SOCs Test Scheduling using TAM Switch

Lung-Jen Lee[1], Wang-Dauh Tseng, Rung-Bin Lin, and Zheng-Han Zhang
Department of Computer Science & Engineering
Yuan Ze University, Chung-li, Taiwan, 32026, ROC
[1]s959101@mail.yzu.edu.tw

***Abstract*—The progress of the SOC technologies has inspired the requirements of complex circuits. The potential complexity of SOCs results in the difficulty of testing and long testing time. The testing time of a SOC is determined by the total test bandwidth of TAM and the test sequence of cores. High bandwidth TAM provides faster test data access of cores and thus reduce the testing time of SOCs, but it also results in high area overhead. Besides, under the constraint of the total test bandwidth, the test sequence of cores affects the SOC testing time significantly. In this paper, we propose a TAM Switch based test scheduling algorithm for SOCs, under the constraint of fixed TAM width. Experimental results show the proposed approach obtains low testing time for all cases in SOC d695 and SOC p93791 benchmarks**
***Index Terms*—X-filling, capture power, LCP X-filling.**

1. INTRODUCTION

Modern semiconductor technologies enable the design of a complete system on a single chip (SOC). In order to reduce time-to-market, more pre-designed modules (cores), are used in the design of SOCs. But the testing of core-based SOCs is facing a serious challenge [2]. Because cores are deeply embedded in SOC, we can't directly access test data from chip I/Os. In order to transport test data and test responses between cores and chip I/Os, a special test access path is built to increase the test accessibility of SOCs. This test path is called test access mechanism (TAM). The SOC testing time is directly related to TAM width. The larger the TAM width is, the less the SOC testing time will be. But the increase of TAM width will lead to an increase of area overhead.

Furthermore, the test sequence of cores will affect the SOC testing time significantly. The purpose of this paper is to minimize the total testing time by arranging the test order among cores and fulfilling all the given constraints.

Scheduling the tests in a system means that the start time and end time for each test are determined in such a way that all constraints are satisfied and the total test time is minimized. The constraints usually considered in SOC test scheduling are listed as follows:

1). Power consumption [5][6]: In order to reduce the SOC testing time, as many cores are required to be tested concurrently as possible. However, that too many cores are tested simultaneously results in the increasing of power consumption and may thus damage SOC due to overheating. To avoid such situation, the power limit of the SOC must be specified in advance to prevent the sum of power consumed by the cores tested concurrently from exceeding the threshold at any instance during test.

2). Test bandwidth [7]: The bandwidth of TAM directly influences the SOC total testing time. High bandwidth TAM provides faster test access of cores and thus reduces the SOC testing time; however, large test bandwidth results in high area overhead and will affects system performance and reduces the yield of chips.

3). Test resource [8]: In built-in self-test (BIST) cores, the designers must allocate extra test hardware to enhance the controllability and observability. The extra test hardware will lead to a large area overhead. In order to reduce area overhead, cores with similar functionality can share test resources. But it will lead to conflicts if the cores sharing the same test resources are tested concurrently.

4). Preemptive and non-preemptive test scheduling [9][10]: If a new test is allowed to start

before the current test session is completed, the kind of test scheduling is called preemptive, otherwise called non-preemptive.

Several TAM architectures have been proposed [1]-[4]. Aerts and Marinissen described three scan-based test access architectures [4]: (1) multiplexing architecture, (2) daisychain architecture, and (3) distribution architecture. In the multiplexing architecture, only one core can be accessed at one time, so cores must be tested sequentially in the multiplexing architecture. The total testing time of SOC is the sum of the test time of individual core in the multiplexing architecture. The daisychain architecture transports test patterns to every core simultaneously. An efficient way of using the daisychain architecture is to test as many cores as possible in parallel, and turn on a core's bypass as soon as it runs out of test patterns. In the distribution architecture, the total TAM width is divided into several test buses and each is distributed to the cores under test. The TAM architectures illustrated above can't change the test bandwidth of a core during test once the test bandwidth for the core is determined in test design stage. Consequently, part of test bandwidth will be wasted if the test bandwidth assigned for the core is large than that of the required. The TAM Switch architecture proposed in [12] allows dynamically reconfiguring the test bandwidth of each core during test. To adjust the test bandwidth, a switch is typically placed between the core under test and test bus. Figure 1 illustrates the TAM switch architecture. TAM switch is a simple configurable switch which connects P wires out of the N test bus wires to the core test input terminals and collects P wires from the core test output terminals to the test bus. Where, N presents the SOC total test bandwidth; P is the test bandwidth that will be assigned to a given core. The switch is used to choose P wires among the N test bus wires that will be connected to the test pins of the core. The input pins of SOC will be connected to the input pins of

