High Performance and Energy Efficient Architecture for PRESENT Cipher and its FPGA Implementation

M.Tech. Thesis

By

Himanshu Soni

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2019

(This page is intentionally left blank)

High Performance and Energy Efficient Architecture for PRESENT Cipher and its FPGA Implementation

A THESIS

Submitted in partial fulfillment of

the requirements for the award of the degree

of

Master of Technology

VLSI Design and Nanoelectronics

by

Himanshu Soni

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2019

(This page is intentionally left blank)

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **High Performance and Energy Efficient Architecture for PRESENT Cipher and Its FPGA Implementation** in the partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** and submitted in the **DISCIPLINE OF ELECTRICAL ENGINEERING, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from July, 2017 to June, 2019 under the supervision of Dr. Santosh Kumar Vivhvakarma, Associate Professor, Indian Institute of Technology Indore and Dr. Jai Gopal Pandey,Senior Scientist, at CSIR-Central Electronics Engineering Research Institute, Pilani.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the student with date
HIMANSHU SONI

This is to certify that the above statement made by the candidate is correct to the best of my/our knowledge.

Signature of the Supervisor of
M.Tech. thesis # 1 (with date)
Dr. Santosh Kumar Vishvakarma

Signature of the Supervisor of
M.Tech. thesis # 2 (with date)
Dr. Jai Gopal Pandey

HIMANSHU SONI has successfully given his M.Tech. Oral Examination held on **1st July, 2019**.

Signature(s) of Supervisor(s) of M.Tech. thesis
Date:

Convener, DPGC
Date:

Signature of PSPC Member # 1
Date:

Signature of PSPC Member # 2
Date:

(This page is intentionally left blank)

Acknowledgements

First of all, I would like to express my sincere gratitude to my thesis supervisor **Dr. Santosh Kumar Vishvakarma** and **Dr. Jai Gopal Pandey** for their constant support, encouragement and guidance during my Master's study and thesis work. Furthermore, I would like to express my deep gratitude to my PSPC members **Dr. Vivek Kanhangad** and **Dr. Aruna Tiwari** for their valuable suggestions and feedback. I would also like to thanks to **Dr. Srivathsan Vasudevan** to develop a zeal of practical learning in me.

I would also like to thank all the faculty members and the staff at IIT Indore for their cooperation throughout my study and thesis work. I would like to thank the Discipline of Electrical Engineering for providing all the facilities, resources, and research environment required for the completion of this work.

I would like to thank all the members of Analog and Mixed Signal Research Lab for their assistance at various levels of this work.

I am grateful to all my colleagues for their constant support, fruitful discussions, and making my stay at the institute enjoyable.

I am especially grateful to my parents, sisters and brothers for their invaluable support and strong belief in me. Their sacrifices and unbounded love motivates me to remain focused and determined, which always pushes me to strive for excellence. I dedicate my Master's thesis to my great parents for their countless sacrifices.

Dedicated to my family

Abstract

Lightweight cryptography plays a crucial role in the emerging authentication-based omnipresent computing applications in the resource-limited domain. In this paper, a high-performance, resource and energy-efficient VLSI architecture for PRESENT block cipher has been proposed and named it as p_opt-80. Here, a 16-bit data-path based architecture that supports pipelined input of the next block with the output of the current block has been used. The proposed architecture takes 42 clock cycles to compute the first block (64-bit) of data and 37 clock cycles for further blocks which makes the effective latency of 37 clock cycles. In order to compare the architecture with existing 16-bit architectures, the architecture has been implemented on a set of FPGA devices that include Xilinx Virtex-4 xc4vlx24-12ff668, Virtex-5. At 13.56 MHz RFID frequency, the proposed architecture provides a throughput of 23.46 Mbps and consumes around 72% lesser energy in comparison to existing 16-bit architectures that suits for most of the Internet-of-things (IoT) application. In this architecture, we can either have fixed input or can have varied input key. The proposed design is best suitable for modern devices for area and performance metrics. Parameters like, Throughput-per-slice and Energy-per-bit also improved to a great extent which makes it an optimized architecture for speed and energy consumption with the area. *s-box* is implemented using combinational. So it does not requires any extra memory to store input of *s-box*. It is also synthesizable for ASIC implementations.

