Majority-Based Test Access Mechanism for Parallel Testing of Multiple Identical Cores
Taewoo Han, Inhyuk Choi, and Sungho Kang

Abstract—The increased use of multicore chips diminishes per-core complexity and also demands parallel design and test technologies. An especially important evolution of the multicore chip has been the use of multiple identical cores, providing a homogenous system with various merits. This paper introduces a novel test access mechanism (TAM) for parallel testing of multiple identical cores and identifying faulty cores to derate the chip by excluding it. Instead of typical test response data from the cores, the test output data used in this paper are the majority values, that is, the typical test responses from the cores. All the cores can thereby be tested in parallel and test costs (in both test pins and test time) are exactly the same as for a single core. The proposed TAM can be implemented with on-chip comparators and majority analyzers. The experimental results in this paper show that the proposed TAM can test multiple cores with minimal test pins and test time and with hardware overhead of <0.1%.

Index Terms—Homogeneous, multicore, parallel test, test access mechanism (TAM).

I. INTRODUCTION

ADVANCES in chip design and test technology have allowed for the integration of a large number of cores into a system-on-chip (SoC). However, the resulting system complexity makes it a major challenge to design and test reliable systems. Multicore system designs exacerbate power concerns and add test complexity, but also offer exciting design-for-test (DFT) opportunities [1]. Many difficulties in testing arise from the use of deeply embedded cores. For these designs, a test access mechanism (TAM) is needed that allows the core to be efficiently accessed and tested [2]. Using a general embedded core test methodology, such as IEEE 1500, cores can be wrapped to prevent interaction with outside data sources, allowing a test for a single core to be generated and applied to each of the instances of that core [3]. IEEE 1500 provides an efficient solution for testing cores by allowing parallel test access. Main problem with IEEE 1500 is how to transfer test data between the system test sinks/sources and the wrappers [4]–[6].

Recently, multicore designs have evolved to include multiple identical cores. This trend has been increasingly observed for CPU cores in modern microprocessor designs [7]. The use of multiple identical cores achieves several goals; in addition to the benefits of multiprocessing, some cores can be used as redundant cores to guarantee a highly reliable system. A modular test development approach can be used to reduce the test generation time and data volume [8]. The advantage of testing multiple identical cores is that each core responds identically to a given pattern, provided that it is isolated to prevent interaction with outside data sources. This allows us to compare the core’s output responses either to the expected response or to the other cores’ responses using on-chip comparators [9]. Using a broadcast-based TAM to test all the cores in parallel has been described previously. The AZSCAN architecture tests multiple identical cores in parallel, and the responses are compared with the expected data in the chip [10]. A pipeline-based TAM allows for a great deal of flexibility in test applications, and the pipelining helps to improve test time and to reduce the capture power requirements [11]. A partitioning scheme enables the broadcast TAM to achieve high diagnostic resolution. This scheme requires the addition of a few extra chip output pins, which allow core-level failures to be deduced from the chip-level automated test equipment (ATE) measurements [12].

These various and valuable parallel TAM methodologies have been researched by EDA vendors; one recent paper about design and test for a multicore system by a processor-IP vendor gives a good description of the application of parallel TAM [13]. The test responses of homogeneous cores are expected to be identical when there are no faults in the cores. Therefore, the parallel TAMs described in previous research establish a reference based on nonfaulty cores and compare the other cores to that reference. The broadcast-based TAM uses expected data from the outside of chip as the reference for nonfaulty cores, but the necessity of reference cores adds other costs; namely, additional test pins are required for the expected data. The pipeline-based TAM uses primary core data from the inside of chip. The pipeline strategy can reduce the cost of testing, often making the test cost of multiple cores equivalent to that of a single core. However, additional test time is required when the primary core is faulty, because the pipeline-based TAM turns off the faulty primary core and restarts the test process after assigning another core to a new primary core.

However, if more than half the cores have faults, users in most practical situations will not be interested in salvaging the usable cores, instead derating the multicore system and considering it unreliable. In addition, when the number of
redundant cores is increased to nearly half the number of total cores, the effect of yield improvement which is related to the number of redundant cores decreases [14].

