Synthesis Based Constant Propagation Attack on Logic Locking

Anita Titus[1], Preethi.K[2], Shreya.V[2], Swethapriya.P[2]

Professor[1], Student[2],

[1,2]Department of Electronics and Communication Engineering, Jeppiaar Engineering College , Chennai, Tamil Nadu, India.

Abstract:

Hardware intellectual property (IP) piracy and misuse have introduced new challenges in the semiconductor industry as untrusted parties in the IP's life cycle may clone, reverse-engineering, or extract important design secrets from an IP. A promising solution to protect a hardware IP against such attacks is to perform logic locking, where additional logic controlled by a secret key is inserted in strategic locations of an IP to lock the functionality when the correct key is not available. SCOPE is oracle-less and requires no knowledge about the locking algorithm or the locked design by an attacker. The introduced attack performs a synthesis-based analysis on each individual key-input port and looks for meaningful design features that may help derive the correct key value. SCOPE offers two attack modes with varying complexity and effectiveness, a linear regression test, and an unsupervised machine- learning analysis. This performs SCOPE to a number of existing locking techniques and demonstrate that the average attack accuracy is 84.13% with high scalability in terms of design size. SCOPE is enhanced to be unsupervised, as it does not require any training data. These constraints are analyzed using machine learning algorithms in IP core mapping.

Keywords: Hardware security, IP protection, logic locking, obfuscation, security evaluation.

1. Introduction

The semiconductor industry has grown rapidly in critical applications, such as automobiles, military equipment, health-care applications and the Internet of Things (IOT). Due to this growth, the process of hardware intellectual property (IP) development has been considerably globalized. Many fab-less design houses outsource the manufacturing process of their developed systems to third-party facilities. Logic locking has received considerable interest as a prominent technique for protecting the design intellectual property from untrusted entities, especially the foundry. Recently, machine learning (ML)-based attacks have questioned the security guarantees of logic locking and have demonstrated considerable success in deciphering the secret key without relying on an oracle, hence, proving to be very useful for an adversaryin the fab. Such ML-based attacks have triggered the development of learning-resilient locking techniques. Although this practice is widely adopted, security challenges when sending a design to an untrusted party are a major concern. Attackers may perform malicious activities, such as reverse engineering, counterfeiting, and backdoor modification to the IP. In addition, attackers may insert hardware Trojans, which can be used by the attacker to leak information, corrupt a function, or degrade the performance of the IP [9]. To mitigate these security threats, logic locking has been introduced as a design modification process that integrates key-based locking gates to mask the functionality and hide the design intent of the IP. In this the main focus is to develop this attack with minimal requirements for the attacker to achieve key extraction with high accuracy. Hence, SCOPE is created as an oracle-less attack, as it does not requirean unlocked circuit, nor any knowledge about the type of the locking mechanism used or any familiarity with the design. In other words, SCOPE treats the locked netlist as a black box. Although the basic concept is similar, SCOPE is unsupervised, as it does not require any training data, while surpassing, or performing similar to sweep.

2. Literature Review

2. Literature Review

Jain, A. et al. [7] proposed a TAAL attack is based on implanting a hardware Trojan in the netlist which leaks the secret key to an adversary once activated. Thangam et al. [3] proposed a approach increases security level and hence applied in a cryptographic algorithm. Montgomery algorithm is the cryptographic algorithm which will be tested by the logic locking technique. Chiang et al. [4] proposed a new cyclic logic locking method to invalidate attacks simultaneously. Yasin et al. [1] have discussed about various SOC vulnerable attacks. Since the globalization of integrated circuit (IC) supply chain and the emergence of threats, such as intellectual property (IP) piracy, reverse engineering, and hardware Trojans, have forced semiconductor companies to revisit the trust in the supply chain. Rostami et al. [2] have discussed in detail about the multinational, distributed, and multistep nature of integrated circuit production supply chain. Sirone et al. [5] have analyzed the FALL attacks and detection of functional faults in digital circuits. Rahman et al. [6] Proposed a multi-layered defense structure to establish the defense-in-depth in the integrated circuits, “Defense-in-Depth: A Recipe for Logic Locking to Prevail.” addressing the challenge of incorporating the multi-layer mechanism with significant logical obfuscation.

