INVESTIGATION OF DPA RESISTANCE OF BLOCK RAMS IN FPGAS

by

Shaunak Shah
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Computer Engineering

Spring Semester 2010
George Mason University
Fairfax, VA

Investigation of DPA Resistance of Block RAMs in FPGAs

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Shaunak Shah
Bachelor of Engineering
University of Mumbai, 2006

Director: Dr. Jens-Peter Kaps, Professor
Department of Electrical and Computer Engineering

Spring Semester 2010
George Mason University
Fairfax, VA

Dedication

I dedicate this thesis to my parents Sunil and Grishma Shah, and to my loving and caring sister Krishna Shah.

Acknowledgments

In an endeavor to successfully complete this thesis, I received inspiration, guidance and support from many people. I take opportunity to thank all those who helped me directly or indirectly along the way to achieve this success.

First and foremost I would like to thank my advisor Dr. Jens-Peter Kaps; without his knowledge, support, time, effort and strict deadlines, it would had been impossible to complete my thesis.

I would like to thank Dr. David Hwang for sharing his knowledge, ideas and tricks with me; and Dr. Kris Gaj for his valuable inputs and suggestions towards my thesis.

I would also like to thank all the Cryptographic Engineering Research Group (CERG) members specially Rajesh for making things easier; friends and family members for their support and believing in me.

Not to mention the great divine force and his blessings who helped me throughout in many known/unknown ways and situations.

Jai Shree Krishna!!!

Table of Contents

List of Tables

List of Figures

Abstract

INVESTIGATION OF DPA RESISTANCE OF BLOCK RAMS IN FPGAS
Shaunak Shah, M.S.
George Mason University, 2010
Thesis Director: Dr. Jens-Peter Kaps

Security at low cost is an important factor for cryptographic hardware implementations. Unfortunately, the security of cryptographic implementations is threatened by Side Channel Analysis (SCA). SCA attempts to discover the secret key of a device by exploiting implementation characteristics and bypassing the algorithm's mathematical security. Differential Power Analysis (DPA) is a type of SCA, which exploits the device's power consumption characteristics. Several countermeasures to DPA have been proposed, however, all of them increase security at the cost of increased area which in-turn leads to increased power consumption and reduced throughput.

FPGAs are popular due to their reconfigurability, lower development cost, off-the-shelf availability and shorter time to market. Block RAMs are large memories in FPGAs that are commonly used as ROM, FIFO, Look-up tables, etc. In this paper we explore the DPA resistance of Block RAMs and verify if their usage can improve the security of block ciphers such as the Advanced Encryption Standard (AES). We implemented a small test circuit comprised of elements from AES on Xilinx Spartan 3E FPGA and discovered that moving essential parts of AES from look-up tables (LUT) and distributed RAMs to Block RAMs yields about 26 times increase in DPA resistance without any increase in the area. On the

contrary it reduces the LUT based area consumption by a factor of 4 and increases speed 1.4 times. Subsequently the same techniques when applied to a standard S-Box and a T-Box implementation of AES showed similar results. The security increased about 9 times, slice area got reduced about 4 times and speed increased about 1.18 times.

Chapter 1: Introduction

Hardware implementations of cryptographic algorithms are preferred over software implementations because hardware implementations provide better speed and security [1]. Cryptographic devices are prone to various kind of attacks. One category of the attacks that pose a serious threat to modern cryptographic implementations is Side Channel Analysis (SCA). SCA is capable of bypassing the algorithmic and mathematical security leaving the implementation unsecured.

1.1 Side Channel Analysis

The term Cryptography can be briefly described as a science of hiding information. Along with it, there is another inevitable term called Cryptanalysis, which can be briefly described as a science of recovering the hidden information. The cryptographic devices are prone to various kind of attacks that try to recover the hidden information. Primarily, these attacks can be classified based on access/manipulation of the cryptographic device. They are Active attacks and Passive attacks.

- **Active Attacks** are the attacks where the attacker modifies the cryptosystem. In case of hardware implementation, the device is manipulated such that it behaves abnormally in an intended manner. This modification assist in revealing the required information.

- **Passive Attacks** are the attacks where the device is predominantly operated normally or mostly within its specification. The information is revealed by observing the working device and/or its physical properties. In other words, passive attacks resembles the nature of eavesdropping without touching or modifying the device.

Side channel analysis falls under the category of passive attacks. In SCA the hidden information is recovered by analyzing the characteristic data collected from the physical implementation of the cryptographic hardware. SCA involves observing or recording the information revealed by the cryptosystem, by exploiting its electrical characteristics such as power, current, radiations etc. Another type of classification of attacks is based on features exploited to reveal the information. They are Timing Analysis, Power Analysis and Electromagnetic Field Analysis.

- **Timing Analysis** [2] consists of an attack methodology where the time taken to perform different computations or multiple instances is measured. This data is then analyzed to recover the hidden information.
- **Power Analysis** [3] involves measuring the instantaneous power consumed by the device during its operation, followed by analyzing and processing the measurements to reveal the hidden information.
- **Electromagnetic Field Analysis** [4] consists of attacks where the electromagnetic field radiation leakages are recorded and that data is analyzed to reveal the hidden information.

In power analysis attack, the instantaneous power consumption of the cryptographic device is recorded. The attacker performs analysis on this data to recover the hidden information. In most cases the hidden information is a secret key used in a cryptographic algorithm implemented in hardware. In Power analysis, the attacker exploits the fact that the instantaneous power consumption depends on the operations being performed and the corresponding data being processed in the device. Power analysis is further classified into two types i.e. Simple Power Analysis (SPA) and Differential Power Analysis (DPA).

1.1.1 A Leakage Model in an FPGA

The total power consumption (P$_{tot}$) in a Complementary Metal Oxide Semiconductor (CMOS) gate is comprised of three basic dissipation sources. They are dynamic power consumption, short circuit power consumption and static power consumption (leakage power). The dynamic power consumption (P$_{dyn}$) is due to the charging and discharging of load capacitances. The power consumption due to short circuit (P$_{sc}$) is a result of the short circuit currents which exist for a very short duration of time, when both the transistors PMOS and NMOS are ON simultaneously. The static power consumption (P$_{stat}$) is due to the flow of current when there is no switching activity. This is the leakage current flowing through the reverse biased junctions of the transistors. Details are well explained in [5]. The total power consumption (P$_{tot}$) can be summarized by the Eq. (1.1).

$$\begin{gathered} P_{tot} = P_{dyn} + P_{sc} + P_{stat} \\ P_{tot} = C_L V_{DD}^2 P_{0\to 1} f + V_{DD} I_{peak} t_{sc} f + V_{DD} I_{leak} \end{gathered} \tag{1.1}$$

C$_L$ is the load capacitance, V$_{DD}$ is the voltage of power supply, $P_{0\to 1}$ is the probability of a clock event resulting in a $0 \to 1$ bit transition, f is the operating frequency, t$_{sc}$ is the time both transistors are conducting, I_{peak} is the peak current and I_{leak} is the leakage current.

Consider the CMOS inverter in Fig. 1.1, assume currently the input is unchanged HIGH. The NMOS (bottom transistor) is ON and PMOS (top transistor) is OFF and hence the output is LOW and remains low till the input is changed. Similarly if the input in unchanged LOW, the output remains HIGH. During these time there is a leakage current due to reverse biased transistors resulting in static power consumption (P$_{stat}$). Now, when the input changes from HIGH to LOW, the PMOS starts conducting and NMOS stops conducting. The load capacitance is charged from the power supply V$_{DD}$ and the output changes from LOW to HIGH. This results in dynamic power consumption. Similarly when the input changes from LOW to HIGH, the NMOS starts conducting and PMOS stops conducting

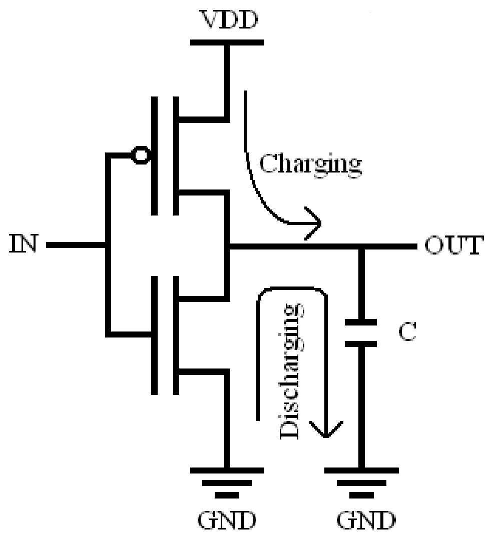


Figure 1.1: Power Consumption in a CMOS Inverter

and the load capacitance discharges again resulting in dynamic power consumption (P$_{dyn}$). During either of these transitions i.e. from HIGH to LOW or from LOW to HIGH, for a short amount of time both the transistors are conducting simultaneously, this results in power consumption due to short circuit (P$_{sc}$).

Another important point to note is that the dynamic power consumption is highly dependent on the input data (or output transitions). In other words, if the input data remains unchanged ($0 \rightarrow 0$ or $1 \rightarrow 1$), then the output remains unchanged, and hence there is no dynamic power consumption. If the input data changes ($0 \rightarrow 1$ or $1 \rightarrow 0$), then the output value changes accordingly (charging or discharging); thus dynamic power is consumed.

The dynamic power consumption is very important with respect to the side channel analysis. Dynamic power consumption depends on the switching activity of the gates, and also it depicts the relationship between internal data flow and the observed power consumption. Similarly the dynamic power consumption of an FPGA is due to the switching activity of FPGA's elements. The power model of an FPGA can be generated by computing the change in values of the FPGA cells before and after the clock edge. The number of bits changed in one clock equals the hamming distance between the data values before and after the clock pulse. This hamming distance is directly proportional to the dynamic power consumption. Details are well explained in [6]. Hamming distance is discussed in detail in chapter 4.

1.1.2 Simple Power Analysis

Simple Power Analysis (SPA) is a type of SCA which is well explained by Kocher et al. in [3] as "a technique that involves directly interpreting power consumption measurements collected during cryptographic operations." A power trace refers to the measured instantaneous power consumption of the device or FPGA. The power trace depicts the activity inside a device, refer to Fig. 1.2. This activity mainly depends on the data being processed and the operations being performed on the data. Therefore each power trace contains data dependent information. This dependency is exploited by the attacker to recover the hidden information. The goal of SPA is to reveal the secret information using only a few power traces. In extreme cases the attacker tries to find the secret information from one power trace and that too by observation only. Therefore SPA is used when only a few power traces are available and each power trace has significant noticeable distinctions. The attacker is assumed to have access to the attacked device. The attacker should have good detailed knowledge about the system. In SPA, the power trace is observed across the time axis (i.e. how the power is consumed over a period of time).

