Remote Testing and Diagnosis of System-on-Chips Using Network Management Frameworks[1]

Oussama Laouamri & Chouki Aktouf
DeFacTo Technologies, 167 rue de Mayoussard, 38 430 Moirans, FRANCE

Abstract

This paper presents a new approach that allows remote testing and diagnosis of complex (Systems-on-Chip) and embedded IP cores. The approach extends both on-chip design-for-test (DFT) architectures and network management protocols to take full benefits from existing networking infrastructures. By running intensive experimentation on ITC'99 and ITC'02 design benchmarks, the efficiency of the proposed testing and diagnosis methodology is analyzed..

1. Introduction

The SoC design paradigm has been widely accepted and implemented in practice during the last few years as an enabling design methodology for the improvement of productivity and the increase of design functionality. The main issues to be faced when testing an SoC are essentially related to the new challenges offered by the growing design complexity. Semiconductor technology feature sizes are getting deep in the submicron area, thus allowing the integration of many complex cores within a single design.

In the DSM (Deep Sub-Micron) arena, testing and diagnosis become a major concern for quality and cost. Testing costs are already significant part of the chips total production cost and represent a bottleneck in the successful completion of a design project and its final market success. Unlike other silicon manufacturing costs, the test cost has not benefited as much from the overall downward trend over time. The cost of large automated test equipments (ATE) have steadily risen to multi-million dollars for cutting-edge capabilities. The length and number of test vectors per design are increasing, resulting in each design consuming more time on the testers. The International Technology Roadmap for Semiconductors (ITRS) reported in its 2001 analysis that for some products in certain market segments, test may account for more than 70% of total manufacturing cost, or process technology. The solution to this predicament is reducing the reliance on big testers and instead utilizes more low-cost testers that are DFT-aware.

[1] This research work has been conducted under the support of the Institut National Polytechnique de Grenoble (INPG)

Besides traditional manufacturing testing needs, diagnosis and silicon debug can be required at various stages in the lifecycle of an electronic product: during development, qualification, production ramp-up, or in maintenance-like activities. The root cause of the problem-to-be-debugged can lie in the technology used, as well as in the design implementation (or the combination of these); the debugging process has to cover both aspects.

Although it is clear that software tools and DFT are gaining in importance, with the advent of SoC, diagnosis and physical debug methods have still an important role to play.

Given an SoC which is already integrated within an electronic system, any mechanism that allows (i) remote application of test vectors for any of the SoC blocks and (ii) gathers related test results, is very beneficial. Beyond simplifying the access to CUT (Chip Under Test) blocks from external I/O pins, a remote access should be possible during an SoC live time. Only network infrastructure can allow such easy access. Making an SoC TCP/IP compliant extends testing and diagnosis possibilities. Such a compliance can take benefits from TCP/IP network management protocols such as SNMP (Simple Network Management Protocol).

Today, available network infrastructures allow secure data transfer. Such infrastructure can serve as a powerful vehicle to drive test vectors from a test engine to chips and gather test results for a deep analysis. Furthermore, the large experience in network management of electronic and computer systems can assist the testing community in monitoring the behavior of chips during execution of real-life applications. Indeed, network management for local TCP/IP networks is mainstream. To maintain and deliver high service quality to end users, network performance and reliability must be constantly monitored.

To date, several research works have addressed hardware-based solutions using network protocols and applications. In the Applied Research Lab (ARL) at Washington University, a set of hardware components for research in the field of networking, switching, routing and active networking have been developed [1]. However, hardware components of layered protocol wrappers (UDP/IP wrappers) [1] have been proposed which process Internet packets in reconfigurable hardware. Hence, several network applications which use this wrapper library [1] have been developed. For instance, an Internet router or a firewall are important applications that use the

978-3-9810801-2-4/DATE07 © 2007 EDAA

wrapper library to route and filter packets [2, 3]. A single chip has been used to filter internet SPAM and to guard against several types of network intrusion. In such research, work has not addressed a hardware-based SNMP solution at the application layer. This is important since such a feature has to be considered at the chip level. SNMP is considered as an application layer protocol which uses a TCP/IP suite (in practice UDP is used). In this work, an SNMP agent is developed on a wrapper library described in [1]. This agent is developed within an SoC to help the external testing of the overall SoC.