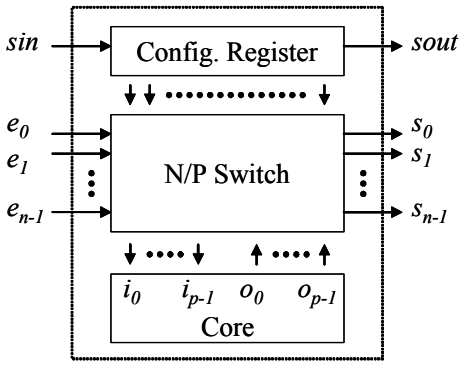

Figure 1. TAM Switch Architecture

TAM Switch (e0…en-1). We also use TAM Switch to select the inputs (i0…ip-1) that will be connected to the core for test. We export the test response (o1…op-1) via the output pins of TAM Switch (s0…sn-1).

2. PRELIMINARIES

We can use rectangles to model the test of cores; this notion is proposed in [1]. As show in Figure 2(a), the height of the rectangle represents the test bandwidth of core i and the width represents the test time of core i at the given test bandwidth. Therefore, as show in Figure 2(b), we can use 2-tuple (W(i) , T(W(i))) to represent the test bandwidth assigned and the test time consumed for core i. The maximum test bandwidth for each core must be specified before test schedule is performed. If core i has m functional inputs, n functional outputs, b bi-directional I/Os, and s internal scan chains in which the maximal scan chain length is lm, the maximum test bandwidth for each core i is given by:

$$W\max(i) = \begin{cases} s + \lceil (\max(m,n) + b / l_m \rceil, \\ \qquad \text{for core i with scan chains} \\ \max(m,n) + b, \\ \qquad \text{for core i without scan chain} \end{cases}$$

Assume that the number of test patterns for core i is pi. When assign the maximum test bandwidth Wmax(i) to core i, the required testing time in cycles is given by:

$$T_i(i) = \begin{cases} (p_i + 1) \times l_m + p_i, & \text{for core i with scan chains} \\ p_i, & \text{for core i without scan chain} \end{cases}$$

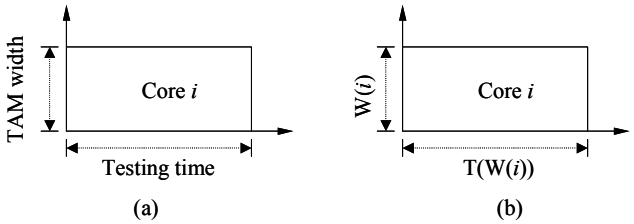

Figure 2. The relation between the test bandwidth and the test time

For a given core, the testing time of the core varies with TAM width as a staircase function. The testing time decreases only when the TAM width exceeds core-specific thresholds. We can regard these threshold points as pareto-optimal points, and are formally defined in [11]. As illustrated previously, we use 2-tuple (W(i), T(W(i))) to represent the test schedule of core i. We regard a solution (W(i), T(W(i))) as the pareto-optimal if and only if there does not exist a solution (W(k), T(W(k))), such that W(k) ≤ W(i) and T(W(k)) ≤ T(W(i)).

3. PROPOSED METHOD

3.1. Assumptions

Before detailing the proposed approach, the assumptions used in this paper are stated as follows. Assume that the SOC consists of N cores, for each core i, $1 \le i \le N$, there are mi functional inputs, ni functional outputs, bi bi-directional I/Os, si internal scan chains and pi test patterns. In addition, the length of the maximal scan chain is lmi. Then the maximal allowed test bandwidth for each core i is given by $Wmax(i) = si + \lceil (max(mi, ni) + bi) / lmi \rceil$. If the assigned test bandwidth for core i is less than its maximal allowed test bandwidth, test data serialization will be required at core I/O. The amount of test data serialization required at the inputs and outputs is determined by Wmax(i), which influences the testing time of core i, because if the core requires serialization at its I/Os, more testing time is needed to complete a test. If the test bandwidth assigned to core i is less than Wmax(i), the test time for this core is given by $Ti = (pi + 1) \times$ (the test time of a test pattern) + pi.

