A Hierarchical Test Scheme for System-on-Chip Designs

Jin-Fu Li, Hsin-Jung Huang, Jeng-Bin Chen, Chih-Ping Su, Cheng-Wen Wu,
Chuang Cheng*, Shao-I Chen*, Chi-Yi Hwang*, and Hsiao-Ping Lin*
Department of Electrical Engineering
National Tsing Hua University
Hsinchu, Taiwan
*Faraday Technology Corp.
Hsinchu, Taiwan

Abstract

System-on-chip (SOC) design methodology is becoming the trend in the IC industry. Integrating reusable cores from multiple sources is essential in SOC design, and different design-for-testability methodologies are usually required for testing different cores. Another issue is test integration. The purpose of this paper is to present a hierarchical test scheme for SOC with heterogeneous core test and test access methods. A hierarchical test manager (HTM) is proposed to generate the control signals for these cores, taking into account the IEEE P1500 Standard proposal. A standard memory BIST interface is also presented, linking the HTM and the memory BIST circuit. It can control the BIST circuit with the serial or parallel test access mechanism. The hierarchical test control scheme has low area and pin overhead, and high flexibility. An industrial case using this scheme has been designed, showing an area overhead of only about 0.63%.

1. Introduction

With the advent of the core-based system-on-chip (SOC) and reuse methodologies, cores from different sources can be integrated into a single chip. Compared with the traditional multi-chip system-on-board, benefits of the SOC include higher performance, lower power consumption, smaller size, etc. Different types of core (such as CPU, DSP, SRAM, flash memory, ADC, DAC, and PLL) are usually incorporated into a single SOC design [4]. Moreover, cores sometimes come in hierarchical compositions, i.e., one complex core is composed of multiple simple ones [13]. Although the SOC design process is analogous to the board design process, their manufacturing test methods are quite different [12]. For a board, the ICs from the providers are already tested. Normally only the interconnects on the board need to be tested for manufacturing defects. In an SOC design, however, the cores are not yet manufactured and tested. The core integrator is responsible for manufacturing and testing the chip, including the cores on it. SOC testing includes core internal test, core external test, core test knowledge transfer, test access, test integration and optimization, etc. [13]. Apparently, testing SOC designs is more challenging than testing board designs.

Recently, research results have been reported for testing SOC using IEEE 1149.1-compliant test architectures [11, 3, 7]. In [11], a systematic solution for accessing embedded JTAG (IEEE 1149.1) cores hierarchically has been proposed, where a TAP linking module was designed to handle the interaction between the upstream TAP controller and downstream TAP controllers. In another work, a Hierarchical Test Access Port (HTAP) architecture has been reported [3]. This architecture supports test access to embedded JTAG cores with a snoopy TAP. The pin requirement and behavior of the TAP controller of this design is fully compatible with IEEE 1149.1. However, these methods do not handle the test control of the IEEE P1500 cores. In [2], a central TAP controller consisting of an 1149.1-like TAP finite state machine and a counter is used to control P1500 and TAPed cores. Also, a hierarchical test control mechanism reported in [7] provides hierarchical test capability for 1149.1 and P1500 cores, but ten additional pins are required to operate the major component, Central Test Controller, and only one core can carry out the testing task at a time. In all these works, controlling memory cores with built-in self-test (BIST) was not considered.

The purpose of this paper is to present a hierarchical test architecture for managing the test operations of diversified cores, including the 1149.1 wrapped, P1500 wrapped, and BISTed memory cores. An 1149.1-based hierarchical test manager is proposed, which also provides the P1500 test control signals. A memory BIST interface is also developed, providing both serial and parallel access ports for the BIST circuits. This approach has the advantages of low area and pin overhead, and high flexibility. The proposed SOC test scheme has been implemented on an industrial case. The area overhead is only about 0.63%, which is very small as compared with that of the IEEE P1500 test wrappers for the cores (5.1%).

2. IEEE P1500 Scalable Architecture

To solve the problems mentioned above and for easy test automation, a standard test interface for the cores is

required. A scalable architecture [9] was proposed by the IEEE P1500 Standard Working Group. The IEEE P1500 Standard proposal tries to standardize the Core Test Wrapper and the Core Test Language (CTL). The architecture consists of the user-defined parallel test access mechanism (TAM) for delivering the test patterns and responses in parallel, standard core test wrappers that can isolate the cores and provide different test modes, and a user-defined test controller for controlling the wrapper and TAM [9]. Serial test access can always be done by using the Serial Interface Layer (SIL) provided by the P1500 Wrapper, which is mandatory.