Contents

List of Figures

List of Tables

List of Abbreviations

AES	Advanced Encryption Algoritm
ASIC	application-specific integrated circuit
CLB	Configurable Logic Blocks
CMT	Clock Management Tile
CPS	Cyber Physical Systems
DCM	Digital Clock Manager
DFS	Digital Physical Synthesizer
DLL	Delayed Locked Loop
DSP	Digital Signal Processing
FPGA	Field Programmable Gate Array
FSM	Finite state Machine
HDL	Hardware Descriptive Language
IOB	Input output Block
IoT	Internet of Things
ISE	Integrated Synthesis Environment
LUT	Look Up Table
MUX	Multiplexer
NCD	Native Circuit Description
NGD	Native Generic Description
PCF	Physical Constraint File
PLL	Phase Locked Loop
PS	Phase Shifter
RAM	Random Access Memory
RFID	Radio Frequency Identification
RTL	Register Transfer Level
SAIF	Switching Activity Interchange format
SCL	Semi-Conductor Laboratory
UCF	User Constraint File
VCD	Value Change Dump
VLSI	Very Large Scale Integration

VHDL	Very High Speed Integrated Circuit Hardware Descriptive Language
XPA	Xilinx Power Analyzer

Chapter 1

Introduction

In this new era, with the expeditious growth of cyber-physical systems (CPS) and the Internet-of-Things (IoT) technologies; devices that can work in a constrained environment is very much needed [1]. The constraints can be in terms of high-performance, low-cost, low-area, low-power, low-energy along with a sufficient level of security are needed. As these devices are being connected to the Internet, it raises security concerns [2]. Here cryptography and associated algorithms are very much sought that can provide security under constrained environment yet having a small device footprint. This is the prime reason that a broad variety of architectures are required for lightweight cryptography[3],[4]. Nowadays, the scope of this type of devices is pervasive. Lightweight cryptography can provide solutions of security in modern IoT devices that are expected to have many applications including on-vehicle devices, biomedical devices, radio frequency identification (RFID) tags, remotely-accessed devices, and acreage, etc [1,5]. The greedy demands in terms of constraints and expectations create more challenges in the task of the development of the system. So we need to do an intelligent trade-off between them according to our requirement in application and consumer expectations.

We can classify devices in two different categories - one active smart devices and the other is passive smart devices. Where active smart devices have their own power supply while passive smart devices do not have their own power supply. So in the active devices, our aim is to reduce total energy and execution time whereas in passive devices,

along with total energy and execution time, power consumption also plays a crucial role to decide the goals of the design. In the emerging IoT applications such as smart cities, connected cars, RFID Tags, etc., secure communication is necessary. For securing electronic data communication, cryptography is a crucial technique. It is a technique of store and then transmits the data in a specific form so that only an authorized person can access and process it. Basically, cryptography is used to authenticate data and protect it from unauthorized access. The cryptographic technique is used for authentication in many emerging IoT applications [1]. For the constraint limited environment in terms of energy and area requirements, hardware-based solutions which supports symmetric key concepts are very much suitable for the IoT devices [5],[2].

In the proposed work a resource and energy-efficient optimized very large scale integration (VLSI) architecture of PRESENT block cipher has been presented. The architecture supports the key size of 80-bit and the size of a data block 64-bit. The architecture follows the 16-bit data-path and support pipelining. The proposed architecture has been compared with the set of existing 16-bit architectures [6,7]. By experimental results, it has observed that the proposed architecture outperforms with the existing architectures.