In this work, a new DFT methodology named SNMP/1500 which makes complex SoCs easily testable and diagnosable is presented. By using existing network infrastructure, the proposed methodology can interoperate with existing DFT methodologies and network technologies [4]. Using the IEEE 1500 DFT methodology [5, 6], test logic is extended and made compliant with SNMP TCP/IP management protocol.

SNMP is known as a simple and a very powerful management protocol. It embeds a set of features that allow the management of heterogeneous and complex networks. In this research work, SNMP is considered within SoCs to allow either remote testing or real-time monitoring of embedded cores.

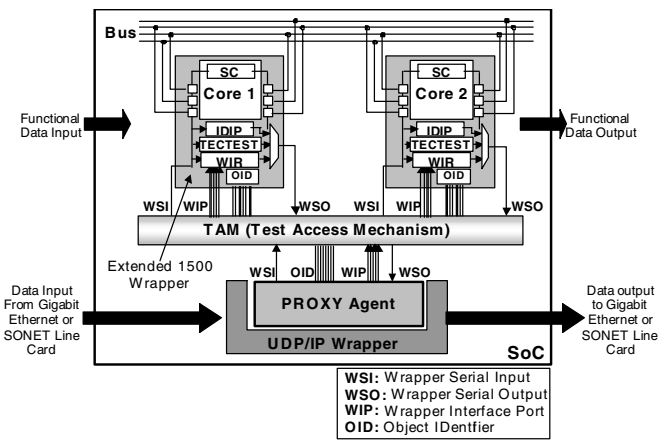
IEEE 1500 standard [5, 6] enhances testability of SoCs. It helps isolating blocks or IP cores which allows targeting their individual test or monitoring. The proposed architecture embeds two kinds of interfaces. The first interface extends a traditional IEEE 1500 wrapper. The second interface embeds a SNMP proxy agent at the SoC level. Starting from SNMP requests sent by a test engine such as an ATE via TCP/IP network, the on-chip SNMP agent is made capable to perform IEEE 1500 boundary scan operations at the level of an embedded core. This allows support for testing, diagnosis and monitoring of an SoC or a block of an SoC with a full compliance to both the IEEE 1500 and SNMP standards. It is noteworthy that the proposed architecture reuses the existing Test Access Mechanism (TAM).

The rest of the paper is organized as follows: Section 2 presents the SNMP/1500 compliant testing architecture. Section 3 presents software considerations that help implementing the proposed approach. In section 4, the main approach which combines both management and testing standards is presented. Section 5 summarizes the implementation results and performance evaluation on monitoring and testing operations. Finally, conclusions are given in section 6.

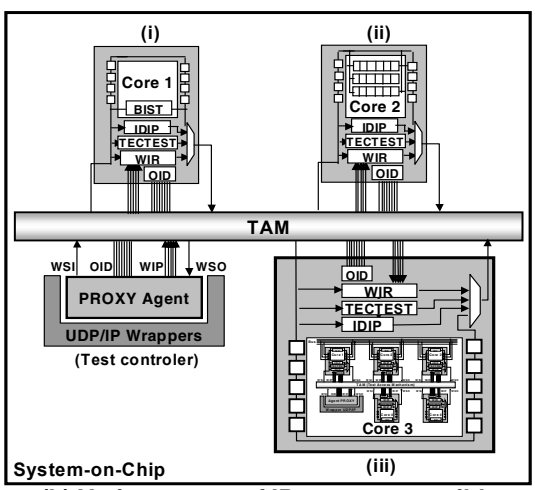
2. SNMP/1500 Test architecture

In this architecture, IEEE 1500 wrappers are extended and represent SNMP agents. An SNMP agent which is illustrated in Figure 1.a is managed by a proxy agent. Test data is carried between IP cores and the proxy agent via a TAM. The proposed SNMP/1500 architecture is compliant with existing TAM structures.

