Enhancement in IEEE 1500 Standard for at-speed Test and Debug

Ghazanfar Ali[1], Fawnizu Azmadi Hussin[1], Noohul Basheer Zain Ali[1], Nor Hisham Hamid[1], Raja Mahmud Adnan[2]
[1]Centre for Intelligent Signal and Imaging Research (CISIR),
Electrical and Electronic Engineering Department,
Universiti Teknologi Petronas (UTP), Perak, Malaysia.
[2]Intel, Penang, Malaysia.
ghazanfarali@ieee.org, {fawnizu, noohul.z, hishmid}@petronas.com.my, mahmud.adnan@intel.com

***Abstract*—IEEE 1500 standard provides the facility to test and debug embedded cores with the use of an on-board or off-board tester. So far all the developments in IEEE 1500 standard are for testing application in the test mode. No development in IEEE 1500 standard is proposed where IEEE 1500-compliant cores can be tested in functional mode of operation. In this paper, an enhancement of the IEEE 1500 standard for functional test and debug is proposed. As a case study, the proposed enhanced IEEE 1500 standard is implemented and validated on a SAYEH processor using embedded Software Based Self-Testing (SBST) technique. The case study demonstrated that the enhancement in IEEE 1500 standard enables it to be used for at-speed test and debug with increased observability.**

Keywords— IEEE 1500 Standard; Functional Test; Functional Debug; Software Based Self-Testing (SBST)

I. INTRODUCTION

System on Chip (SOC) provides the possibility to design a large and complex system of digital and analog cores on a same silicon dye; however the complexity of the design makes testing of the SOC a very difficult task. To improve this test access issue, IEEE 1500 has been introduced in 2005 to standardize design for testability (DFT) strategy of SOC [1]. This standard allows the testing of embedded cores by isolating them from their functional environment and test vectors can be applied with the help of internal or external Tester i.e. Automatic test equipment (ATE). The core response is then compared with the predetermined response to determine if there is any fault in the core. In addition to core testing, this standard also enables the testing of logic circuit between two wrapped cores.

Different techniques have been developed to test IEEE 1500-compliant SOC. In [2], an embedded Micro-Tester architecture has been proposed by Tuna *et-al*. that can test at-speed, IEEE compliant cores by configuring them in test mode. In [3], Turbo1500 has been proposed by Laung-Terng *et-al.* that served as an automation tool to connect all the compliant cores automatically. In [4], Kuen *et-al.* proposed an on-chip at-speed test platform for IEEE 1500-compliant cores to eliminate the use of ATE. The proposed platform has the capability to handle different types of embedded cores that include scan-based logic cores, built in self-test (BIST) based memory cores, BIST-based mixed-signal devices, and hierarchical cores. In the proposed platform a centralized test access mechanism generate all the control signals for different cores. In [5], a novel test controller, System on Chip Embedded Core Test (SoCECT), has been proposed by Higgins *et-al.* that can concurrently test multiple cores by configuring them in test mode. In the proposed design IEEE 1149.1 JTAG state machine was used to operate the test controller.

In [6], a new embedded delay framework has been proposed by Po-lin *et-al.* In the framework, a new architecture for Wrapper Boundary Register (WBR) of IEEE 1500 standard for at-speed delay testing was proposed. In this technique although on-chip phase-locked-loops (PLLs) are used to generate functional clocks but to improve the accuracy of delay tests, the test clock is applied from the ATE. In [7], new WBR cell architecture has been proposed that can protect the internal scan chains and the primary inputs and outputs of the IEEE 1500 compliant cores. In the proposed design the WBR's flip flips also serve as linear feedback shift registers (LFSR) to generate the golden key.

Based on the literature review it is found that all the developments to test IEEE 1500 compliant cores are in the test mode. So far no development in IEEE 1500 standard is proposed where IEEE1500-compliant cores can be tested in functional mode of operation like by using software based self-testing (SBST) technique. Although there are techniques that have been developed to perform at-speed test and debug like in [2], [4] and [6], but the dependency on external tester is still present.

In this research, an enhancement to the IEEE 1500 standard for functional testing is proposed that will allow the testing of CUT in its functional mode of operation, facilitating the testing for faults that crop up only in real environment. The proposed modified IEEE 1500 standard is used with the SBST which is one of the very special on-chip functional testing techniques [8]. In SBST, the test vectors are generated by instruction sequences and looking at the responses, the processor itself detects the possible faults [9], [10]. Using IEEE 1500 standard together with SBST will allow the core to be tested by using its own functional clock and it can also reduce the test complexity and test cost by reducing the dependency on the traditional ATE.

