Title of the Experiment: Hardware Obfuscation
Group Members' Names: Sujan Kumar Saha, Pankaj Bhowmik
Date: 06/12/2019

Abstract

Hardware obfuscation is a method to hide the logic in a circuit design. By definition, obfuscation is the technique of obscuring or hiding the true meaning of a message or the functionality of a product, in order to protect the intellectual property inherent in the product. The objective of hardware obfuscation is to have a functionally equivalent but structurally different design. The design is modified in a way that it implements different logic functions and it is not possible to retrieve the correct logic equation by reverse engineering. A locking mechanism must be incorporated which ensures the design becomes functionally equivalent upon the correct unlocking process. This experiment aims to achieve these goals for a sequential and combinational circuit.

Experiment Details

Goals: The goal of this experiment is to understand the hardware obfuscation and apply this method on a benchmark combinational circuit and sequential hardware obfuscation on a state machine.

Experimental Setup: We used the MAX10 FPGA on Hardware Hacking (HaHa) board, USB-blaster, Quartus Prime Software.

Experiment Steps:

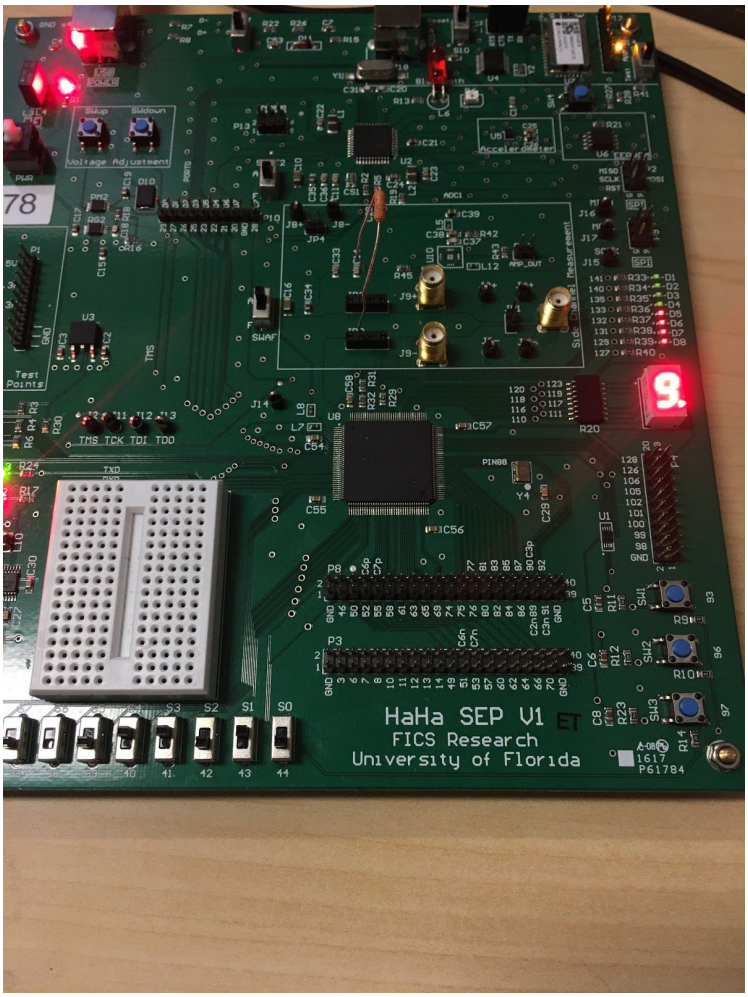
1. In the first part of the experiment, we download the given sof file for group 5. After loading the binary into the HAHA board, we extract the key by different combinations of SW1 through SW5.
2. Combinational hardware obfuscation is implemented on the given benchmark code. We simply apply XOR and not gate to obfuscate the output. We test the implementation using a testbench. The testbench includes the original benchmark code and the obfuscated combinational Verilog. Both results will be the same only if the key that is assigned to group 5 is applied to the keyinput of the second Verilog.

3. Finally, an experiment is performed on the sequential circuit. A sequential circuit is a state machine and we apply another state machine as a key. If anybody has the key then he can unlock the system. Otherwise, enter to the unreachable state.

Results and Analysis:

Part I:

We load the given sof file and manually change the switches sw5-sw1 to obtain the



unlocking key. The key we obtain is 01101. We present the image as proof of this statement. At the initial state, there was no signal on the seven segment display. After that, we apply different key values by altering the switches. To observe each pattern it took us 10s approximately to observe each output. Hence, if the key is 256 bits instead of 5 bit, then it should take 1.1579209e+77x10s or 3.6717431e+70 years.