1.1 Challenges

When we are going to design any system there are always some trade-offs between design metrics like low cost, low power consumption, high speed, etc. These greedy demands on the design metrics of the system create more challenges in the task associated with the development of systems. So we need to do an intelligent trade-off between them according to our requirement in application and consumer expectations. So there are many pillars of design attributes. As a cipher is proposed here which is basically an encryption algorithm so security is one of the concern. As our target devices are mainly related to IoT. So an adequate level of security is sufficient for our device. Like if our code can be broken in several hundreds of years that will be sufficient for us. while designing cipher for the security-sensitive applications, like military applications code should not be broken for several thousands of years. As in IoT devices, power

consumption is also a major concern so lightweight cryptography is desirable instead of conventional methodology. Lightweight cryptography is basically the cryptographic techniques for power-constrained devices. So low power consumption is also one of the points which we should keep in our mind. Especially for passive devices, this is a very crucial requirement as they do not have their own power supply. Low cost and less occupied area are the greedy demands of modern applications. On the modern FPGAs, a slice contains more flip flops(FFs) and LUTs, etc. So on modern FPGAs design can be implemented using fewer slices which reduces the occupied area. High speed of encryption is also a desirable attribute in IoT devices. So basically in order to design cipher for the IoT devices desired attributes are high speed, less occupied area, low cost, low power, and energy consumption along with an adequate level of security. The three major pillars are as shown in Fig. 1.1 are as follows.

(a) High Speed

(b) Low Area/ Low Cost

(c) Adequate level of Security

To increase security there are mainly two ways either increase the number of iterations to encrypt the data or add more circuitry in the architecture. So if the number of iterations increases are increased to provide high security then execution time increase which will lead to a decrease in the speed of encryption. On the other hand, if more circuitry is added to increase security then this we lead to an increase in area. So by that trade-off of security with area and speed can be understood. By some methods like unrolling the loop and parallel processing speed can be increased but at the same time area occupied by the device will be increased. On the other hand in iterative methods, the same block is being reused multiple times by which area is considerably decrease but at the same time, it will reduce the speed of operation.

So basically an intelligent trade-off between design metrics is required according to our application. An optimal architectural solution can be obtained by an intelligent trade-off between design metrics.

Figure 1.1: Challenges in the Design of Cipher.

1.2 Existing Work

The elucidation of different architectures like round-based, parallel and serial for PRESENT block cipher has been provided in [8]. Additionally, an analysis of the architectural design space exploration on Spartan-III xc3s400 device is provided in [9]. Two different implementations based on random acess memory (RAM) of PRESENT cipher have been given in [10] in which the main motive was to reduce slice count by utilization blocks of existing RAM in FPGA devices to store internal states. One of the implementations with 64-bit data-path has been given in [6], which occupies 74 slices in the xc6slx16-3csg324c FPGA device. Here, a throughput of 429.83 Mbps is acquired at a maximum clock frequency of 221.63 MHz along with the latency of 33 clock cycles. Likewise, an architecture with 64-bit data-path consumes 87 slices on the xc5vlx50 FPGA device in [11]. Here, a throughput of 341.64 Mbps is acquired at a maximum frequency 221.64 MHz along with latency of 47 clock cycles.

A high-performance and resource-efficient architectures for the PRESENT block cipher has been given in [12]. Compare iterative and serial architectures of Clefia, advanced encryption algorithm (AES) as well as PRESENT block cipher [11]. Discussed seven different architectures of PRESENT in [7]. These Seven different architectures are based on different design goals like area, power consumption, and speed, etc. Beside that both proposed architectures in [7] and [6] are using 16-bit data-path. So it is very much suitable for comparison with our architecture. An 8-bit data-path based architecture is given in [4] that takes 49 clock cycles in order to get an output of each 64-bit block. The architecture has been implemented on Xilinx Virtex-5 along with an application-specific integrated circuit (ASIC) implementation in semi-conductor laboratory (SCL) 180 nm technology. An implementation of the PRESENT block cipher has been given in [13].

1.3 An Overview of the PRESENT Block Cipher

The algorithm involves typically four functions, which are: *key scheduling*, *add_round_key*, *s-box layer*, and *permutation layer*. These functions are drawn in Fig. 1.2.

