INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

A SECURE TEST WRAPPER DESIGN USING VLSI TECHNOLOGY

P.Karthikeyan[1]

Asst. Professor, karthickcnp@gmail.com

Abstract: - This project presents a secure test wrapper (STW) design that is compatible with the IEEE 1500 standard. STW protects not only internal scan chains but also primary inputs and outputs, which may contain critical information (such as encryption keys) during the system operation. To reduce the STW area, flip-flops in the wrapper boundary cells also serve as the LFSR to generate the golden key. We propose to design a Cipher block chain based test wrapper design. We also propose the testing methodology based on Golden keys. We perform testing outside the module which provides high security. The designed module or block can be ported in to FPGA Processor. The system will be designed using Verilog HDL and simulated using Modalism Software.

Keywords - Design for testability, scan, security

1. Introduction

In the system-on-chip (SoC) era, many embedded cores are purchased from external Intellectual Property (IP) vendors. To ensure the secure operation of SoCs, it is very important for SoC integrators to prevent critical data stored in IP cores from being hacked. Data protection is especially important for encryption/decryption IP cores, such as Advanced Encryption Standard (AES) decoders/encoders, in communication applications. Security has become an important concern for modern SoC designers as well as computer-aided design (CAD) tools [1]–[3]. [1]–[3]. In scan mode, SoCs are especially vulnerable to attack because internal data can be controlled and observed through scan chains [4],[5].

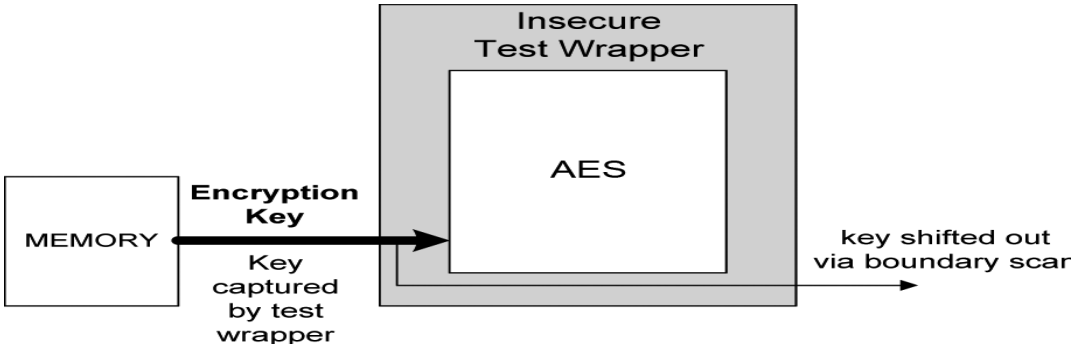


Figure 1: Security problem of a test wrapper

Traditional secure design for testability (DFT) techniques tries to protect internal scan chains from being controlled or observed by unauthorized test access. Traditional secure DFT techniques may be suitable for single chips but not for SoC because the primary inputs/outputs of cores are not protected. Fig. 1 illustrates a potential attack via insecure test wrapper. During critical system operations, important data (such as AES encryption keys) are transferred from memory to the AES core via primary inputs or outputs. Those important data, when captured by the test wrappers, may be shifted out and observed in boundary scan mode. The original IEEE 1500 test wrapper does not provide any security protection for PI/PO. It is therefore very important to have a secure test wrapper designed for critical IP cores, such as AES. The proposed STW technique has the following advantages. First of all, STW is designed and implemented by SoC integrators so it provides security in the system level. Once a core is wrapped by STW, not even the original (external) IP provider can attack the core, which increases the confidence of SoC costumers. (Direct memory dump is too large and the key position is unknown to the IP designer. However, the exact location of the key in the IP core is known to the IP designer.) Second, unlike other techniques which change the original circuit design, STW requires no modification in the original IP cores. SoC integrators can easily apply STW without any knowledge of the IP internal design. Third, STW is fully compatible with the IEEE 1500 standard. STW wrapped cores works seamlessly fine with original IEEE 1500 wrapped cores in the same SoC. Of course, only STW wrapped cores are protected in scan mode, the others are not. Finally, the STW key length can be easily extended using LFSR without extra area overhead. On the contrary, using a fixed seed is not flexible, and the area overhead increases with the size of key. The following assumptions are made. First of all, the LFSR polynomial is kept secret and only the SoC integrator knows the goldenSTWK not even the original IP provider could unlock the STW. This is a valid assumption because the SoC integrator is in charge of testing the SoC while IP providers are not. Secondly, a functional test of STW is proposed based on the RTL design generated by our tool, *STW compiler*. As long as the STW is synthesized with the RTL code provided by the STW compiler, no RTL faults will unlock the core without receiving the correct STWK. Since the actual gate level implementation of STW depends on the synthesis library, it is impossible to propose a structural test for all possible implementations of STW.