For example, squaring and multiplication operations are easily distinguished in case of RSA public key cryptosystem. In [7] the author describes an SPA attack on a private RSA exponentiation and an attack on a DES key schedule.

1.1.3 Differential Power Analysis

In 1998, Kocher introduced and demonstrated a powerful technique for cryptanalysis by measuring power consumption called Differential Power Analysis (DPA) [3]. DPA is a sophisticated and effective method of power analysis. The attacker uses statistical method analysis on multiple sets of collected power traces to recover the key. The attacker is assumed

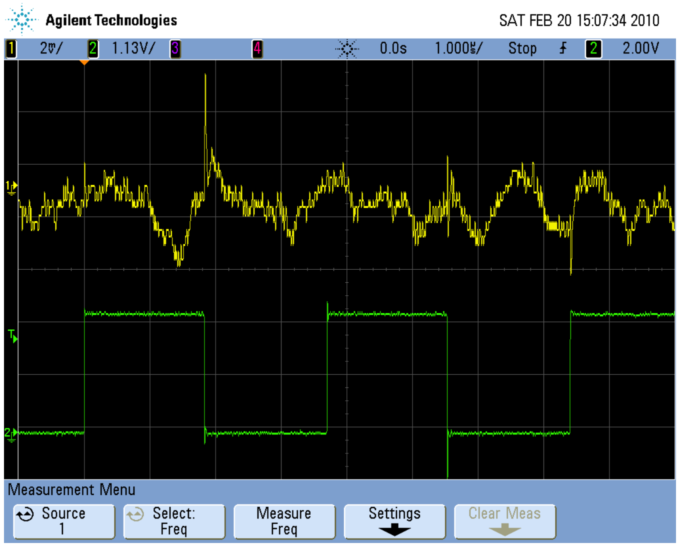


Figure 1.2: A Power Trace

to have frequent access to the attacked device. DPA attack differs from SPA attack in many ways. Unlike SPA, DPA requires multiple power traces and statistical analysis for an attack. In case of DPA, the power trace is analyzed at fixed moments of time (importance is given to the data being processed at that particular point of time) [8]. DPA attacks are considered to be more powerful than SPA attacks. One of the strengths of DPA is that, they are effective even if the power traces are extremely noisy. Another advantage of DPA over SPA is that you don't need to have detailed knowledge about the device, just the knowledge of the implemented algorithm is sufficient to perform an attack.

1.2 Countermeasures against DPA

Since Kocher et.al. demonstrated DPA as a threat against cryptographic implementations, there has been significant advances towards DPA countermeasures. The DPA attack works because the instantaneous power consumed by the device is highly data dependent. The main goal of the countermeasures is to eliminate or suppress this data dependency. All the countermeasures that have been published so far can be classified into three groups Protocol, Hiding and Masking.

1.2.1 Protocol

Protocol countermeasures focus on using session keys. A session key is a temporary symmetric key used for encryption during a session. Use of session keys makes power analysis more difficult because the attacker will be able to analyze only few power traces. Naccache describes such a countermeasure in his patent [9] using a dynamic secret key algorithm against DPA attacks. However implementing such a countermeasure is difficult and impractical in cryptosystems that natively do not use session keys. Further more this countermeasure is not considered strong enough to prevent power analysis attacks. Kocher explains the countermeasure's flaw in [10] stating "Unfortunately, most protocols today are not designed to withstand leakage. For example, conventional protocols that compute session keys by encrypting a counter or hashing a key with a nonce can be attacked using DPA because each counter/nonce value potentially reveals new information about the key to the adversary". Therefore hiding and masking schemes are preferred over this countermeasure.

1.2.2 Masking

The concept of masking is to randomize the intermediate data values. Since the power consumption is dependent on the intermediate data values, if these values are randomized before being processed then the resultant power consumption will be independent of the actual data values. In masking [8], the intermediate value v during algorithmic computations, is concealed by a random value m called as mask and a new value is generated v_m. This value is then processed and hence the information leakage will be independent of the original data values. Masking can be summarized by the equation:

$$v_m = v * m \tag{1.2}$$

The operation * is generally XOR, but sometimes it varies depending on the algorithm. Addition or multiplication operators can also be used. The masking scheme that uses the XOR as an operator is called **Boolean Masking**, while the scheme that uses addition or multiplication operators is called **Arithmetic Masking**. Sometimes the arithmetic operators are used along with modulus.

The most important aspect in this technique is the mask. The mask has to be chosen carefully and should be applied to all the intermediate values at every stage of the algorithm. The mask should also be applied to the values which are being derived from any previous intermediate values. The mask can be applied to a single bit or multiple bits of the intermediate values. The value of the mask and the operation of the mask should remain unknown to the attacker because the security of this countermeasure depends on the mask. Generally the mask is generated inside the device and hence it remains unknown to the attacker.

Masking can be implemented at gate level or at algorithmic level. Some of the examples of gate level masking implementations are Masked Dual-rail Precharge Logic (MDPL) [11], Random Switching Logic (RSL) [12] and Dual-rail Random Switching Logic (DRSL) [13]. All these masking styles use a single mask per intermediate value and are based on secret sharing schemes on two shares v_m and m [14]. In [15] Oswald et.al. demonstrated a masking implementation of the AES S-Box by shifting the computation of the finite field inversion in the AES S-Box down to $GF(4)$.

Researchers have implemented these masking countermeasures on various different algorithms evaluating the security. Suzuki et.al. [12] implemented RSL on AES, Kamoun et.al. [16] demonstrated a DPA secure masked S-Box of AES algorithm as described by Canright et.al. [17].

1.2.3 Hiding

The concept of hiding is to modify the power consumption characteristics of the device such that, the power consumption is independent of the data. Most of the hiding schemes modify power consumption characteristics at the gate level such that the instantaneous power consumed by the device remains constant in each operation. Some of the hiding schemes involve techniques that make the instantaneous power consumption of the device completely random. In other words, hiding focuses on increasing the noise or equalize/balance the power consumption in each operation. Hiding techniques are well recognized and are considered to be more successful than other methods. Hiding countermeasures involves gate level modifications. The major types of hiding styles are dual-rail precharge, asynchronous and current mode logic styles [14].

Dual-rail Precharge (DRP)

In DRP, the concept of dual rail and precharge logic are used together to achieve constant power consumption. The dual-rail technology implies that for each signal there are two wires. One wire carries the signal's original value and the other wire carries the signal's complementary value. This is also known as differential encoding. Thus, dual rail technique hides the original value of the signal. The dual rail cells have complementary wires for inputs and outputs.

As discussed earlier, the dynamic power consumption during the transitions ($0 \rightarrow 0$, $0 \rightarrow 1$, $1 \rightarrow 0$ and $1 \rightarrow 1$) is not always same. Therefore this vulnerability can be exploited by the attacker. Precharge logic is used to fix this vulnerability. Precharge breaks the signal sequence and manages the transitions in signal values. It ensures that for every signal in the circuit there is always going to be one transition form 0 to 1 and one transition from 1 to 0. Precharge implementation generally involves gate level modification.

A constant power consumption can be achieved by balancing wires between the cells and balancing the internal structure of the cells. Therefore DRP ensures that the total power consumption of the device is always constant and hence independent of the data values. DRP logic styles increase overall power consumption. A few examples of DRP implementation are Sense Amplifier Based Logic (SABL) [18] [19], Simple Dynamic Differential Logic (SDDL) [20] [21], Wave Dynamic Differential Logic (WDDL) [20] [22], Divided Wave Dynamic Differential Logic (DWDDL) [20], etc.

Asynchronous

In this case the circuits are designed as asynchronous circuits. The underlying concept is that, the asynchronous nature of the circuit induces randomization based on the data flow paths and gate delays. This randomization leads to randomized power consumption and thus increases the security. Fournier et.al. [23] demonstrated the security of asynchronous circuits using a DPA resistant asynchronous processor test chip. However research shows that the randomization is quite data dependent and hence there is no significant increase in security. Eventually asynchronous circuits are implemented in conjunction with DRP logic styles to increase security.

Current Mode Logic (CML)

Unlike DRP, in case of CML the differential encoding is applied to the current flowing through different paths in the circuit instead of the voltage. The underlying concept is that the total sum of currents flowing through the circuit remains the same and almost independent of the output values and hence more secure against DPA. A few examples of CML are Metal-oxide-semiconductor Current Mode Logic (MCML) and Dynamic Current Mode Logic (DyCML) [24]. Regazzoni et.al. [25] demonstrated the DPA resistance of a MCML implementation. One drawback of MCML is that its implementation increases the power consumption by a huge amount and hence DyCML is used instead of MCML. Toprak et.al. [26] implemented a low power design for MCML, which consumes power comparable to classical CMOS cells. Mace et.al. [27] has proposed a DPA resistant design with DyCML.

The advantage of masking over hiding is that masking can be applied at the algorithmic level and gate level whereas hiding can be applied only at gate level. Another advantage is that power consumption characteristic of the device need not be forcefully changed. On the other hand, Hiding techniques are usually considered to be more secure than masking techniques, based on the number of samples required to break the key and also as cited by the author in [28]. Chen et.al. [29] demonstrate an attack on arithmetic masking scheme. Various possible attacks on hiding and masking schemes are discussed in detail in [8].

1.3 Drawbacks of Countermeasures

All the countermeasures that increase security of the implementation against DPA involve either one or more of the following: modification at the algorithmic level, changing the basic gate design, adding extra logic or noise. These techniques ultimately results in increased logic area, power consumption and time delay. Suzuki et.al. [12] reports his implementation results of various masking techniques. These results clearly show an increase in area ranging from 100% to 400% along with an increase in critical path delay by a factor of 1.5 to 2.3. Kamoun el.al. [16] reports his implementation results of masking S-Box technique on AES and his results show an increase in area by about 60.1% along with an increase in critical path by about 4%. Velegalati et.al. [21] implemented SDDL countermeasure and they were able to improve the security by about 9 times over that of the original design but at the cost of increased area by about 2.1 times and increase in the delay by about 2 times. Tiri et.al. [20] compared area utilization and critical path delay of three algorithms namely Kasumi, Data Encryption Standard (DES) and Advanced Encryption Standard (AES) for a normal single ended design and a WDDL implementation. The results show an increase in area (in terms of gates) for the WDDL implementations compared to normal single ended designs by a factor ranging from 3.2 to 3.6 with a small increase in critical path delay. Yu et.al. implemented their prototype design on FPGA in [30] and their results show an increase in slice count for WDDL and DWDDL implementations compared to single ended designs by a factor of 5.8 and 11.6 respectively. Thus gain in security is achieved at the cost of drastic increase in area consumption and critical path delay.