The IEEE P1500 Core Test Wrapper contains the following elements: 1) the *wrapper instruction register* (WIR) that can handle various test modes defined by mandatory and user instructions; 2) the *wrapper interface port* (WIP) containing the wrapper control signals; 3) the *wrapper bypass register* (WBY, normally 1-bit) connecting directly the *wrapper serial input* (WSI) to the *wrapper serial output* (WSO), used for bypassing the current core when we are testing others; 4) the *wrapper boundary register* (WBR) consisting of the *wrapper cells* that wrap the normal IO pins of the core, providing control, observation, and isolation for the core, in addition to its normal function; and 5) optional TAM input/output ports.

3. Hierarchical Test Scheme for SOC

3.1. Hierarchical Testing

Figure 1 shows the top-level architecture of the proposed hierarchical test scheme by a system chip example, which consists of a 1149.1 core, a P1500 core, a BISTed memory core, and a hierarchical core with two P1500 cores in itself. Four components are needed to perform the test task.

1. A *hierarchical test manager* (HTM) is used to handle the test operations at each level. Its upstream IOs are compatible with the IEEE 1149.1 TAP [6]. They are the TAP control signals (collectively denoted as TCS_UP), including TCK_UP (test clock), TMS_UP (test mode selection), TRST_UP (test reset), and the serial test access IOs (TDI_UP and TDO_UP). Its downstream IOs consist of the P1500 control signals (PCS), TCS_DN (including TCK_DN, TMS_DN, and TRST_DN), the serial test access IOs (TDI_H and TDO_H) for the HTMs at the next level, and the serial test access IOs (TDI_C and TDO_C) for the cores at the same level. The details will be described in Sec. 3.2.

2. A *TAM* provides a higher test data transport capacity for the P1500 and BISTed memory cores.

3. A *wrapper control interface* (WCI) is used to decode the PCS signals into the WIP signals for operating the test wrapper. It is composed of a decoder and a register. The registers of all the WCIs at the same level are serially connected into a *Selection Register*, which is one of the data registers of the HTM. The details will be described in Sec. 3.3.

4. A *memory BIST interface* (MBI) handles the BIST operations through the serial test access port or TAM. It

can perform the BIST operations for multiple memory cores concurrently, reducing the effort in chip-level test plan, test pin requirement, test data volume, etc. The details will be discussed in Sec. 4.
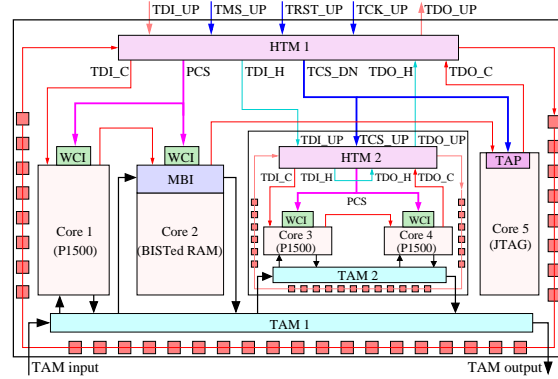

Figure 1. The proposed hierarchical test scheme.

A simple test procedure is shown below, assuming conformance and integrity tests have been done. Note that the corresponding HTM instructions (to be described later) must be updated into the HTM instruction registers before performing each step of the test procedure.

1. *Test Configuration*: the instructions of the wrappers and MBIs are loaded, configuring each core into a specific test mode or the bypass mode.

2. *TAM Specification*: the user needs to specify the cores to be tested by the TAM. This is done by shifting a binary sequence into the Selection Register of each HTM. The most significant bit (MSB) of the Selection Register in a lower-level HTM is to specify whether or not the lower-level TAM is connected to the TAM at the current level. Other bits are used for specifying the connection between the cores and the TAM at the same level. For example, there is a three-bit Selection Register (bit2, bit1, bit0) in the HTM2 of Fig. 1: bit2 specifies whether TAM2 is connected to TAM1 or not, and bit1 and bit0 specify whether Core3 and Core 4 are connected to TAM2 or not, respectively. Note that the TAM selection also can be specified by the instructions of the HTMs, wrappers, or MBIs. In that case, the Selection Register can be removed and this test step can be omitted.