In this paper, a novel solution is proposed to allow true parallelism in testing, that is, the testing of multiple cores as a single core. Using the proposed TAM, a multicore system in which fewer than half of the total cores are faulty will be salvaged as usable. Also, the proposed TAM does not require a reference of nonfaulty cores; rather, cores are deemed nonfaulty based on their agreement with the majority vote of the multiple cores, a vote that is carried out within the chip itself. The basic concept of testing cores through majority voting has been used in previous research, but the methods used have focused on online tests and the cores were regarded as known-good cores, which had already passed manufacturing tests. Therefore, it was assumed that only one core can have a fault in each comparison, and partial cores (three cores among multiple cores) were targeted for voting [15], [16]. This paper includes manufacturing testing in addition to online testing. The proposed TAM can be used to perform a complete core-level diagnosis in the case of faults by multiple cores, and can obtain the majority value in one scan shift clock cycle.

The objectives and properties of the proposed TAM clearly show that it is an optimized solution.

1) It can test multiple cores for the cost of a single core.
2) Only the majority values of the proposed TAM are analyzed in the ATE, while in other TAM methods, all test response data are analyzed in the ATE.
3) The proposed method is related to the delivery of test response data of cores, and can be compatible with the conventional DFT technologies.

The rest of this paper is organized as follows. Section II describes the background information on existing parallel TAMs and outlines the concept of the proposed TAM. Section III provides detailed architectures and a flow diagram of the majority-based TAM. An extended version of the proposed TAM is also presented. Section IV describes the implementation of a multicore system with the majority-based TAM, and gives the results of several experiments that verify the effectiveness of the proposed TAM. Finally, Section V concludes this paper.

II. BACKGROUND INFORMATION

On-chip access to wrapped cores embedded in an SoC is provided by a TAM. The TAM is used to transport the test stimuli data from a test pattern source to the core under test and to transport the test response data from the core under test to a test pattern sink. IEEE 1500 does not prescribe a specific interface connection at the SoC level. In designing an SoC TAM architecture, the main focus is typically to optimize cost factors, such as test time, test pins, and hardware overhead.

A typical TAM architecture for a single core has test input pins and test output pins. The core accesses test stimuli patterns with the test data input (TDI) and exports its test response data with the test data output (TDO). In this paper, the number of pins used for TDI is represented by W; this number is equivalent to the test channel bandwidth of the TDI, as well as that of the TDO. For a multicore system, TAM can be implemented in various ways, which can be classified into two methods. The first method is more intuitive, increasing the number of test pins to allow simultaneous testing of multiple cores. In the second method, the same number of test pins is used as for a single core, and the cores share the TDI and TDO. The first method allows multiple cores to be tested in the same time it would take to test just one, but requires the use of many test pins, increasing in proportion to the number of cores. Conversely, the second method allows multiple cores to be tested using the same number of test pins as would be required to test just one, but requires more test time, increasing in proportion to the number of cores.

The use of an on-chip comparator is an attractive solution [9] for reducing the costs of testing homogeneous cores, and helps in the implementation of a parallel TAM. Fig. 1(a) shows a simple diagram of a typical parallel TAM architecture with an on-chip comparator. The test response data of multiple identical cores are compared with the expected data in the chip, and if any differences appear, 1-bit data 1 will be recorded in sticky-bit registers (gray blocks in Fig. 1). After the test processes are finished, only the data in the sticky-bit registers are read by the ATE; if they are read as 1s, the related core is considered faulty. To diagnose a specific core, only one TDO is used; it has the same number of pins as that of a single core. In this paper, this parallel TAM is called a broadcast-based TAM; it can test multiple cores in the same test time as for a single core, but needs more test ports to accommodate the expected data input (EDI). To reduce the number of test pins to that of a single core, a pipelined TAM [11] has been proposed, as shown in Fig. 1(b). In this design, the test response data of multiple identical cores are compared on chip with the test response data from a primary core. The test response data of the primary core are compared with the expected data in the ATE; if they agree, the primary core is considered nonfaulty, and any other cores whose test response data differ from that of the primary core are thus considered faulty. In this paper, this parallel TAM is called a pipeline-based TAM; it can test multiple cores with the same number of test pins as a single core, but requires additional test time when the primary core is faulty.