2. Proposed Work

2.1 Architecture of STW

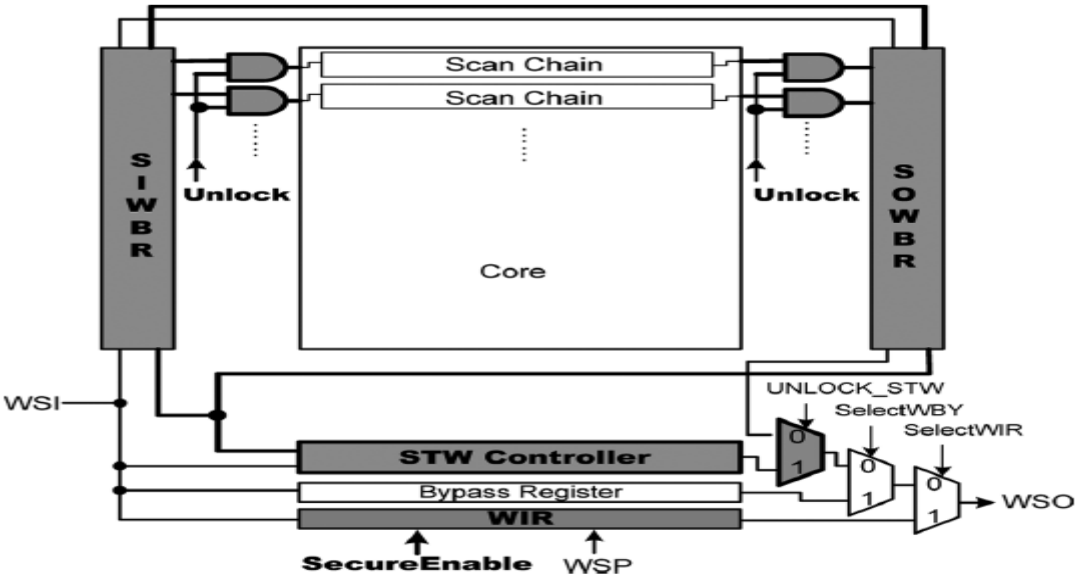
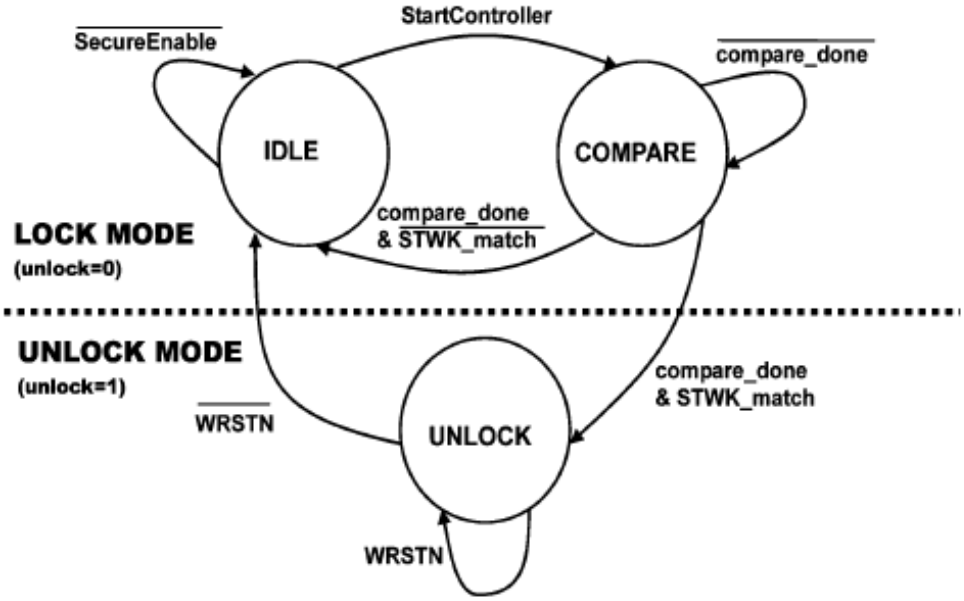