3. *Test Transportation*: the test patterns are imported to the cores under test and their test results are exported by the TAM and serial IOs according to the test configuration.

3.2. Hierarchical Test Manager

The HTM architecture is depicted in Fig. 2, which consists of a *Test Manager* and a *Hierarchical Test Interface*. The Test Manager is extended from the TAP Controller [6].

It is composed of the Finite State Machine (FSM), Instruction Register, Bypass Register, Activation Register, Boundary Register, and Wrapper Control Encoder (WCE). The main difference between the TAP Controller and the Test Manager is the WCE. It generates the P1500 control signals (i.e., PCS, including PCS0, PCS1, and PCS2) according to the instruction and the FSM state. It greatly reduces the number of control signals for an SOC with many P1500 cores. The Hierarchical Test Interface consists of a Switch Box, which specifies the connection of the serial test access IOs according to the control signals from the Test Manager.
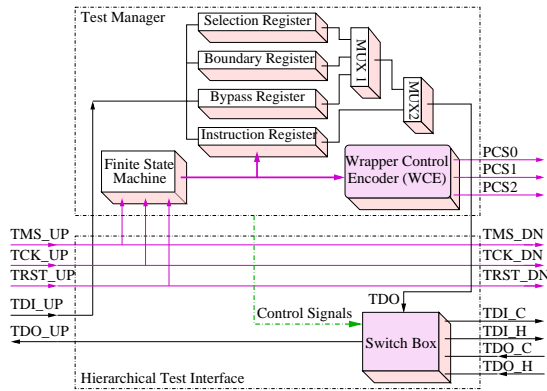

Figure 2. The Hierarchical Test Manager.

The state diagram of the FSM is depicted in Fig. 3 [6]. By controlling the TMS_UP input sequence, different states of the FSM can be reached. In the figure, the states in Group 2 control the operations of the HTM Instruction Register, while the states in Group 1 can be defined in a different way under different instructions. The key instructions are described next.
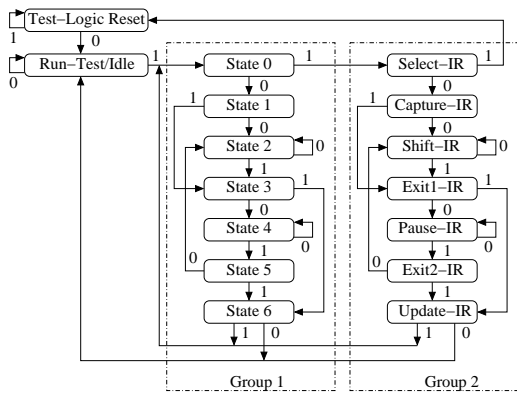

Figure 3. The FSM state diagram.

The BYPASS, EXTEST, and SAMPLE/PRELOAD instructions are the same with the 1149.1 mandatory instructions [6]. When these instructions are used, the states in Group 1 of the FSM are interpreted as (State 0, State 1,..., State 6) = (Select-DR, Capture-DR, Shift-DR, Exit1-DR, Pause-DR, Exit2-DR, Update-DR), i.e., the function of the HTM is the same as the TAP Controller.

The LSELECTWIR, LTAMSELECT, and LSELECTWR instructions are the local P1500 instructions. Under these instructions, the Wrapper Boundary Register is formed only by the cores at the same level with the HTM, done by configuring the Switch Box (to be discussed later). The LSELECTWIR instruction forces the HTM into the Test Configuration phase, where (State 0, State 1,..., State 6) = (Shift-WIR, Capture-WIR, Shift-WIR, Shift-WIR, Update-WIR, Capture-WIR, Update-WIR), and the Bypass Register is selected. The WCE encodes the PCS signals according to these HTM state. The wrappers and MBIs shift and update the instructions into the Instruction Registers by controlling the TMS_UP. The LTAMSELECT instruction enables the Selection Register, such that a binary sequence can be shifted into the register to specify the cores connected to the TAM. The LSELECTWR instruction is for test transportation, under which (State 0, State 1,..., State 6) = (Shift-WR, Capture-WR, Shift-WR, Shift-WR, Update-WR, Capture-WR, Update-WR), and the Bypass Register is selected. The WCE encodes the PCS signals according to these states, and the WCI (to be discussed in Sec. 3.3) decodes the PCS signals into the WIP signals. The test patterns can then be transferred to the cores under test by controlling TMS_UP.

