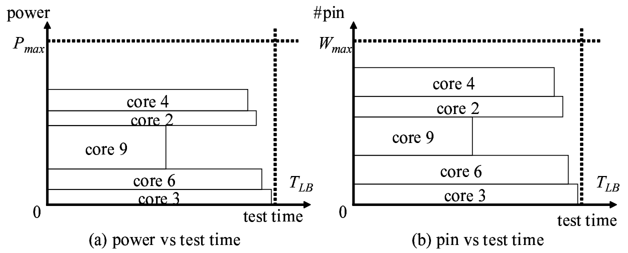

Figure 4. Test schedule after Step 2.

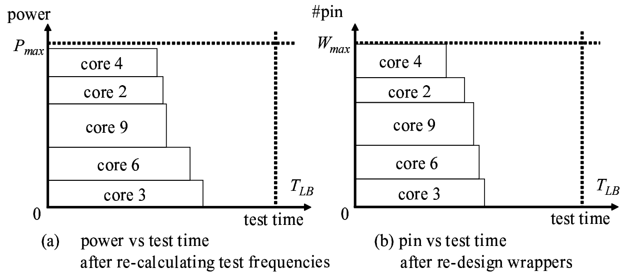

Figure 5. Test schedule after Step 3.

Lower bound $T_{LB}^{c_i}$ on the core test time and $TotalDate$ are defined as follows.

$$T_{LB}^{c_i} = cycle(r_{ij})/freq(c_i) \text{ s.t.}$$

$$1.\ pin(r_{ij}) \text{ is maximized, and}$$

$$2.\ pin(r_{ij}) \leq W_{max} \cdot f_{ATE}/freq(c_i). \quad (7)$$

$$TotalData = \sum_i pin(r_{ij}) \cdot cycle(r_{ij}) \text{ s.t.}$$

$$pin(r_{ij}) \text{ is minimized.} \quad (8)$$

Then, we determine test frequency $f_{c_i}^{test}$ for c_i as follows.

$$f_{c_i}^{test} = \begin{cases} freq(c_i)/m_{c_i} & \text{if } atspeed(c_i) = no \\ freq(c_i) & \text{if } atspeed(c_i) = yes \end{cases} \quad (9)$$

Step 2: *determine cores which start their tests at time 0*

First, we sort cores in the descending order based on its $T_{LB}^{c_i}$. Then, we schedule a core c_i in the above order at time 0 with wrapper r_i^{test} and test frequency $f_{c_i}^{test}$. This process repeats until 1) the power consumption at time 0 (P_0) does not exceed P_{max}, 2) the pin usage at time 0 (W_0) does not exceed W_{max}, and 3) $T_{LB}^{c_i}$ is less than $T_{LB}/10$. The third condition can prevent us from scheduling cores with small amount of test data to time 0. Instead of scheduling such small cores at time 0, Step 3 re-designs wrappers and re-calculates test frequencies for the cores scheduled in this step to reduce the overall test time of the SoC.

Figure 4 shows a current test schedule generated after Step 2. In Figure 4(a), the horizontal axis denotes the test time, and the vertical axis denotes the power consumption used in each test time. In Figure 4(b), the horizontal axis denotes the test time, and the vertical axis denotes the number of test pin used in each test time.

Step 3: *re-calculate test frequencies and re-design wrappers for cores scheduled at time 0*

There exists a case where P_0 (power consumption at time 0) does not reach P_{max} after Step 2 (Fig. 4(a)) since Step 2 stops the above three conditions. In this case, we find a core c_i that satisfies all the following conditions.

$$1.\ cycle(r_i^{test})/f_{c_i}^{test} \text{is maximized} \quad (10)$$

$$2.\ m_{c_i} > 1 \quad (11)$$

$$3.\ P_{max} \geq P_0 - power(c_i)\{1/m_{c_i} + 1/(m_{c_i} - 1)\} \quad (12)$$

$$4.\ P_{max}/2 \geq power(c_i)/(m_{c_i} - 1) \quad (13)$$

If there exists such a core c_i, we update m_{c_i} to $m_{c_i} - 1$, and reduce the test time of c_i by increasing $f_{c_i}^{test}$ according to equation (9). The fourth condition (equation(13)) can prevent one core from dominating power consumption, and help us to increase the test concurrency at time 0. This process repeats until 1) P_0 does not exceed P_{max} and 2) there exists a core that satisfies the above conditions. Figure 5(a) shows a result where we apply this process to the current schedule generated after Step 2 corresponds to Figure 4. In this figure, frequencies for core 2, 3, 4 and 6 are increased. Consequently, the test time for these cores are reduced.