Figure 2: Architecture of STW

2.1.1 Hardware Architecture

Figure 2, shows the architecture of the proposed STW, with the modified components highlighted. The inputs and outputs of internal scan chains are gated by the unlock signal, which is generated by the STW controller. Both IWBR and OWBR are replaced by their secure versions, SIWBR and SOWBR, respectively. In addition to the original wrapper serial ports (WSP), an extra input Secure Enable is needed. The WIR is also slightly modified to decode one extra instruction, *UNLOCK_STW*, and generate STW control signals.
Figure 3, shows the state diagram and output signals of the STW controlled. When powered up, the STW controller is in the IDLE state. When the Init LFSR signal is asserted, the LFSR is loaded with a predefined seed that generates the golden STWK. When Start Controller=1, the controller enters the COMPARE state, in which the input bit stream from the WSI is compared with the golden STWK.

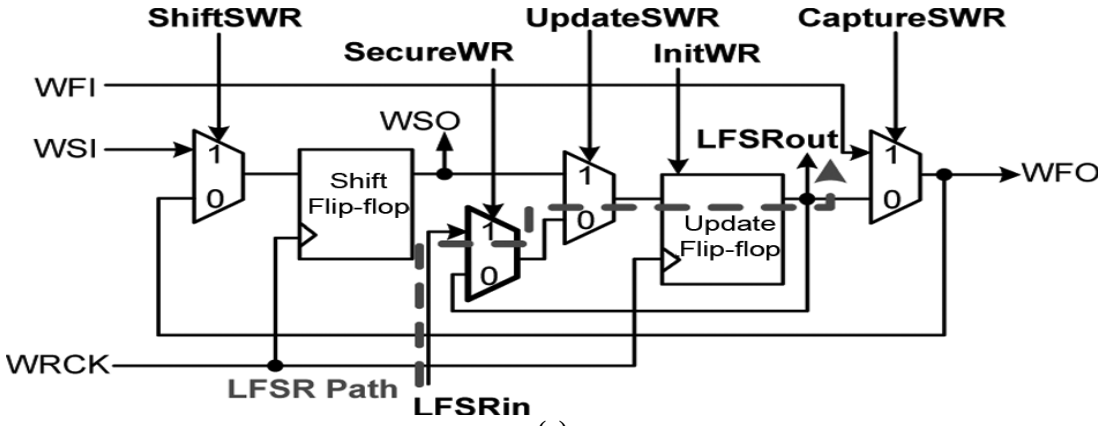


output signals	IDLE state	COMPARE state	UNLOCK state
InitLFSR	1	0	0
EnLFSR	0	1	0
unlock	0	0	1