Table 1 shows the WCE outputs (PCS0, PCS1, PCS2) for the respective instructions and FSM states. Note that PCS0 is dependent on the instructions, while PCS1 and PCS2 are determined by the FSM states, such that we can specify the operations by TMS_UP. If it is an 1149.1 mandatory instruction, then (PCS0, PCS1, PCS3) = (0, 0, 0).

Table 1. PCS encoding.

Instruction	FSM State	PCS0	PCS1	PCS2
LSELECTWIR	Shift-WIR	1	0	0
GSELECTWIR	Update-WIR	1	1	0
	Capture-WIR	1	0	1
LSELECTWR	Shift-WR	0	0	0
GSELECTWR	Update-WR	0	1	0
	Capture-WR	0	0	1
LTAMSELECT		0	1	1
GTAMSELECT		0	1	1
1149.1		0	0	0

Note that for P1500 instructions, only three different states are defined in Group 1, i.e., Shift-WR, Update-WR, and Capture-WR. They are enough for the test wrappers. For example, to test a core with internal scan, we can use the loop (State 2→State 3→State 4→State 5→State 2) to apply the tests.

Finally, the GSELECTWIR, GTAMSELECT, and GSELECTWR instructions are global P1500 instructions. Their functions are similar to those of the local P1500 instructions discussed above, except that they are for all HTMs.

Table 2 lists all possible configurations of the Switch Box. For example, if the BYPASS, EXTEST, SAMPLE/PRELOAD, or LTAMSELECT instruction is loaded, then the TDO is connected to TDO_UP, both TDI_C and TDI_H are 0, and both TDO_C and TDO_H are disconnected (don't-care). The configurations for LTAMSELECT and GTAMSELECT are different from those of other P1500 instructions. The reason is that only the Selection Registers of the HTMs are needed to be configured. Note that when the FSM is in a Group 2 state, the Switch Box is

forced to the same configuration as GTAMSELECT. All HTMs are serially connected into a chain so that we can load the desired instructions into the HTMs by controlling the TMS_UP pin.

Table 2. Switch box configurations.

Instruction	Configuration
BYPASS, EXTEST, SAMPLE/PRELOAD, LTAMSELECT	TDO→TDO_UP, 0→TDI_C, 0→TDI_H, TDO_C→x, TDO_H→x
GTAMSELECT	TDO→TDI_H, 0→ TDI_C, TDO_H→TDO_UP, TDO_C→x
LSELECTWIR, LSELECTWR	TDO→TDI_C, TDO_C→TDO_UP, 0→TDI_H, TDO_H→x
GSELECTWIR, GSELECTWR	TDO→TDI_C, TDO_C→TDI_H, TDO_H→TDO_UP

3.3. Wrapper Control Interface

The Wrapper Control Interface (WCI) decodes the PCS signals into the Wrapper signals—SelectWIR, ShiftWR, UpdateWR, and CaptureWR. The WCI is composed of a 1-bit register and a decoder. All the registers of the WCIs at the same level are connected serially into the Selection Register. Each bit of the register specifies whether the corresponding core is connected to a TAM or not. The PCS0 signal is directly connected to SelectWIR. The ShiftWR, UpdateWR, CaptureWR signals are 1 when (PCS1, PCS2) = (0,0), (1,0) or (0,1). This reduces the routing overhead since the PCS signals are broadcast to the cores from the HTM.

4. Memory BIST Interface (MBI)

Figure 4(a) shows the proposed MBI, which consists of the Instruction Register, Bypass Register, Monitor Register, Status Register, and Programmable Switch. The MBI Serial Input (MSI) and MBI Serial Output (MSO) are used to transfer the test data to and from the registers. The MBI Interface Port (MIP) contains similar control signals and has a similar function as the P1500 WIP. Although the P1500 Wrapper can wrap embedded memories, the MBI does not wrap the functional I/Os since they are already isolated by the BIST circuit, removing the cost of WBR. The Programmable Switch determines whether the BIST IOs are handled by the TAM or the MBI. In the latter case, the Monitor Register and Status Register are used to observe the BIST outputs. The Monitor Register is for monitoring the error flag (indicating whether a memory fault is detected or not) or exporting the diagnostic data (consisting of the faulty cell/word address, March syndrome, and Hamming syndrome [10, 8]) on-the-fly. The Status Register records key status values, such as the FAIL (go/no-go) output from the BIST circuit.