3. Proposed System

In the proposed system, evaluation on in-build prediction model is focused. The system comprises of machine learning algorithm that holds the various attack constraints as attributes and tries to reveal the attack status as early as possible. For achieving that, the system works in two modes, in which the fully automated mode, keeps running inside the on-chip until active condition. The second mode is the sleep- mode accelerator that works in the static condition of the chip extracting the feasible probing attacks heldwith the IC. A customized SCOPE architecture to search and find out the attacks present in the logical systems are evaluated [6]. Proposed work considers test circuit to deploy the logic attack detection framework. Pattern based search mechanism for detecting the abnormal activity in the logical circuits are implemented. Probing attack, Forbidden attack and corruption attacks are implemented in the proposed system. The concept of probing, forbidden replay, Data corruptions are clearly shown Figure 1.

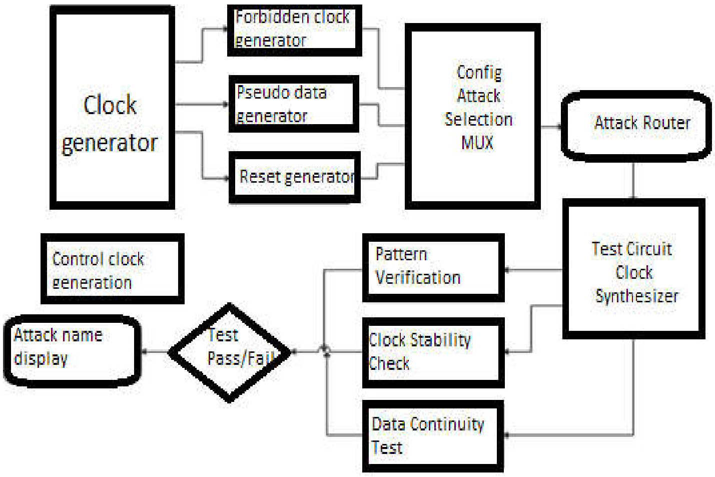


Figure 1. Block Diagram

In this block diagram clock pulse signals are generated using the clock pulse generator and different pulses are generated and passed into the attack selection MUX where the attacks are fetched into the pulse signals and then passed through the attack router which pushes it into the test circuit where different types of tests are conducted and if an attack is detected, it is displayed on the attack name display.

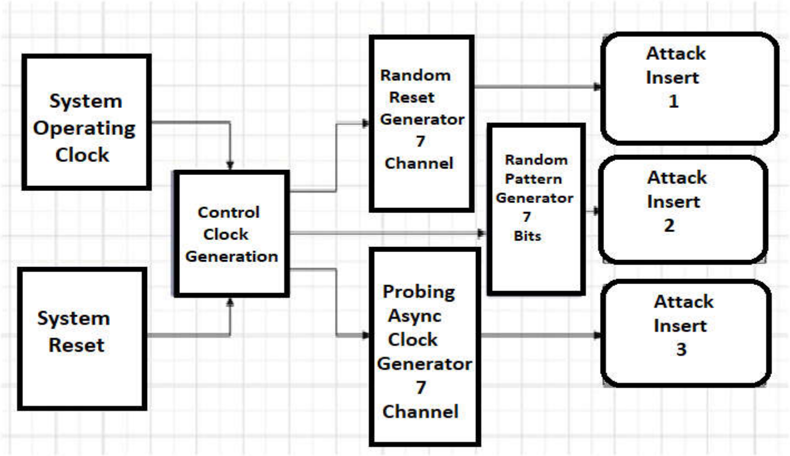


Figure 2. Block Diagram of Attack Generation

Figure 2 has seven input pins which is taken as the input. When the reset value is zero, the system starts to function. The system operating clock and the system reset are connected to thecontrol clock generation from which clock signals are generated and passed to the random resetgenerator which consists of seven channel and is split into seven bits of random pattern generator in which three types of attacks have been inserted. [9]