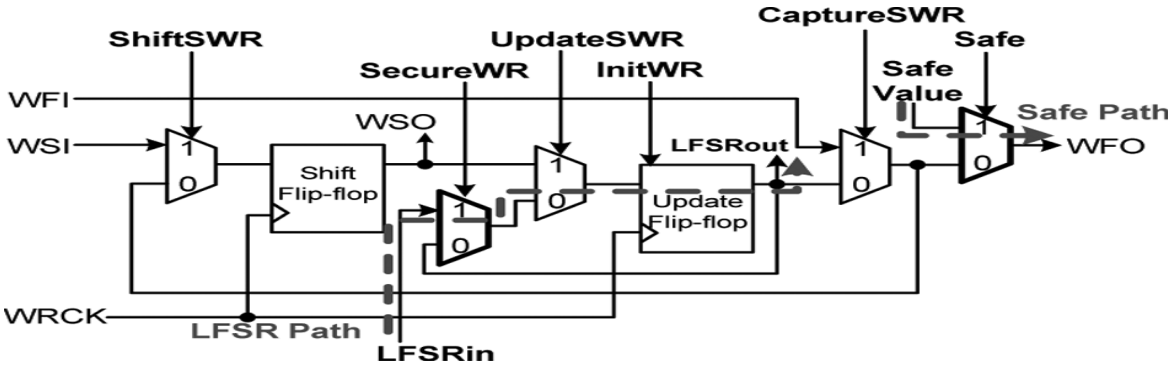
Figure 3: STW Controller

After the comparison is done (compare_done=1) , if every bit of the input bit stream matches the golden STWK(STWK_match=1) , the wrapper enters the UNLOCK state. The unlock signal rises to one in the UNLOCK state. If there is any mismatch during the key comparison, STW returns to the idle state without unlocking. Only after STW enters the UNLOCK state are the internal chains allowed to shift and wrapper boundary registers allowed to capture. When WRSTN=0, the STW controller resets and then returns to the IDLE state. For simplicity, the comparator and counter are not shown in this figure. This is a Moore finite state machine, so glitches on the unlock signal are prevented during state transitions.

In the COMPARE state, the golden STWK is generated by an on-chip LFSR. Longer LFSR provides better security with the cost of larger area. To reduce the area overhead, the LFSR can be embedded in the wrapper boundary registers. Since some WBCs may have more than one flip-flop the update flip-flops can serve as the LFSR beforeSTW is unlocked.



(a)



(b)

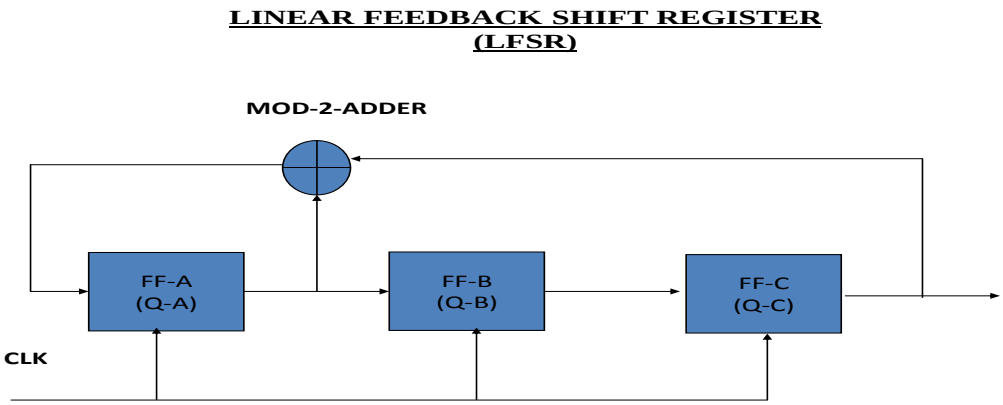
Figure 4: (a) SIWBC. (b)SOWBC

Figure 4(a), shows an implementation of the secure input wrapper boundary cell (SIWBC). Compared with the original WBC, the SIWBC has one additional MUX (bolded).

When InitLFSR is asserted, the update flip-flop is initialized to either a one or zero, which is predetermined by the LFSR seed solver .When the LFSR is enabled (EnLFSR=1), the LFSR path is connected from LFSRin to LFSRout through the update flip-flop. The other paths of the SIWBC remain unchanged. When CaptureSWR=1,the function path is connected from WFI to WFO. When ShiftSWR=1, the shift path is connected from WSI to WSO through the shift flip-flop.

In total, five new control signals and two I/O signals are needed for this SIWBC. Fig. 4(b) shows an implementation of the secure output boundary cell (SOWBC). Compared with the original WBC, the proposed SOWBC has two additional multiplexers (bolded). The LFSR path is connected from LFSRin to LFSRout through the update flip-flop. The safe path provides a safe output so that the core under test does not generate hazardous outputs to the subsequent cores. In total, six new control signals and three I/O signals are added to thisSOWBC.

From Figures 4(a) and 4(b), it is shown that SIWBC is smaller than SOWBC because the former does not require a safe path. Therefore, to reduce the area of STW, the SIWBC has higher priority to be used for LFSR than the SOWBC. That means, if the degree of LFSR is smaller than the number of primary inputs, only the SIWBC is used. SOWBC is used only when the degree of LFSR is larger than the number of SIWBC. The degree of LFSR must be smaller than the total number of WBCs; otherwise, additional flip-flops are needed to implement the LFSR. Usually, this limitation is not a problem because most cores have hundreds of I/O pins, which is plenty enough for a very long LFSR.



