Introduction:

Like software Trojans, a hardware Trojan is a malicious modifications made to a piece of hardware to produce unintended functionalities. The objective of this lab is to create hardware Trojans during the design phase in the HDL code. It is also assumed that the attacker has the ability to bypass detection mechanisms before the code is implemented in an IC. The other goal of this lab is to study the behavior of these Trojans.

A Trojan can be classified based on its insertion phase, abstraction level, activation mechanism, effect, and location. Some of these terms will be used to study the behavior of the Trojans.

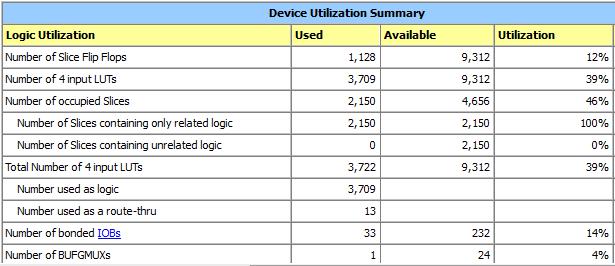
Hardware Description:

Alpha is a DES crypto-core implemented using on an FPGA board. Alpha will display encrypted messages on the 7-segment LED display. During normal operation, a message is first selected to be encrypted. Then the encryption process is triggered by pressing the start button. After the encryption is complete, the resulting ciphertext is displayed on the 7-segment display. Since the result is 16 bytes long and 7-segment display can only display 4 bytes at a time, the select button can be used to select the portion of the ciphertext to be displayed. The device will contain a predefined private key. To transmit a message, a soldier must do the following:

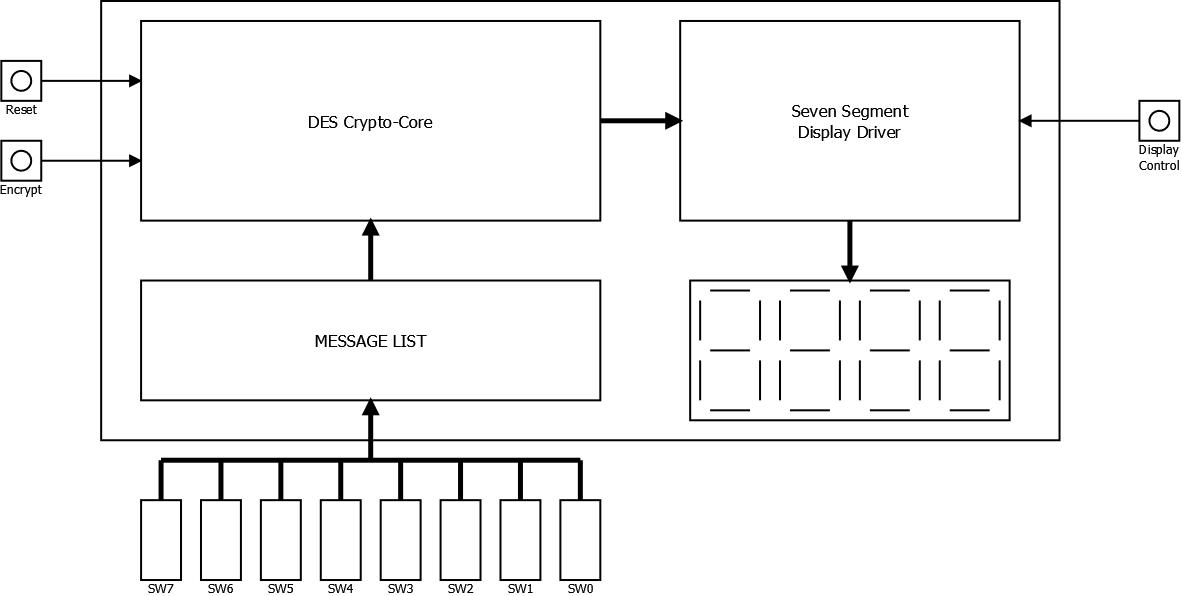
1. The device contains up to 256 predefined ‘messages’ (input plaintext), which is selected using the slide switches.
2. Press the "Start Encryption" button to encrypt. When this key is pressed the encryption engine encrypts the plaintext and displays the ciphertext into the 7-segment LED.