In general, in order to use the on-chip comparator mentioned previously, it is necessary to use a mask scheme to mask unknown (X) values. Various masking methods have been developed that do not require additional test pins for masking data [11], [17]. The proposed method reuses these masking
In a multicore SoC, multiple identical cores can be tested in parallel using broadcasted test patterns, and if there are no faults, the test response data are expected to be the same among each of the cores. It is expected that most cores will not have faults, so the proposed TAM analyzes the test response data and finds the majority value. A core that produces test response data different from the majority value can then be considered faulty. Naturally, the majority value is then tested by the ATE to determine whether it matches the expected value or is faulty. The MA module calculates the majority value of the test response data of all the cores. Given the same test patterns, all the cores from Core 1 to Core N are expected to produce the same test response data. If more than half the cores produce the same test response data M, then M is the majority value, and when M is equal to the expected data, it means that more than half the cores are not faulty. If the test response data of a core is different from the majority value, that core is recorded in the error registers as a faulty core. When the majority value is different from the expected data, it means more than half the cores were faulty. In this case, the nonfaulty cores would instead be recorded in the error registers, but this multicore chip would be discarded.

In addition, we introduce an extended version of the majority-based TAM that allows the continued testing of a multicore chip even when fewer than half the cores are good. In this version, when the majority value is the same as the expected test data, the testing of multiple cores is completed in one test process, just like in the basic version of the majority-based TAM; however, if the majority value differs from the expected test data, the faulty cores are excluded and testing restarts with the nonfaulty cores. To achieve this, more registers are needed than are needed for the basic majority-based TAM architecture. In previous research, cases of low yield in the multicore systems require additional test time. While the majority-based TAM completes the test in one cycle, the extended version of the majority-based TAM could possibly require several test cycles; however, we will show that this is unlikely.

III. PROPOSED TAM SCHEME

A. Architecture of the Majority-Based TAM

In this section, we explain the proposed TAM scheme in detail. Recall Fig. 3, which shows the architecture of the majority-based TAM for three identical cores. While the broadcast-based TAM compares the response data to the expected data in the chip, the majority-based TAM compares the response data to the majority data in the chip. In Fig. 3 example, it is supposed that each core has two scan chains, so TDI and TDO each require two test pins. Majority analyzer (MA) 1 represents the MA module for Scan chain 1 of each core; its output is connected to the first pin of TDO. The MA is a bit-wise architecture and the test responses of Scan chain 1 from every core are compared with the output of MA 1 by XOR gates. If any of the test response data in a scan chain differ from the majority data, the core producing the different data is regarded as a faulty core and is recorded in that core’s E register. During the test process, if the ATE confirms that the TDO differs from the expected data, this means that more than half the cores have faults and thus the multicore system under test will be discarded. On the other hand, if there are no differences between the TDO and the expected data throughout the test process, this means that the multicore system can be salvaged and used. When the whole test process is finished, the ATE reads the E1, E2, and E3 registers, allowing the recorded cores to be registered as faulty and the remaining cores as nonfaulty.
The proposed TAM includes a bit-wise MA designed for simple implementation and fast operation. Consider the general case in which there are $N$ cores, and at least $G$ good cores are required to sell the chip. In addition, we define $F$, as $N-G$. The MA can obtain the majority value by using the number of 1s in the test response data and comparing it to $F$. Because $G$ is larger than $N/2$ and $F$ is smaller than $G$, MA uses $F$ as a criterion for selecting the majority value. When the number of cores having 1 as the test response data is larger than $F$, the majority value of cores is 1. If the expected data is 1, the cores which have 0 as test response data are faulty cores and since the number of faulty cores is less than or equal to $F$, this chip can be used. On the other hand, if the expected data is 0, the cores which have 1 as test response data are considered as faulty cores and the number of faulty cores turns out to be larger than $F$, so this chip cannot be used. Therefore, according to the number of 1s in the test responses of cores and to $F$, the majority value is only 0 or 1, allowing for simple implementation.