For length-*m* registers they produce a sequence of length 2^*m* − 1 . The duty cycle of the STWK Sequence is

c=[M-1]/[N-1];

Where, N=Length of STWK Sequence and M=Number of ones in STWK-Sequence.

Balance Property:-Of the (2^m)-1 terms,2^(m-1) are one and {2^(m-1)}-1 are zero.

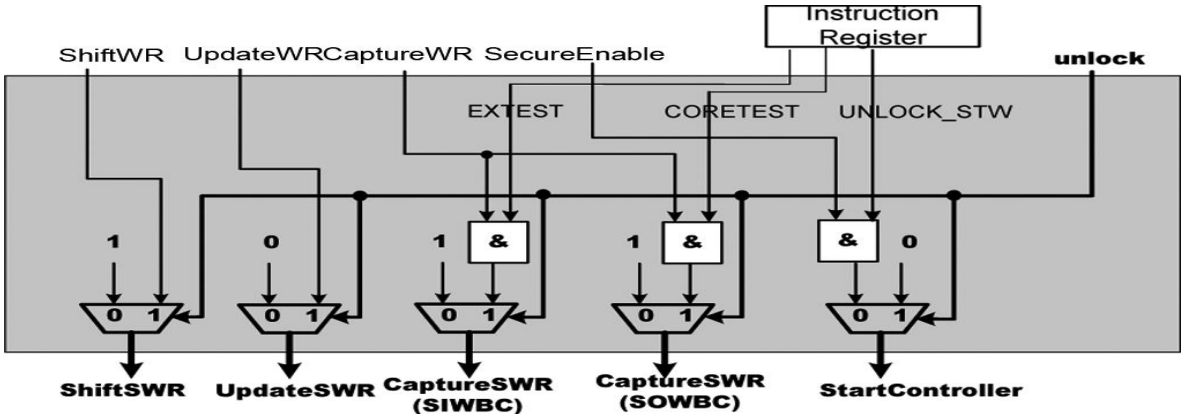


Figure 5: Secure WIR

Figure 5, shows how control signals are generated in the secure version of WIR. The ShiftSWR is selected by the unlock signal. Before STW is unlocked (unlock=0), secure boundary wrapper registers are allowed to shift (ShiftSWR=1) but no update (UpdateSWR=0) is allowed. This configuration ensures that the secure WBR is never controlled or observed through the boundary scan chain before being unlocked. The STW controller starts (StartController=1) if an instruction *UNLOCK_STW* is loaded and SecureEnable=1. Only after the wrapper is unlocked (Unlock=1) do ShiftSWR and UpdateSWR operate normally. When the user loads an *INTEST* instruction, then CaptureSWR of the SOWBC is connected to CaptureWR. When the user loads an *EXTEST* instruction, then CaptureSWR of the SIWBC is connected to CaptureWR.

2.1.2 Security Analysis

The security of STW is determined by two factors: the length of STWK(k) and the degree(d) of LFSR . To quantitatively analyze the security of STW, three possible scenarios are discussed in the following. First of all, if the hacker knows nothing about k or d , then he has to try exhaustively all bit sequences of lengths 1 to .Security is defined as the reciprocal of the probability to unlock STW by random trial. Second, if the hacker knows k but not d, he has to try exhaustively all bit sequences of length. The security of scenario two is equal to the number of all possible bit sequences of length.

$$\mathbf{Security_{scenario\text{-}2} = 2^{k}}$$

In the last scenario, if the hacker knows both k and d, then his search space is slightly smaller than that of the second scenario. The security of scenario three can be estimated as

$$\mathbf{Security_{scenario\text{-}1} = 2^{k+1}}$$

k and 2d. Normally, k is no less than because it is a waste of area to design an LFSR longer than the key. It is seen that STW provides very good security for a reasonable number d .