As previously detailed in [7, 8], such an architecture embeds new registers such as IDIP (IDentifier of IP core), TECTTEST (TECHnique of TEST) and OID (Object Identifier). The Wrapper Instruction Register (WIR) controls wrapper operations. Given control inputs, WIR operations are directly controlled (WIP) by the proxy agent. Furthermore, WIR is extended by new instructions such as WS_GETREQUEST and WS_SETREQUEST. Hence, a SNMP operation is indicated by two parameters: PDU (kind of operation: get-request, set-request...) and OID [9]. The same behavior is extrapolated at the level of the IP cores test interfaces (extended 1500 wrapper) which ensure a full compliance with the IEEE 1500 standard. During a SNMP operation, the semantics of a IEEE 1500 instruction is completed by a flattened OID, which is the equivalent to a hierarchical one.



(a) SNMP/1500 Architecture



(b) Various types of IP cores compatible with SNMP/1500 architecture

Fig. 1– Proposed SNMP/1500 architecture

Through the use of network layered protocol UDP/IP wrappers [1], a test operator has the ability to manage the SoC infrastructure by using an SNMP proxy agent module (Fig. 1.a). The proxy agent monitors and controls the embedded cores under test. The proxy agent is used to translate information between SNMP and IEEE std. 1500 protocols. It provides a protocol conversion function which allows a management station to apply a consistent management framework to all SoC and IP cores infrastructures. The proxy agent can be considered as an IP core, which gets SNMP requests coming from the

management station. Such requests are converted into instructions in compliance with the extended 1500 standard. In a similar way, answers from the IP core are converted into an SNMP protocol representation (response). Finally, test results are sent to the ATE as SNMP messages.

Figure 1.b shown various types of IP cores which are compliant with the proposed SNMP/1500 architecture. Figure 1.b.i shows an IP core including an internal Built-In-Self-Test (BIST) structure. The SNMP/1500 architecture operates also with the IP cores using the scan test infrastructure (Fig. 1.b.ii). To ensure hierarchical testing, any IP core can be considered as an SoC with a full compliance to the proposed SNMP/1500 architecture (Fig. 1.b.iii).

3. Software design considerations

A MIB (Management Information Base) [9] data model represents a software interface between SNMP Framework and the design under test. Hence, a correspondence must exist between the MIB knowledge available to the manager (ATE) and what is really implemented within the agent (SoC). The manager can only carry out the operations which are envisaged in the MIB.

Name	OID	Description
socIdentifier	X.1.1.1.0	SoC Identifier
ipCoreIdentifier	X.2.2.1.2.ipCoreIndex	IPcore Identifier
techniqueTest	X.2.2.1.3.ipCoreIndex	test technique
functionalTestVT	X.2.2.1.4.ipCoreIndex	Functional test
exTestVT	X.2.2.1.5.ipCoreIndex	External test
simpleCoreTestVT	X.2.2.1.6.ipCoreIndex	Simple Internal test
coreBISTEnable	X.2.2.1.8.ipCoreIndex	Built-In Self-test

Tab. 1– Definition of main managed objects

A MIB basic element [7, 8] is called "**mibSoCTest**". Any new module is identified by the OID "**1.3.6.1.4.1.X**", for which **X** is a reference given by Internet Engineering Task Force (IETF). Given an IP core or an SoC under test, the MIB describes both features of the implemented test techniques which are associated to the IEEE 1500 wrapper and information related to the testing process. The MIB is divided into two parts: the information at the SoC level and those at the level of IP cores. The first part of the MIB is dedicated to the SoC: SoC identifier, configuration of basic components, etc. The second part of the MIB is dedicated to the IP cores. For instance, the table called "**ipCoresWrappedP1500Table**" is related to the information regarding IEEE 1500 test architecture of each IP core. The index of this table is called "**ipCoreIndex**". It represents the logical address of IP cores in the SoC environment. The table 1 shows examples of managed objects.

