Notice of Violation of IEEE Publication Principles

"Area Efficient TAM Controller and Wrapper Design for Embedded Cores"
by G. Rohini and S. Salivahanan
in the 2009 International Conference on Control, Automation, Communication and Energy Conservation

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper copied substantial portions of text and figures from the papers cited below. The original text was copied without attribution (including appropriate references to the original author(s) and/or paper title) and without permission. The misconduct was committed specifically by G. Rohini (First author), and her actions were done without the knowledge or approval of S. Salivahanan (Second author).

Due to the nature of this violation, reasonable effort should be made to remove all past references to this paper, and future references should be made to the following article:

“Developing Test Environment for Embedded Cores-Based System-on-a-Chip (SOC)
By S. Das, D. Biswas, E.M. Petriu, M.H. Assaf and M. Sahinoglu
in the Instrumentation and Measurment Technology Conference - 2005

Area Efficient TAM Controller And Wrapper Design For Embedded Cores

G. Rohini, S. Salivahanan

Abstract– IEEE 1500 is a standard under development which intends to improve ease of test reuse and test integration with respect to the core-based SoCs. The subject paper proposes developing test environment and test methodologies for digital embedded cores based system-on-a-chip (SoC). The digital cores used in the study were constructed from ISCAS 85 combinational and ISCAS 89 sequential benchmark circuits. The wrapper that separates the core under test from other cores is assumed to be IEEE 1500-compliant. The test access mechanism plays an important role in transporting the test patterns to the desired core and the core responses to the output pin of the SoC. The faults were injected using a fault simulator that generates tests for the core. The cores and test access mechanism were described using VHDL. The test access mechanism (TAM) provides the connection between the test sources, cores, and test sinks, and is crucial in any SoC design. The outcome was the fault coverage of all the cores being tested. Area overhead and power consumption are taken into account in our scheme. Some experiment results based on a sample SoC are reported, showing the effectiveness of the proposed approach in terms of area overhead.

Keywords – Built-in self-testing (BIST), embedded cores-based system-on-a-chip (SoC), sequential circuits, test access mechanism (TAM),test pattern generator(TPG), VHDL, wrapper.

1 INTRODUCTION

Large-scale integration has added enormous complexity to the process of testing modern digital circuits. Besides, during the past several years, integrated circuit technology evolved from chip-set philosophy to embedded cores based system-on-a-chip (SoC) concept[1], which simply refers to an IC, designed by stitching together multiple stand-alone VLSI designs to provide full functionality for an application. Though many aspects of these embedded cores-based systems and SoC are still evolving, they are revolutionizing the electronics industry. These innovations are already on their way to the next generation of cell phones, multimedia devices, and PC graphics chipsets. The cores-based design, justified by the necessity to decrease time-to-market, has created a host of challenges for the design and test community[4],[9]. Unlike traditional test design approaches, SoC makes it impossible to establish the demarcation lines between design and test. For mixed-signal devices and complex digital cores, engineers must use design tools that let them incorporate testability[7] early in the design process. Though the use of cores in an SoC serves a wide range of applications, they are far too complex to be tested by traditional methods. In practice, combinations of test methodologies such as functional test, full-scan test, logic BIST, RAM BIST, IDDQ, etc. are normally used for SoCs. But difficulties surround implementation of the methods for testing cores and SoCs as a whole. A wide range of circuits traditionally comes within the definition of SoCs including microprocessors and microcontrollers. The core test integration is a complex problem – the chip integrator can modify the test and add some design for test (DFT) and built-in self-test (BIST) features, if necessary. The fundamental items of interest in core test is access, control, and isolation, and these are the issues which were addressed by the IEEE Technical Council on Test Technology Working Group 1500[2] entrusted with the responsibility of developing standard architecture for their solution. The embedded core test requires, in general, hardware components like wrapper around the core, a source and a sink for test patterns (on-chip or off- chip) and an on-chip test access mechanism (TAM) to connect the wrapper to the source or sink. The cores could be without boundary scan or with boundary scan. For design and test reuse, ASIC manufactures have suggested certain characteristics. In general, different DFT and BIST schemes like scan, partial scan, logic BIST and scan-based BIST are used to test various logic blocks within a SoC like microprocessor or microcontroller. However, the main problem is still the resulting area overhead and performance penalties. Structural test methods like scan and BIST are desirable for test reuse, portability, and test integration into the SoC test set. The TAM includes on-chip test generation logic for cores with BIST. In this paper, test methodologies are proposed for embedded cores-based system-on-a-chip (SoC) digital systems comprising of wrapper and TAM. The cores