B. Architecture of the MA

The MA used in the majority-based TAM is shown in more detail in Fig. 4. In the figure, $I[0]$ is the least significant bit of an input port that transfers the test response data of Core 0. Likewise, $I[i]$ is the $i$th input port of the MA; it receives the test response data of Core $i$. The main operation of the MA is to count the number of 1s at input I and then to output 1 when the number of 1s at input I is larger than $F$. In this case, the total number of cores is three ($N = 3$), so at least two cores need to be nonfaulty to salvage the multicore system ($G = 2$, $F = 1$). The output value $O$ is 1 when the number of 1s of input I is larger than one. Therefore, the output value of MA is 1 if $\{I[0], I[1], I[2]\}$ is 11X, 1X1 or X11 (here, X is a don’t care bit), and 0 otherwise. The functions of the MA are to count the number of 1s in the input port and to compare it with $F$; it can be implemented using two AND gates and two OR gates as shown in Fig. 4. The hardware complexity of the MA increases with the number of cores, but it can be implemented simply. Table I shows the truth table of the MA; it can be designed similarly to an $N$-to-1 decoder. In addition, one input port can be connected to the output port according to the mux selection signal MA_sel, allowing one specific core to be directly connected to the ATE for diagnosis or other purposes.

In the scan of one core, if any one test response differs from the majority data, that core is considered faulty and the comparison results in the XOR gates get through the OR gate to record the fault in the corresponding $E$ register ($E1$, $E2$, and $E3$ in Fig. 3). There is no functional difference between the proposed $E$ registers and the sticky-bit registers used in previous research. The $E$ registers record whether any mismatch occurs during an entire test process. After the test process ends, the data recorded in the $E$ registers are shifted out to the ATE. The information about faulty cores will be transmitted over a 1-bit wire and it can be shared with JTAG or TDO because there is no data in the test ports when the test process finishes.

C. Flow of the Majority-Based TAM

A flow diagram of the majority-based TAM is given in Fig. 5 to explain its processes and to clarify the chip’s internal functions (gray boxes in Fig. 5) and the external functions (other boxes) processed by the ATE. At the start of a test process, the ATE stimulates the design under test (DUT) with test patterns, and the TAM in the chip broadcasts these test patterns to multiple identical cores. The cores are tested by these test patterns, and the MA in the proposed TAM determines the majority value of the test responses. The TAM compares the individual test responses to their majority value and, if they are different, records this in the core’s error register as a fault. At the same time, the ATE compares the majority value to the expected test data, discarding the DUT, and ending the test process if these values do not match.

Otherwise, this test process is repeated until all the test patterns have been used, and if the test process ends without any of the majority values differing from the expected test data, the ATE then reads the error registers. Since the majority values have now been confirmed to match the expected values, any core with an error register entry of 1 is considered faulty; if the number of nonfaulty cores is counted to be larger than $G$, the DUT is considered usable. In this way, multiple cores can be tested in a way similar to the testing of a single core, that is, by observing the test response data at external test equipment, here replacing the single core’s test outputs with the majority values of the multiple cores. As shown in Fig. 5, the proposed method provides conditions for early termination of the test, saving testing time when it is determined early that the chip should be discarded.
**D. Extended Version of the Majority-Based TAM**

The extended version of the majority-based TAM is proposed for use with low-yield systems, or for cases in which aggressive derating and repairing policies are used. It can be used to test chips and to detect faulty cores exactly, even if fewer than half the total cores are good. The extended version of the majority-based TAM can complete the testing of multiple cores in one test process in cases where the majority value is always the same as the expected test data. If the majority value differs from the expected test data, the proposed TAM extracts the majority-response cores as faulty and restarts testing on the remaining cores. To achieve this, some registers are added to each of the XOR gates in the majority-based TAM architecture.

Fig. 6 shows a simple diagram of the extended version of the majority-based TAM. The added registers, denoted as \( R \) (\( R_1 \), \( R_2 \), ..., \( R_6 \)) in Fig. 6, store the test response data of the cores. In normal cases, these registers operate as buffering registers from the XOR gates to OR gates. However, if the majority value differs from the expected test data, the \( R \) registers are converted to shift registers; these shift lines are represented as dotted lines in Fig. 6. When the majority value does not match the expected test data, the output data from the \( R \) registers are compared with the expected test data in the ATE to find the faulty cores. At the same time, the output data from the \( E \) registers indicate which cores were found to be faulty before the present mismatch. The \( E \) registers are sticky-bit registers; thus, they are refreshed to restart the test process. While the MAs use TDO ports during the test, \( R \) registers use TDO ports after finishing the test. Therefore, TDO ports are shared by the MAs and \( R \) registers.

In conclusion, if more than half the cores fail concurrently in response to a specific test pattern, the extended version of the majority-based TAM restarts the test process with the nonfaulty cores, which can be identified by reading the \( E \) and \( R \) registers.