4. Remote test protocol

4.1. Overview

The proposed SNMP/1500 interface (UDP/IP wrapper and Proxy Agent) controls the internal and wrapper boundary scan (WBR of IEEE std. 1500) via the TCP/IP network. This interface assumes the connection of input and output terminals of the scan chains (including the boundary-scan chain), test control pins, and clock pins to ATE channels (managed by existing TCP/IP bandwidths).

At the level of every IP core, the access to the rest of the functional pins is achieved via the IEEE 1500 wrapper boundary-scan chain. In this work, both the UDP/IP wrapper and the Proxy Agent constitute an SNMP/1500 wrapper around an SoC (fig. 1). In this case, the DFT for the SoC can be designed without even knowing about the target ATE. Later, by using the re-configurable logic, the number of scan chains and their length can be modified in the embedded cores according to the ATE specification. The basic idea behind the combination of DFT techniques and the SNMP standard is to provide remote access not only to the functional terminals, but also to the internal scan chains via the boundary-scan architecture (IEEE 1500 wrapper), in order to enable even greater scalability of the SoC-ATE interface. Given such an extension, the SoC becomes capable of understanding SNMP requests. SNMP requests (get-request, set-request…) retrieve or modify the value of any managed objects (e.g. IP core identifier, SoC identifier, test vector, tests techniques, etc.) at an SoC level. Our proposed SNMP/1500 wrapper around an IC truly converts the fixed number of external test inputs and outputs, which represent a TCP/IP network interface, into IEEE std. 1500 dedicated internal test inputs and outputs.

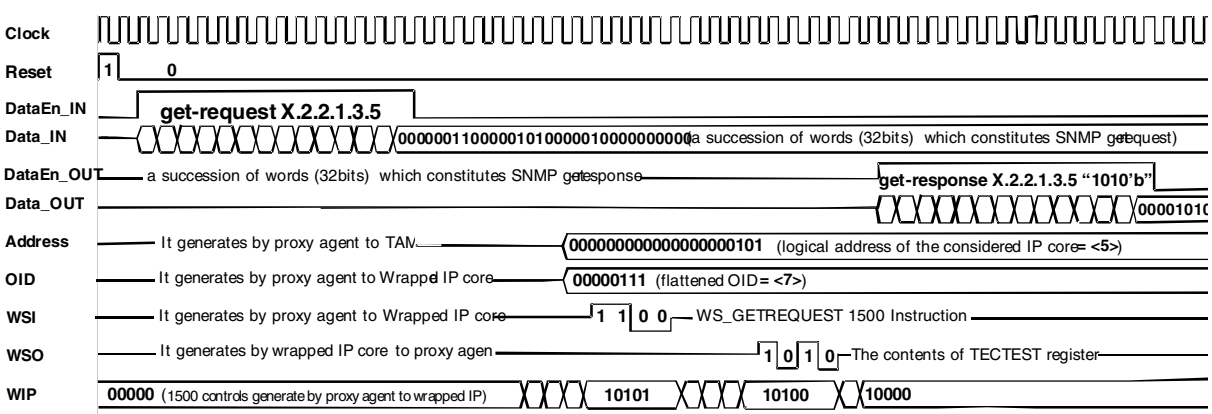


Fig. 2– Functioning of SNMP/1500 architecture

The SNMP set-request message (set-request OID TV) applies test vectors on the IP cores where both object identifiers (OID) and test vectors are specified. The OID distinguishes the type of the applied test. The SNMP get-request message (get-request OID) retrieves information of test or monitoring (e.g. IP core identifier, SoC identifier, tests techniques, monitoring registers, etc.) of either the IP core or the SoC by specifying the identity of an instance of the managed object.