Chapter 2: Elements of an FPGA

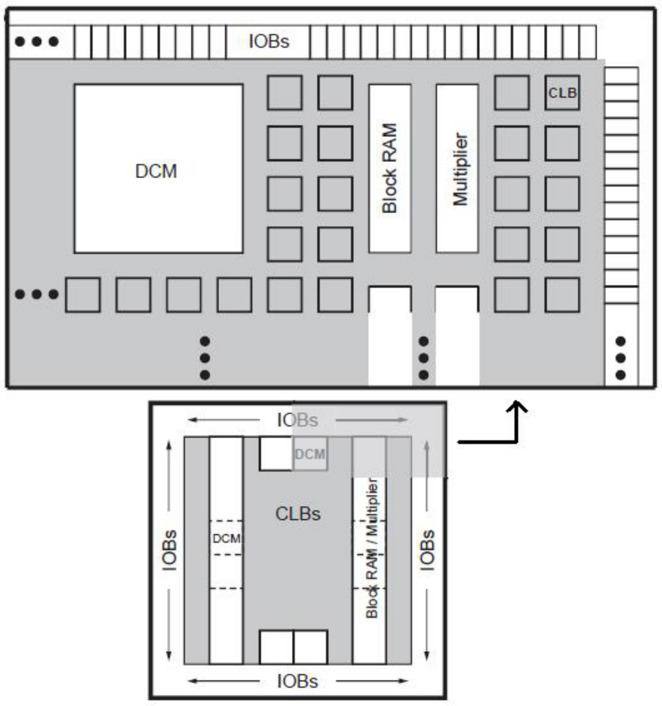
Field Programmable Gate Array (FPGA) is a re-programmable integrated circuit. It can be configured by the designer to realize any digital logic. FPGAs consists of programmable logic elements to implement the logic, and programmable switch matrix for interconnections. There are several manufacturers of FPGAs including Xilinx, Altera, Actel, QuickLogic, etc. They use different technology for manufacturing of FPGAs. For Example Xilinx, Altera, Lattice Semiconductor uses SRAM based technology, Actel uses antifuse technology. FPGAs are preferred over ASICs and full custom design chips due to their programmability, off-the-shelf availability, lower development cost and shorter time to market.

This thesis discusses in detail the Xilinx Spartan 3 family FPGAs [31], which consists of the following five programmable parts: Logic Blocks, I/O Blocks, Memory Blocks, Multiplier Blocks and Clock Manager Blocks. These blocks are shown in Fig. 2.1.

2.1 CLBs, Slices and LUTs

Configurable Logic Block (**CLB**) is the basic block that implements a wide range of sequential and combinational logic functions, as well as stores data. Each CLB consist of four interconnected slices that are grouped in pairs. Each slice contains two flexible Look-Up Tables (**LUTs**) that implement logic, two dedicated storage elements that can be used as flip-flops or latches and few logic elements. The pairs of slices are arranged in form of columns. The left hand pair of slices is called **SLICEM** (Fig. 2.2). They are capable of implementing both logic and memory functions. The right hand pair of slices is called **SLICEL**. They can support implementation of only logic functions. Therefore, a slice contains:

- Two 4-input LUTs



[a]

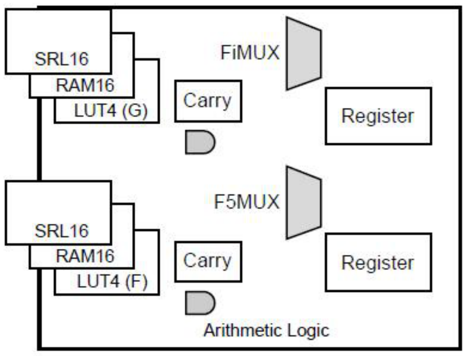
Figure 2.1: The FPGA Fabric

[a]From Xilinx datasheet

- Two storage elements for implementing a latch or a flip-flop
- Two wide function multiplexers
- Carry and arithmetic logic elements
- Two 16x1 distributed RAM blocks instead of 4-input LUTs (Only in **SLICEM** slices)
- Two 16 bit shift registers instead of 4-input LUTs (Only in **SLICEM** slices)

2.1.1 LUTs as Distributed RAMs

The 4-input LUTs in the SLICEM can be programmed as distributed RAM, capable of storing 16 bits of data. Therefore, one CLB block containing two SLICEMs, is capable of storing 64 bits of data in Distributed RAMs. It can also be configured as 32 bits of Dual-ported RAM. This memory is different from the dedicated memories called as Block



[a]

Figure 2.2: SLICEM

[a]From Xilinx datasheet

RAMs (BRAMs). These RAMs use slice area. Distributed RAMs are fast, localized, and best suited for implementing small data buffers, FIFOs, or register files. In distributed RAM data write is synchronous while the data read is usually asynchronous. Synchronous data read can be achieved by using the available register in LUT.

2.1.2 LUTs as Shift Registers SRL16

The 4-input LUTs in the SLICEM can also be programmed as a 16-bit shift register. Therefore, one CLB block containing two SLICEMs, is capable of implementing one big 64-bit Shift Register. The advantage of this implementation is that, a large shift register can be implemented without using flipflop chains. The associated flip-flop can be used along with shift registers to achieve synchronous output.

2.2 I/O Blocks

Input/Output Blocks (IOBs) controls the dataflow between the I/O pins(external world) and the internal logic implemented in the device. IOBs support programmable uni-directional or bi-directional data flow with a wide variety of interface standards. It also supports 3-state logic when the output is high impedance.

2.3 Multiplier Block

Dedicated 18x18 bit multipliers are useful for implementing fast DSP logic. They are available only in selected FPGAs of the Spartan-3 family. The Spartan-3A DSP family of FPGAs has the high performance DSP multiply-accumulate blocks known as DSP48. Each block accepts two 18-bit numbers (signed/unsigned) as inputs and calculate their product. This block is also capable of performing other arithmetic functions such as 48-bit addition or subtraction. These DSP blocks share the dedicated routing resources with Block RAMs. These blocks also have dedicated registers for pipelining and other useful DSP implementations.

2.4 Digital Clock Manager Block

Digital Clock Manager (DCM) Block provides advanced clocking facilities. DCMs help resolve many of the common clocking issues and also help in improving performance, as discussed below:

- DCM helps to **Eliminate Clock Skew** within the device or even to the external components, thus improving overall performance of the system and eliminating the delays due to clock distribution.
- It facilitates complete or partial **Phase Shift** by a fixed fractional amount of the incoming clock signal, mainly used for delaying the clock.
- It is capable of **Frequency Synthesis** by multiplying or dividing the input clock, or even generate a new frequency by a mixture of clock multiplication and division operations.
- It also **Conditions the Clock**, ensuring a clean output clock with a 50% duty cycle.
- It can also **Mirror, Forward, or Rebuffer** the input clock. It helps converting the incoming clock signal to a required I/O standard, minimize skew, etc.

The DCM can perform any or all the above functions simultaneously.

2.5 Dedicated Multiplexers

A multiplexer (Mux) is a very basic block used in the digital logic designs. The Spartan-3 family FPGAs support Wide-function multiplexers which efficiently combine the LUTs inside the slice and across the slices in order to achieve more complex logic operations. Each slice has two 4-input LUTs. Each LUT can implement a 2:1 mux. The dedicated mux **F5MUX** present in the bottom portion of the slice, combines the two LUTs in a slice implementing a 4:1 mux. The dedicated mux **FiMUX** in the top portion of the slice combines these 4:1 muxes across the slices. Therefore, a 16:1 mux can be implemented in a CLB without using any routing resources and thus avoiding routing delays.

2.6 Carry and Arithmetic Logic

Most of the digital system design circuits requires basic blocks such as adders, counters, comparators, etc. Although these all can be implemented using the LUTs; dedicated resources are available to improve the performance and reduce the routing delays. The carry chain consist of dedicated gates, multiplexers and connections with other logic gates. It facilitates fast and efficient implementations of adders.

2.7 Memory Blocks

These are dedicated memory elements present in an FPGA. There are many applications that require memories for storing data, configurations, etc. In case of Spartan-3 family there are dedicated memory blocks with their dedicated routing resources. These memory blocks are called **Block RAMs** (BRAMs). Unlike Distributed RAMs, BRAMs are completely synchronous. The BRAMs are discussed in detail in the next chapter.

Chapter 3: Block RAMs

These days memory blocks are an inevitable part in a system's design. They have many applications such as FIFO, circular buffers, delay lines, complex state machines etc. These days BRAMs have gained popularity for their ability to reduce slice area consumption in certain implementations, which are discussed later in this chapter.

BRAMs are generally located close to IOBs and CLBs and have their own dedicated routing resources. BRAMs are organized in form of columns. The Block RAM consists of fast static SRAM cells. Each BRAM can store upto 18,432 (18K) bits, of which 16,384 (16K) is allocated for data and 2,048 (2K) is allocated for parity or additional data. The total number of BRAMs and memory bits depends on a particular FPGA. Each BRAM supports many aspect ratios and can be configured depending upon the number of input address lines and output data lines. Physically, the BRAMs are dual ported having two completely independent access ports. Each BRAM is completely symmetrical, and both ports are interchangeable. Both ports support independent data read and write operations having their own clock, clock enable and write enable.

Unlike Distributed RAMs, BRAMs are completely synchronous. The writes as well as the reads are synchronous with their clock edge. BRAMs can be operated in any of three different write modes. The first one is WRITE_FIRST mode, in this case the data is written into the destination memory and simultaneously provided on the data-out port. The second one is READ_FIRST mode, in this case the current data of the target memory location appears on the output port, and the new data is written into the memory location. The third one is NO_CHANGE mode. In this mode memory read is not performed, the data on output port remains unchanged from the previous clock cycle, and new data is written into the memory location.

BRAMs can be realized using either of the two methods i.e. inference or instantiation. In case of inference, the tool infers the BRAM directly from the HDL code, (provided all the options in the tool are enabled properly). In case of instantiation a core format is provided and it is modified as required. Some sample codes are available in the manual [31] [32]. Some tools also have core generators. Also, there are some attributes available, which force the compiler to implement a particular HDL code using a BRAM.