The paper is organized as follows; in section 2, the methodology is explained that includes explanation of development of each block of IEEE 1500 standard. In section 3, a case study is explained where arithmetic logic unit (ALU) of SAYEH (Simple Architecture Yet Enough Hardware) processor is wrapped with proposed modified IEEE 1500 standard and is tested by executing a test program. The results are validated by using Vertex-6 FPGA.

978-1-4799-5924-2/14/$31.00 ©2014 IEEE

TABLE I. TABLE OF INSTRUCTIONS USED IN FUNCTIONAL MODE TESTING

Instruction Name	OPCODE	Control Signals							
		se	hold enable	wir snooping	wir shift	wir test	wir test out	wir debug	wir debug out
WS_ F_TEST	1101	0	0	1	0	1	0	0	0
WS_F_ TEST_OUT	1110	x	x	1	x	0	1	0	0
WS_F_ DEBUG	1111	0	0	1	0	0	0	1	0
WS_F_ DEBUG_ OUT	0101	x	x	x	x	0	0	0	1

II. METHODOLOGY

Current IEEE 1500 wrapper design has three major components namely Wrapper Instruction Register (WIR), Wrapper Bypass Register (WBY) and Wrapper Boundary Register (WBR) [1]. Complete architecture level diagram of the proposed modification of the IEEE 1500 standard is shown in Fig. 1. The highlighted modules are new proposed addition to the standard. A new mode of testing has been introduced to test the circuit under test (CUT) in functional mode, called as "functional testing mode". In order to make test and debug possible, "functional testing mode" is further divided into 'functional test mode' and 'functional debug mode'.

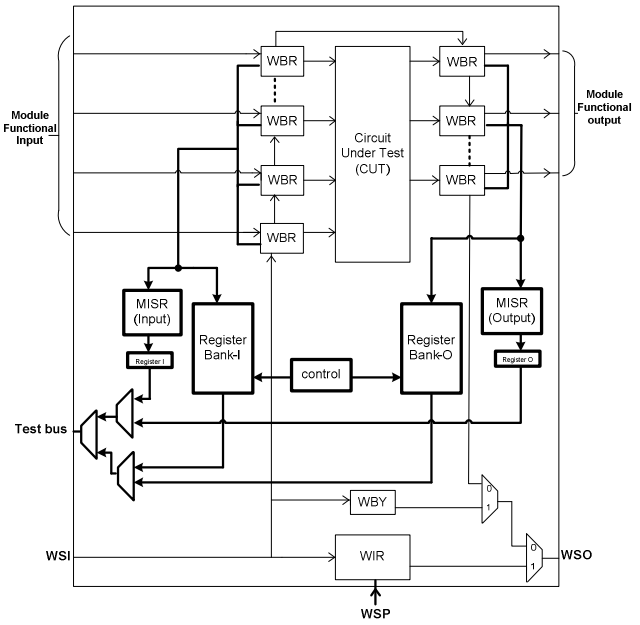

Fig. 1. Enhanced IEEE 1500 standard architecture for at-speed test and debug

In functional test mode, all the data snooped by the WBR goes to "Multiple Input Signature Register (MISR)" modules that can generate signature for each input. This generated signature can be compared on-chip or off-chip with predetermined signature. In functional debug mode, all the data snooped by the WBR is stored in the register banks and sent out for the comparison purpose to determine the faulty location. To handle the on-chip and off-chip frequency difference, the register banks are loaded at-speed with the functional data, and then unloaded at lower speed through the IEEE 1500 interface.

A. Wrapper Instruction Register (WIR)

WIR generates all the control signals to control IEEE 1500 wrapper's components i.e. WBY, WBR, MISR blocks, Register Banks and the multiplexer (MUXES). First a 4-bit OPCODE is serially shifted into WIR through the WSI port which is decoded and control signals are generated when "wir_update" signal is applied. These control signals set the IEEE 1500 wrapper in intest, extest, bypass [1] and functional testing mode. In order to set the wrapper in "functional testing mode", four new instructions have been introduced as shown in Table I. To configure the IEEE 1500 standard in a required mode of operation, specific OPCODE from the table I is shifted into the WIR by using the same protocol defined for the IEEE 1500 standard in [1].

B. Wrapper Boundary Register (WBR)

WBR is a key component in IEEE 1500 standard as it enables the wrapper to execute functional mode, test mode or functional testing mode. It remains transparent in "functional mode" and isolates the CUT from the functional environment in "test mode". In "functional testing mode", the incoming and outgoing data is snooped on every clock cycle and sent to the 'MISR' block or "Register bank" via CFTO port, to test or debug the CUT in its functional mode of operation. The modified WBR architecture is shown below in Fig. 2.