This is the target for the Trojan injections within this lab.

Here are the resources that are used by the unmodified design.



Block Diagram for the Design:

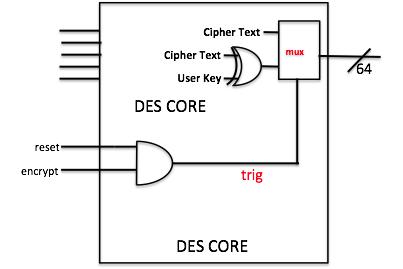


Hardware Trojan Descriptions:

Trojan 1:

This Trojan takes advantage of the undefined functionality of certain input combination. As described in the specification, the operation modes does not cover all the input patterns of the input buttons. Therefore, it is possible to exploit one of the unused input combination and use it as a trigger to achieve our goal. More specifically, our Trojan uses input combination of reset and encryption signals both being activated as our trigger. When this input combination is detected, the output will be xor-ed with the user key. The key can then be calculated by xor the normal output with the xor-ed output.

Block Diagram:



Trigger:

Reset and encrypt both being ‘1’.

Payload:

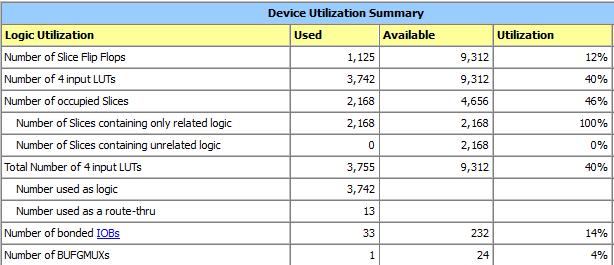
The key is xor-ed with the user key.

Probability of false trigger:

Attacker has control over when to trigger this Trojan. The trigger is not defined by the specification and is unlikely to be tested.

Overhead (Area) :

Here is the resource used by the design.



The overhead is (3742-3709)/3709=0.89%

Active Regions (Normal Operation & Attack):

The xor module and the mux will be active during an attack. The combination block to detect the trigger will always be active.

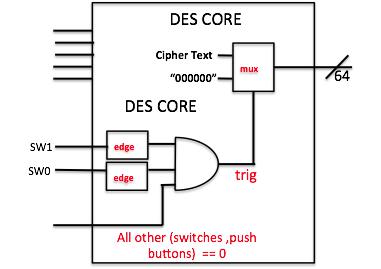
Type of Trojan:

Leak Information.

Trojan 2:

Similar to Trojan 1, this Trojan takes advantage of the undefined input pattern. Unlike Trojan 1, this Trojan is triggered on the falling edges of switch 0 and switch 1. Once this is detected, the Trojan will lock the 7-segment display.

Block Diagram:



Trigger:

Falling edges of switch 0 and switch 1. If any other button or switch is used, the trigger will be reset.

Payload:

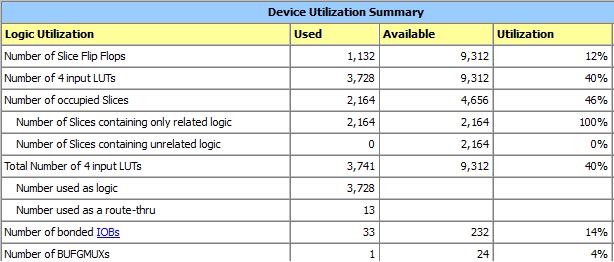
Lock down of 7-segment display.

Probability of false trigger:

The trigger is only activated with switch 0 and switch having a falling edge. Since during test, all the signal are expected to change simultaneously, the false trigger is unlikely to occur.

Overhead (Area) :

Here is the resource used by the system.



The overhead is (3728-3709)/3709=0.51%

Active Regions (Normal Operation & Attack):

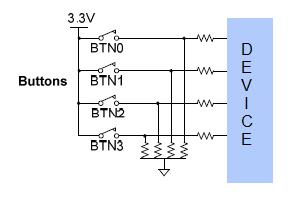
The detection block which detects the trigger is always on. The circuit which blocks the 7-segment display is only on when trigger is activated.