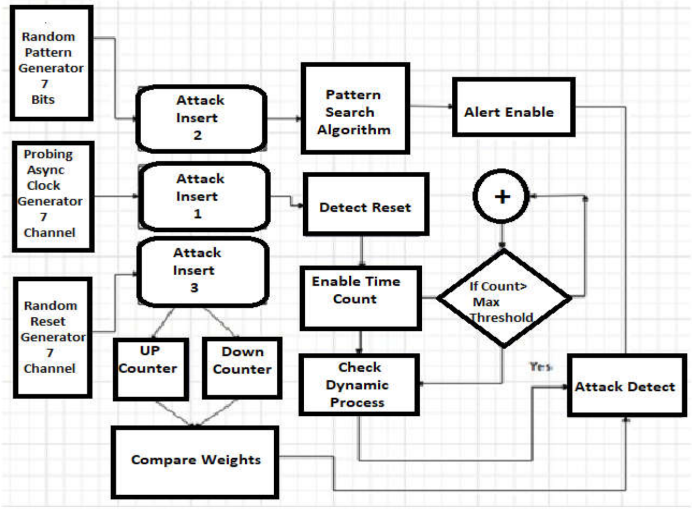


Figure 3. Block Diagram of Attack Detection

Figure 3 has the reset values from random reset generator using which it detects the replay attack (attack insert 3) from which the up counter and down counter weights have been compared and it has been fetched to the attack detector block. From the random pattern generator, the data insertion attack (attack insert 2) using the pattern search algorithm enables the alert and is given into the attack detector block [10].

Probing Asynchronous clock generator is fetched into the information leaking attack block (attack insert 1) from where reset is detected and enables the time count and then checks the dynamic process, If the threshold is maximum, the attack is detected or else it rechecks the presence of any other attack.

4. Results and Discussion

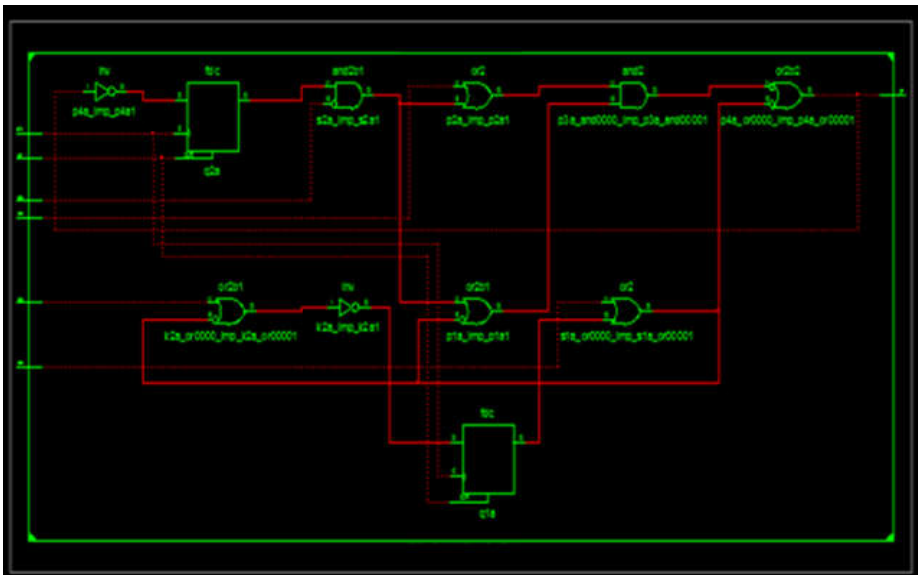
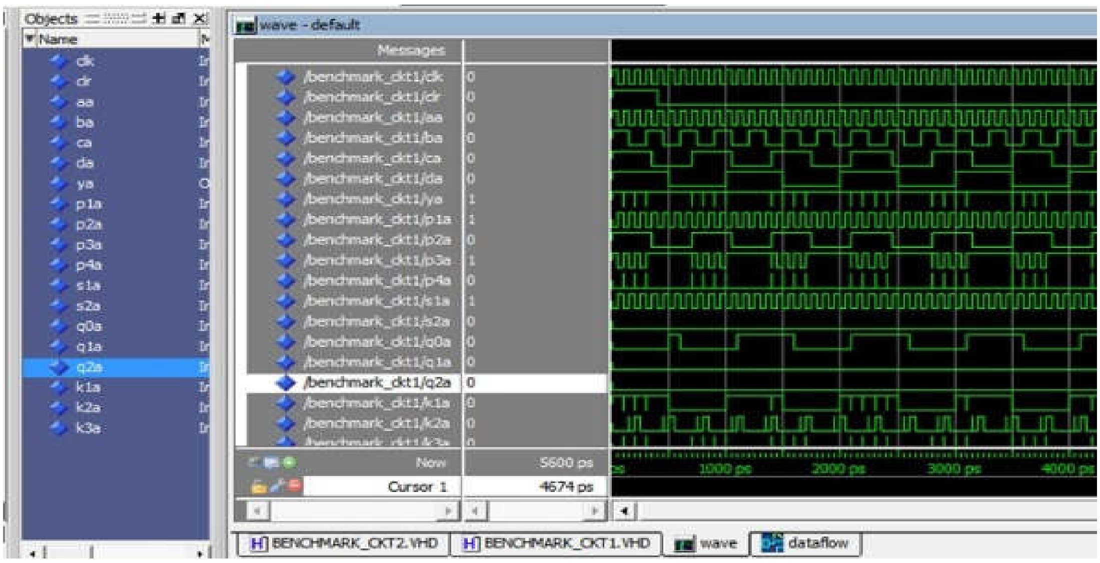