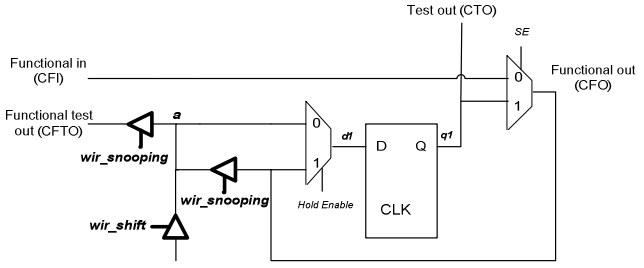

Fig. 2. Modified WBR cell

C. Multiple Input Signature Register (MISR)

MISR blocks have been introduced in the architecture to observe the values of the inputs and outputs of the CUT for a given test program. When 'WS_ F_TEST' is decoded in WIR, a control signal is enabled to initialize the MISR from a known

state using a seed value. The output of MISR blocks can be controlled by loading instruction 'WS_F_ TEST_OUT' into the WIR. In order to send the signature data over 'Test bus' one by one (due to the limitation of test bus band width), a 3-bit state 'Ctrl_misr' is generated when 'wir_test_out' signal is set to logic value 'high'. The state transition diagram for 'Ctrl_misr' is shown below in Fig. 3. This state transition diagram has been generated for 3 MISRs e.g. A, B and C. For 'Ctrl_misr=001', 'Ctrl_misr=010' and 'Ctrl_misr=011', signature value of A, B and C are sent over 'Test bus' respectively.

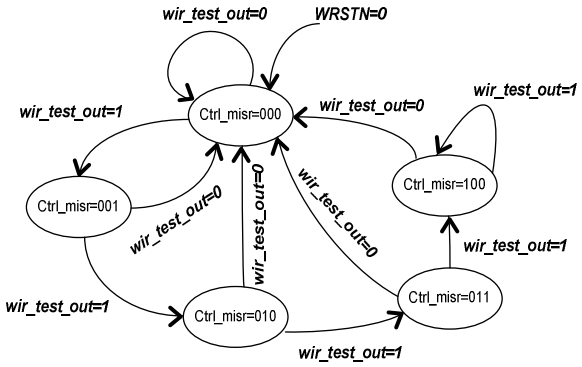

Fig. 3. State Transition Diagram for Ctrl_misr

D. Register Bank with Controller

Introducing register banks in IEEE 1500 standard provides the facility to efficiently debug the circuit at-speed. The purpose of 'Register Bank' is to temporarily store the data on each clock cycle coming either on the input port or the output port in debug mode. In debug mode since the objective is to locate the fault location so capturing information from each bit is important. There are two register banks shown in the general architecture (Fig. 1) of enhanced IEEE 1500 wrapper, one for the input and other for the output. Each register bank is controlled by a memory controller.

There are two major parts of writing and reading these register banks i.e. address generation for incoming data and address generation for the sending the data on test bus. When 'WS_F_DEBUG' instruction is loaded through WSP, it sets the value of 'wir_debug' from '0' to '1'. The controller generates the address for the vacant position from FF[0] to FF[31]. On each positive clock edge it stores the incoming data and increments the memory address by 1. When 'WS_F_DEBUG_OUT' instruction is loaded through WSP, it sets the value for 'wir_debug_out' from '0' to '1'. To send the data of each register bank one by one on the test bus, control signal 'Ctrl_ShiftReg' is generated, e.g. for 3 register banks, let's say X, Y and Z. Figure 4 shows that the value of 'Ctrl_ShiftReg' is controlled with control signal 'wir_debug_out'. When 'Ctrl_ShiftReg=00', test bus is disconnected from register modules, for 'Ctrl_ShiftReg=01', 'Ctrl_ShiftReg=10' and 'Ctrl_ShiftReg=11', the data of bank X, Y and Z is sent over 'Test bus' one by one. Complete architecture of the 3 register banks with controller is shown below in Fig. 5.