- *G.Rohini is with St.Joseph's College of Enginering, Chennai, PO 600119 . E-mail: rohini_sneha23@yahoo.co.in.*
- *SSalivahanan is with SSN College of Engineering, Kalavakkam, PO 603110. E-mail: salivahanans@.hotmail.com*

considered in the study were constructed from ISCAS 85 combinational and ISCAS 89 sequential benchmark circuits. The fault model used is the conventional single stuck-fault model. The wrapper separates the core under test from other cores, which is assumed to be IEEE 1500-compliant[2]. The TAM, a vital role in transporting the test patterns to the desired core and the core responses to the output pin of the SoC. The faults were injected using a fault simulator that also generates the test sets for the cores. The TAM was implemented as a plain signal transport medium, which is shared by all the cores in the SoC. The automatic fault simulator generates tests for the core described at the gate and flip-flop level. The simulation process is completely automatic, and requires no intervention from the designer during the test generation process. This paper describes the architecture of the wrapper and test access mechanism, applications of the fault simulator, together with models of the SoCs being used, based on application environment. Some partial simulation results using the designed cores are also provided to demonstrate the feasibility of the proposed implementations.

2 1500-BASED SoC TEST INTEGRATION ARCHITECTURE

The IEEE 1500 Std wrapper[2][3] has various modes of operation. There are modes for functional (nontest) operation, inward facing (IF) test operation, and outward facing (OF) test operation. Different test modes also determine whether the serial test data mechanism (WSI–WSO) or the parallel test data mechanism (WPI–WPO), is being utilized. Instructions loaded into the WIR, together with the IEEE 1500 wrapper signals, determine the mode of operation of the wrapper and possibly the core itself. All instructions that establish test modes that utilize the parallel port WPI and WPO are optional, as the presence of this port is optional. Furthermore, IEEE Std 1500 also allows for user-defined instructions. IEEE Std 1500[2] has a set of instructions that are defined to use only the serial interface (WSP) and a corresponding set of instructions that are defined for the parallel interface. IEEE Std 1500 must allow accessibility to test the core.

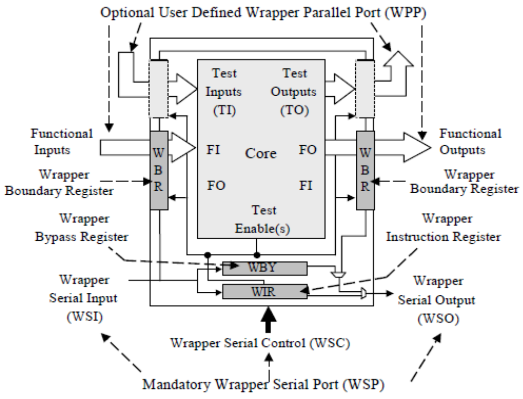
Optional User Defined Wrapper Parallel Port (WPP)
Test Inputs (TI)
Test Outputs (TO)
Functional Inputs
Functional Outputs
W B R
W B R
FI
Core
FO
FO
FI
Wrapper Boundary Register
Wrapper Boundary Register
Test Enable(s)
Wrapper Bypass Register
Wrapper Instruction Register
WBY
WIR
Wrapper Serial Input (WSI)
Wrapper Serial Output (WSO)
Wrapper Serial Control (WSC)
Mandatory Wrapper Serial Port (WSP)

Fig..1: Standard IEEE 1500 wrapper components

There is one main core test instruction—Wx_INTEST (user-specified core-test instruction)—that is flexible enough to allow any core test to execute. There are two other instructions that are mandatory: an instruction for functional mode (WS_BYPASS) and an instruction for external test mode (WS_EXTEST). WS_BYPASS puts the wrapper into the bypass configuration and allows access to all functional terminals of the core shown in Fig 1. WS_EXTEST is the serial EXTEST configuration of the wrapper. Even if there is a WP_EXTEST mode (for parallel access), there must still be a WS_EXTEST instruction capability. The signal connected to the WRCK terminal is a dedicated clock used to operate IEEE 1500 functions.