Part2:

For this part, we attached the obfuscated Verilog file with Testbench. The following figure represents shows the simulation result of *I,Ow,Keywrong* and *I,Or,Keyright*. In the test bench file, we perform XOR of the outputs of the obfuscated result and the benchmark result. After performing the XOR, we will get zero output when the two results are the same. In the testbench, we put the original key 16'b1011111010100100 and then the key values are random. From this figure, we can see that only the first block gives us 0 and some values after that. That means, in the rest of the blocks, the obfuscated outputs are wrong. Hence, our properties are properly justified.

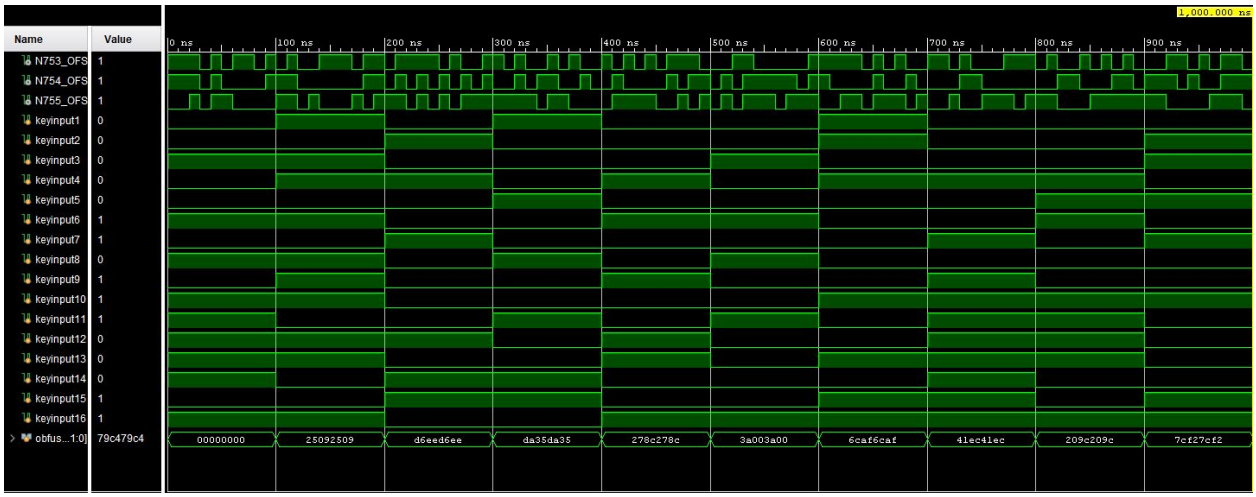
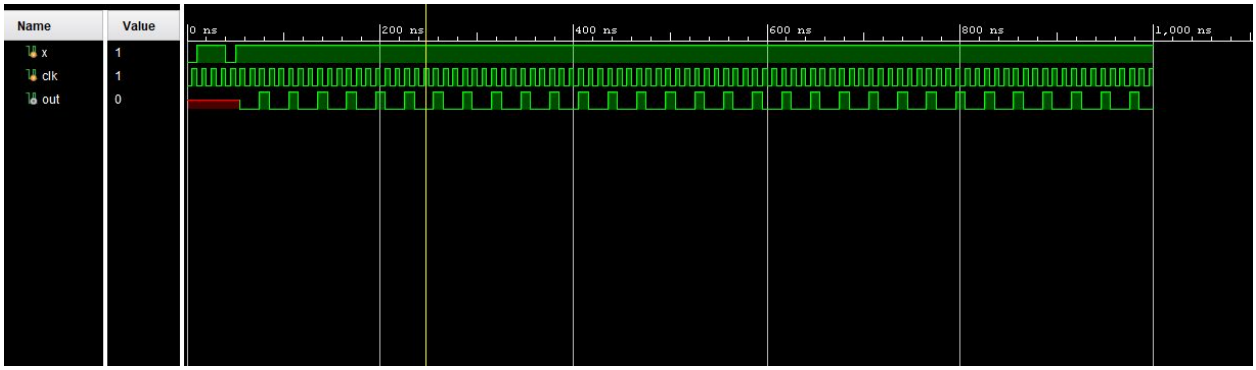


Figure: Testbench result for part 2.