2.1.3 Testing STW

It is very important to test the STW and make sure that no security holes are induced by any defects in the STW. Five STW components have to be tested thoroughly: the STW controller, AND gates of internal scan chains, SIWBC, SOWBC, and WIR. Since the STW controller generates control signals, it must not be tested by scanning. Furthermore, the STW controller can be implemented using different cell libraries so it is impossible to generate a structural test for all possible implementations. In test A, the UNLOCK_STW instruction is loaded and an incorrect STWK is applied. The STW should not be unlocked in this test. The correct value of unlock is zero and the faulty value is one. An additional multiplexer (highlighted) is inserted to observe the unlock signal at the WSO output. After that, standard IEEE 1500 instructions, such as *EXTEST* and *INTEST*, are loaded and executed. Neither the internal chains nor the external chains are supposed to shift in the meantime. Test A detects stuck-at one faults on the unlock signals in the WIR as well as AND gates of the internal chains. In test B, the UNLOCK_STW instruction is loaded again but this time a correct STWK is applied to unlock STW. After that, IEEE 1500 instructions are loaded and executed. Test B detects the stuck-at zero fault on the unlock signal in the WIR as well as AND gates of the internal chains. In test C, an alternating "101010…" pattern is shifted into the wrapper boundary cells, followed by an immediate UpdateSWR and CaptureSWR. For a fault-free circuit, the shift-out pattern is identical to the shift-in pattern. However, if EnLFSR is stuck at one, then the shift out pattern would be altered. Every RTL fault in the STW controller is detected by the proposed functional test so the RTL fault coverage is 100%. STW controller is very small in size so the structural fault coverage of STW controller is almost negligible compared to the whole circuit. After unlocking STW, both the SIWBC and the SOWBC can be fully tested by regular scan tests. The test patterns of SIWBC and SOWBC can be easily generated by ATPG. Similarly, faults in WIR can also be detected by ATPG patterns because the WIR can be scanned. It is the SoC integrators' responsibility to generate valid ATPG test patterns.

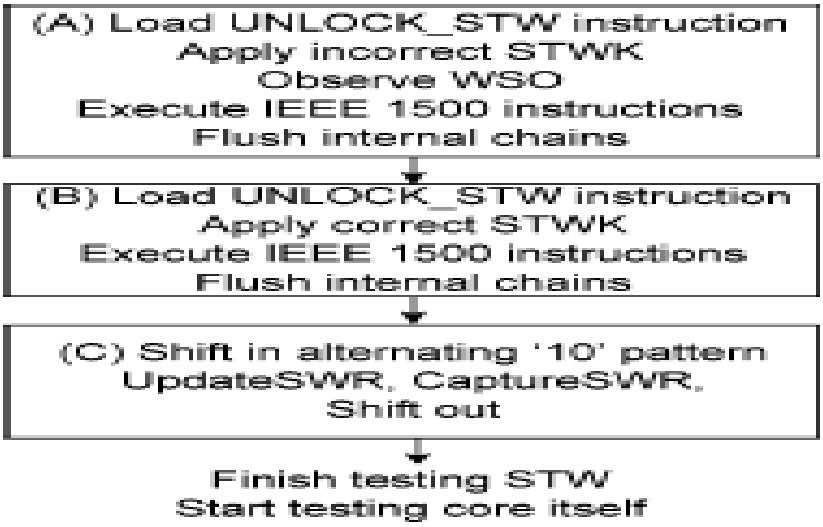
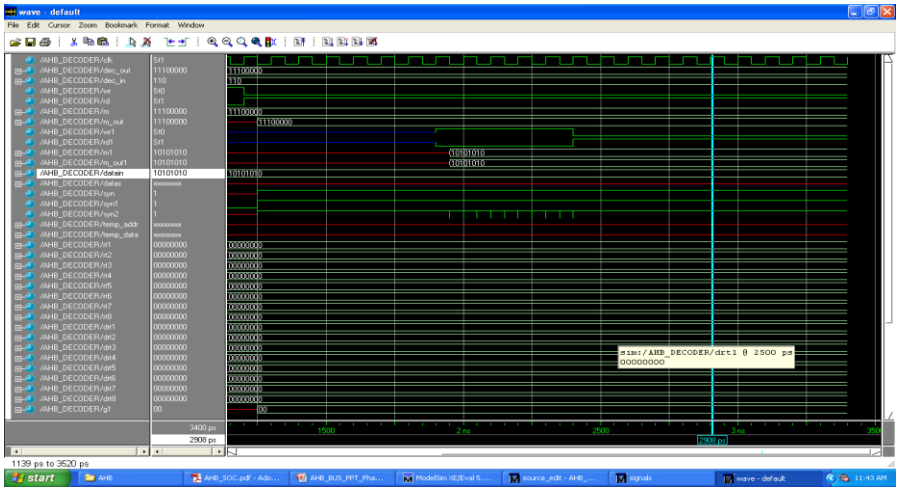


