International Journal of Computer Applications Technology and Research
Volume 2– Issue 6, 738 - 742, 2013

Fault Injection Test Bed for Clock Violation

E. Kavitha
Dept. of ECE
MRE College
Secunderabad, AP, India

P.S. Indrani
Dept. of ECE
MRE College
Secunderabad, AP, India

M. J. C. Prasad
Dept. of ECE
MRE College
Secunderabad, AP, India

Abstract: In this paper, the International Data Encryption (IDEA) algorithm synthesis models will be used as test encryption algorithm. The Xilinx Digital clock manager component will be used for generation of clocks for different frequencies and phase shifts. The encryption output with faults introduced and without faults introduced is compared as a function of ratio of used clock frequency and maximum frequency of operation reported by synthesis tool. The clock generation, clock switching, interface adopter to IDEA core and UART interface will be realized and tested in FPGA hardware in integrated form. FPGA based test bed is realized for injecting faults through clock glitches, to result in setup and hold violations. The UART interface is realized on FPGA to provide PC based controlling for this fault injection. Xilinx chip scope tools will be used for verifying the output at various levels in FPGA hardware.

Keywords: International Data Encryption algorithm (IDEA), UART, FPGA, Digital clock Manager (DCM), PLL.

1. INTRODUCTION

To increase performance, a lot of cryptographic algorithms are implemented in hardware for that purpose FPGAs are frequently used for this purpose. Such implementations are however prone to various types of attacks intended to compromise their security. One possible way to perform such an attack is to inject transient faults affecting the normal circuit operation. One of the cryptographic algorithms most commonly used is the International Data Encryption algorithm (IDEA) is a block cipher [2]. The mentioned algorithm works on 64-bit plain text and cipher text block (at one time). For encryption, the 64-bit plain text is divided into four 16-bits sub-blocks. I denote these four blocks as X1 (16 bits), X2 (16 bits), X3 (16 bits) and X4 (16 bits). Each of these blocks will perform of operation 8 ROUNDS and one OUTPUT TRANSFORMATION phase.

A long term objective of our research is to develop an efficient method for protecting FPGA-based implementations of cryptographic algorithms through effective concurrent testing of various types of faults, including faults injected by the attackers [3]. An essential part of this research is to develop a method and tool for the evaluation of susceptibility of FPGA based circuits to fault injection attacks. In this paper, I present such a method and tool. It allows us to examine an FPGA-based circuit, in particular an implementation of a cryptographic algorithm, subjected to a fault injection attack based on clock glitching [7].

2. EXPERIMENTAL SETUP

The circuit under test (CUT) and the tester are both implemented on a low cost FPGA Spartan 3E development board. I used VHDL for defining custom components and Xilinx chip scope which is the system on-chip building tool, for creating standard library components and connections.

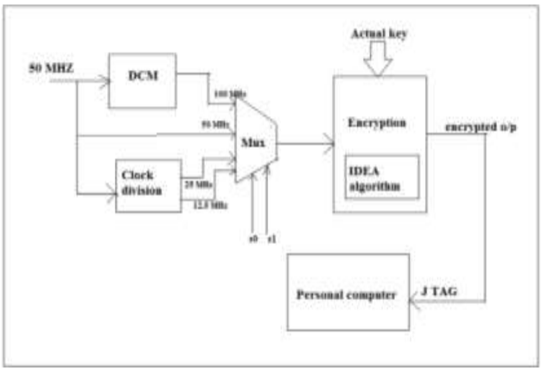


Figure 1: Experimental setup

A simplified diagram of experimental setup implemented in the device is shown in above Fig 1.

2.1 Analysis of IDEA

In IDEA algorithm, I taken input text of size 64 bits at a time and divide it in evenly; i.e., 64 bit plain text is divided into 4 sub-blocks, each of 16 bits in size [1]. The basic operations needed in the entire process for 8 rounds are

1. Multiplication modulo 2^**16** +1.
2. Addition modulo 2^**16**.
3. Bitwise XOR.

And, operations needed in the OUTPUT TRANSFORMATION phase –

1. Multiplication module 2^**16** +1.
2. Addition modulo 2^**16**.

All the above mentioned operations are performed on 16 bit sub-blocks. For simplicity of expressing the operations. Now, let us take a look on the key generation for the encryption process while using the 25-bit circular left shift operation on the original key, it produce other subsequent sub-keys, used in different rounds [2]. For instance, among the total no. of 52 keys- Sub-key Z1 is having first 16bits of the original key, sub-key Z2 is having the next 16 bits, and so on till sub-key Z6; i.e., for ROUND1, sub-keys Z1 to Z6 use first 16x6= 96 bits of the original cipher key. In the ROUND2, sub-key Z7 &Z8 take the rest of the bits (bits 97 to 128) of the original cipher key. Then we perform circular left shift (by 25bits) operation on the original key.