3.2. TAM Switch Based Test Scheduling Algorithm

In this section, we detail the proposed algorithm. The Problem to solve in this paper is formally stated as follows: given N cores and total bandwidth W for a SOC, the object is to determine an optimal test schedule based on TAM switch Architecture such that the SOC total test time is minimum. The proposed test schedule algorithm is shown in Figure 3. The first step of the proposed algorithm is to calculate the pareto-optimal width for each core and to initialize the input parameters. If the available TAM width, w_avail, is greater than the test bandwidth of a group of cores, then the core among the group, say i, with the longest testing time has the highest priority to schedule. The TAM width assigned to core i is width(i) and the testing time for core i is Ti(width(i)). Otherwise, we assign the highest pareto-optimal width w to core i, where w ≤ w_avail. Hereafter, two cases, as illustrated in Figure 4, are discussed. If the end time of core i is less than next_time, as show in Figure 4(a), we assign the TAM width, w_avail, to core i. In this case, the available TAM width is not increased during the test of core i. Therefore, the test bandwidth assigned for core i is not changed. If the end time of core i is greater than next_time, as show in Figure 4(b), the available TAM width are increased during the test of core i, because of the releasing of test bandwidth by the core which has completed its test. In this case, the increased test bandwidth is added to core i at this point, next_time. Note that the test bandwidth assigned to core i can't exceed the maximal test bandwidth of core i. Here, we use Figure 4(c)(d) to explain the case that the assigned test bandwidth for core i varies to the change of the available TAM width. As show in Figure 4(c), the available TAM width is increased at point a; therefore, the test bandwidth assigned for core i can be increased with the addition of available TAM width. When assigning the test bandwidth for core i, we must consider the pareto-optimal width of core i. As show in Figure 4(c), although the available TAM width is increased at point b, the test bandwidth for core i is not changed, if the higher pareto-optimal width for core i can't be found at this point.

As show in Figure 4(d), if the higher pareto-optimal width for core i at point b exist, the test bandwidth for core i can be changed with the increase of the available TAM width at this point.

In the following, an example is used to illustrate the proposed algorithm. The benchmark SOC used in this example is SOC d695 [13]. There are 10 cores in SOCd695. We use Ri = (a, b) to represent the rectangle of core i, where a is the maximal test bandwidth for core i and b is the test time of core i when the test bandwidth assigned for core i is a. The test bandwidth W is 64. Let the maximal test bandwidth and testing time for each core of SOCd695 be as follows. Figure 5 shows the detailed process.

R1=(32, 12); R2=(207, 73); R3=(3, 2507);

R4=(5, 5829); R5=(39, 5105); R6=(20, 9869);

R7=(21, 3359); R8=(6, 4605).

As show in Figure 4(d), if the higher pareto-optimal width for core i at point b exist, the test bandwidth for core i can be changed with the increase of the available TAM width at this point.

In the following, an example is used to illustrate the proposed algorithm. The benchmark SOC used in this example is SOC d695 [13]. There are 10 cores in SOCd695. We use Ri = (a, b) to represent the rectangle of core i, where a is the maximal test bandwidth for core i and b is the test time of core i when the test bandwidth assigned for

Algorithm ***TAM-switch-based test scheduling***

```
begin
    Calculate the pareto-optimal width for each core and initialize test parameters;
    while Core list is not empty
    begin
        Pick the core, say i, with longest testing time;
        If available test width w_avail i ≤ the test width of Core then
            assign w_avail to Core i;
        else
            If the end time of Core i is less than next_time then
                assign pareto(w_available) to Core i;
            else
                assign pareto(w_available + released test width) to Core i;
            end (if)
        end (if)
    end (while)
end (algorithm)
```