3 TEST ACCESS MECHANISM (TAM) AND WRAPPER

The design of test access mechanism (TAM)[4] and test wrapper is of critical importance in terms of system integration since they directly impact hardware overhead, test time, and tester data volume. The main issues in this context are wrapper optimization, core assignment to TAM wires, sizing of the TAMs, and TAM wire routing. There are two important concepts related to TAM, viz. test pattern source and sink, and core wrapper. The test pattern source is responsible for generating the test vectors or test stimuli for the desired core under test. The test pattern sink compares the fault-free response to the faulty response of the core under test. The test pattern source and sink can be built on-chip or off-chip. There are several ways to design and implement a TAM[5]. Some of the common TAM architectures are: 1) daisy chained TAMs (that use serial shifting of test data); 2) bussed TAMs (based on use of complex protocols); 3) direct access TAMs (for cores with many inputs and outputs); and 4) multiplexer (MUX)-based direct access TAMs. In this paper, MUX-based direct access TAM architecture, as shown in Fig. 2 is implemented. The TAM is used to drive the test vectors from the test source[5][6], that is, from the fault simulator to the desired core under test and to transport the test response from the core back to the fault simulator. The selection of the core in the SoC was implemented as part of the TAM architecture. The width of the TAM by the core that has the maximum

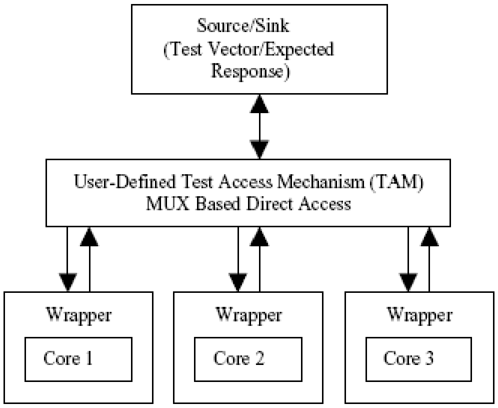
Source/Sink (Test Vector/Expected Response)
User-Defined Test Access Mechanism (TAM) MUX Based Direct Access
Wrapper
Core 1
Wrapper
Core 2
Wrapper
Core 3

Fig .2: TAM Architecture

number of input/output (I/O) pins within the SoC is determined. There are other issues for consideration at this phase, viz. the bandwidth of the TAM versus the cost of extra wires needed for its implementation, total test time depending on the TAM bandwidth, test vectors from the source, and ultimately test data for the individual core. The obvious mechanism to make embedded cores testable from the IC pins[10] is to make the core under test directly accessible from the IC inputs. Though this approach is mostly practiced for embedded memory cores, many block-based ASICs also use this test strategy.

4 TAM CONTROLLER DESIGN

As shown in Fig. 3, TAM controller is used to provide the dynamic control signals to wrapper[4], i.e., WIP. WIP is composed of six signals: WRCK, WRSTN, SelectWIR, ShiftWR, CaptureWR and UpdateWR. At the exception of WRSTN, all WIP signals are active high.

- WRCK (Wrapper Clock): The dedicated test clock.
- WRSTN (Wrapper Reset): The dedicated asynchronous reset signal which resets wrapper instruction register (WIR) and puts the wrapper in normal operation mode.
- SelectWIR: Selects whether WIR or one of wrapper data registers (WDR) is exclusively connected between WSI and WSO.
- ShiftWR, CaptureWR: When the corresponding signal is asserted, a shift or capture operation will occur on the rising edge of WRCK.
- UpdateWR: When it is asserted, an update operation will occur on the falling edge of WRCK.
- TransferDR: TransferDR is required when the WBR includes cells with a transfer capability.