The BYPASS instruction is also the default instruction of the MBI, as in P1500. The RUN_BIST instruction runs the BIST circuit in the test mode. The Monitor Register is connected between the MSI and MSO. The RUN_DIAGN instruction forces the BIST circuit to be operated in the diagnosis mode. The diagnosis data is exported through the
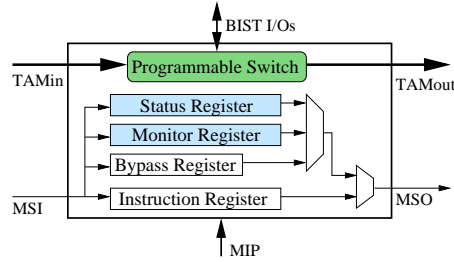

Figure 4. The MBI architecture.

Monitor Register. Note that only one memory core can be in the diagnosis mode at any time. However, if the TAM is used to handle the BIST operations, then the width of the TAM determines the number of memory cores that can be diagnosed concurrently. The EXPORT_STATUS instruction is used to export the content of the Status Register, and the TAM_CONTROL instruction configures the Programmable Switch to connect the BIST IOs to the TAM.

5. Experimental Results

We now estimate the gate count overhead of the hierarchical test scheme: Total gates = (HTM gates × No. HTMs) + (MBI gates × No. MBIs) + (WCI gates × No. WCIs) + (TAM gates × No. TAMs) + (Gates in test wrappers). The gate counts of the HTM and MBI are affected by the number of registers and their widths. The routing overhead usually dominates the TAM area. The TAM architecture [1] also affects the result. We have implemented a Wrapper cell library, providing different kinds of wrapper cell for various applications [5]. We found that the test wrapper overhead is mainly determined by the IO pins of the core. We have implemented the proposed hierarchical test scheme in an industrial design containing three cores—two P1500 cores (i.e., Core 1 and Core 2) and a hierarchical core (i.e., HCore)—using a 0.25μm CMOS technology. Core 1 and Core 2 have one and two BIST circuits for their embedded memories, respectively. HCore itself contains two P1500 cores. The logic circuits are all tested by internal scan. Core 1, Core 2, and HCore have 8, 16, and 2 scan chains, respectively. Also, an 8-bit daisy-chain TAM [1] is used to transport the test data. Table 3 shows the gate count of each hierarchical test component. The gate count of the HTM does not include the Wrapper Boundary Register. Here, only the logic circuit of the TAM is estimated. The 4-bit and 3-bit Instruction Registers are used for the HTM and MBI, respectively.

Table 3. Gate counts of the hierarchal test components.

	HTM	WCI	MBI	TAM
Gate count	828	33	198	183

We define the hardware overhead HO = (wrapped core area−core area)/(core area). The clocks, reset signals, etc., are normally not wrapped. Only the wrapped IOs are considered. Table 4 shows the HO values for the core test wrappers. As shown in the table, the total number of IOs (IO) in the design is 1209, and there are 1135 wrapped IOs (WIO). The single-flip-flop cells are used to implement the wrappers. The fourth row lists the core area (CA), and the next row shows the corresponding wrapped core area (WCA). Core 1 has the smallest HO (about 1.4%). Core 2 has the largest number of IOs, resulting in the largest HO.

Table 4. Hardware overhead of the wrappers.

	Core 1	Core 2	HCore	Total
IO	224	885	100	1209
WIO	183	864	88	1135
CA	254087	146169	17266	417822
WCA	257660	161836	19330	438826
HO	1.4%	10.7%	12%	5.1%

Statistics of the internal scan chains are shown in Table 5. The scan length denotes the length of the longest scan chain in each core. Core 1 has the lowest fault coverage and the most test patterns. In contrast, Core 2 has the highest fault coverage and fewer test patterns. The hierarchical test scheme uses two HTMs, six WCIs, and three MBIs. The area overhead (in terms of gate count) of all the components is about 2631 gates. Therefore, the HO is about 0.63%. This is very small compared with the HO of the wrappers (5.1%). Note that the fault coverage numbers are low because the designs have many latches that are not scanned. This scheme only requires 5 extra control pins. If the number of test pins is W, then the number of bits left for TAM is $(W-5)$. In [7], 10 control pins are required, i.e., the TAM width is $W-10$. In comparison, our scheme reduces the test time with the same number of test pins. Moreover, in [7], concurrent testing of multiple cores can not be done.