Figure 4. Logical Schematic of Clock Synthesizer

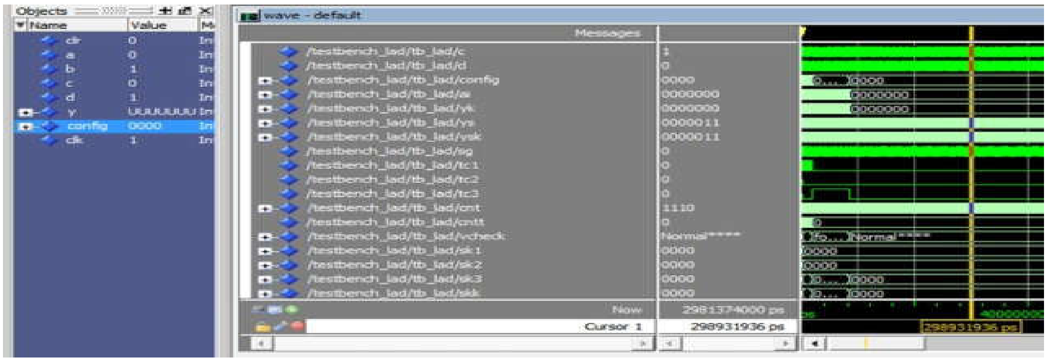
Figure 4 shows the logical schematic generated from the RTL design of test circuit named clock synthesizer.The clock synthesizer consists of low power clock gated flip flops with positive edge trigger mechanism adoptable for clock signals. The clock synthesizer works with reconfigurable mechanism.



```
/benchmark_ckt1/clk   0
/benchmark_ckt1/clr   0
/benchmark_ckt1/aa    0
/benchmark_ckt1/ba    0
/benchmark_ckt1/ca    0
/benchmark_ckt1/da    0
/benchmark_ckt1/ya    1
/benchmark_ckt1/p1a   1
/benchmark_ckt1/p2a   0
/benchmark_ckt1/p3a   1
/benchmark_ckt1/p4a   0
/benchmark_ckt1/s1a   1
/benchmark_ckt1/s2a   0
/benchmark_ckt1/q0a   0
/benchmark_ckt1/q1a   0
/benchmark_ckt1/q2a   0
/benchmark_ckt1/k1a   0
/benchmark_ckt1/k2a   0
/benchmark_ckt1/k3a   0
            Now   5600 ps
       Cursor 1   4674 ps
```