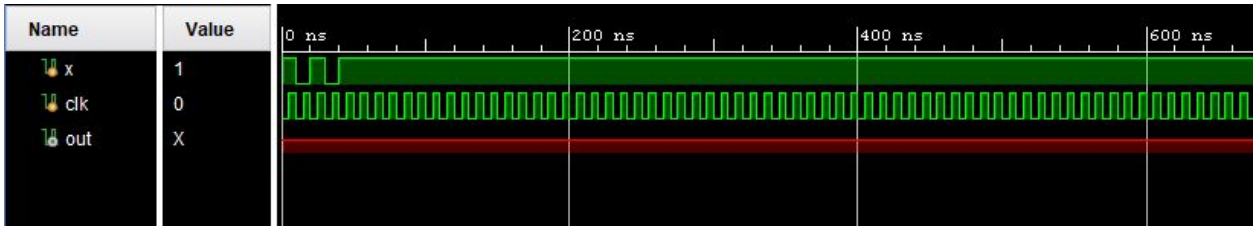
Part 3:

In the part, we implement a circuit with sequential obfuscation. Here we initiate five more states and if anyone has the right key which is 5'b01110, will get into the original FSM and finally generate output 1. Without having the original key, they will reach into the black hole state. The Verilog file and the test bench are attached to the report. We provide the screenshot with the original key is given below.

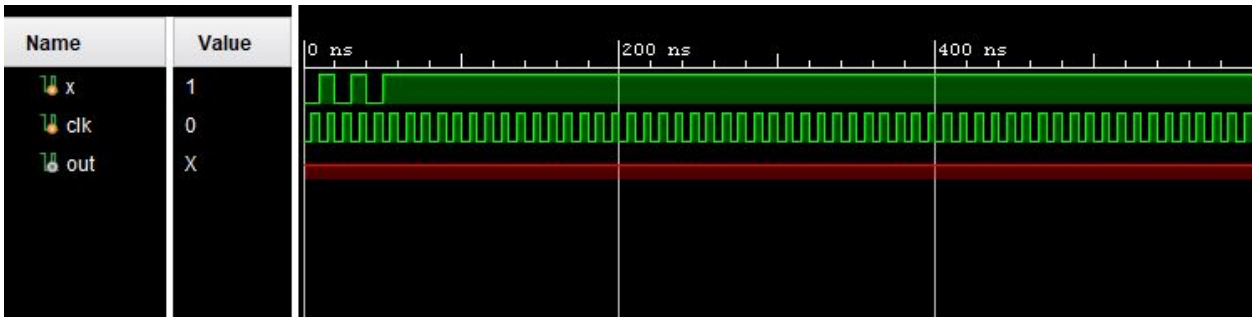


As we can see, we get output 1 after applying the write key.

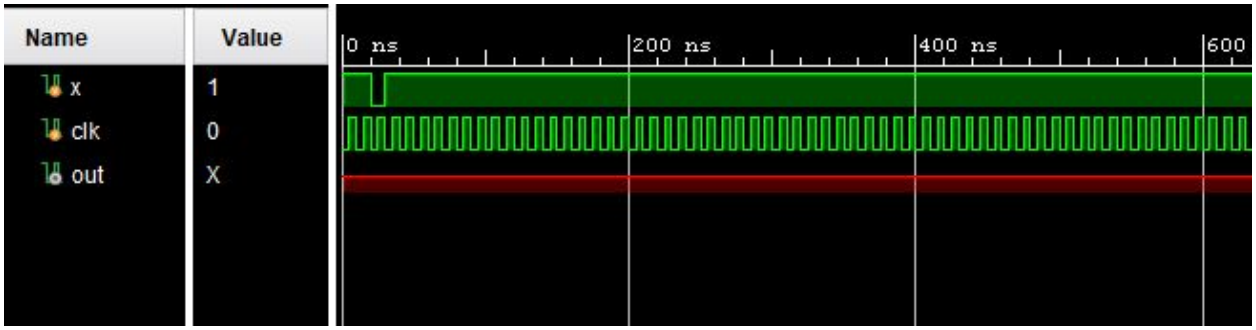
To validate our model, we apply three random key values from the figures below, we observe that they remain in always in a black hole and can not reach to the sequential block.



Key=10101



Key = 01010



Key = 11011

Figure: Blackhole example with wrong keys.

Summary

This experiment gives a hand-on experience of hardware obfuscation. We learned about both combinational and sequential hardware obfuscation. Hence, we can easily secure a circuit with a key using this method.