Figure 5: Functional test of STW

3 Simulation Results

In proposed architecture, the higher throughput, less power consumption and less area are achieved .The architecture is implemented using spartan3E family and XC3S500E device in Xilinx 9.2i.The proposed system is written in Verilog HDL language and synthesized in Xilinx 9.2i and stimulated using Modalism 5.7. Dynamic power is defined as amount of power consumed by switching activities of FF, where as static power is power consumed by leakage current. In 200MHz operation the Coprocessor consumes 79mW in static and 96mW in dynamic in the total summation of 175mW.



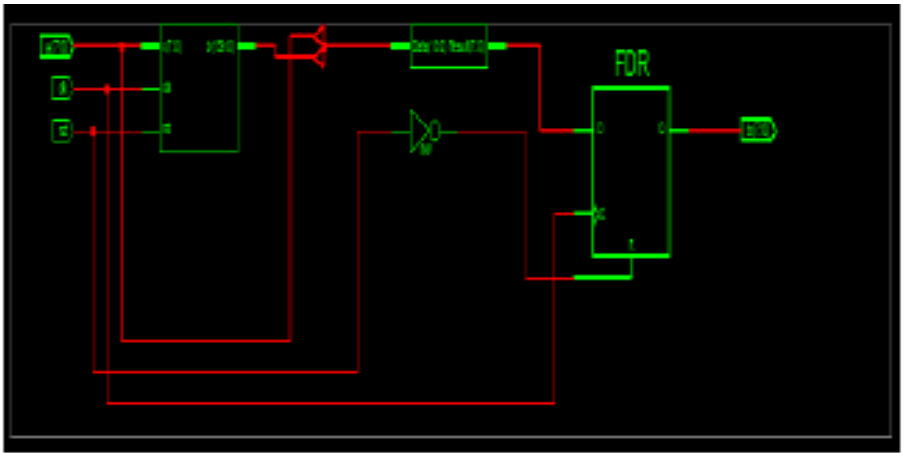


Figure 6: RTL View of Proposed Architecture

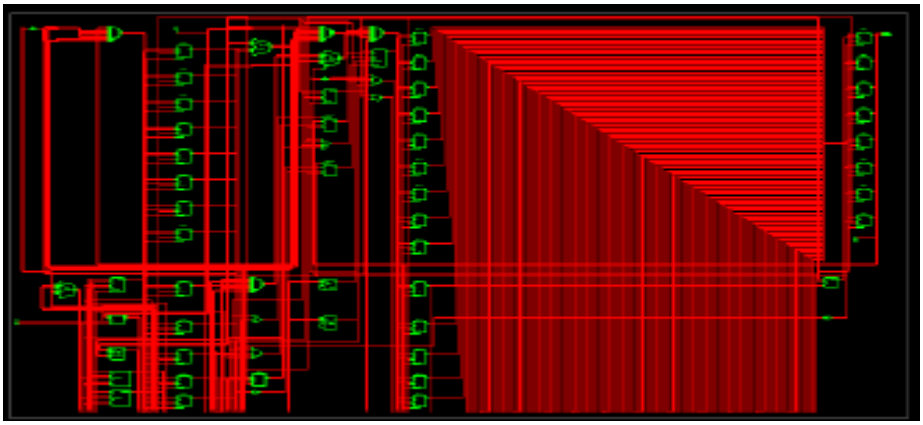


Figure 7: Technology Schematic View

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	682	9,312	7%	
Number of 4 input LUTs	1,987	9,312	21%	
Logic Distribution				
Number of occupied Slices	1,034	4,656	22%	
Number of Slices containing only related logic	1,034	1,034	100%	
Number of Slices containing unrelated logic	0	1,034	0%	
Total Number of 4 input LUTs	2,008	9,312	21%	
Number used as logic	1,987			
Number used as a route-thru	21			
Number of bonded IOBs	18	232	7%	
IOB Flip Flops	8			
Number of GCLKs	1	24	4%	
Total equivalent gate count for design	22,689			
Additional JTAG gate count for IOBs	864			

Figure 8: Device Utilization Results

REFERENCES

[1] P. Schaumont and A. Raghunathan, "Guest editors' introduction: Security and trust in embedded-systems design," IEEE Des. Test Comput., vol. 24, no. 6, pp. 518–520, Dec. 2007.