4.2. SNMP-IEEE 1500 relationships

The relationship between the SNMP requests and those of IEEE 1500 is implemented at the level of the proxy agent. The proxy agent converts SNMP requests into IEEE 1500 instructions. For example, the SNMP request "**get-request X.2.2.1.3.5**" is used to recover the contents of the TECTEST register (4 bits which identify the used test technique). This request is converted into **WS_GETREQUEST** IEEE 1500 instruction (Fig. 2) with a flattened OID that equals "**7**". This flattened OID relates to the hierarchical OID "**X.2.2.1.3**". Given some IP cores, the last number (**ipCoreIndex**) of the hierarchical OID represents the logical address (number "**5**") of the considered IP core. To distinguish the type of applied tests at the IP core level the flattened OID is considered instead of the hierarchical OID. This choice is motivated by the need for minimizing the processing logic of the hierarchical OID for each IP core. Figure 2 illustrates this example. In this figure, the data enable signals (**DataEn_IN** and **DataEn_Out**) indicate if 32 bit databusses **Data_IN** and **Data_OUT** are a valid payload of the SNMP message or not.

5. Simulation results

The considered design flow is based on Synopsys® tools. Using a 0.18 µ CMOS technology, implementation of a 200 MHz proxy agent requires 16369 gates. Given a million gate SoC **f2126** from ITC'02 [10] that embeds four cores, **2%** of the area overhead is estimated for the SNMP/1500 architecture adapted to this SoC benchmark. This seems reasonable knowing the number of added features. The proxy agent analyzes at **33 MHz** to **100 MHz** or **200 MHz**, suited to the production lines of electronic component manufacturers, who require high throughput. Therefore, the proxy agent can operate on high-speed networks. The theoretical maximum network throughput that can be supported is **3.2 Gb/s** for a **100 MHz** proxy agent. However, several experimentations at IP core level have been conducted using twenty-two design benchmarks known as ITC'99 benchmarks (b01 to b22) for estimating the cost of area overhead of the extended IEEE std. 1500 wrapper. In summary, the area cost of the extended wrapper depends on the size of the core, as well as the number of core terminals. We reported 1% additional silicon area for the extended wrapper at IP core level, on top of **4.5%** area costs in order to make all cores fully testable with internal full scan.

A queueing model is used in the performance analysis of the proposed approach. The OMNeT++® [11] is considered for analyzing the performance of both monitoring and testing. The implemented queueing network (Fig. 3) is the model M/G/1/N FIFO queue system. It has a Poisson arrival distribution, a general service time, a single server and a finite queue (N: system capacity). The service time is calculated for each SNMP request received starting from its contents. The parameters used in the calculation of the service time are: service type, test type, size of monitoring registers, size of test patterns, number of tests per message, chip speed (clock frequency of test architecture), length of each core-internal scan chain, etc. The setting of these parameters typically influence the number of clock cycles needed to apply one test pattern or to recover one monitoring register.

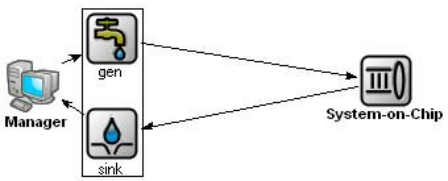


Fig. 3– Queueing network of the proposed approach

As shown in figure 3, three main components are considered in the model that is proposed in this work: system under test or monitoring, the generator of SNMP requests and the system (sink) that processes and analyzes SNMP responses. These two last components constitute the testing station (ATE).

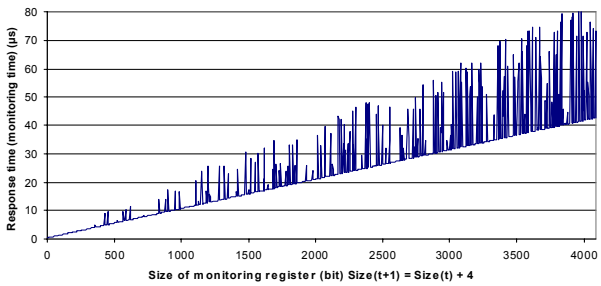
5.1. Monitoring performance analysis

Within larger digital systems, you often find a large number of hardware registers (monitoring registers). Generally, these kinds of registers control and monitor hardware functions within the system. It is common practice to spelling registers from the functional blocks (FB) of each IP cores, and interconnect them with the extended P1500 logic proposed in this approach. These facilities allow complete remote management and monitoring of the IP cores activities via simple SNMP requests (get-request, etc.).