Figure 5. Simulation Result of Clock Synthesizer

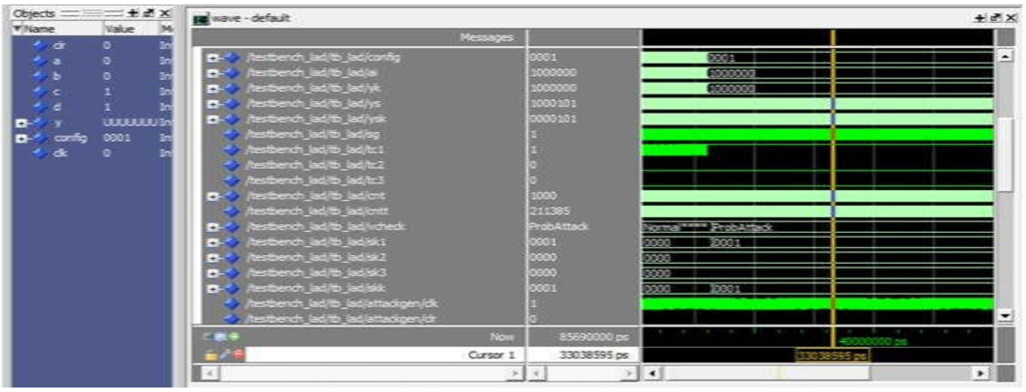
Figure 5 shows the simulation result in MODELSIM 6.3 G ALTERA software waveform window showing the reconfigurable clock generator outputs. It is clearly shown, that at some regions of clock signals, the occurrence of glitches and jitter comes. Clock synthesizer is one of the important circuits in any digital systems in which the generation of stable clock signals are focused.



```
Objects                     wave - default
Name    Value     M                                        Messages
clr     0         In        /testbench_lad/tb_lad/c        1
a       0         In        /testbench_lad/tb_lad/d        0
b       1         In        /testbench_lad/tb_lad/config   0000
c       0         In        /testbench_lad/tb_lad/ai       0000000
d       1         In        /testbench_lad/tb_lad/yk       0000000
y       UUUUUUU   In        /testbench_lad/tb_lad/ys       0000011
config  0000      In        /testbench_lad/tb_lad/ysk      0000011
clk     1         In        /testbench_lad/tb_lad/sg       0
                            /testbench_lad/tb_lad/tc1      0
                            /testbench_lad/tb_lad/tc2      0
                            /testbench_lad/tb_lad/tc3      0
                            /testbench_lad/tb_lad/cnt      1110
                            /testbench_lad/tb_lad/cntt     0
                            /testbench_lad/tb_lad/vcheck   Normal****
                            /testbench_lad/tb_lad/sk1      0000
                            /testbench_lad/tb_lad/sk2      0000
                            /testbench_lad/tb_lad/sk3      0000
                            /testbench_lad/tb_lad/skk      0000
                                                     Now   2981374000 ps
                                                Cursor 1   298931936 ps
```