Fig. 7 is a flow diagram of the extended version of the majority-based TAM, most of the steps are the same as the majority-based TAM described previously, and the added steps are represented by dotted boxes. When the majority value and the expected test data are different in the ATE, the test process is stopped, and the ATE reads both the \( E \) and \( R \) registers to determine which cores are faulty. If the number of nonfaulty cores is larger than \( G \), the test process is restarted with only the nonfaulty cores. If not, the test process is finished and the DUT is discarded.

**E. Support for Diagnosis**

Typically, diagnosis is used to find out the location where the fault has occurred, which is important when the yield is low or when a manufacturer wants to know about the reason. For the test, TAM exports only whether the cores have a fault or not. However, for diagnosis, the test responses which are related with the fault must be exported for analyzing and finding a location of the fault. The proposed TAM can support the diagnosis in two steps: first step is to figure out which core is failing, and the second step is to run diagnosis. As described in Section III, the majority-based TAM allows one specific core to be directly connected to the ATE for diagnosis or for other purposes. Furthermore, the existing
diagnosis technologies can be applied to the proposed TAM since it offers the direct channel for the diagnosis.

IV. EXPERIMENTAL RESULTS

Several experiments were performed to verify the effectiveness of the proposed parallel TAM. The experimental results included comparisons of the proposed majority-based TAM to the schemes used in previous studies, and clearly showed the effectiveness of the majority-based TAM; the costs of testing the multicore chip with the MA are the same as the costs of testing a single core chip. The CPU cores in an OpenSPARC T2 [18] were synthesized for analyzing the hardware overhead in a real multicore system.

A. Expected Experiments

To estimate the required test time of the TAM, we use the expressions $E$ and $E_{\text{max}}$ [11]; $E$ indicates the average number of experiments that are necessary to determine the pass/fail status of each core, while $E_{\text{max}}$ denotes the maximum number of experiments to pass or fail all of the cores on all of the sellable chips.

Fig. 8 shows the worst-case test time for each of the parallel TAMs: 1) the broadcast-based TAM (Broadcast); 2) the pipeline-based TAM (Pipeline); 3) the majority-based TAM (Majority); and 4) the extended version of the majority-based TAM (Majority_ex). Both $E$ and $E_{\text{max}}$ of the majority-based TAM are always 1, since it completes the test process after one test process, finding the chip usable if less than $N/2$ cores fail and discarding it otherwise. Thus, the following experimental results of the expected experiments focus primarily on the extended version of the majority-based TAM. The extended version of the majority-based TAM can filter out at least $N/2$ cores at once; thus, in the worst case, testing $N$ cores requires $\log_2 N$ experiments. When $\log_2 N - \log_2 N_G$ experiments are conducted in which each of the directly observed cores fails, then only one last experiment will be needed to determine whether all the remaining $G$ cores are good; therefore

$$E_{\text{max}} = \log_2 N - \log_2 N_G + 1. \quad (1)$$

When $G$ is larger than $N/2$, the extended version of the majority-based TAM is essentially equivalent to the majority-based TAM, and thus the values of both $E$ and $E_{\text{max}}$ are 1. For the pipeline-based TAM, the worst case is when the primary core fails continuously, thereby requiring $N$ tests for $N$ cores. Therefore, the pipeline-based TAM is suggested to compose multitrack method for reducing the required number of tests, but the single-track has best expected experiments values when the yield of each core is larger than 70%. To show that the proposed TAM has $E$ less than that of the pipeline-based TAM, the experimental probability of the extended version of majority-based TAM is represented in

$$E(N, G, Y, P) = 1 + \sum_{i=1}^{N_G} C_N^i Y^i (1-Y)^{N-i} P_{N-i} E(i, G, Y, P)$$

when $N$ is odd

$$1 + \sum_{i=1}^{N_G} C_N^i Y^i (1-Y)^{N-i} P_{N-i} E(i, G, Y, P)$$

when $N$ is even

$$+ \frac{1}{2} C_N^G Y^G (1-Y)^{N-G} P_{N-G} E\left(\frac{N}{2}, G, Y, P\right).$$