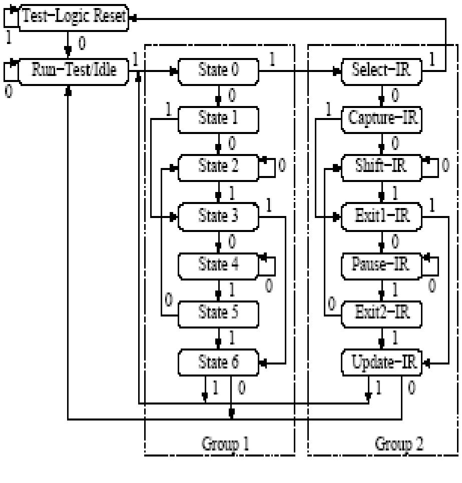

Fig. 3: TAM Controller State Diagram

TAM controller can be used, as long as its outputs fully conform to the requirement of WIP. TAM controller is a Finite State Machine (FSM) in nature. Since WIP signals mimic the output of TAP controller of IEEE 1149.1[9], here the same state-diagram as TAP (Fig. 3) is used and the output logic of FSM is modified in order to make it suitable for WIP. The three inputs (tclk, tms, trst) of TAM controller have just the same meaning as in TAP, that is, TAM controller changes its state on the rising edge of tclk according to tms. In our design, tclk is also used as WRCK.

5 WRAPPER DESIGN

P1500 wrapper consists of wrapper instruction register (WIR), wrapper bypass register (WBY), wrapper boundary register (WBR), gating control logic, mandatory serial path, and optional parallel paths. The serial path is used both for wrapper control by loading instructions into WIR, as well as for low-bandwidth test data access to WBR. Parallel paths are used for internal scan-chains and high-bandwidth test data access to WBR. A minimal library of WBR cells is presented and the design of WBR is not depicted in this paper. WBY used in our wrapper is just an ordinary flip-flop. Bypass registers both for the serial path and parallel paths are inserted. WIR is composed of shift register, update register and decoding logic (Fig. 1). At the start of every test cycle, a new instruction is shifted into WIR through the serial path. Consequently, the operation of the wrapper is controlled by both WIP signals, as well as the presently active instruction in WIR. WIR outputs two static control signals for the wrapper. Sel identifies whether WBR or WBY is connected between WSI and WSO and M2_Mode determines whether the test data in WBR apply to system logic or not. WRSTN resets the content in the update register and puts the wrapper in normal operation mode.

6 CORE TEST GENERATION AND TEST ENVIRONMENT

The cores we used for our purpose were constructed from ISCAS 85 combinational and ISCAS 89 sequential benchmark circuits[8],[9]. The problems related to testing sequential circuits are much more complicated than those for the combinational circuits. The basic reason is the presence of storage elements in the realization of sequential circuits. The testing time is also generally higher in the case of sequential circuits. The test generation was focused at the gate level based on single stuck-fault model[11]. The first step in our SoC test generation procedure is to obtain specified sets of test vectors using high-level descriptions of circuit cores. All the SoCs used in the paper were designed to operate in Altera Max Plus II development environment. Not only was the SoC cores, the TAM also designed to work described in VHDL. The process of fault injection for the different SoC cores was carried out by the designed fault simulator. The fault simulator injects faults at the input lines, then at the output lines, which were followed by fault injection at the internal wires. The nature of faults is single stuck faults. Thus each line can have only two types of stuck faults: stuck-at-1 and stuck-at-0. Simulation based fault injection scheme is used to evaluate the dependability of the system based on the percentage of fault coverage. The fault simulator makes a fault list of all stuck-faults on the inputs and outputs of gates and functional blocks. The simulator then simulates the circuit core with the selected faults by using the given test vectors. It keeps a record of the detection data (time and output of detection) for each fault and also stores the true-value response. Each vector set is run through the simulator by using the list of undetected faults up to that point. At the end of a run, the simulator can store the internal states of the circuit core in a checkpoint data set for use by the next vector set. The data set is provided to the test program and the fault coverage is finally calculated.

7 IMPLEMENTATION RESULTS

To demonstrate the feasibility of the developed test environment and test methodologies for embedded cores- based SoCs, independent simulations were conducted on the cores of the various SoCs that is constructed based on ISCAS 85 combinational and ISCAS 89 sequential benchmark circuits. We first provide here some simulation data on an SoC that was constructed from the sequential circuits in ISCAS 89 benchmark circuits list. Next, simulation experience on an SoC comprised of the combinational circuits in ISCAS 85 benchmark circuits list is given. Finally, we consider SoCs comprised of both sequential and combinational circuits from ISCAS 89 and ISCAS 85 benchmark circuits lists, respectively. Table 1 shows the number of injected faults, detected faults, and fault coverage for the first SoC (SoC_seq) using pseudorandom test vectors. Table 2 shows the result for core area with wrapper for our experimental SoC. Table 3 shows clock gating and glitch reduction techniques to core data path for reducing its power consumption.