Figure 6 Simulation Result of Attack Generator

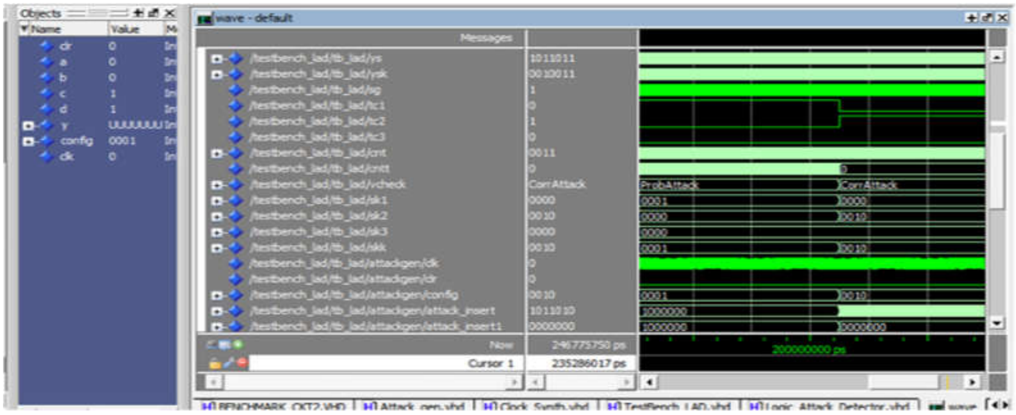
Figure 6 shows the Simulation results of attack generator in which the RTL code is derived to achieve low power circuitry, generating the probing attack generator, SOC replay attack generator and Data insertion attack generator. Each attack is fetched into the test circuit with respect to the 4-bit configuration value.



```
Objects                     wave - default
Name    Value     M                                                  Messages
clr     0         In        /testbench_lad/tb_lad/config             0001
a       0         In        /testbench_lad/tb_lad/ai                 1000000
b       0         In        /testbench_lad/tb_lad/yk                 1000000
c       1         In        /testbench_lad/tb_lad/ys                 1000101
d       1         In        /testbench_lad/tb_lad/ysk                0000101
y       UUUUUUU   In        /testbench_lad/tb_lad/sg                 1
config  0001      In        /testbench_lad/tb_lad/tc1                1
clk     0         In        /testbench_lad/tb_lad/tc2                0
                            /testbench_lad/tb_lad/tc3                0
                            /testbench_lad/tb_lad/cnt                1000
                            /testbench_lad/tb_lad/cntt               211385
                            /testbench_lad/tb_lad/vcheck             ProbAttack
                            /testbench_lad/tb_lad/sk1                0001
                            /testbench_lad/tb_lad/sk2                0000
                            /testbench_lad/tb_lad/sk3                0000
                            /testbench_lad/tb_lad/skk                0001
                            /testbench_lad/tb_lad/attackgen/clk      1
                            /testbench_lad/tb_lad/attackgen/clr      0
                                                               Now   85690000 ps
                                                          Cursor 1   33038595 ps
```

Figure 7. Probing Attack Detection

Figure 7 shows the Probing attack detection result coming from Model sim 6.3 waveform window. Probing attack detection after the process of Pattern search algorithm is highlighted with string named “Prob Attack” with the variable name v check.



```
Objects                     wave - default
Name    Value     M                                                       Messages
clr     0         In        /testbench_lad/tb_lad/ys                      1011011
a       0         In        /testbench_lad/tb_lad/ysk                     0010011
b       0         In        /testbench_lad/tb_lad/sg                      1
c       1         In        /testbench_lad/tb_lad/tc1                     0
d       1         In        /testbench_lad/tb_lad/tc2                     1
y       UUUUUUU   In        /testbench_lad/tb_lad/tc3                     0
config  0001      In        /testbench_lad/tb_lad/cnt                     0011
clk     0         In        /testbench_lad/tb_lad/cntt                    0
                            /testbench_lad/tb_lad/vcheck                  CorrAttack
                            /testbench_lad/tb_lad/sk1                     0000
                            /testbench_lad/tb_lad/sk2                     0010
                            /testbench_lad/tb_lad/sk3                     0000
                            /testbench_lad/tb_lad/skk                     0010
                            /testbench_lad/tb_lad/attackgen/clk           0
                            /testbench_lad/tb_lad/attackgen/clr           0
                            /testbench_lad/tb_lad/attackgen/config        0010
                            /testbench_lad/tb_lad/attackgen/attack_insert 1011010
                            /testbench_lad/tb_lad/attackgen/attack_insert1 0000000
                                                                    Now   246775750 ps
                                                               Cursor 1   235286017 ps
BENCHMARK_CKT2.VHD | Attack_gen.vhd | Clock_Synth.vhd | TestBench_LAD.vhd | Logic_Attack_Detector.vhd | wave
```

Figure 8 Data Insertion Attack Detection

Figure 8 shows the data insertion attack detection or normally called as FPGA corruption attack. The simulation results above clearly depicts the detection of corruption attack.