Type of Trojan:

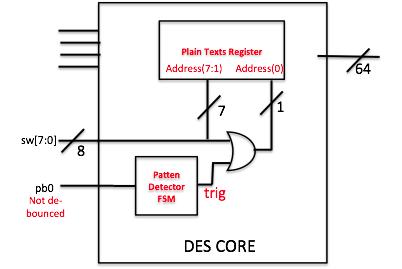
Denial of Service.

Trojan 3:

Trojan 3 exploits the uncertainty in the input device. The pushbutton used in the board has bouncing effects, meaning that when the button is pushed, the signal generated is not just a simple rising edge. On the contrary, the voltage level will jump between ‘0’ and ‘1’ before the signal stabilize at the desired level. When such an event happens, a random bit pattern generated from this uncertainty will be added to the text, thus changes the functionality of the design.



Block Diagram:



Trigger:

The Trojan is triggered by the bouncing effect of the push button 0: byte select.

Payload:

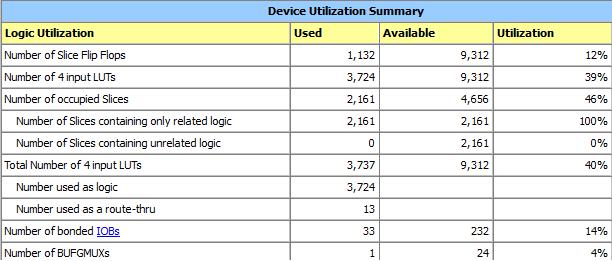
The goal is to randomize the text to be encrypted thus resulting in incorrect output.

Probability of false trigger:

Since bouncing is only a part of the device, simulation will not trigger this Trojan. On the other hand, the on-board test has a low probabilityto trigger suchTrojans depending onthe quality of the button.

Overhead (Area) :

Here is the resource used by the device.



The overhead is: (3724-3709)/3709=0.40%.

Active Regions (Normal Operation & Attack):

The trigger detection region is always active. The corruption of the plain-text is only active when triggered,