This recursive function reflects the fact that if an experiment is executed and more than half the cores turn out to be
If \( N \) is even and exactly \( N/2 \) number of cores are faulty, the majority value can be either the expected value or the faulty value, so the last term in (2) is divided by 2. This formula facilitates direct comparison with previous research, but with one more term \( P_x \) added to represent a concurrency characteristic of the proposed TAM, described as follows. Table II shows example cases of testing multicore chips using the extended version of the majority-based TAM. The results of Test1 and Test2 are same insofar as Core1 passes, while Core2 and Core3 fail. However, in Test2 only, more than half the cores fail concurrently at TP1. In Test1, the majority value always matches the passing value and thus testing can be completed in one test process. On the contrary, the Test2 example requires one more test process for testing Core1. \( P_x \) reflects this concurrency of the proposed TAM, indicating the probability that when \( x \) cores fail, they all fail concurrently. In general, considering thousands of test patterns, \( P_x \) could be a very low probability; however, it is plausible that a number of homogeneous cores could fail the same test pattern. The following simulation results are for the worst case in which the value of \( P_x \) is 1; despite this handicap, the proposed method is shown to minimize the expected experiments.

**Table II**

**Example of Testing by an MA**

<table>
<thead>
<tr>
<th>Test cube</th>
<th>Core1</th>
<th>Core2</th>
<th>Core3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP0</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>TP1</td>
<td>P</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>TP2</td>
<td>P</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>Results</td>
<td>P</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>Test2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP0</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>TP1</td>
<td>P</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>TP2</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Results</td>
<td>P</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Fig. 9. Expected experiments by the extended majority-based TAM \((N = 8)\).

procedure is required when \( G \) is larger than \( N/2 \). While the number of expected experiments for the pipeline-based TAM increase rapidly with decreasing yield, the number of expected experiments for the extended version of the majority-based TAM increases slowly and finally decreases with decreasing yield. A worst case for the extended version of the majority-based TAM is always when exactly half the number of cores fail. Figs. 10 and 11 are similar to Fig. 9 but for 16 cores and 32 cores, respectively, with similar scaling of \( G \). In Fig. 11, when \( Y \) is 0.4, the proposed TAM is expected to need more experiments than the previous TAM. However, the case of 40% yield of each core is not significant, and considering that \( P_x \) can be much smaller than 1, the proposed TAM can be an outstanding solution to reduce the test time from that of the previous parallel TAM.

**B. Hardware Overhead**

In order to compare the hardware size of parallel TAMs, they were designed in RTL code and synthesized. On-chip comparators and error registers are commonly used in all parallel TAMs. The broadcast-based TAM is only composed of the two components. However, the pipeline-based TAM adds pipeline registers and the majority-based TAM includes MA modules. The pipeline registers have a large part of the area size. Also, additional logics are required to control the pipelining. On the other hand, MAs are simple combinational
logic since the area overhead of the MAs is smaller than that of the pipeline registers.

Figs. 12–14 show the hardware size of the parallel TAMs in terms of NAND gates, according to various cases of N and W. The majority-based TAM and the extended version of the majority-based TAM required more hardware size than the broadcast-based TAM and less than the pipeline-based TAM. The extended version of the majority-based TAM required more hardware area than the majority-based TAM due to its additional shift registers. All components of parallel TAMs (comparators, registers, and MAs) are bit-wise architecture, so the hardware size of parallel TAM is proportional to the width of TAM. The parallel TAMs for multiple cores are composed of the duplicated architecture of the components for a single core, excluding MA. Therefore, the hardware size of parallel TAMs is also proportional to the number of cores.

### TABLE III

<table>
<thead>
<tr>
<th>N</th>
<th>spc(s)</th>
<th>IEEE 1500</th>
<th>MA (W=32)</th>
<th>Overhead (c/a+b+c)*100</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2184356</td>
<td>20604</td>
<td>1181</td>
<td>0.054</td>
</tr>
<tr>
<td>8</td>
<td>4368712</td>
<td>41208</td>
<td>2665</td>
<td>0.060</td>
</tr>
<tr>
<td>16</td>
<td>8737424</td>
<td>82416</td>
<td>5423</td>
<td>0.061</td>
</tr>
<tr>
<td>32</td>
<td>17474848</td>
<td>164832</td>
<td>14204</td>
<td>0.081</td>
</tr>
</tbody>
</table>

### TABLE IV

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Test pins</td>
<td>3W</td>
<td>2W</td>
<td>2W</td>
</tr>
<tr>
<td>Test time (max)</td>
<td>T</td>
<td>(N-G+1)+T</td>
<td>T</td>
</tr>
<tr>
<td>Hardware size</td>
<td>A</td>
<td>4.73A</td>
<td>2.09A</td>
</tr>
</tbody>
</table>

Meanwhile, the hardware size of MA has a dependency on the number of cores since each MA calculates the majority test response value of multiple cores, rather than one. However, the area overhead of MA is negligible and increases linearly as the number of cores.