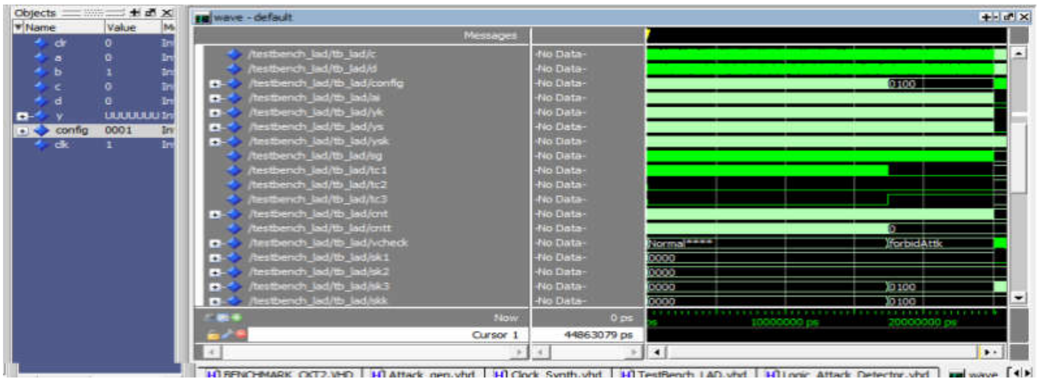


Figure 9. SOC Replay Attack Detection

Fig 9 shows the SOC replay attack detection using pattern search algorithm which is highlighted with string “forbid attack”, since SOC replay attack is also represented as forbidden attack since the replay attackdoes not depend on any of the stable signals and it occurs in unexpected sequences.

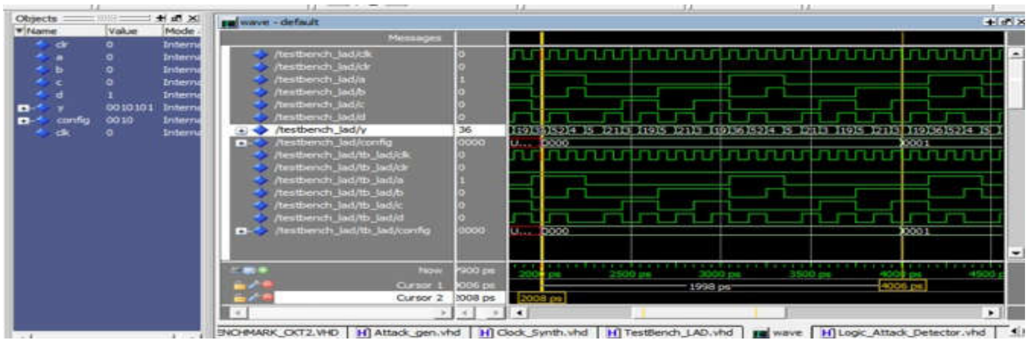


Figure 10 Pattern of Input Sequence

Figure 10 shows the test pattern of input sequence fetched and displayed as a single frame. The same pattern isrepeated at every test. The test pattern is reconfigured for various test data. The outcome of the pattern searchalgorithm validates each data in the pattern frame with respect to the edge triggered stable operating clock.

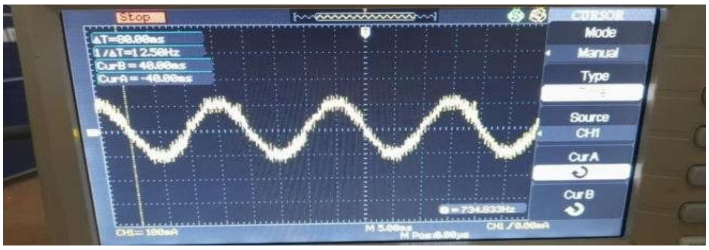


Figure 11. Output Waveform of Attack Detection

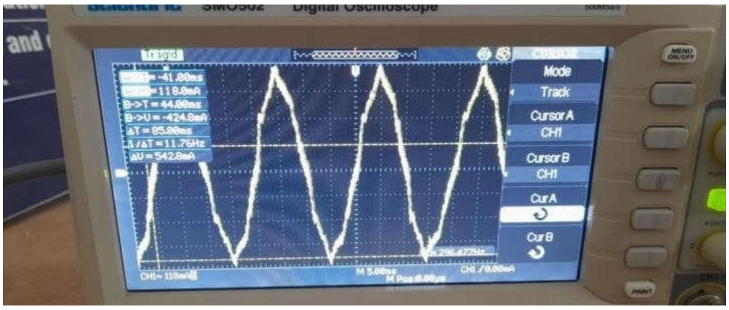


Figure 12. Output Waveform

The output waveform of the clock pulse signal that was generated was fetched with the attack and passed to the test circuit to identify the type of attack. This has been displayed in Figures 11 and 12.