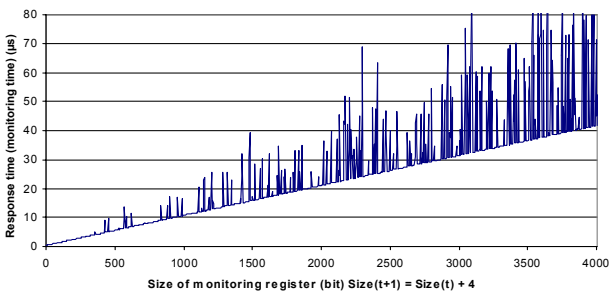
According to the size of monitoring registers that are embedded within the SoC, the monitoring performance has been estimated. The performance characteristics are evaluated in terms of the response time (delay), traffic intensity and loss rate. The proposed results also show the influence of system parameters such as: queueing capacity, chip speed and interarrival rate (**1/λ**, versus of arrival rate **λ**).

As discussed earlier, an extensive simulation work has been conducted to show how many parameters for modeling are necessary to influence the performance of monitoring operations. For instance, let us assume that a mean arrival rate is **λ=10000** SNMP messages per second, i.e. on average one message appears every **1/λ=1/10000= 0.00001** second. This implies that the interarrival times have an exponential distribution with an average interarrival time of **0.00001** second. Moreover, when the arrival rate **λ** increases the time of inter-arrival **1/λ**

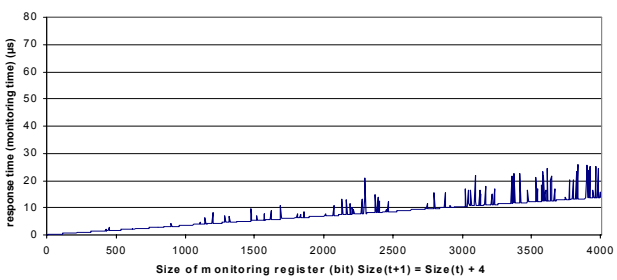
decreases. The system capacity (**N**) represents the number of SNMP messages supported. **Cq** is the capacity of the queue, therefore **N=Cq+1**. Figure 4 shows the end-to-end response time (monitoring time) of the system according to the size of monitoring register during the time.



(a) λ =10000, Speed=100MHz and Cq=1



(b) λ =10000, Speed=100MHz and Cq=2



(c) λ =10000, Speed=300MHz and Cq=2

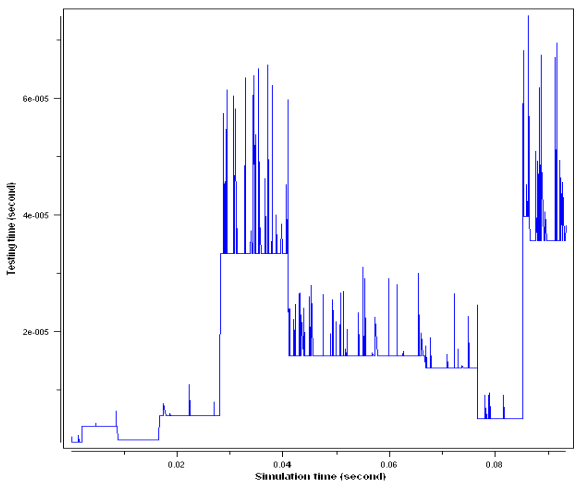
Figure 4 -End-to-end response time(µs) analysis

Figure 4 shows high and intensive peaks which explain the progression of the occupancy rate of the SoC under monitoring. If a new message arrives at the input ports of an occupied SoC, then the considered message is placed at the tail of its queue from where it will possibly be withdrawn. The latency between arrival and service is that which constitutes the peaks shown in figure 4. When the arrival rate λ increase, the number of peaks explodes. Also, the monitoring register lengths are a determining factor to regularize the occupancy rate of the SoC.