Table 1: Test Data for the cores

Circ uit name	No. of Test vectors for Simulation	No. of fault inject ed	No of faults detected	Fault Coverag e (%)	Execution time of the Processor(in µs)
C17	20	6	5	83.33	0.00492
S382	512	32	29	90.77	0.00607
S386	128	24	22	91.6	0.00649

Table 2: Core Area with Wrapper

Core	Slices	Core Area With Wrapper in this work	Conventional Core Area With Wrapper
C17	2	109	117
S382	4	109	116
S386	7	87	94

Table 3: Power Consumption for test circuit

Circuits	Without Clock Gating	With Clock Gating
C17	48.45 mW	44.76 mW
S382	55.06 mW	53.90 mW
S386	58.06 mW	51.06 mW

8 CONCLUSION

Embedded cores-based design paradigm has evolved from the necessity to increase design productivity and decrease time-to-market, but as a result has created numerous challenging problems for the test design community. This paper provides approaches to developing test environment and test methodologies for digital SoCs. A 1500-compliant wrapper and TAM controller co-design scheme is presented in this paper. In this scheme, only one TAM controller is required at SoC level hence costing low area overhead. The test access mechanism which plays an important role in transporting the test patterns to the desired cores and the resulting responses to the output pins of the SoC was implemented as a plain signal transport medium, being shared by all the cores in a given SoC. The faults were injected using fault simulator that generates tests for the core under test. Testing power consumption of the wrapper is also taken into account by using gated clocks to operate wrapper registers. The effectiveness of the proposed scheme has been confirmed by experiment results based on a sample SoC.

REFERENCES

[1] Yervant orian, Erik Jan Marinissen, and Sujit Dey, "Testing Embedded-Core-Based System Chip", *IEEE Transaxtions on Computer*, 32(6):52–60, 1999.

[2] IEEE P1500 WebSite.http://grouper. IEEE.org/groups/1500/index.html

[3] Erik Jan Marinissen, Rohit Kapur, and Yervant orian, On Using IEEE P1500 SECT for Test Plug-n-Play, *In Proceedings IEEE International Test Conference(ITC)*, pp.770 –777, Atlantic City, NJ, Oct 2000.

[4] Vikram Iyengar, Krishnendu Chakrabarty, and Erik Jan Marinissen, Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip, *In Proceedings IEEE International Test Conference(ITC)*, pp. 1023–1032, Baltimore, MD, 2001.

[5] Vikram Iyengar, Krishnendu Chakrabarty, and Erik Jan Marinissen, Test Access Mechanism Optimization, Test Scheduling, and Tester Data Volume Reduction for System-on-Chip, *IEEE Transactions on Computers*, 52(12): 1619-1632, 2003.

[6] R. Rajsuman, System-on-a-Chip: Design and Test. Boston, MA: Artech House, 2000.

[7] K. Chakrabarty, V. Iyengar, and A. Chandra, Test Resource Partitioning for System-on-a-Chip. Boston, MA: Kluwer, 2002.

[8] K. Chakrabarty (Editor), SoC (System-on-a-Chip) Testing for Plug and Play Test Automation. Boston, MA: Kluwer, 2002.

[9] S. Mourad and Y. Zorian, Principles of Testing Electronic Systems. New York: Wiley, 2000. Circuit

[10] T. W. Williams and K. P. Parker, "Testing logic networks and design for testability", Computer, vol. 21, pp. 9-21, Oct. 1979.

[11] S. R. Das, C. V. Ramamoorthy, M. H. Assaf, E. M. Petriu, and W.B. Jone, "Fault tolerance in systems design in VLSI using data compression under constraints of failure probabilities", *IEEE Trans. Instrum. Meas.*, vol. 50, pp. 1725-1747, Dec. 2001.