3.1 Block RAM Applications

There many applications of BRAMs, described in detail in [33]. A few of them are explained below:

- **First-In First-Out (FIFO)**, is one of the most common applications for BRAMs. It is used for efficiently ordering, resynchronizing and manipulating data as required in the application.
- **Circular Buffers**, are implemented using BRAMs and a counter. They are used in many Digital Signal Processing (DSP) applications such as Finite Impulse Response (FIR) filters, cross-correlations, etc.
- **Fast Complex State Machines**, since the BRAMs can be initialized with any set of values, they are an excellent choice for implementing complex state machines containing its next state sequence along with the state outputs. It can also be configured using BRAM's dual port nature and thus separating its next state from its state outputs.
- **Content-Addressable Memory (CAM)**, commonly known as associative memory. It is a very important and well known application for most embedded systems and a variety of networking and DSP applications.
- **Shift Registers**, using the READ_FIRST mode and a counter the designer can implement a fast shift register.

Recently, Block RAMs have gained popularity for implementations involving reduced slice area usage. In 2003, Chodowiec and Gaj [34] demonstrated the most compact 32-bit datapath architecture for AES-128 utilizing 222 slices and 3 BRAMs; following them Rouvroy et.al. in 2004 [35] used similar concept and achieved better results utilizing 163 slices and 3 BRAMs; and finally in 2007, Huang et.al. [36] implemented the same architecture maximizing BRAM usage and achieved results utilizing 148 slices and 11 BRAMs. Chaves et.al. [37] demonstrates an efficient use of BRAMs in a reconfigurable memory based co-processor. In [38] Chang et.al. explains the use of BRAMs to save on area and implements an AES with 8 bit datapath using only 130 slices and 4 BRAMs. Drimer et.al. [39] shows an efficient method to maximize the use of BRAMs and other elements while minimizing the use of LUTs, implementing a 32-bit AES T-Box design using only 93 slices with 2 BRAMs and 4 DSP blocks. These implementations clearly demonstrates that careful usage of BRAMs leads to drastic reduction of slice area consumption.

3.2 Security Analysis of BRAMs

In the world of cryptographic hardware implementations, there are two very important factors to consider; security and low area. As discussed previously, all countermeasures improve the security of the cryptographic hardware implementations at the cost of increased area. Most of them also cause an increase in critical path delay making the circuit slower. Many countermeasures involve modifications at the cell level which is very difficult to realize in different technologies and leads to increased area and power consumption. Therefore a good countermeasure would provide better security, occupy less area, consume less power and would be easy to realize.

Several features of BRAMs indicate that their use might lead to implementations that are more resistant to DPA attacks than implementations in other resources such as LUTs or Distributed RAMs.

3.2.1 Reduced Glitches

A 8x8 S-Box (2048 bits) implementation using LUTs, occupies nearly 64 Slices, similarly a 8x32 T-Box (8192 bits) implementation using LUTs, occupies about 256 Slices. Two of these S-Boxes or even T-Boxes can fit in one single BRAM. Therefore it is common practice to use BRAM implementations in order to conserve slice area. This area reduction leads to less utilization of corresponding routing resources and avoids glitches. Furthermore BRAMs are inherently glitch free.

Glitches are unexpected output transitions due to hazards, resulting from combinational logic gates delays and routing delays. Therefore, glitches are data dependent and influence the dynamic power consumption. This results in information leakage which can be exploited by DPA. BRAM cells do not have any combinational path from address to the output. Further more the output ports are latched with a self-timed circuit providing glitch-free read operations.

3.2.2 BRAMs are Fast Static SRAM Cells

BRAMs consist of fast static SRAM cells. Konur et.al. [40] explain in detail the structure of an SRAM cell, its operation and power consumption leakage during memory read and write operations. After preforming various experiments, they concluded that the power consumed by an SRAM cell during memory read operation remains the same, irrespective of reading 0 or 1. Therefore, using a BRAM as a ROM should provide security against power analysis attacks.

Because of reduced slice area consumption, glitch free property and SRAM cell structure of BRAMs, the implementations using BRAMs are expected to be less susceptible to DPA. In order to explore the DPA resistance of BRAMs, we implement designs by varying BRAM usage towards both possible extremities.

Chapter 4: Attack Methodology

4.1 Hamming Distance Model and Correlation Attack

As discussed in Chapter 1, the power consumption of FPGA is directly proportional to the total number of gate outputs changing from 1 to 0 (capacitance discharging) or 0 to 1 (capacitance charging).

Hamming Weight is the total number of non zero characters present in a given set, string or word. Digital circuits consist of binary systems, thus hamming weight is the total number of bits that are set (ON) in a particular string or word. Therefore the hamming weight model can be used for analysis of system at a given particular instant with a known output, without being referenced to any previous or next output values. In other words, this model is more favorable for SPA attacks. According to the power model of FPGAs, the dynamic power consumption depends on the change of bits and not just on the final value of bits, therefore hamming distance model is more suitable for our analysis. Hamming weight model can also be applied to DPA, but it is a weaker attack compared to hamming distance model.

Hamming Distance is the total number of positions, where the characters differ after comparing two strings of equal length. Digital circuits consist of binary systems, thus it is the total number of bits that change value. As discussed earlier, the power model of an FPGA is generated by computing the change in values of the FPGA cells before and after the clock edge. Thus hamming distance model resembles the leakage model of FPGAs and hence it can be used to create an expected power model. The power model may differ from design to design depending on the position of the attack and elements under attack. This power model is then correlated with the actual power consumption recorded by the oscilloscope.

Correlation can be described as the degree to which two variables are related to each other, or to measure the degree of dependence. We use correlation as a statistical method to analyze the relationship between the actual power trace and calculated power model. Correlation power analysis is also used as an attack method [41], it is based on hamming distance calculations. In this thesis, Pearson product-moment correlation coefficient, commonly known as Pearson's correlation is used to compute correlation. This method is sensitive to linear dependence between two variables. The correlation is calculated by Eq. (4.1)

$$r_{xy} = \frac{n\sum_i^n X_iY_i - \sum_i^n X_i \sum_i^n Y_i}{\sqrt{n\sum_i^n X_i^2 - (\sum_i^n X_i)^2}\sqrt{n\sum_i^n Y_i^2 - (\sum_i^n Y_i)^2}} \tag{4.1}$$

where r$_{xy}$ is the Pearson's correlation coefficient for variables X and Y, X is the actual power consumption and Y is the calculated hypothetical power model.

The absolute value of Pearson's correlation coefficient ranges from -1 to +1. The +1 value signifies that the two variables are perfectly linear dependent on each other and related in a positive direction i.e. if one variable increases then the other variable also increases. The -1 value signifies that the two variables are perfectly linear dependent on each other but related in a negative direction i.e. if one variable increases then the other variable decreases. The 0 value means there is NO linear relation between the two variables.

4.2 Experiment Designs

Several designs are implemented by varying the utilizations of LUTs, Distributed RAMs and BRAMs. The designs are divided into two groups, small scale implementations (Test Design) and real world implementations (AES-128 cipher). The advantage of small scale implementations is that, they are very easy to control, manipulate and analyze. This Test Design is similar to designs used by Yu et.al.[30] and Velegalati et.al.[21] for WDDL and SDDL countermeasures respectively. The Test Design incorporates essential components of the block cipher AES. The real world implementations are that of AES-128 bit cipher

[42] with a standard S-Box design and a T-box design. Analysis on larger-scale/real-world implementations allows us to relate and confirm whether the use of BRAMs leads to more DPA resistant implementation. These designs are discussed in detail in the next chapters.

4.3 Measurement Setup

The experiments were performed using a Xilinx Spartan 3E starter kit with a XC3S500eFG320-4 FPGA. We removed the capacitors around the FPGA and the core voltage net in order to obtain the unfiltered power signals. An external DC power supply was used to power the FPGA core. Power consumption was measured using Tektronics CT-1 current probe and an Agilent DSO6054A oscilloscope, which has a bandwidth of 500MHz and can record samples up to 4GSa/sec. We applied an input clock frequency between 100KHz-500KHz.

4.4 DPA Attack

DPA attacks were performed on all designs discussed in this thesis. It A DPA attack involves various steps which can be summarized as follows:

1. **Analyze the design and derive equations**: The design is analyzed and a particular register or memory element is selected for the attack. The step size/bit size (number of bits attacked simultaneously) from the total number of bits is decided. This is important because all the mathematical computations and complexity depends upon the attack bit size. The higher the bit size, better is the attack because more bits are used for correlation but it also results in larger computation complexity. Usually the number of bits attacked (step size) is 8. Therefore there are 2^8 different possible combinations for the unknown key. The equations are then derived from the previous and next values at the point of attack in a clock event.

2. **Generate the expected power model**: The hamming distance is calculated using the previously derived equations, for all possible values of the key, and a power model is calculated for that design.

3. **Implement on FPGA**: The HDL code is synthesized and implemented on the FPGA chip.

4. **Measure Power**: The experimental setup is used to measure the instantaneous power consumption when algorithm is being executed on the FPGA.

5. **Correlation**: The actual recorded power trace and the calculated power model are correlated using the Pearson's correlation method. The result shows the correlation of each assumed key value with the actual power trace.

6. **Calculate MTD**: The security of design against DPA attacks is determined by the number of measurements required to recover the key, also known as *Measurements To Disclosure* (MTD). This is the minimum required number of measurements after which, the correlation of a particular key value dominates all other key values. Data from one encryption operation is considered one measurement.

Chapter 5: Test Design

The Test Design consists of a synchronous (Sync.) S-Box whose input is connected to an 8-bit LFSR and output is xored with an 8-bit Key. The result is stored in a register. The block diagram of this circuit is shown in Fig. 5.1.

A sync. S-Box is a S-Box look-up table (8x8) followed by a register. It can be implemented as one block using BRAMs, absorbing the register. A sync. XOR block is a 8-bit logic XOR gate followed by a register. To implement this block using BRAMs, the logic gate is replaced by a precomputed look-up table followed by a register. We attack the design at the output of the LFSR. The hamming distance equation for this attack is shown in Eq. (5.1)

$$P_{est.} = \mathrm{HD}(lfsr_{(i-1)}, \mathrm{SBOX}^{-1}(k_{guess} \oplus Q_i)) \tag{5.1}$$

Apart from the block diagram design, the actual implementation has a wrapper circuit containing additional counters and multiplexers (muxes). The main purpose of the counter is to keep track of the number of encryptions. A multiplexer is used for selecting different key values. The counter and some muxes are also used to perform other functions such as: control and verify the final outputs on FPGA board I/Os, generate a trigger signal that will indicate the start of an encryption which is used as a reference point for the DPA attack, control logic to start and stop the encryption, etc. The main purpose of wrapper circuit is to manage the number of I/O ports required by the design with the available I/O ports on a given FPGA.