Figure 3. TAM Switch Test Scheduling Algorithm

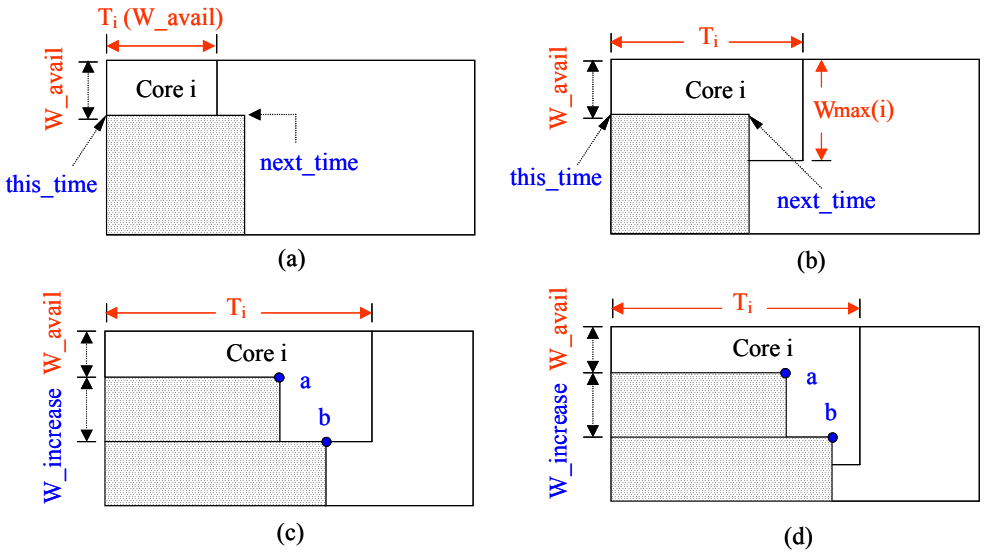

Figure 4. Dynamic test bandwidth assignments

core i is a.The test bandwidth W is 64. Let the maximal test bandwidth and testing time for each core of SOCd695 be as follows. Figure 5 shows the detailed process.

R1=(32, 12); R2=(207, 73); R3=(3, 2507);

R4=(5, 5829); R5=(39, 5105); R6=(20, 9869);

R7=(21, 3359); R8=(6, 4605); R9=(38, 714);

R10=(34, 3863).

4. EXPERIMENTAL RESULTS

The proposed algorithm is implemented in C++ and executed on Windows 2000. Two benchmarks d695 and p93791 are used as sample circuits to compare the testing time between the proposed approach and the approach in [1]. The two SOCs are part of the ITC 2002 SOC test benchmarks initiative [13]. Table 1 shows experimental results for SOC d695 and SOC p93791. The symbol Told represent the SOC testing time, in clock cycles, obtained by the approach in [1], and Tnew represent the testing time obtained by the proposed approach. As can been, the testing time of the proposed approach is less than that of the approach in [1].

5. CONCLUSIONS

Based on the TAM switch test architecture which allows the test bandwidth of cores be adjusted dynamically during test according to the varies of available TAM width, this paper propose a scheduling algorithm to minimize the testing time for SOC. Due to the features of TAM Switch, the test bandwidth can be used more efficiently and thus reduce the idle time as well as the testing time for SOC significantly. The experimental results show that the testing time obtained by the proposed approach is less than that of the approach in [1] in terms of the two benchmarks d695 and p93791.

REFERENCES

[1] V.Iyengar, K.Chakrabarty and E.J.Marinissen, "On Using Rectangle Packing for SOC Wrapper/TAM Co-Optimization," Proceedings of IEEE VLSI Test Symposium, 2002, pp. 253-258.

[2] M.Benabdenbi, W.Maroufi and M.Marzouki, "Testing TAPed cores and wrapped cores with the same test access mechanism," Proceedings of Design Automation and Test Conference in Europe, Mar. 2001, pp. 150-155.

[3] M.Benabdenebi, W.Maroufi and M.Marzouki, "CAS-BUS: a scalable and reconfigurable test access mechanism for systems on a chip," Proceedings of Design Automation and Test Conference in Europe, Mar. 2000, pp. 141-145.

[4] J.Aerts and E.J.Marinissen, "Scan Chain Design for Test Time Reduction in Core-Based Ics," Proceedings of International Test Conference, Oct. 1998, pp. 448-457.

[5] E.Larsson and Z.Peng," Test Scheduling and Scan-Chain Division Under Power Constraint," Proceedings of Asian Test Symposium, Nov. 2001, pp. 259-264.

[6] V.Muresan, X.Wang, and M.Vladutiu, "The Left Edge Algorithm and the Tree Growing Technique in Block-Test Scheduling under Power-Constraints," Proceedings of IEEE VLSI Test Symposium, May 2000, pp. 417-422.

[7] K.Chakrabarty," Design of System-on-a-Chip Test Access Architectures using Integer Linear Programming," Proceedings of IEEE VLSI Test Symposium, May 2000, pp. 127-134.

[8] G.Chandra, C.P.Ravikumar and A.Verma, "A Polynomial-Time Algorithm for Power Constrainted Testing of Core Based Systems," Proceedings of Asian Test Symposium, Nov. 1999, pp. 107-112.

[9] Y.Zorian, "A Distributed BIST Control Scheme for Complex VLSI devices," Proceedings of IEEE VLSI Test Symposium, Apr. 1993, pp. 4-9.

[10] V.Muresan et al., "A Comparison of Classical Scheduling Approaches in Power -Constrained Block-Test Scheduling," Proceedings of International Test Conference, Oct. 2000, pp. 882-891.

[11] V.Iyengar, K.Chakrabarty and E.J.Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," Proceedings of International Test Conference, Oct. 2001, pp. 1023-1032.