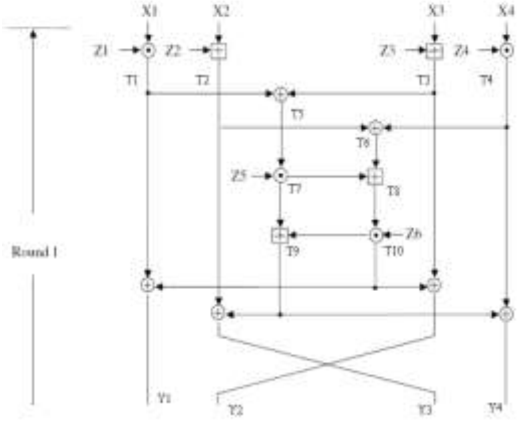


Figure 2: IDEA Encryption/Decryption sub key generation Architecture

As a result the 26th bit of the original key shifted to the first position and becomes the first bit (of the new shifted key) and the 25th bit of the original key is moves to the last position and becomes the 128th bit (after first shift). This process continues till ROUND8, and also in the OUTPUTPUT TRANSFORMATION phase; i.e., after the ROUND8, the key is again shifted left by 25 bits and the first 64 bits of the shifted key is taken for use, and used as sub-keys Z49 to Z52 in the OUTPUT TRANSFORMATIONMATION phase [2].

2.1.1 Output transformation stage

The final round of IDEA algorithm is also called output transformation stage. It only uses 4 sub-keys. The block diagram of final round is given below. The VHDL code for IDEA final round module is given.

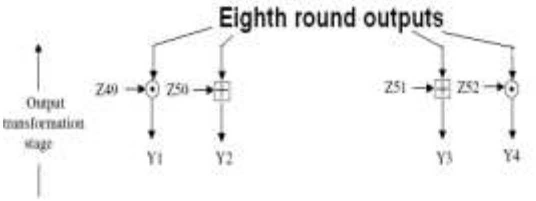


Figure 3: Block diagram of output transformation stage

The general IDEA architecture uses eight rounds with total 48 keys and final output transformation round with 4 sub-keys is implemented in VHDL using structural modeling style. The IDEA Decryption module also uses the same hardware, but the Decryption sub-keys are different. The encryption followed by decryption module is used for testing the complete IDEA algorithm with the following input.

Main Key: Z = (5a14 fb3e 021c 79e0 6081 46a0 117b ff03)
64-bit plaintext: X = (X1, X2, X3, X4) = (7fa9, 1c3, ffb3, df05)

The same text is used in simulating the other IDEA architectures.

2.2. Fault injection

The basic idea of our implementation of the fault injection based on clock glitching is to switch from a normal operation clock to a faster clock; so that one clock cycle is slightly shorter than CUT can handle [6].This idea is depicted in Figure 4. In order to generate single faults, the frequency of the faster clock has to be adjusted very precisely, more accurately than can be achieved using an on-chip PLL circuitry for clock generation or phase shifting. An external clock signal generated by Tektronix AWG 5002B Arbitrary Waveform Generator is used instead. The external clock is going to feeds an internal PLL circuitry where it is divided by 4 to produce the slower clock. It also passes through unchanged to produce the faster (high speed) clock.

To switch clocks, I using a Clock Control Block, the dedicated clock management built-in component available in the device. The result of the operations on clock signals is shown in Figure 4, as “output clock”. The last trace in Figure 4 is the real output clock registered by the 1 GHz Tektronix oscilloscope. The faster clock frequency is 150 MHz and it can be noticed that the signal is not distorted too much [7]. Moreover, additional measurements, made by the MXG-9810AVolcraft frequency counter show that the faster clock has the same frequency (with accuracy of 1 Hz) as the clock supplied to the FPGA by the waveform generator.