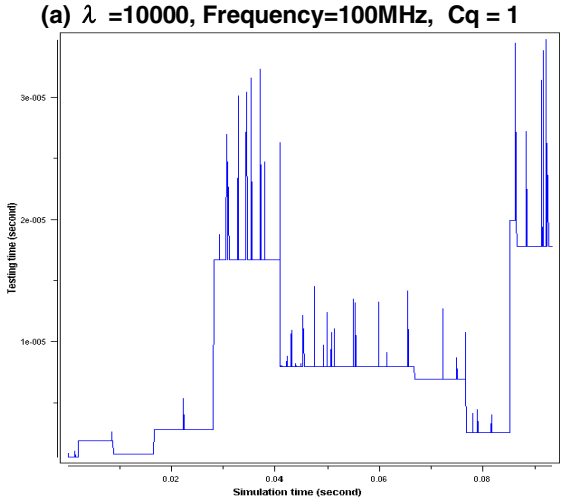
Each time the length of the monitoring registers grow, the SoC load increases which causes increasingly higher latencies. Note that the second curve (Fig. 4.b) presents peaks higher than the first curve. This is due to the waiting capacity **Cq**. With an increased capacity for waiting, there is less loss, giving higher response times. In figure 4.c, the clock frequency of the on-chip DFT is increased to 300MHz. Note that the intensity of the specific peaks decreases by a third which is why the clock frequency constitutes an important parameter in enhancing the monitoring performance of the proposed SNMP/1500 architecture.

5.2. Testing performance analysis

Several simulations have been conducted using several ITC'02 SoC Test Benchmarks [10]. The proposed results also show the influence of system parameters such as: queueing capacity (**Cq**), chip speed (clock frequency of test architecture) and interarrival rate (SNMP messages per second).



(a) λ =10000, Frequency=100MHz, Cq = 1



(c) λ =10000, Frequency =200MHz, Cq = 1

Fig. 5– Instantaneous testing time for SoC d695 (Duke University)

In figure 5, the instantaneous testing time is shown according to characteristic of SoC test benchmark **d695** (number of test patterns, number and length of scan chains, etc.). The test of the SoC **d695** is not complex and its traffic intensity converges towards zero because it does not use very long scan chains. Simulations of figure 5 show specific peaks of differing intensity. Such peaks express the latency of the messages during the functioning of the on-chip DFT. The clock frequency of the architecture influences considerably the performance of the test processes (Fig. 5.a and 5.b). Traffic intensity is

a measure of the congestion of the system. If it is near to zero then there is very little queueing, and in general as the traffic intensity increases (to near 1 or even greater than 1) the amount of queueing increases.

Table 2 gives simulation results of three ITC'02 SOC Test Benchmarks: **u226**, **p22810**, **p34392**. This table is organized as follows. Column 1 gives the names of the SoCs. In Column 2, the number of modules is listed. Column 3 shows the total number of input/output terminals in the SoC; this is the sum of input/output counts for all modules. Column 4 shows the total number of scan flip-flops in the SoC; this is the sum of scan chain lengths of all modules. Column 5 lists the sum of test pattern counts of all tests. Column 6 gives the test data volume (in Kbytes) generated by the ATE; this is the data volume which is needed for a **100%** faults coverage, etc. The next columns give the results in two cases: SoC without a queue (**Cq=0)** and SoC with a queue where its capacity is only one SNMP message (**Cq=1**). Column 7 presents the rates of test vectors processed by the SoCs under test. The fault coverage can be deduced from the number of test vectors processed. The testing time (in ms) is given in the last column.

It is noteworthy that the benchmarks used have differing test complexities. It is clear that a high pattern count does not directly imply a large test time. The testing time is dependent on the total test pattern time (the number of clock cycles that it takes to load and unload one test pattern). Some tests have fewer test patterns, but utilize very long scan chains, whereas other test have many patterns, but do not use scan chains at all.