CPU cores in an OpenSPARC T2 [18] were synthesized by the Synopsys 90-nm generic library [19] for analyzing the hardware overhead in a real multicore system. Table III shows the hardware overhead of the majority-based TAM. The spc (a) is the SPARC processor core module; the values given are the number of gates. IEEE 1500 (b) represents the size of the IEEE 1500 standard wrapper for the spc module, which has 477 ports. MA (c) is the size of the proposed TAM, when its width is 32 bits. The overhead in Table III indicates the percentage of hardware overhead of the proposed TAM, considering both the IEEE 1500 standard wrapper for the spc module and the proposed TAM in the total amount. The hardware overhead of the proposed TAM is 0.06% when there are eight cores and the TAM has a width of 32 bits. This experimental result shows that the area overhead of the proposed TAM is negligible compared with the number of gates of a modern multicore processor system, which is much more than a million.

In Table IV, W is the number of test pins for a single core, T is the test time for a single core, and A is the number of gates for the broadcast-based TAM. The majority-based TAM and the extended version of the majority-based TAM need the same number of test pins as the TAM for a single core (2W = W for TDI + W for TDO), but the broadcast-based TAM needs more test pins for receiving the expected data (3W = W for TDI + W for TDO + W for EDI). The majority-based TAM needs the same test time as the TAM for a single core, but the pipeline-based TAM needs more test time if the primary core has any faults. The extended version of the majority-based TAM needs more test time when more than half
the cores have faults. The hardware size represents the relative size of parallel TAMs when \( N = 8 \) and \( W = 32 \); as mentioned previously, the majority-based TAM and the extended version of the majority-based TAM are larger than the broadcast-based TAM and smaller than the pipeline-based TAM. A synthesized clock of the proposed TAM, which means the available test clock speed of the TAM without slack, is 450 MHz when \( N = 8 \) and \( W = 32 \). This clock speed is enough to perform scan testing since the general scan shift clock cycle is much lower than 100 MHz. The clock speed of the proposed TAM has no problem and no relation to use a fast capture clock cycle for at-speed test or use other DFT applications.

V. Conclusion

In this paper, we described a novel parallel TAM for parallel testing of a multicore system. All the cores can be tested simultaneously using the same test pins and test time as required for testing a single core. An MA module is designed that uses the majority value to test multiple identical cores; if the test response data of one core differs from the majority value, that core is considered faulty. From the standpoint of the ATE, the majority-based TAM allows multicore chips to be tested for only the cost of a single-core chip, while requiring few modifications. Experimental results show that the proposed TAM has a minimized number of test pins and test time with sufficient operational speed and negligible hardware overhead. The majority-based TAM is only related to the delivery of test response data and it can be compatible and improved with the existing DFT technologies. Since the proposed TAM is flexible in design, configuration, and application, it is an attractive and efficient solution for testing multiple identical cores.

References


Taewoo Han received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2008, where he is currently pursuing the Ph.D. degree. He has been involved in BISR design and the design of a navigator path search accelerator. His current research interests include memory test and repair, multicore test and repair, and reliable design methodologies.

Inhyuk Choi received the B.S. degree in electrical and electronics engineering from Yonsei University, Seoul, Korea, in 2009, where he is currently pursuing the M.S. and Ph.D. degrees in electrical and electronics engineering. His current research interests include SoC design, design for testability, and system-level test and validation.

Sungho Kang received the B.S. degree in control and instrumentation engineering from Seoul National University, Seoul, Korea, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin, Austin, TX, USA, in 1992. He was a Research Scientist with the Schlumberger Laboratory for Computer Science, Schlumberger Inc., Houston, TX, USA, and a Senior Staff Engineer with Semiconductor Systems Design Technology, Motorola Inc., Schaumburg, IL, USA. Since 1994, he has been a Professor with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul. His current research interests include VLSI/SOC design and testing, design for testability, and design for manufacturability.