[2] I.M. R. Verbauwhede, Ed., Secure Integrated Circuits and Systems. Berlin, Germany: Springer, 2010.

[3] K. Tiri, "Design for side-channel attack resistant security ICs," Ph.D. dissertation, Electr. Eng. Dept., Univ. of California, Los Angeels, 2005.

[4] R. Goering, "Scan design called portal for hackers," EE Times 2004 [Online]. Available: http://www.eetimes.com/electronics-news/4050578/Scan-design-called-portal-for-hackers

[5] R. Kapoor, "Security versus test quality: Are they mutually exclusive?," in Proc. IEEE Int. Test Conf., 2004., p. 1414.

[6] G.-M. Chiu and J. C. M. Li, "IEEE 1500-compatible secure test wrapper for embedded IP cores," in Proc. IEEE Int. Test Conf., 2008, p. 1, Poster 4.

[7] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing scan design using lock and key technique," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst., 2005, pp. 51–62.

[8] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "A low-cost solution for protecting IPs against scan-based side-channel attacks," in Proc. IEEE VLSI Test Symp., 2006, pp. 94–99.

[9] O. Kommerling and M. G. Kuhn, "Design principle for tamper resistant smartcard processors," in Proc. USENIX Workshop Smartcard Technol., 1999, pp. 9–20.

[10] K. Hafner, H. C. Ritter, T.M. Schwair, S.Wallstab, M. Deppermann, J. Gessner, S. Koesters, W. D. Moeller, and G. Sanweg, "Design and test of an integrated cryptochip," in Proc. IEEE Des. Test Computer., 1991, pp. 6–17.

[11] R. Karri, K. Wu, and P. Mishra, "Fault-based side-channel cryptanalysis tolerant architecture for Rijndael symmetric block cipher," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst., 2001, pp. 427–435.

[12] B. Yang, K. Wu, and R. Karri, "Secure scan: A design-for-Test architecture for crypto chips," in Proc. IEEE/ACM Des. Automatic Conf., 2005, pp. 135–140.
[13] D. Hely,M.-L. Flottes, F. Bancel, B. Rouzeyre, N. Berard, and M. Renovell, "Scan design and secure chip," in Proc. Int. On-Line Test Symp., 2004, pp. 219–226.
[14] D. Mukhopadhyay, S. Banerjee, D. RoyChowdhury, and B. B. Bhattacharya, "Cryptoscan: A secured scan chain architecture," in Proc. Asian Test Symp., 2005, pp. 348–353.
[15] S. Paul, R. S. Chakraborty, and S. Bhunia, "Vim-scan: A low overhead scan design approach for protection of secret key in scan-based secure chips," in Proc. IEEE VLSI Test Symp., 2007, pp. 455–460.
[16] C. Liu and Y. Huang, "Effects of embedded decompression and compaction architecture on side-channel attack resistance," in Proc. IEEE VLSI Test Symp., 2007, pp. 461–468.
[17] IEEE Computer Society, IEEE Standard Testability Method for Embedded Core-Based Integrated Circuits, IEEE Std. 1500-2005.
[18] J. Daemen and R. Rijmen, The Design of Rijndael: AES—The Advance Encryption Standard. Berlin, Germany: Springer-Verlag, 2002, pp. 31–62.
[19] A. Biryukov and D. Khovratovich, "Related-Key Cryptanalysis of the Full AES-192 and AES-256," [Online]. Available: https://cryptolux.org/mediawiki/uploads/1/1a/Aes-192-256.pdf
[20] Y. B. Zhou and D. G. Feng, "Side channel attacks: Ten years after its publication and the impacts on cryptographic module security testing,"2005 [Online]. Available: http://eprint.iacr.org/2005/388
[21] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Side channel cryptanalysis of product ciphers," in Proc. Eur. Symp. Res. Comput. Security, 1998, pp. 97–110.
[22] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," Lecture Notes in Computer Science, vol. 1294, pp.513–527, 1997.
[23] S. Ravi, A. Raghunathan, and S. Chakradhar, "Tamper resistance mechanisms for secure embedded systems," in Proc. Int. Conf. VLSI Design, 2004, pp. 605–611.