[12] S.Basu, D.Mukhopadhay, D.Roychoudhury, I.Sengupta and S.Bhawmik, "Reformatting test patterns for testing embedded core based system using test access mechanism (TAM) switch,"Proceedings of Asia and South Pacific and the 15th International Conference on VLSI Design, Jan. 2002, pp. 598-603.

[13] E.J.Marinissen,V.Iyengar and K.Chakrabarty, "ITC 2002 SOC test benchmark initiative," http://www.extra.research.philips.com/itc02socbenchm.

Table 1. Experimental results for d695 and p93791

W	SOC d695		SOC p93791	
	approach in [1]	proposed approach	approach in [1]	proposed approach
	T_{old} (Clock cycles)	T_{new} (Clock cycles)	T_{old} (Clock cycles)	T_{new} (Clock cycles)
16	43723	43716	1851135	1839051
24	30317	30199	1248795	1240642
32	23021	22201	975016	939296
40	18459	17798	794020	747133
48	15698	15535	627934	616930
56	13415	13410	568436	537966
64	11604	11207	511286	472752

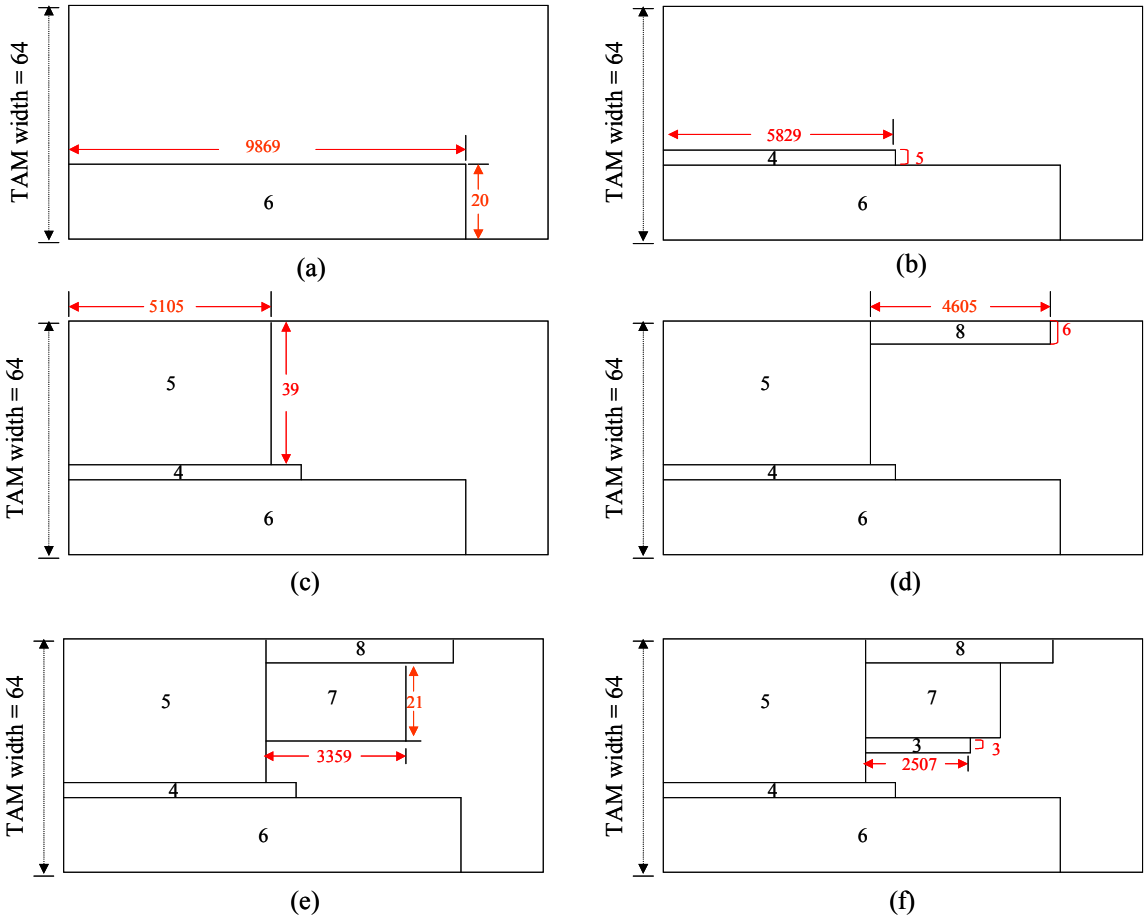

Figure 5. The illustration of the TAM Switch based test scheduling for SOC d69