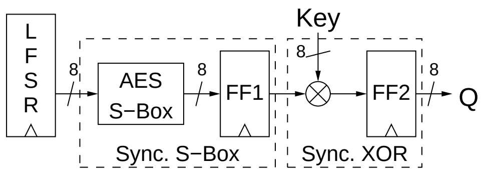


Figure 5.1: Block Diagram of Test Design

Table 5.1: Various Implementations of Basic Test Design

Circuit No.	S-Box in	XOR in
1.	LUTs	LUTs
2a.	Distributed RAMs	LUTs
2b.	LUTs	Distributed RAMs
2c.	Distributed RAMs	Distributed RAMs
3a.	BRAMs	LUTs
3b.	LUTs	BRAMs
3c.	BRAMs	BRAMs

5.1 Basic Test Design Circuits

The Test Design was implemented using various resources such as 1.) LUTs, 2.) Distributed RAMs and 3.) BRAMs; for individual components such as S-Box and XOR, which are also the basic components used in many cryptographic algorithms. The various possible implementations are summarized in Table 5.1.

5.1.1 Results

All the 7 circuits were implemented, and the post-place-and-route results are summarized in Table 5.2.

The best results for minimum area are achieved by circuits 3a and 3c. Both circuits use BRAMs for the implementation of S-Box, thus resulting in slice area reduction by over a factor of 5 compared to LUT implementations. Implementing the XOR in BRAM does not lead to any significant reduction in slice area because it was occupying only a few slices originally.

Table 5.2: Implementation Results of Basic Test Design Circuits

Circuit No.	Circuits	Slices	FFs	4 input LUTs	BRAMs	Minimum Delay	MTD
1.	S-Box and XOR in LUTs	85	24	161	0	7.737 ns	> 456
2a.	Only S-Box in Distributed RAMs	95	23	191	0	7.727 ns	> 256
2b.	Only XOR in Distributed RAMs	117	32	219	0	9.115 ns	> 256
2c.	S-Box and XOR in Distributed RAMs	128	32	247	0	6.695 ns	> 256
3a.	Only S-Box in BRAMs	16	16	29	1	5.710 ns	$> 13,000$
3b.	Only XOR in BRAMs	86	24	157	1	8.350 ns	> 256
3c.	S-Box and XOR in BRAMs	15	16	25	2	5.569 ns	$> 13,000$

Circuits 3a and 3c are also the fastest implementations. The critical path of a LUT based S-Box implementation consist of multiple LUTs and corresponding connections, leading to a slower design.

The security of design against DPA attacks is determined by the number of measurements required to recover the key, also known as *Measurements To Disclosure* (MTD). It is clearly visible that S-Box in BRAM implementations (circuits 3a and 3c) have about 26 times higher MTD compared to S-Box in LUTs, hence provide an increased resistance against DPA. It is an important point to note that, implementing only XOR in BRAM does not increase the security of the design.

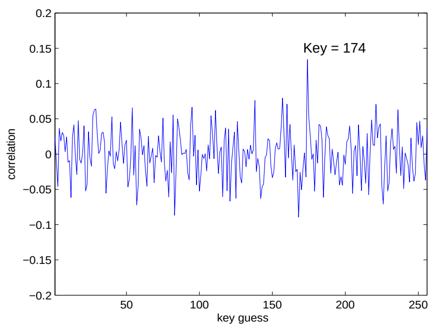
Another important point to note is that the position of the XOR and S-Box does not affect the security at all. All the 7 circuits mentioned above were implemented with the positions of XOR and S-Box swapped with each other. The MTD and area results were almost same.

5.1.2 MTD Graphs

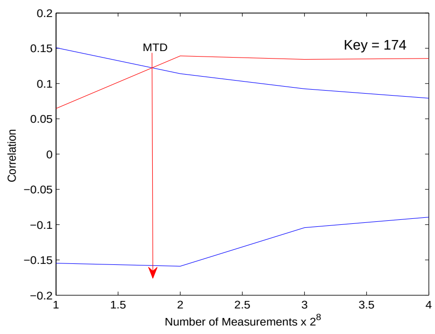
The various correlation plots and MTD plots for all Basic Test Design circuits are shown below. A fixed key of value decimal 173 was selected during the experiments to verify the results. The MATLAB figures show the correct key as 174 because in case of MATLAB, the key guess file starts with value decimal 1 and goes till decimal 256 instead of 0 to 255. The power trace data sample collected from oscilloscope contained 1024 data samples for correlation, which was sufficient to crack the key for most of the circuits as seen in Table 5.2. For two circuits involving S-Box in BRAM, a single power trace containing 1024 encryptions was not sufficient to crack the correct key. Therefore multiple power traces were collected and appended together for computations. The MTD plot shows the value after which the correct key dominates all other keys, in correlation with the actual power consumption data. All the MTD results were also verified with different set of keys.

5.2 Duplicate Test Design Circuits

The results from basic design implementations show that circuits 3a and 3c have the best resistance against DPA, occupy the least slice area and have minimum critical path delay. Hence, an argument can be made that the low area consumption which leads to lower dynamic power consumption might be the cause for the higher DPA resistance. Therefore, in order to verify this claim, we increased the area consumption through replication of the circuit 3a. Circuits 4a, 4b and 4c are duplication, triplication and quadruplication of circuit 3a respectively, as shown in Table 5.3. In order to create circuits 4a, 4b and 4c, and to exactly replicate circuit 3a, Xilinx Design Language (XDL) was used. A slight variation is observed in the slice area because circuit 3a has a few slices which are not required to be replicated.

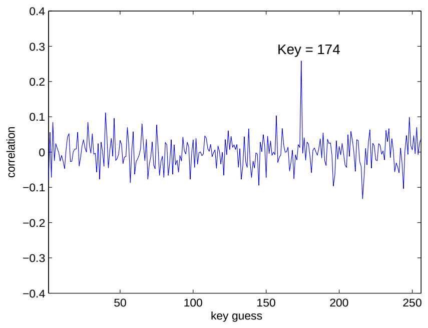


(a) Correlation after 1024 measurements for Circuit 1

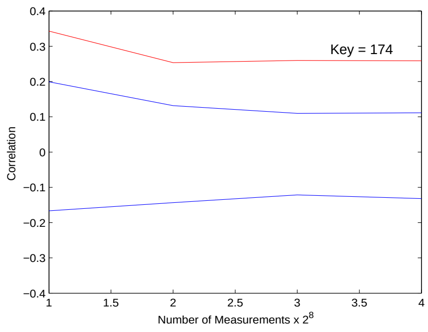


(b) MTD Plot of Circuit 1

Figure 5.2: Diagrams for Everything in LUTs (Circuit 1)

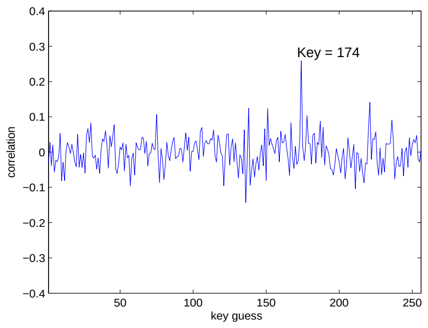


(a) Correlation after 1024 measurements for Circuit 2a

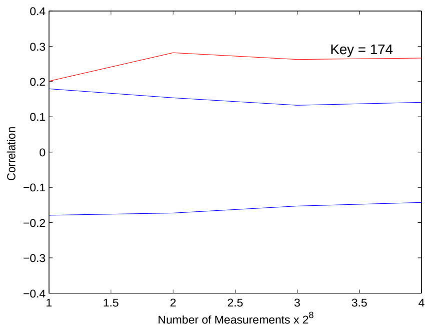


(b) MTD Plot of Circuit 2a

Figure 5.3: Diagrams for only S-Box in Distributed RAM (Circuit 2a)

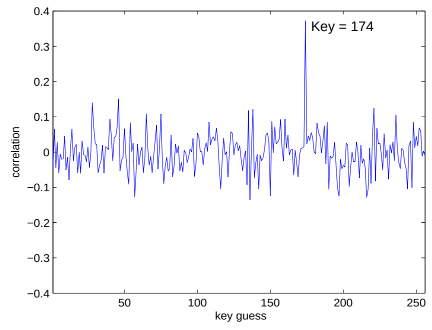


(a) Correlation after 1024 measurements for Circuit 2b

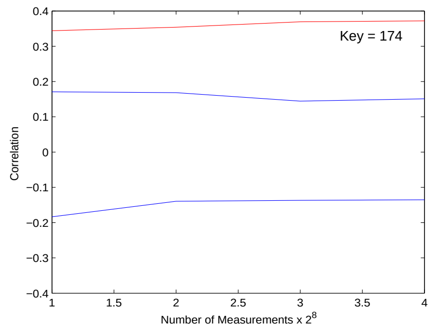


(b) MTD Plot of Circuit 2b

Figure 5.4: Diagrams for only XOR in Distributed RAM (Circuit 2b)

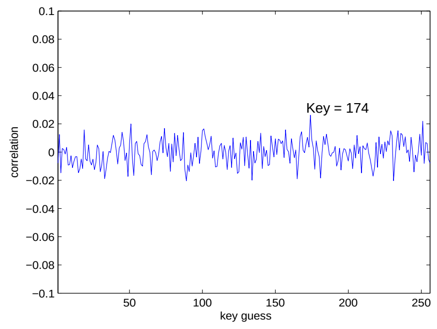


(a) Correlation after 1024 measurements for Circuit 2c

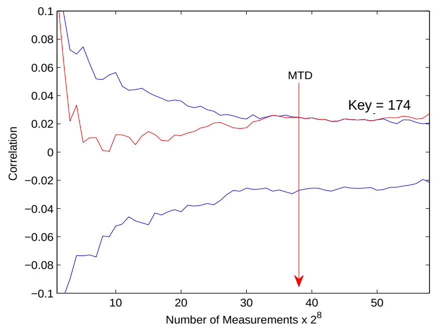


(b) MTD Plot of Circuit 2c

Figure 5.5: Diagrams for both S-Box and XOR in Distributed RAM (Circuit 2c)

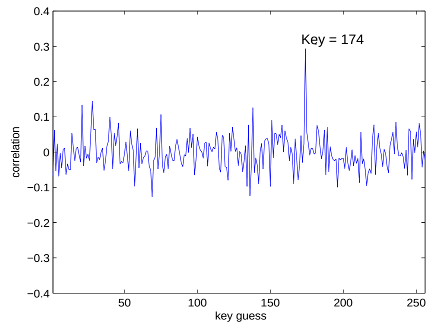


(a) Correlation after 14848 measurements for Circuit 3a

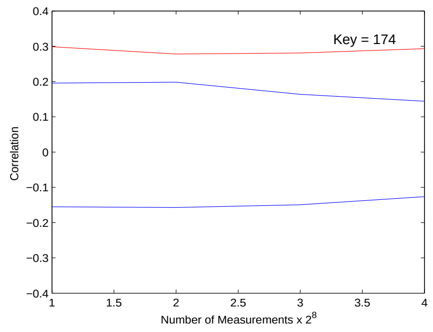


(b) MTD Plot of Circuit 3a

Figure 5.6: Diagrams for only S-Box in BRAM (Circuit 3a)