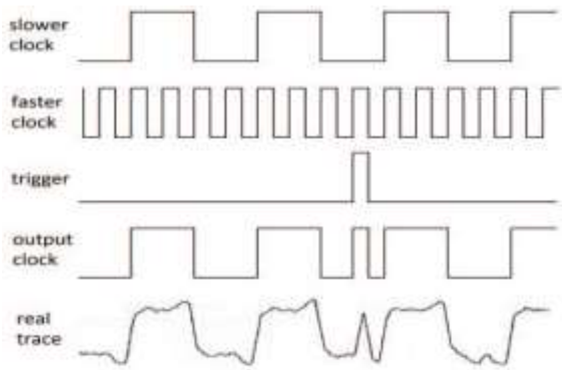


Figure 4: Clock glitch generation

3. TESTING PROCEDURE

Steps involved in project work

To minimize the test application time, the IDEA circuit is performed using the FPGA Spartan 3E, by running a dedicated application. Only the final results are sent through the host computer for display. Implementation of IDEA algorithm in VHDL coding. Implementation of sub key generation module from 128 bit key in VHDL and the basic Arithmetic and logic blocks in VHDL. The VHDL structural modeling of 9 rounds for IDEA encryption module is programmed.

VHDL structural modeling of 9 rounds for IDEA decryption module. Writing test bindles for individual components and also for top level modules and Simulation using modelsim. Verifying the modelsim outputs with expected results. Synthesizing the developed IDEA modules on Xilinx Spartan 3E FPGA using Xilinx ISE tool. Downloading the IDEA encryption and Decryption modules on Spartan 3E development board using the IMPACT tool. The Faults will be introduced and corresponding output results are observed and the Clock generator is going generate the clock and oscilloscope is going to generate the different clock frequencies [6-7]. The FPGA Spartan 3E kit is connected to host computer as shown in below figure.

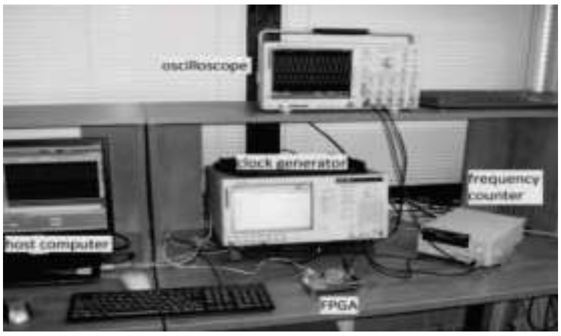


Figure 5: Measurement setup

4. OBSERVATIONS AND RESULTS

4.1. Chip scope resullts

IDEA algorithm has a maximum freqency of approximately 19MHZ frequency. So the clock freqency applied should be less than this freqency or approximately around this frequency. If we apply the freqency more than this freqency then the metastability condition takes place and the output will be corrupted depending on the applied input clock freqencies. The chipscope results of idea encryption algorithm with the corresponding different clock freqencies as shown in figure.

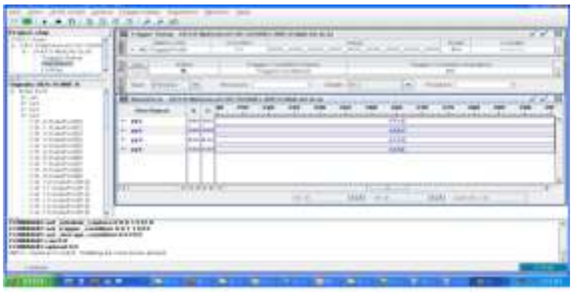


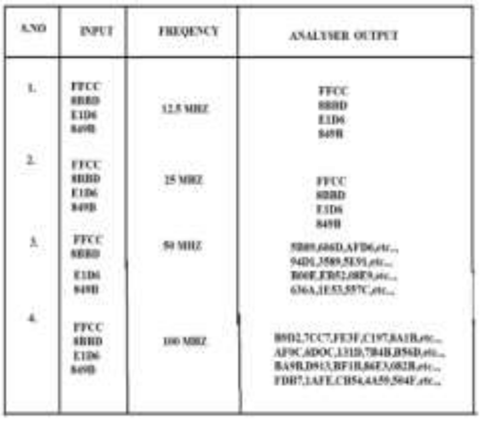
Figure 6: clock frequency of 12.5 MHZ frequency

The below chipscope results show how the idea encryption algorithm is behaving with the corresponding 100 MHZ clock frq.The chipscope results show how the idea encryption algorithm is behaving with the corresponding 50 HZ clock freqency.