Similarly, there exists a case where W_0 (pin usage at time 0) does not reach W_{max} after Step 2. In this case, we find a core c_i with maximum test time, then assign 1 test pin to c_i. This process repeats until W_0 does not exceed W_{max}. Figure 5(b) shows a result where we apply this process to the current schedule corresponds to Figure 5(a).

5.3 Test scheduling for remaining cores based on BFD (Stage 3)

In this stage, we determine a test schedule for the remaining cores based on BFD heuristic. First, we pick a core c_i in the descending order based on $T_{LB}^{c_i}$. Then, we find the best start time, wrapper design and test frequency for c_i such that the total test time of the given SoC is minimized as follows.

1. Let S be a set of start time candidates that consists of the end time of scheduled cores in the current schedule. For each candidate $s \in S$, we calculate available power consumption P_s and available test pin W_s from the current schedule.

2. For each candidate $s \in S$,

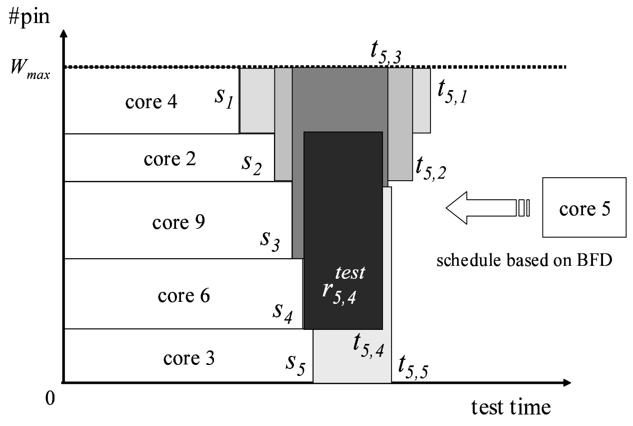

Figure 6. An example of test scheduling for core 5.

(a) Determine a maximum test frequency $f^{test}_{c_i,s}$ such that $power(c_i) \cdot f^{test}_{c_i,s} / freq(c_i)$ does not exceed P_s.

(b) Determine a wrapper design $r^{test}_{i,s}$ such that 1) $pin(r^{test}_{i,s})$ does not exceed $W_s \cdot f_{ATE}/f^{test}_{c_i,s}$ and 2) $pin(r^{test}_{i,s})$ is maximized.

(c) Calculate the end time $t_{i,s}$ when c_i starts its test at time s with wrapper $r^{test}_{i,s}$ at frequency $f^{test}_{c_i,s}$.

3. Schedule c_i at time s with wrapper $r^{test}_{i,s}$ at frequency $f^{test}_{c_i,s}$ such that 1) $t_{i,s}$ is minimized and 2) the test of c_i does not overlap the tests of cores already scheduled in the current schedule.

Figure 6 shows an example of the test scheduling for core 5. Here, a set of start time candidates S consists of five elements: s_1, s_2, s_3, s_4, s_5. For each candidate $s \in S$, we calculate a end time $t_{5,s}$ by determining a test frequency $f^{test}_{c_5,s}$ and a wrapper design $r^{test}_{5,s}$ shown as a rectangle in Figure 6. In this example, core 5 is scheduled to start its test at time s_4 with a wrapper $r^{test}_{5,4}$ at frequency $f^{test}_{c_5,4}$ since the end time $t_{5,4}$ has a minimum value.

This process repeats until all the remaining cores are scheduled in the descending order based on $T^{c_i}_{LB}$. Through the above processes, we can generate a final test schedule.