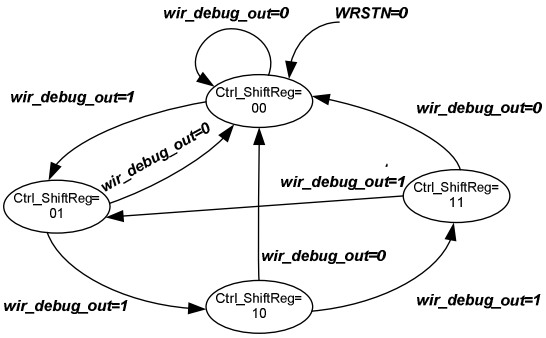

Fig. 4. State Transition Diagram for 'Ctrl_ShiftReg'

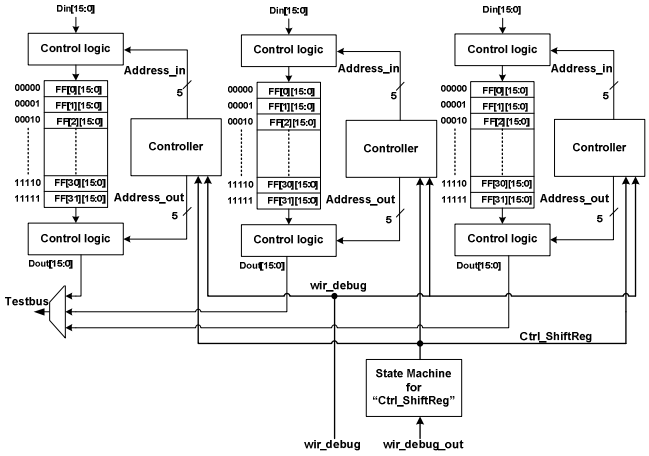

Fig. 5. Complete architecture of 3 register banks with Controller

III. CASE STUDY

For the validation of our proposed modification in the IEEE 1500 standard, the proposed DFT has been applied on SAYEH, a 16-bit free opencore processor [11] for educational purpose. The complete system level diagram has been shown in Fig. 6. A test program is developed which also serves as a debug program. This test program is loaded in the "Program Memory" to be executed by the SAYEH processor. 'Arithmetic Unit (AU)' of the processor is wrapped with the proposed IEEE 1500 wrapper and connected with another embedded module 'iTester'. The purpose of 'iTester' module is to send the test/debug data over "data bus" that it receives over "Test bus" from the IEEE 1500 wrapper. 'eTester' on the other hand generates all the WSP controls to configure the wrapper in functional testing mode.

The complete system is validated by using "Vertex-6 ML605 evaluation board". Results show that the proposed technique can be used for at-speed functional testing. The objective of this case study is to test the AU only, so all the instructions used in the test program involve only the AU operations. 'A', 'B' and 'aluout' are the signals that appear on the inputs and output of the AU. These signals are snooped by the modified IEEE 1500 standard and a signature is generated for each IOs of the AU during the functional test mode. In the functional debug mode, three separate register blocks are used

to store the functional data that appear on the three IOs of the AU as discussed earlier in section 2 of this paper.

Table II summarizes the synthesis results for the Vertex-6 FPGA for two cases. The percentage overhead increase of the proposed technique as compared with the IEEE 1500 standard is shown in column 5 of the table II. The result shows that the current implementation of the proposed technique suffers with area overhead as compared to the IEEE 1500 standard. This is mainly because of the small processor SAYEH that is used for the case study. However, for large designs like RISC, ARM or even real processors, the area overhead may be tolerable because of their own design size. Apart from the overhead, there are several advantages with the use of this modified IEEE 1500 standard as it increases the observability during the functional testing, especially in functional debug mode, helping at-speed test and debug. This increase in observability enables the modified IEEE 1500 to capture and transport out the internal states of processors. Because of this ability, simple test and debug programs can be used to functionally validate the CUT that requires complex test/debug programs for propagation of faults using other SBST techniques. In addition, if the CUT is unable to load the test program, the modified IEEE 1500 standard also provides the facility to test and debug it with on-board tester 'eTester' which can reduce the dependency on the traditional ATE.

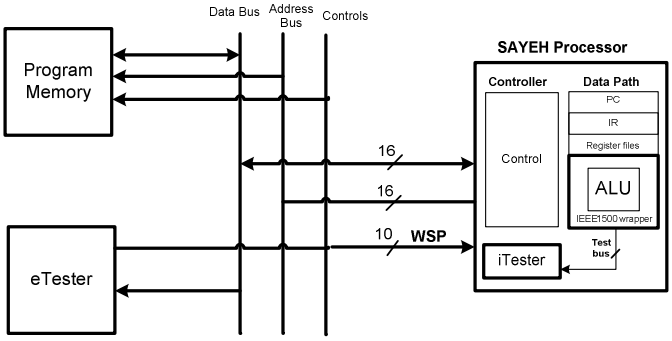

Fig. 6. Complete system level architecture

TABLE II. SYNTHESIS RESULTS FOR VERTEX-6 FPGA

ML605 platform	SAYEH	SAYEH with IEEE 1500 standard	SAYEH with modified IEEE 1500 standard	Overhead Percentage
Occupied LUT	102	194	316	62.8%
Slice LUT	286	561	901	60.6%
Slice Registers	49	112	321	184.6%
Timing Specs	10.587ns	10.716ns	11.446ns	6.8%