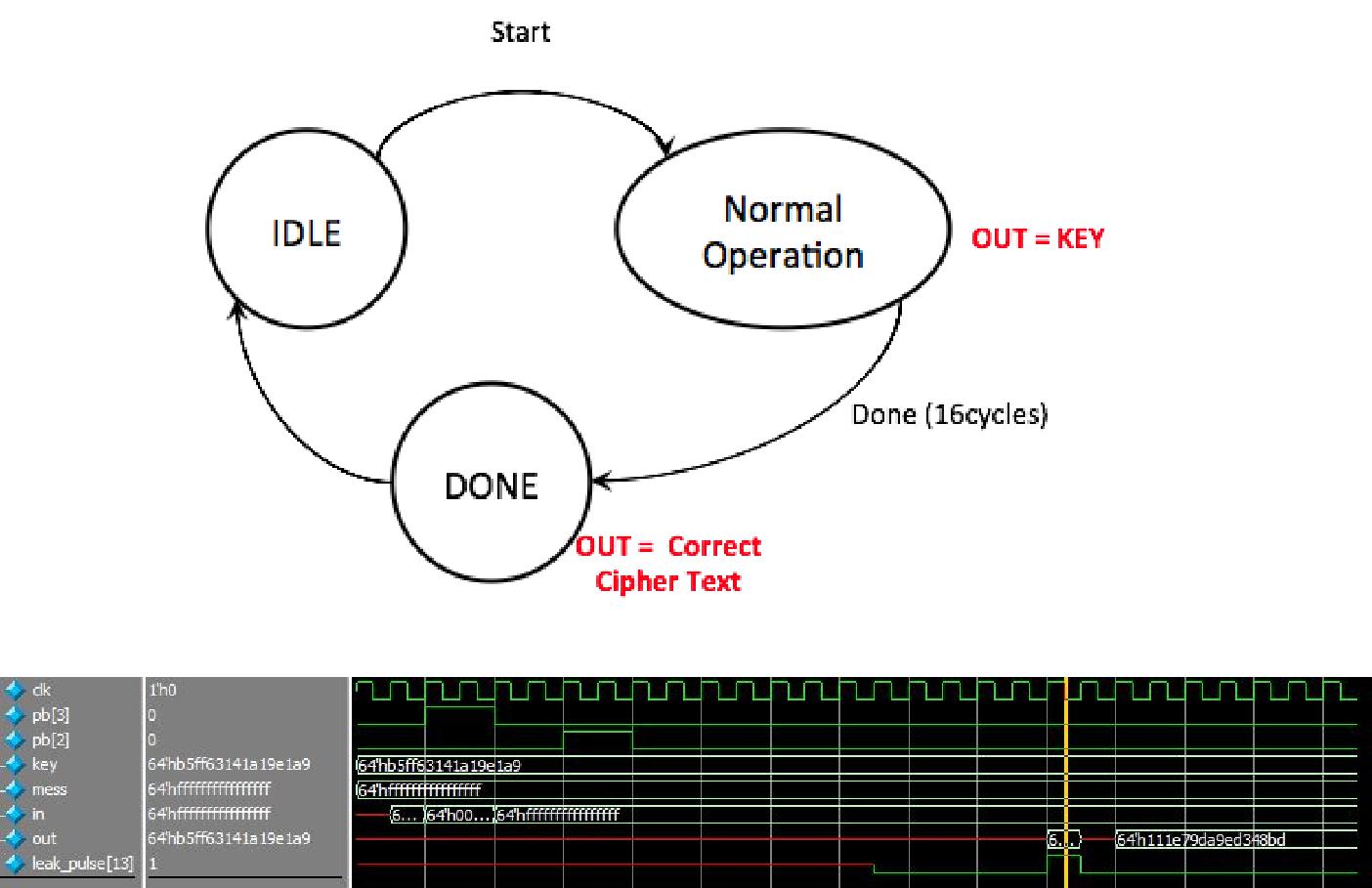
Type of Trojan:

Change of Functionality.

Trojan 4:

Trojan 4 takes advantage of the unused time where encryption is not complete. It displayed the user key at the time.

Block Diagram:



It can be seen that the output on the 13th encryption cycle is equal to the key value.

Trigger:

There is no trigger this Trojan is always on

Payload:

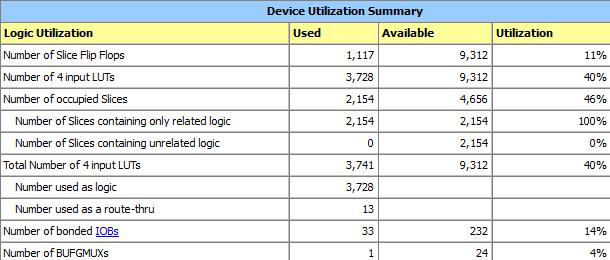
The Trojan will display the secret key during round 13 of the data encryption of the key generation for a single clock cycle.

Probability of false trigger:

The Trojan is always active.

Overhead (Area):

Here is the resource used by the system.



The overhead is: (3728-3709)/3709=0.78%.

Active Regions (Normal Operation & Attack):

The Trojan is always active.

Type of Trojan:

Leak Information that is the secret user key in our case.

Conclusion:

Four different Trojans were created in this lab. Those Trojans achieved different purposes and bypass the testing system using different techniques. The target functionality for the Trojans was to Leak Information (Trojan 1 & 4), Change Functionality (Trojan 3), and Deny Service to the User (Trojan 2). To perform these functions the following properties were exploited:

* Trojan 1 - Operations not Defined by the Design Specification but still valid input
* Trojan 2 - Sequential Operation of the device when properly being used
* Trojan 3 - A random event caused by the physical devices used in the final product
* Trojan 4 - Properties of the Encryption Algorithm in leaking the key when the output is not supposed to be valid.

All of the implemented Trojans add less than 1% of the amount of logic to that of the original design. This means that the amount of power and cost difference should be neglectable to the entity creating the chips for production.