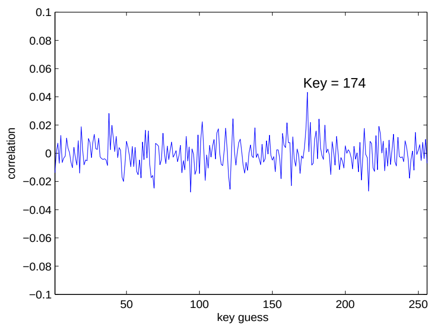


(a) Correlation after 1024 measurements for Circuit 3b

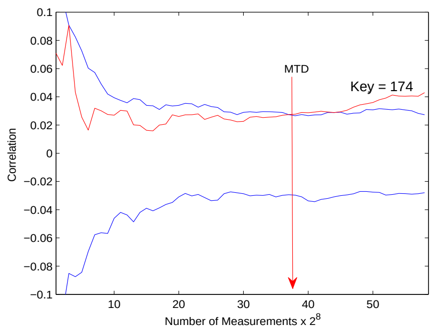


(b) MTD Plot of Circuit 3b

Figure 5.7: Diagrams for only XOR in BRAM (Circuit 3b)



(a) Correlation after 14848 measurements for Circuit 3c



(b) MTD Plot of Circuit 3c

Figure 5.8: Diagrams for both S-Box and XOR in BRAM (Circuit 3c)

Table 5.3: Various Implementations of Duplicate Test Design

Circuit No.	Circuit
4a.	Duplicate Circuit 3a
4b.	Triplicate Circuit 3a
4c.	Quadruplicate Circuit 3a
4d.	Dummy Circuit 3a

Table 5.4: Implementation Results of Duplicate Test Design

Circuit	Slices	FFs	4 input LUTs	BRAMs	Minimum Delay	MTD
3a. Only S-Box in BRAMs	16	16	29	1	5.710 ns	$> 13,000$
4a. Duplicate 3a Circuit	26	32	41	2	5.710 ns	$> 6,000$
4b. Triplicate 3a Circuit	35	48	53	3	5.710 ns	$> 3,000$
4c. Quadruplicate 3a Circuit	44	64	65	4	5.710 ns	> 500
4d. Dummy 3a Circuit	101	32	194	1	6.839 ns	$> 13,000$

5.2.1 Results

All 4 circuits were implemented, and the post-place-and-route results are summarized in Table 5.2.

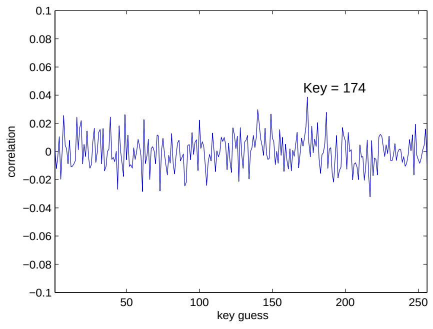
From the implementation results shown in Table 5.4, a drastic reduction in MTD is observed for circuits 4a, 4b and 4c because of increase in related logic. This might imply that the earlier claim is true and the increase in security is due to low area consumption. On the other hand, by replicating circuit 3a, an unfair scenario was created, because same data inputs were applied and signal strength was increased along with data dependent power consumption. Therefore in order to further investigate this drastic drop in MTD, circuit 4d was created.

In circuit 4d, the area consumption of circuit 3a was increased by adding a LUT based S-Box whose inputs are connected to an LFSR with different feedback coefficients. Hence this LFSR produces independent data from the original circuit. The results of this circuit show that even after increasing the area, and inevitably the total dynamic power consumption, the circuit still maintains the same DPA resistance as the original circuit 3a.

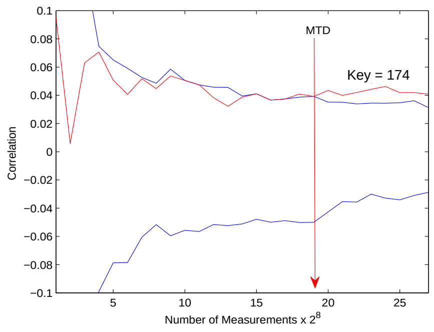
The results in Table 5.4 show that adding unrelated logic to circuits does not impact the security. This confirms that the resistance to DPA does not depend on the total amount of power consumption but on the power consumed by related logic.

5.2.2 MTD Graphs

The various correlation plots and Measurement To Disclosure (MTD) plots for all Duplicate Test Design circuits are shown below.

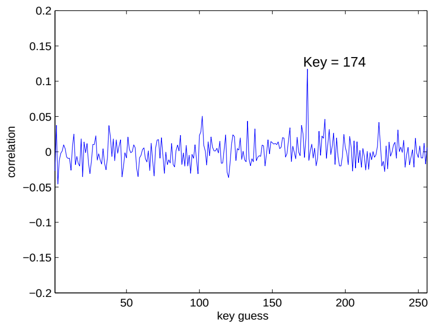


(a) Correlation after 6912 measurements for Design 4a

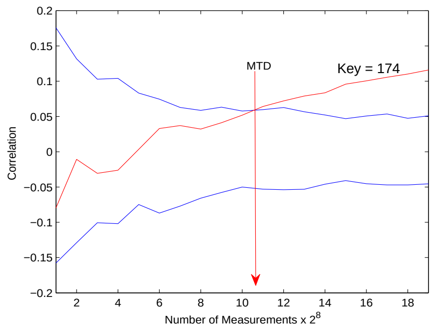


(b) MTD Plot of Design 4a

Figure 5.9: Diagrams for Duplicate Design (Design 4a)

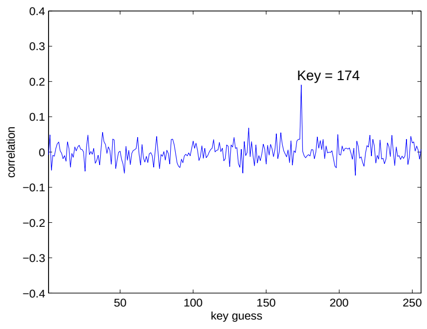


(a) Correlation after 4864 measurements for Design 4b

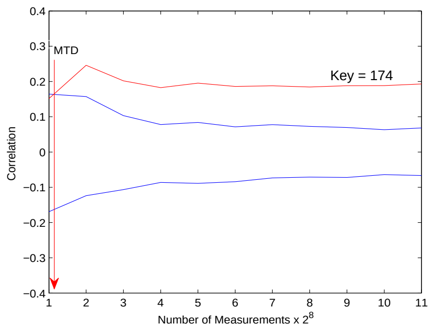


(b) MTD Plot of Design 4b

Figure 5.10: Diagrams for Triplicate Design (Design 4b)



(a) Correlation after 2816 measurements for Design 4c



(b) MTD Plot of Design 4c

Figure 5.11: Diagrams for Quadruplicate Design (Design 4c)

Chapter 6: AES 128-bit Design

Advanced Encryption Standard (AES) [42] is a block cipher and the current encryption standard adopted by the U.S. government. In 2001, AES was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) and then it became a federal government standard in 2002. Three out of five cipher options were selected as standards i.e. AES-128, AES-192 and AES-256 from originally published Rijndael. Each AES cipher has a similar architecture with a datapath of 128-bits and key sizes of 128, 192 and 256 bits, respectively. Various implementations such as high speed, high throughput, low power etc, have been reported by the researchers based on different datapath sizes (8, 16, 32, 64, 128-bit).

AES is an iterative cipher executing a round function several times. The number of rounds varies depending on the size of key. The round function consists of four different transformations: SubBytes, ShiftRows, MixColumns and AddRoundKey.

- **SubBytes** is a non-linear operation where each input byte is replaced by another byte based on the look-up table defined in the AES specification. This look-up table is commonly known as substitution box (S-Box).
- **ShiftRows** is a transpositional step where the bytes in a row are shifted along the row by a specific number of places.
- **MixColumns** is a mixing operation where the four bytes in a column are combined by modulo multiplication with a fixed polynomial.
- **AddRoundKey** is a step where the output is bitwise xored with the round key.

Each round uses an intermediate key called “round key” which is derived from the original key through key scheduling. The different round keys can be pre-computed and stored in a

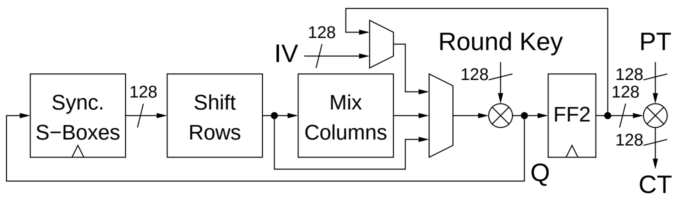


Figure 6.1: Block Diagram of AES-128 test Circuit

RAM or can be computed on the fly during the round.

We have implemented an AES cipher with 128-bit key length and 128 bit wide datapath. It is an encryption only design with on-the-fly key scheduling and it requires 11 clock cycles for one encryption. The SubBytes function is realized through 16 sync. S-Boxes. The cipher is implemented in Output Feedback (OFB) mode and can therefore generate new outputs without requiring new plaintext. This facilitates easy collection of multiple power samples for DPA. The block diagram for this design is shown in Fig. 6.1.

We attack the design at the output of the sync. S-Boxes, after the last round of encryption. The equation for calculating Hamming Distance is shown in Eq. (6.1) where Q_{11} is the output of the last round which xored with the plaintext PT to produce the ciphertext CT. Because of the OFB mode, the output of the last round Q_{11} is the same as the input to the next round Q'_1.

$$\begin{aligned} P_{est.} &= \mathrm{HD}(\mathrm{SBOX}(CT \oplus PT), \mathrm{SBOX}(k_{guess} \oplus Q'_1)) \\ P_{est.} &= \mathrm{HD}(\mathrm{SBOX}(Q_{11}), \mathrm{SBOX}(k_{guess} \oplus Q_{11})) \end{aligned} \tag{6.1}$$

From the basic Test Design Circuit's results it can be concluded that LUT based implementations and distributed RAM based implementations have similar results. The reason is that the distributed RAMs are implemented by LUTs. On the other hand the MTD is largely affected by varying the BRAM usage with respect to S-Box implementations. Therefore in AES designs, the comparison is done between the implementation using Distributed RAMs and implementations maximizing BRAM usage.