Figure 7: clock frequency of 50 MHZ frequency

4.2. Observed output results

The analyzer outputs are observed for different clock frequencies (12.5 MHZ, 25 MHZ, 50 MHZ, and 100 MHZ). The input is FFCC, 8BBD, E1D6, 849B which is generated by input generator.



S.NO	INPUT	FREQENCY	ANALYSER OUTPUT
1.	FFCC 8BBD E1D6 849B	12.5 MHZ	FFCC 8BBD E1D6 849B
2.	FFCC 8BBD E1D6 849B	25 MHZ	FFCC 8BBD E1D6 849B
3.	FFCC 8BBD E1D6 849B	50 MHZ	5B89,686D,AFD6,etc.., 94D1,3589,5E91,etc.., B00F,EB52,08F9,etc.., 636A,1E53,557C,etc..,
4.	FFCC 8BBD E1D6 849B	100 MHZ	89D2,7CC7,FE3F,C197,8A1B,etc.., AF9C,ADOC,131D,7B4B,B56D,etc.., BA9B,D913,BF1B,86E3,082B,etc.., FDB7,1AFE,CB54,4A59,504F,etc..,

Table 1: output comparison table

5. CONCLUSION

The presented method and tool for injecting faults in an FPGA Spartan 3E circuit, based on Clock glitching and it has some unique features that allow us to thoroughly examine and analyze the impact of such faults on the operation of the circuit. The IDEA (International Data Encryption Algorithm) is a strong block-cipher text.

Though there are many arithmetic operations involved in the entire algorithm, only three different of operations are involved (as mentioned above). As the cipher key size is 128bits, in that respect IDEA is too strong (having taken care for weak keys).In particular, through recise adjustment of the frequency of an external clock Generator; we can control the number of faults occurring at the output of the circuit under Test.

6. FUTURE SCOPE

The presented solution is intended for injecting a single fault (single clock glitch) during an encryption operation. For more complex circuits the more complex experimental setup can be implemented, to allow dynamic configuration of fault injection conditions .In particular, the trigger unit could be redesigned and interfaced with the Avalon bus, so that it could be reconfigured by the software processor, depending on the testing scenario.

In order to decrease the testing time, the NIOS II processor can be clocked by an independent clock freq as faster than the external clock freq is divided by 4 and the solution requires some changes to the interface adaptor unit to account for a difference in clock frequencies for the NIOS II processor and the circuit under test. The proposed approach can be used not only in the case when the CUT is implemented in the same FPGA; Although it appears that only small changes need to be done to our experimental setup, no attempt has been made to verify this idea. It makes the algorithm more secure and less susceptible to cryptanalysis.

7. Acknowledgment

I E. Kavitha would like to thank P. S. Indrani Associate professor, who guided me through out to complete my work successfully. I would like to thank my HOD (ECE Dept.) Dr. M. J. C. Prasad for providing us constant support and providing us the resources needed.

8. REFERENCE

[1]. Blum M. and Gold wasser S., “An efficient probabilistic public-key encryption scheme which hides all partial information,” Advances in Cryptology-CRYPTO’84, Lecture notes in computer science (Springer-Verlag), pp.289-299, (1995).

[2]. Biryukov, Alex; Nakahara, Jorge Jr.; Preneel, Bart; Vandewalle, Joos, "New Weak-Key Classes of IDEA", Information and Communications Security, 4th International Conference, ICICS 2002.

[3]. William Stalling “Cryptography and Network Security”.

[4]. Bruce Schiener “Applied Cryptography “.

[5] Chang H.S., “International Data Encryption Algorithm” CS-627-1 Fall, 2004.

[6] K. Bousselam, G. Di Natale, M-L.Flottes, B. Rouzeyre, "Evaluation of concurrent error detection techniques on the Advanced EncryptionStandard", Proc. 16th IEEE On-Line Testing Symposium, 2010.

[7] J. Balasch, B. Gierlichs and I. Verbauwhede, "An In-depth and Black-box Characterization of the Effects of Clock Glitches on 8-bit MCUs",Proc. Workshop on Fault Diagnosis and Tolerance in Cryptography,2011.