6 Experimental Results

In Section 6.1, we show experimental results for a multi-clock domain SoC with power constraint. Section 6.2 presents experimental results for single-clock domain SoCs with power constraint (“d695” and “h953” from ITC’02 SoC benchmarks [22]) in order to show the effectiveness of our approach compared to previous works. All the experimental results can be obtained within 0.01 sec. on a SunBlade 2000 workstation (1.05 GHz with 8GB RAM).

6.1 Results for a multi-clock domain SoC

Table 2 shows the multi-clock domain SoC $MCDS_1$ used in this experiment. This SoC consists of 14 cores. First 10 cores are from “d695” in ITC’02 SoC benchmarks. “flexible(≥ 2)” in column “wrapper list” denotes that we can design any wrapper (wrapper with any number of test pins) by the procedure proposed in [2, 3]. We use the same power consumption shown in [14], and assume that $freq(c_i) = 50MHz$ and $atspeed(c_i) = no$ for these 10 cores. The wrappers for core 11 and core 12 are already designed (i.e., 64 pins, 32 pins, respectively). We assume that these two cores are tested at higher frequencies than other cores, and $atspeed(c_i) = yes$. Core 13 and core 14 are copies of core 7 and core 5, respectively. However, we assume that these two cores are tested at lower frequencies than other cores.

For this SoC, Table 3, 4, 5 show test time results when f_{ATE} = 200MHz, 100MHz, 50MHz, respectively. In this tables, the test time results are shown as “μsec.”, and “untestable” denotes that there exists no solution for the given parameters. In this SoC, since core 11 should be tested at 100MHz with 64 pins, we observe that there exists no solution for three cases: 1) f_{ATE} = 100MHz and W_{max} = 32, 2)f_{ATE} = 50MHz and W_{max} = 32, and 3) f_{ATE} = 50MHz and W_{max} = 64. We also observe that test time depends on the product of f_{ATE} and W_{max}. Therefore, when we use a high speed ATE, we can test SoCs with small number of test pins. On the other hand, even when we use a low speed ATE, we can achieve the same test time by using more test pins. Moreover, for all cases, we cannot achieve a 50% (75%) test time reduction compared to the results for W_{max} = 32 even if we change W_{max} to 64 (128), respectively. From these results, we can say that for this SoC it is more effective to do multi-site testing where a small number of ATE test pins are assigned to each SoC. For example, we consider the case when f_{ATE} = 200MHz and P_{max} = 1500, and total test time for 4 SoCs. When we test this SoC with 128 pins, the test time is 427.61 μs for one SoC. Therefore, the total test time for 4 SoCs is 1710.44 μs. On the other hand, when we test this SoC with 32 pins, the test time is 431.05 μs. However, the total test time for 4 SoCs is also 431.05 μs since we can test 4 SoCs at the same time.

6.2 Comparison with other approaches

Table 6 and 7 show the test time results for “d695” and “h953” in ITC’02 SoC benchmarks. For “d695”, we use the same power consumption shown in [14]. In this experiments, we assume that these two SoCs are single-clock domain SoCs (i.e., $f_{ATE} = 50MHz$, and $freq(c_i) = 50MHz$ and $atspeed(c_i) = no$ for all core $c_i \in C$) in order to compare the previous approaches [14, 15] which are applicable only to single-clock domain SoCs. In this

Table 2. An multi-clock domain SoC $MCDS_1$.

core	at-speed requirement	wrapper list (pins)	test freq. (MHz)	power (unit)
1	no	flexible(≥ 2)	50	660
2	no	flexible(≥ 2)	50	602
3	no	flexible(≥ 2)	50	823
4	no	flexible(≥ 2)	50	275
5	no	flexible(≥ 2)	50	690
6	no	flexible(≥ 2)	50	354
7	no	flexible(≥ 2)	50	530
8	no	flexible(≥ 2)	50	753
9	no	flexible(≥ 2)	50	641
10	no	flexible(≥ 2)	50	1144
11	yes	fixed (64)	100	480
12	yes	fixed (32)	200	940
13	no	flexible(≥ 2)	20	212
14	no	flexible(≥ 2)	25	345

Table 3. Test time results [μs] for $f_{ATE} = 200$MHz.