6.1 Standard S-Box Implementation using Distributed RAMs

In this design (5a.), the S-Box is implemented using distributed RAMs. The design uses 20 S-Boxes each of size 2^8x8 (2K bits), 16 S-Boxes performing the SubBytes operation on 16 bytes of data simultaneously and 4 S-Boxes performing the SubBytes operation in the key scheduling section.

6.2 Standard S-Box Implementation using Block RAMs

In this design (5b), the sync. S-Box is implemented using BRAM, absorbing the register after the S-Box. Each BRAM is capable of storing 18K bits, with dual port architecture. Therefore 20 S-Boxes can be implemented using 10 partially filled BRAMs. This reduces slice area by approximately 20*(64 slices) = 1280 Slices.

6.3 T-box Implementation using Block RAMs

This is a special design proposed by the authors of AES in [43]. It was initially intended for software implementations on 32-bit micro-processors. Later-on Fischer demonstrated its hardware implementation in [44]. The T-box design features computation of a complete AES round just by using Look-up tables and XOR gates. In T-Box architecture, the SubBytes operation and MixColumns operation is reformulated and implemented using 8x32 look-up tables followed by a large XOR network. The T-box operation and the T-box equations are well explained in [43] and [45].

In this design (5c.), the T-Boxes are implemented using BRAMs. Since each BRAM is capable of storing 18K bits, with dual port architecture, 20 T-Boxes can be implemented using 10 completely filled BRAMs.

Table 6.1: Summary of Implementation Results from AES-128 designs

Design No.	Design	Slices	FFs	4 input LUTs	BRAMs	Minimum Delay	MTD
5a.	Standard AES-128, Distributed RAMs	1,727	422	3,374	0	15.876 ns	300-1,300
5b.	Standard AES-128, BRAMs	412	262	787	10	13.875 ns	$> 9,500$
5c.	T-box AES-128, BRAMs	370	262	705	10	13.461 ns	$> 11,500$

6.4 Results

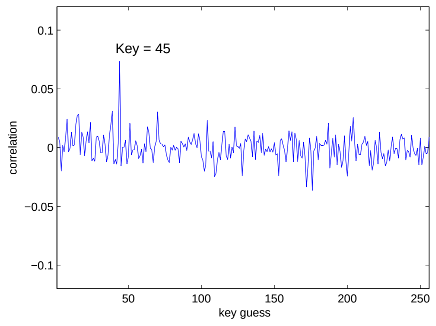
The post-place-and-route results for all three AES-128 implementations mentioned earlier are summarized in Table 6.1. All the results are for the core design only, excluding the wrapper circuit.

The best result for minimum area is achieved by design 5b and 5c. Design 5b implements the S-Box using BRAMs and design 5c implements the S-Box and partial MixColumn operation using BRAMs. Thus both designs demonstrate reduction in slice area by more than 4 times compared to Distributed RAM based implementation, at the cost of additional 10 BRAMs usage.

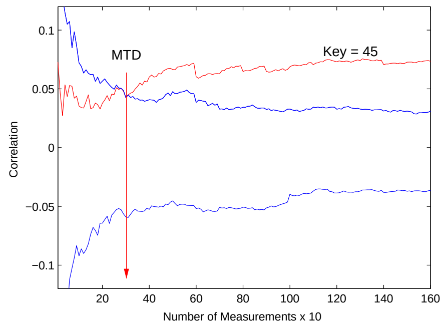
In terms of security, the results follow the same trend as that of the Test design circuits. The MTD results clearly show an increase in DPA resistance with BRAMs implementing more amount of the related logic. The standard AES implementation with its S-Box implemented in BRAM is about 7 times more secure than S-Box in Distributed RAMs. The T-box design which has S-Box and partial mix-columns in BRAMs, is about 9 times more secure. Along with higher security and less area, a small increase in speed is also achieved.

6.5 MTD Graphs

The correlation plots and MTD plots for all AES designs are shown below.



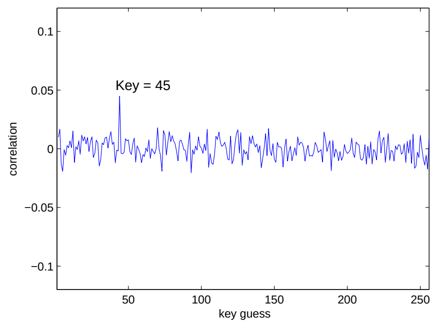
(a) Correlation after 1600 samples for Design 5a



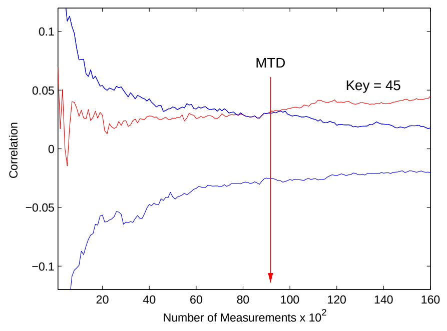
(b) MTD Plot of Design 5a

Figure 6.2: Diagrams for AES-128, S-Box in Distributed RAMs (Design 5a)

The key size for our AES design is 128 bits, which is very long and difficult to break at once. Therefore the attack is done with a step size of 8-bits. The attack position is the last round of the encryption. We attack the first byte of the subkey. Its value in that round is 43. The MATLAB figures show the correct key as 44 as in case of MATLAB the key guess file starts with value decimal 1 and goes till decimal 256 instead of 0 to 255. The power trace data sample collected from the oscilloscope contained 1000 data samples for correlation, which was sufficient to recover the correct key for design 5a, as seen in Table 6.1, but for the other two designs a single power trace containing 1000 encryptions was not sufficient to recover the correct key. Therefore multiple power traces were collected and appended together for computations. The MTD plot shows the value after which the correct key dominates all other keys in correlation with the actual power consumption data. All the answers were also verified with different set of keys.

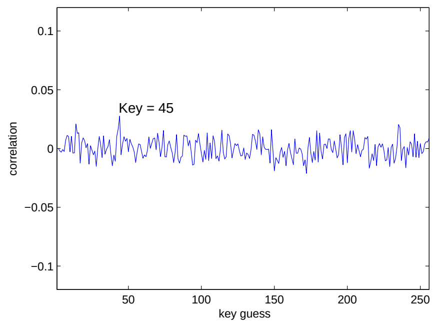


(a) Correlation after 16000 samples for Design 5b

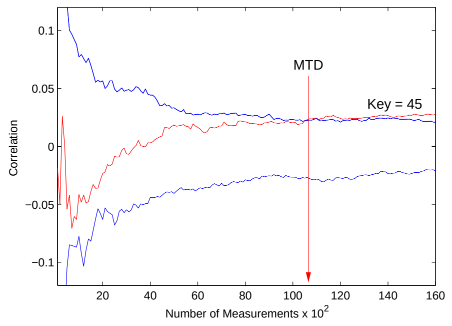


(b) MTD Plot of Design 5b

Figure 6.3: Diagrams for AES-128, S-Box in BRAMs (Design 5b)



(a) Correlation after 16000 samples for Design 5c



(b) MTD Plot of Design 5c

Figure 6.4: Diagrams for AES-128, T-Box in BRAMs (Design 5c)

Chapter 7: Comparison of various Countermeasures

Table 7.1 shows a detailed comparison of various countermeasures for DPA published by other researchers. This table gives a brief idea about other countermeasures, their advantages, disadvantages and a comparison with our designs.

An excellent comparison is possible between Yu et.al. [30], Velegalati and Kaps [21] and our basic test designs, as all three have similar LFSR + S-Box design structure. Yu et.al. [30] did not document MTD but confirmed WDDL being unsecure and DWDDL being more secure. The drawbacks were, increase in area by 484% (5.8 times) & 1068% (11.7 times) and increase in delay by about 390% (4.9 times) & 423% (5.2 times) for WDDL and DWDDL designs respectively. Velegalati et.al. [21] demonstrated (MTD$>$ 10,000) about 10 times increase in security. The drawbacks were, increase in area by 111% (2.1 times) & 107% (2.1 times) and increase in delay by about 100% (2 times) & 94% (1.9 times) for Mux-4 SDDL and Mux-16 SDDL designs respectively. Finally in our case, BRAM implementations have (MTD$>$ 13,000) about 26 times increase in security which is similar in strength compared to Velegalati and Kaps [21]. Along with it we have other advantages such as decrease in slice area by 84% (5.3 times) for LFSR + S-Box design and about 79% (4.7 times) for AES design, and decrease in delay by about 26% (1.4 times) and 15% (1.2 times) respectively.

Table 7.1: Comparison Across Countermeasures

Authors	Technology	Implementations	Area	Minimum Delay (ns)	MTD
Hwang et.al.[22]	TSMC 6M 0.18 μm AES Co-processor	original AES	79 Kgates	3	320 - 8,168
		WDDL AES	245 Kgates	11.7	21,185 - 1,276,186[a]
Yu et.al. [30]	Spartan 3E LFSR + S-Box	SE	70 Slices	3.99	NO
		WDDL	409 Slices	19.54	NO
		DWDDL	818 Slices	20.88	YES
Velegalati et.al.[21]	Spartan 3E LFSR + S-Box	Mux-4 SE	134 Slices	9.08	1,024
		Mux-4 SDDL	283 Slices	18.16	$> 10,000$
		Mux-16 SE	80 Slices	7.51	1,024
		Mux-16 SDDL	166 Slices	14.59	$> 10,000$
This Thesis	Spartan 3E LFSR + S-Box	S-Box Distributed RAM	95 Slices	7.727	> 256
		S-Box BRAM	16 Slices+ 1 BRAM	5.710	$> 13,000$
This Thesis	Spartan 3E AES-128	S-Box Distributed RAM	1,727 Slices	15.876	300-1,300
		S-Box BRAM	412 Slices+ 10 BRAM	13.875	$> 9,500$
		Tbox BRAM	370 Slices+ 10 BRAM	13.461	$> 11,500$

[a]5 Key-bytes were not found after 1.5M measurements

Chapter 8: Conclusion and Future Work

8.1 Conclusion

Previous work by the research community indicates that a good level of hardware security is generally accompanied with a large increase in area and delay. Our experimental results show that moving DPA vulnerable components from LUTs to BRAMs provides a moderate level of security against DPA attacks. The security is comparable to current SDDL implementations on FPGAs [21]. Unlike other countermeasures, the increase in security is without any increase in area or delay, on the contrary it demonstrates significant reduction in slice area and delay, at the cost of BRAMs. The technique is easy to implement and does not require low level design changes such as complimentary logic or symmetric routing.