6. Conclusion

A new approach to remote testing, diagnosis and monitoring of System on Chips and their embedded IP cores is presented. The approach is based on an implementation of a hardware-based network management application called a proxy agent. The proxy agent is a part of a hybrid testing/management solution that is based on a combination of the SNMP and a DFT standard called IEEE 1500. Through the use of network layered protocol wrappers, a test operator has the ability to manage and precisely test the activities of embedded cores (IP cores) by using existing TCP/IP networks. The approach was analyzed at the levels of both the IP core and the SoC.

In future research SNMPv3 capabilities will be used where the approach will consider authentication and privacy features to improve management critical hardware applications. A more extensive validation of the approach is also planned.

References

[1] F. Braun, J. W. Lockwood and M. Waldvogel, "Layered Protocol Wrappers for Internet Packet Processing in Reconfigurable Hardware", Proc. of Hot Interconnects 9 (HotI-9), pp. 93-98, California, USA, Aug 2001.

[2] J. W. Lockwood, C. E. Neely, C. K. Zuver, J. Moscola, S. Dharmapurikar and D. Lim, "An Extensible, System-On-Programmable-Chip, Content-Aware Internet Firewall", Field-Programmable Logic and Applications (FPL'03), pp. 859-868, Lisbon, Portugal, October 2003.

[3] J. Moscola, J. W. Lockwood, R. P. Loui and M. Pachos, "Implementation of a Content-Scanning Module for an Internet Firewall", 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'03), pp. 31-38, California, USA, April 2003.

[4] L. Miclea, Sz. Enyedi, G. Toderean, A. Benso and P. Prinetto, "Agent Based DBIST / DBISR and its Web / Wireless Management", International Test Conference 2003, Charlotte, NC, USA, pp. 952-960, October 2003.

[5] E. J. Marinissen and Y. Zorian, "Challenges in Testing Core-Based System ICs", IEEE Communication Magazine, Vol. 37, No. 6, pp. 104-109, June 1999.

[6] E.J Marinissen, R.Kapur, M. Lausberg, T. McLaurin, M. Ricchetti, and Y. Zorian, "On IEEE P1500's Standard for Embedded Core Test", Journal of Electronic Testing: Theory and Applications, vol. 18, no. 4-5, pp. 365–383, August-October 2002.

[7] O. Laouamri and C. Aktouf, "Enhancing Testability of System on Chips Using Network Management Protocols", In Proc. of IEEE Design Automation and Test in Europe (DATE'04), pp. 1370-1371, Paris, France, February 2004.

[8] O. Laouamri and C. Aktouf, "Towards a More Precise Network Management Through Electronic Design", In Proc. of IEEE 3rd International Conference on Networking, (ICN'04), pp. 45-49, Guadeloupe, France, February 2004.

[9] D. Harrington, R. Presuhn and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", RFC 3411, STD 62, Category Standards Track, December 2002.

[10] E.J. Marinissen, V. Iyengar and K. ChAkrabarty, "A Set of Benchmarks for Modular Testing of SOCs", In Proc. of IEEE International Test Conference (ITC'02), pp. 519-528, Baltimore, MD, October 2002.

[11] OMNeT++ object-orie:nted discrete event simulation system. URL reference: http://www.omnetpp.org, 2004.

SoC	#IP cores	∑ I/Os	∑ SFFs	∑ Test Patterns	ATE Test data (Ko)	Test vectors processed (%)		Testing time (ms)	
						Cq = 0	Cq=1	Cq = 0	Cq=1
u226	10	376	1040	5148569	13.5	99.27	100	7.29	7.64
p22810	29	4283	24723	24890	814.3	96.2	99.16	98.06	137.97
p34392	20	2057	20948	66349	1802.5	97.41	99.03	178.22	295.54

Tab. 2– Performances analysis of testing operations for λ=10000, Clock frequency =100MHz