	W_{max}		
P_{max}	32pin	64pin	128pin
1500	431.05	427.61	427.61
2000	325.29	301.08	301.08
2500	324.05	234.24	221.70

tables, test time results are shown as the number of clock cycles. "NA" denotes that the approach is not applicable for the constraint. "-" denotes that no result is shown for the constraint in the approach.

From Table 6, we observe that the proposed approach can achieve a 6.9% reduction in average test time compared to [14]. Moreover, from Table 7, we observe that the proposed approach can achieve the lower bound (119357) on the SoC test time [13] under all power constraints. From these results, we conclude that the proposed power-conscious TDM technique and test scheduling algorithm are also effective for single-clock domain SoCs.

7 Conclusions

This paper has presented a power-conscious wrapper and TAM design for multi-clock domain SoCs, and proposed a test scheduling algorithm to minimize test time under power constraint. To the best of our knowledge, a test scheduling problem for multi-clock domain SoCs has been addressed and formulated for the first time in this paper. The proposed method can solve the frequency gap between each core and a given ATE by using TDM technique. Therefore, we can achieve at-speed test of cores even when the test frequencies of the cores are higher than that of the ATE, and make use of the ATE capability more effectively. Moreover, we have presented a technique to reduce power consumption of a core during test while keeping the test time by utilizing TDM technique. Experimental results show the effectiveness of our method not only for a multi-clock domain SoC, but also for single-clock domain SoCs with power constraints.

Table 4. Test time results [μs] for $f_{ATE} = 100$MHz.

	W_{max}		
P_{max}	32pin	64pin	128pin
1500	untestable	431.05	427.61
2000	untestable	325.29	301.08
2500	untestable	324.05	234.34

Table 5. Test time results [μs] for $f_{ATE} = 50$MHz.

	W_{max}		
P_{max}	32pin	64pin	128pin
1500	untestable	untestable	431.05
2000	untestable	untestable	301.08
2500	untestable	untestable	234.34

One of our future works is to include various constraints such as test resource conflicts, test precedence between test sets, multiple test sets required by a single core, and routing/hardware overhead. Another future work is to propose a co-optimization technique between test time and area (routing) overhead of wrapper/TAM designs.

Acknowledgments

This work was supported in part by Japan Society for the Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research B(2)(No. 15300018). The authors would like to thank Profs. Kewal K. Saluja, Michiko Inoue, Dr. Satoshi Ohtake and members of Fujiwara Laboratory (Nara Institute of Science and Technology) for their valuable comments.

References

[1] Y. Zorian, E. J. Marinissen and S. Dey,"Testing embedded-core based system chips," *Proc. 1998 Int. Test Conf.*, pp. 130–143, Oct. 1998.

[2] V. Iyengar, K. Chakrabarty and E. J. Marinissen, "Test Wrapper and test access mechanism co-optimaization for system-on-chip," *Journal of Electronic Testing: Theory and Applications*, pp. 213–230, Apr. 2002.

[3] W.Zou,S.R.Reddy,I.Pomeranz and Y.Huang,"SOC Test Scheduling Using Simulated Anneaking," *Proc. 21th VLSI Test Symp.*,pp.325–329,May 2003.

[4] IEEE P1500 web site, *http://grouper.ieee.org/groups/ctl/*.

[5] T. Ono, K. Wakui, H. Hikima, Y. Nakamura and M. Yoshida, "Integrated and automated design-for-testability implementation for cell-based ICs," *Proc. 6th Asian Test Symp.*, pp. 122–125, Nov. 1997.

[6] P. Varma and S. Bhatia, "A structured test re-use methodology for core-based system chips," *Proc. 1996 Int. Test Conf.*, pp. 294–302, Oct. 1998.

Table 6. Test time results (# cycles) for d695.

P_{max}	W_{max}								
	32pin			64pin			128pin		
	3D	EA	proposed	3D	EA	proposed	3D	EA	proposed
1000	NA	NA	**44528**	NA	NA	**27482**	NA	NA	**24707**
1500	45560	-	**42981**	27573	-	**22690**	16841	-	**16239**
2000	43221	-	**42632**	24171	-	**21838**	14128	-	**12753**
2500	43221	-	**42564**	23721	-	**21616**	12993	-	**11180**

Table 7. Test time results (# cycles) for h953.