8.2 Future Work

We want to investigate if BRAM implementations along with some other appropriate hiding or masking countermeasures would provide a better level of security at incremental area cost. We would also like to explore the applicability of this technique to FPGAs from other vendors.

Bibliography

Bibliography

[1] H. Bar-El, "Security implications of hardware vs. software cryptographic modules," Discretix Technologies Ltd., Tech. Rep., Oct. 2002.

[2] P. C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *Advances in Cryptology - CRYPTO 96*, ser. Lecture Notes in Computer Science (LNCS), vol. 1109. Berlin: Springer-Verlag, 1996, pp. 104–113.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology - CRYPTO'99*, ser. Lecture Notes in Computer Science (LNCS), vol. 1666. Berlin: Springer Verlag, Aug 1999, pp. 388–397.

[4] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: Concrete results," in *Cryptographic Hardware and Embedded Systems–CHES 2001*, ser. Lecture Notes in Computer Science (LNCS), c. K. Koç, D. Naccache, and C. Paar, Eds., vol. 2162. Springer-Verlag, 2001, pp. 251–261.

[5] J. M. Rabaey, A. Chandrakasan, and B. Nokolic, *Digital Integrated Circuits, A Design Perspective*, 2nd ed. Pearson Education, 2003.

[6] F.-X. Standaert, *Cryptographic Engineering.* Springer, 2009, ch. Secure and Efficient Implementation of Symmetric Encryption Schemes using FPGAs, pp. 295–320.

[7] M. Joye, *Cryptographic Engineering.* Springer, 2009, ch. Basics of Side-Channel Analysis, pp. 365–380.

[8] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks, Revealing the Secrets of Smart Cards.* Springer, 2007.

[9] D. Naccache, "Countermeasure method in an electronic component using a dynamic secret key cryptographic algorithm," US 7206408B1, April 2007.

[10] P. Kocher, "Design and validation strategies for obtaining assurance in countermeasures to power analysis and related attacks," in *NIST Physical Security Workshop*, September 2005.

[11] T. Popp and S. Mangard, "Masked dual-rail pre-charge logic: DPA-resistance without routing constraints," in *Cryptographic Hardware and Embedded Systems – CHES 2005*, ser. Lecture Notes in Computer Science (LNCS), J. R. Rao and B. Sunar, Eds., vol. 3659. Heidelberg: Springer, 2005, pp. 172–186.

[12] D. Suzuki, M. Saeki, and T. Ichikawa, "Random switching logic: A new countermeasure against DPA and Second-Order DPA at the logic level," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 1, pp. 160–168, January 2007.

[13] Z. Chen and Y. Zhou, "Dual-Rail random switching logic: A countermeasure to reduce side channel leakage," in *Cryptographic Hardware and Embedded Systems - CHES 2006*, ser. Lecture Notes in Computer Science, vol. 4249/2006. Springer Berlin / Heidelberg, Oct 2006, pp. 242–254.

[14] T. Popp, S. Mangard, and E. Oswald, "Power analysis attacks and countermeasures," *IEEE Design and Test of Computers*, vol. 24, no. 6, pp. 535–543, 2007.

[15] M. E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A side-channel analysis resistant description of the AES S-box," in *Fast Software Encryption*, ser. Lecture Notes in Computer Science, vol. 3557. Springer, 2005, pp. 413 – 423.

[16] N. Kamoun, L. Bossuet, and A. Ghazel, "SRAM-FPGA implementation of masked S-Box based DPA countermeasure for AES," in *Design and Test Workshop, 2008. IDT 2008. 3rd International*, December 2008, pp. 74 –77.

[17] D. Canright and L. Batina, "A very compact Perfectly Masked S-Box for AES," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science, vol. 5037/2008. Springer Berlin / Heidelberg, May 2008, pp. 446–459.

[18] K. Tiri, M. Akmal, and I. Verbauwhede, "A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards," in *Solid-State Circuits Conference, ESSCIRC 2002*, 2002, pp. 403–406.

[19] K. Tiri and I. Verbauwhede, "Charge recycling sense amplifier based logic: securing low power security ics against dpa [differential power analysis]," in *Solid-State Circuits Conference, 2004. ESSCIRC 2004.*, Sept. 2004, pp. 179–182.

[20] ——, "A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation," in *Proc. Design, Automation and Test in Europe (DATE'04).* IEEE Computer Society, Feb 2004, pp. 246–251.

[21] R. Velegalati and J.-P. Kaps, "DPA resistance for light-weight implementations of cryptographic algorithms on FPGAs," in *Field Programmable Logic and Applications, FPL 2009*, M. Daněk, J. Kadlec, and B. Nelson, Eds. IEEE, Aug 2009, pp. 385–390.

[22] D. Hwang, K. Tiri, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "AES-based security coprocessor IC in 0.18-μm CMOS with resistance to differential power analysis side-channel attacks," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 781–792, Apr 2006.

[23] J. J. A. Fournier, S. Moore, H. Li, R. Mullins, and G. Taylor, "Security evaluation of asynchronous circuits," in *Cryptographic Hardware and Embedded Systems.* Springer-Verlag, 2003, pp. 137–151.

[24] M. Allam and M. Elmasry, "Dynamic current mode logic (DyCML): a new low-power high-performance logic style," in *IEEE Journal of Solid-State Circuits*, vol. 36, March 2001, pp. 550–558.

[25] F. Regazzoni, S. Badel, T. Eisenbarth, J. Grobschadl, A. Poschmann, Z. Toprak, M. Macchetti, L. Pozzi, C. Paar, Y. Leblebici, and P. Ienne, "A simulation-Based methodology for evaluating the DPA-Resistance of cryptographic functional units with application to CMOS and MCML technologies," in *Embedded Computer Systems: Architectures, Modeling and Simulation, 2007. IC-SAMOS 2007. International Conference*, July 2007, pp. 209 –214.

[26] Z. Toprak and Y. Leblebici, "Low-power current mode logic for improved DPA-resistance in embedded systems," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium*, vol. 2, May 2005.

[27] F. Mace, F.-X. Standaert, I. Hassoune, J.-D. Legat, and J.-J. Quisquater, "A dynamic current mode logic to counteract power analysis attacks," in *In The Proceedings of DCIS 2004*, 2004, pp. 186–191.

[28] R. Velegalati, "Securing light weight cryptographic implementations on FPGAs using dual rail with pre-charge logic," Masters Thesis, ECE Department, George Mason University, Fairfax, Virginia, USA, July 2009.

[29] Z. Chen and P. Schaumont, "Slicing up a perfect hardware masking scheme," in *Hardware-Oriented Security and Trust HOST, IEEE International Workshop.* IEEE Computer Society, 2008, pp. 21–25.

[30] P. Yu and P. Schaumont, "Secure FPGA circuits using controlled placement and routing," in *CODES+ISSS '07: Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis.* New York, NY, USA: ACM, 2007, pp. 45–50.

[31] *Spartan-3 Generation, FPGA User Guide*, Ug331 (v1.2) ed., Xilinx, Inc., Apr 2007.

[32] *Using Block RAM in Spartan-3 Generation FPGAs*, Xapp463 (v2.0) ed., Xilinx, Inc., Mar 2005.

[33] *Creative Uses of Block RAM*, Xilinx, June 2008, white paper 335.

[34] P. Chodowiec and K. Gaj, "Very compact FPGA implementation of the AES algorithm," in *Cryptographic Hardware and Embedded Systems - CHES 2003, Cologne, Germany*, ser. Lecture Notes in Computer Science (LNCS), vol. 2779. Springer, Sep. 2003, pp. 319–333.

[35] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, and J.-D. Legat, "Compact and efficient encryption/decryption module for FPGA implementation of the AES Rijndael very well suited for small embedded applications," in *International Conference on Information Technology: Coding and Computing, ITCC 2004*, vol. 2. IEEE, 2004, pp. 583–587.

[36] C.-W. Huang, C.-J. Chang, M.-Y. Lin, and H.-Y. Tai, "Compact FPGA implementation of 32-bits AES algorithm using Block RAM," in *TENCON 2007 IEEE Region 10 Conference*, 2007, pp. 1–4.

[37] R. Chaves, G. Kuzmanov, S. Vassiliadis, and L. Sousa, "Reconfigurable memory based AES co-processor," in *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, April 2006, p. 8.

[38] C.-J. Chang, C.-W. Huang, H.-Y. Tai, M.-Y. Lin, and T.-K. Hu, "8-bit AES FPGA implementation using block RAM," in *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, Nov. 2007, pp. 2654–2659.

[39] S. Drimer, T. Güneysu, and C. Paar, "DSPs, BRAMs and a pinch of logic: Extended recipes for AES on FPGAs," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2009, submission still under review.

[40] E. Konur, Y. Özelçi, E. Arikan, and U. Ekşi, "Power analysis resistant SRAM," in *World Automation Congress (WAC)*, July 2006.

[41] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems – CHES 2004*, ser. Lecture Notes in Computer Science, vol. 31. Berlin / Heidelberg: Springer, Aug 2004, pp. 135–152.

[42] *Advanced Encryption Standard (AES)*, National Institute of Standards and Technology (NIST), FIPS Publication 197, Nov 2001, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[43] J. Daemen and V. Rijmen, "AES proposal: Rijndael," in *First Advanced Encryption Standard AES Conference*, Ventura, California, USA, 1998, updated version from 1999.

[44] V. Fischer and M. Drutarovsk, "Two methods of rijndael implementation in reconfigurable hardware," in *Cryptographic Hardware and Embedded Systems*, ser. Lecture Notes in Computer Science, vol. 2162/2001. Springer Berlin / Heidelberg, January 2001, pp. 77–92.

[45] K. Gaj and P. Chodowiec, *Cryptographic Engineering.* Springer, 2009, ch. FPGA and ASIC Implementations of AES, pp. 235–294.

Curriculum Vitae

Shaunak Shah received his Bachelor of Engineering in Electronics Engineering degree from K.J. Somaiya College of Engineering, University of Mumbai, India in July 2006. He joined ICICI Bank as an Assistant Manager in Retail Technology department, managing multiple techno-functional projects till July 2007. From August 2007, he started working towards his Master of Science degree in Computer Engineering at George Mason University. He worked as a Teaching Assistant for various undergraduate and graduate courses at George Mason University. He was a recipient of Information Technology and Engineering Graduate Fellowship. He is a Researcher at Cryptographic Engineering Research Group (CERG). His research interest include Cryptographic hardware implementations, Side channel analysis and Efficient hardware implementations of computer arithmetic.