5. Conclusion

Hardware intellectual property (IP) piracy and misuse have introduced new challenges in the semiconductor industry as untrusted parties in the IP's life cycle may clone, reverse-engineer, or extract important design secrets from an IP. A promising solution to protect a hardware IP against such attacks is toperform logic locking, where additional logic controlled by a secret key is inserted in strategic locations of an IP to lock the functionality when the correct key is not available Attack detection in logical circuits is analyzed. Insertion of logical attacks such as Probing, Reverse engineering or corruption attack andForbidden FPGA replay attack inputs are fetched. Dynamic logic attack detector is evaluated and used to detect the attacks with configurable logics. The attack is oracle-less, as it does not require any golden input–output pairs, as well as any knowledge about the input design or the locking algorithm. The attack analysesthe locked circuit using two clustering modes, which vary in attack complexity and effectiveness, with adjustable margin values that can control the level of confidence of the extracted keys.

References

[1] M. Yasin and O. Sinanoglu,(2017) "Evolution of logic locking," IFIP/IEEE International Conference on Very Large-Scale Integration (VLSI-SOC), vol. 27, pp. 1-6, doi: 10.1109/VLSI-SoC.2017.8203496.

[2] M. Rostami, F. Koushanfar and R. Karri, (2014) "A Primer on Hardware Security: Models, Methods, and Metrics," in Proceedings of the IEEE, vol. 102, no. 8, pp. 1283-1295, Aug. 2014,doi: 10.1109/JPROC.2014.2335155.

[3] T. Thangam, G. Gayathri and T. Madhubala, (2017) "A novel logic locking technique for hardware security," 2017 IEEE International Conference on Electrical, Instrumentation andCommunication Engineering (ICEICE), 2017, vol. 101, pp. 1-7, doi: 10.1109/ICEICE.2017.8192439.

[4] H. -Y. Chiang, Y. -C. Chen, D. -X. Ji, X. -M. Yang, C. -C. Lin and C. -Y. Wang, (2020) "LOOPLock: Logic Optimization-Based Cyclic Logic Locking," in IEEE Transactions on Computer- Aided Design of Integrated Circuits and Systems, vol. 39, no. 10, pp. 2178-2191, doi: 10.1109/TCAD.2019.2960351.
[5] Sirone and P. Subramanyan, (2019)"Functional Analysis Attacks on Logic Locking," Design,Automation & Test in Europe Conference & Exhibition (DATE), vol. 36, pp. 936-939, doi: 10.23919/DATE.2019.8715163.

[6] Rahman, M.T., Rahman, M.S., Wang, H., Tajik, S., Khalil, W., Farahmandi, F., Forte, D., Asadizanjani, N., & Tehranipoor, M.M. (2020). Defense-in- Depth: A Recipe for Logic Lockingto Prevail. ArXiv, abs/1907.08863.vol. 42, pp. 3-17.

[7] Jain, A., Zhou, Z. and Guin, U., (2021). TAAL: Tampering Attack on Any Key-based LogicLocked Circuits. ACM journal vol. 26, pp.1-22.

[8] X. Xu, B. Shakya, M. M. Tehranipoor, and D. Forte, (2017) "Novel bypass attack and BDD-basedtradeoff analysis against all known logic locking attacks," in Proc. Int. Conf. Cryptograph. Hardwware Embedded Syst. Cham, Switzerland: Springer,vol. 130, pp. 189–210.

[9] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, (2019) "Full-lock: Hard distributions of SAT instances for

obfuscating circuits using fully configurable logic and routing blocks," in Proc.56th Annu. Design Automat. Conf., vol. 54, pp. 1–6.

[10] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, (2017) "Cyclic obfuscation for creating SAT-unresolvable circuits," in Proc. Great Lakes Symp. VLSI,vol. 58, pp. 173–178.