P_{max}	W_{max}								
	32pin			64pin			128pin		
	3D	EA	proposed	3D	EA	proposed	3D	EA	proposed
5×10^9	NA	NA	**119357**	NA	NA	**119357**	NA	NA	**119357**
6×10^9	122636	122636	**119357**	122636	122636	**119357**	122636	122636	**119357**
7×10^9	**119357**	**119357**	**119357**	**119357**	**119357**	**119357**	**119357**	**119357**	**119357**

[7] E. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg and C. Wouters, "A structured and scalable mechanism for test access to embedded reusable cores," *Proc. 1998 Int. Test Conf.*, pp. 284–293, Oct. 1998.

[8] M. Nourani and C. A. Papachristou, "Structural fault testing of embedded cores using pipelining," *Journal of Electronic Testing:Theory and Applications 15(1-2)*, pp. 129–144, Aug.–Oct. 1999.

[9] S. Ravi, G. Lakshminarayana, and N. K. Jha, " Testing of core-based systems-on-a-chip," *IEEE Trans. on CAD*, Vol. 20, No. 3, pp. 426–439, Mar. 2001.

[10] T. Yoneda, T. Uchiyama and H. Fujiwara, "Area and time co-optimization for system-on-a-chip based on consecutive testability," *Proc. 2003 Int. Test Conf.*, pp. 415–422, Sep. 2003.

[11] Y.Huang et al.,"Resource allocation and test scheduling for concurrent test of core-based SOC design,"Proc. Asian Test Symposium(ATS), pp265-270, 2001.

[12] V. Iyengar, K. Chakrabarty and E. J. Marinissen, "On using rectangle packing for SOC wrapper/TAM co-optimization," *Proc. 20th VLSI Test Symp.*, pp. 253–258, Apr. 2002.

[13] S. K. Goel and E. J. Marinissen,"Effective and Efficient Test Architecture Design for SOCs,"Proc. IEEE International Test Conference(ITC), pp529-538, 2002.

[14] Y. Huang, N. Mukherjee, S. Reddy, C. Tsai, W. Cheng, O. Samman, P. Reuter and Y. Zaidan, "Optimal core wrapper width selection and SOC test scheduling based on 3-dimensional bin packing algorithm," *Proc. 2002 Int. Test Conf.*, pp. 74–82, Oct. 2002.

[15] Y. Xia, M. C. Jeske, B. Wang and M. Jeske, "Using Destributed Rectangle Bin-Packinig Approach for Core-based SoC Test Scheduling with Power Constraints, " *ICCAD'03*, pp.100–105, Nov. 2003.

[16] E. Larsson, K. Arvidsson, H. Fujiwara and Z. Peng,,"Efficient Test Solutions for Core-based Designs,"*IEEE Trans. on CAD*, Vol. 23, No. 5, pp. 758–775, May 2004.

[17] Q.Xu and N.Nicolici, "Wrapper Design for Testing IP Cores with Multiple Clock Domains," *In Proceedings of the 2004 Design,Automation and Test in Europe(DATE)*,pp.416–421,Feb.2004.

[18] A.Khoche,"Test resource partitioning for scan architectures using bandwidth macting,"*Digest of Int. Workshop on Test Resouce Partitioning*,pp.1.4-1–1.4-8, 2001.

[19] A.Sehgal,V. Iyengar,M.D.Krasniewski and K. Chakrabarty, "Test Cost Reduction for SOCs Using Virtual TAMs and Lagrange Multipliers," *In Proc.IEEE/ACM Design Automation Conference*,pp.738–743,Jun.2003.

[20] Q.Xu and N.Nicolici,'Multi-frequency Test Access Mechanism Design for Modular SOC Testing,"*Proc. of IEEE the 11th Asian Test Symposium*,Nov.2004.

[21] P. Girard, "Survey of low-power testing of VLSI circuits," *IEEE Design & Test of Computers*, Vol. 19, No. 3, pp. 82–92, May–June 2002.

[22] ITCf02 SOC Test Benchmarks Web Site, *http://www.extra.research.philips.com*