IV. CONCLUSION

In this paper, an enhanced IEEE 1500 standard for at-speed test and debug has been proposed and evaluated experimentally using ML605 evaluation board. Results show that this modified IEEE 1500 standard can be used for test and debug application in the functional mode as well. In the current implementation, MISR is used only during the "functional test mode" and "register banks" are used in "functional debug mode". To handle large debug programs, due to the limitation of 'register bank' size, the future modification can be using 'MISR' with 'register bank' during "functional debug mode" to identify the portion of debug program that is failing. For the future work, the authors are implementing the proposed technique on 'minisoc', for further performance evaluation.

ACKNOWLEDGMENT

The project was supported in part by the Ministry of Science, Technology and Innovation (MOSTI) under the Science Fund (03-02-02-SF0141) and by Universiti Teknologi PETRONAS under the Graduate Assistantship scheme. The authors would like to acknowledge the Centre for Intelligent Signal & Imaging Research (CISIR) for providing the facilities to conduct this research.

REFERENCES

[1] "IEEE Standard Testability Method for Embedded Core-based Integrated Circuits,"IEEE Std 1500-2005, vol., no, pp.1-117, June 6 2012

[2] Tuna, M.; Benabdenbi, M.; Greiner, A.;,"At-Speed Testing of Core-Based System-on-Chip Using an Embedded Micro-Tester," *VLSI Test Symposium, 2007. 25th IEEE* , vol., no, pp.447-454, 6-10 May 2007

[3] Laung-Terng Wang; Apte, R.; Shianling Wu; Boryau Sheu; Kuen-Jong Lee; Xiaoqing Wen; Jone, W.-B.; Chia-Hsien Yeh; Wei-Shin Wang; Hao-Jan Chao; Jianghao Guo; Jinsong Liu; Yanlong Niu; Yi-Chih Sung; Chi-Chun Wang; Fangfang Li;,"Turbo1500: Toward Core-Based Design for Test and Diagnosis Using the IEEE 1500 Standard," Test Conference, 2008. ITC 2008. IEEE International, vol., no., pp.1-9, 28-30 Oct. 2008

[4] Kuen-Jong Lee; Tong-Yu Hsieh; Ching-Yao Chang; Yu-Ting Hong; Wen-Cheng Huang;,"On-Chip SOC Test Platform Design Based on IEEE 1500 Standard," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.18, no.7, pp.1134-1139, July 2010

[5] Higgins, M.; MacNamee, C.; Mullane, B.;,"SoCECT: System on Chip Embedded Core Test," *Design and Diagnostics of Electronic Circuits and Systems, 2008. DDECS 2008. 11th IEEE Workshop on* , vol., no., pp.1-6, 16-18 April 2008

[6] Po-lin Chen; Jhih-Wei Lin; Tsin-Yuan Chang;,"IEEE Standard 1500 Compatible Delay Test Framework," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.17, no.8, pp.1152-1156, Aug. 2009

[7] Geng-Ming Chiu; Li, J. C -M, "A Secure Test Wrapper Design Against Internal and Boundary Scan Attacks for Embedded Cores," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.20, no.1, pp.126,134, Jan. 2012 doi: 10.1109/TVLSI.2010.2089071.

[8] Sabena, D.; Reorda, M.S.; Sterpone, L.;,"On the development of Software-Based Self-Test methods for VLIW processors,"Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2012 IEEE International Symposium on , vol., no., pp.25-30, 3-5 Oct. 2012

[9] Li Chen; Dey, S.;,"Software-based self-testing methodology for processor cores," Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.20, no.3, pp.369-380, Mar 2001

[10] Ateeq-Ur-Rehman Shaheen, Fawnizu Azmadi Hussin, Nor Hisham Hamid, Noohul Basher Zain Ali, "*A Review on Structural Software-Based Self- Testing of Embedded Processors*", International Review on Computers and Software (IRECOS), May 2014 (Vol. 9 N. 5), pp. 832-846

[11] Opencore: http://opencores.org/project,sayeh_processor.