Table 5. Statistics of scan-based testing.

	Core 1	Core 2	HCore
Scan chains	16	8	2
Scan length	319	1000	452
Test patterns	422	334	103
Fault coverage	79.84%	90.51%	84.96%

6. Conclusions

A hierarchical test scheme for SOC designs has been proposed, which is realized by four major components: the Hierarchical Test Manager (HTM), Test Access Mechanism (TAM), Wrapper Control Interface (WCI), and Memory BIST Interface (MBI). The HTM handles all the test operations of P1500, 1149.1, and BISTed memory cores. The WCI is responsible for decoding the control signals from the HTM and feeding them to the P1500 WIP. The MBI provides a serial/parallel control mechanism for the memory

BIST circuits, and the memory cores can be tested concurrently. It has high flexibility, low pin-count overhead, and low area overhead. An industrial case has been designed using the proposed scheme. Results show that the area overhead of this design is only about 0.63%, which is very small as compared with that of the IEEE P1500 test wrappers for the cores (5.1%).

References

[1] J. Aerts and E. J. Marinissen. Scan chain design for test time reduction in core-based ICs. In *Proc. Int. Test Conf. (ITC)*, pages 448–457, 1998.

[2] M. Benabdenbi, W. Maroufi, and M. Marzouki. Testing TAPed cores and wrapped cores with the same test access mechanism. In *Proc. Design, Automation and Test in Europe (DATE)*, pages 150–155, Munich, Mar. 2001.

[3] D. Bhattacharya. Hierarchical test access architecture for embedded cores in an integrated circuit. In *Proc. IEEE VLSI Test Symp. (VTS)*, pages 8–14, 1998.

[4] R. K. Gupta and Y. Zorian. Introducing core-based system design. *IEEE Design & Test of Computers*, 14(4):15–25, Oct.-Dec. 1997.

[5] H.-J. Huang, J.-F. Li, J.-B. Chen, C.-P. Su, C.-W. Wu, C. Cheng, S.-I. Chen, C.-Y. Hwang, and H.-P. Lin. Test wrapper design automation for system-on-chip. In *Proc. 12th VLSI Design/CAD Symp.*, Hsinchu, Aug. 2001.

[6] IEEE. *IEEE 1149.1 Standard Test Access Port and Boundary-Scan Architecture*. IEEE Standards Department, Piscataway, May 1990.

[7] K.-J. Lee and C.-I. Huang. A hierarchical test control architecture for core based design. In *Proc. Ninth IEEE Asian Test Symp. (ATS)*, pages 248–253, Taipei, Dec. 2000.

[8] J.-F. Li and C.-W. Wu. Memory fault diagnosis by syndrome compression. In *Proc. Design, Automation and Test in Europe (DATE)*, pages 97–101, Munich, Mar. 2001.

[9] E. J. Marinissen, Y. Zorian, R. Kapur, T. Taylor, and L. Whetsel. Towards a standard for embedded core test: An example. In *Proc. Int. Test Conf. (ITC)*, pages 616–626, 1999.

[10] C.-W. Wang, C.-F. Wu, J.-F. Li, C.-W. Wu, T. Teng, K. Chiu, and H.-P. Lin. A built-in self-test and self-diagnosis scheme for embedded SRAM. In *Proc. Ninth IEEE Asian Test Symp. (ATS)*, pages 45–50, Taipei, Dec. 2000.

[11] L. Whetsel. A IEEE 1149.1 base test access architecture for ICs with embedded cores. In *Proc. Int. Test Conf. (ITC)*, pages 69–78, 1997.

[12] Y. Zorian. Test requirements for embedded core-based systems and IEEE P1500. In *Proc. Int. Test Conf. (ITC)*, pages 191–199, Oct. 1997.

[13] Y. Zorian, E. J. Marinissen, and S. Dey. Testing embedded-core-based system chips. *IEEE Computer*, 32(6):52–60, June 1999.