Input for the PRESENT block cipher is a block of a size of 64-bit which is supported by two input key lengths of 80-bit and 128-bit [3]. Because 80-bit provides that much level of security which is required by intended applications, so it is recommended to use an 80-bit key for low security-based systems like IoT devices and RFID tags [3]. Generally, two types of network are used in various encryption algorithms. These are a Feistel network and substitution-permutation network. In the PRESENT algorithm substitution operation is followed by permutation operation and combinedly those operations are made up of 31 encryption rounds along with *XOR* operations. Each round is composed of a *XOR* operation, which needed to instigate a round key K_i for $0 \leq i \leq 31$. Here the last round is used for the post-whitening operation. Furthermore, there is a non-linear bitwise substitution layer followed by a linear *permutation layer-based operation*. Substitution layer is formed by a parallel arrangement of 4×4 *s-box*es.

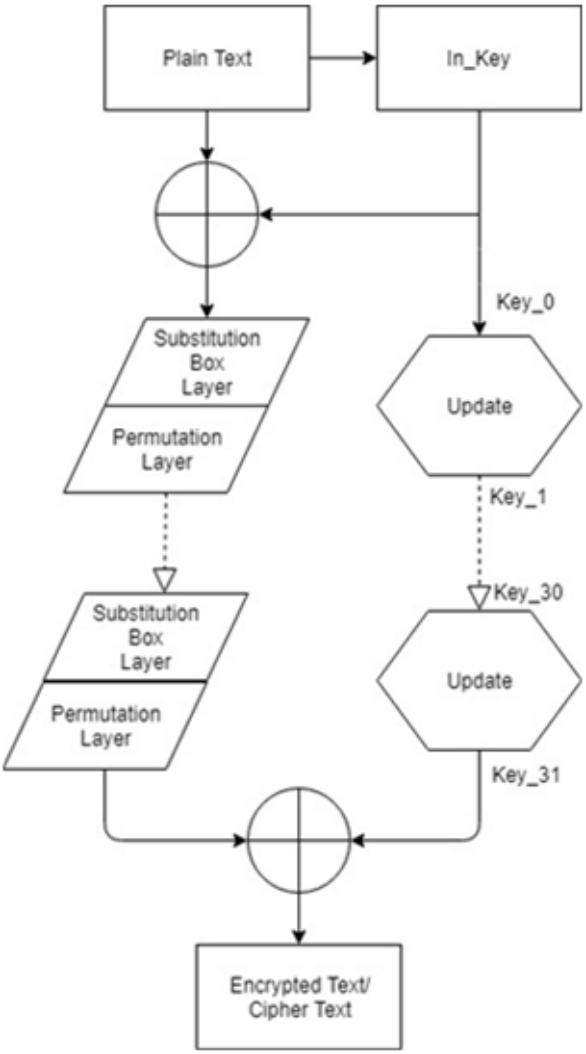

Figure 1.2: An overview of the PRESENT algorithm.

1.4 Organization of thesis

The rest of the thesis is organized as follows:

Chapter 2 presents the detailed description proposed architecture of PRESENT block Cipher with the methodology and our contribution.

Chapter 3 presents a detailed description of the used tools and devices in our work. Chapter 3 presents a description of the coding style used in this work. Chapter 4 presents the results and discussions section. In which, we have presented results of the proposed method. Chapter 5 presents the detailed resource utilization results at multiple frequencies for two different settings. Chapter 6 presents a comparison

of the proposed architecture with a set of existing architectures. Chapter 7 presents the conclusion of the whole work. The directions for future research work are also provided in this chapter.

1.5 Summary

In this chapter, we have discussed the need of the Lightweight Algorithm with increase in the IoT or CPS devices. We also tried to understand the reason behind an increase in the number of such devices. We also tried to understand the security concerns for these devices. After that challenges to provide security by keeping in the mind about the situation of the current market and it's demands. We have also discussed the existing work in this area to understand the need and scope of improvement. These work also useful to grab some novel ideas for the proposed architecture. As we choose PRESENT block cipher for the improvement in the existing architecture so we have tried to see it briefly. This picture of the overview of the PRESENT architecture is very much helpful to understand the flow of this algorithm and the reason behind the evolution of PRESENT algorithm.

Chapter 2

Proposed Architecture for the PRESENT Block Cipher

In this section, an optimized architecture for performance and energy with Area parameters is proposed for the PRESENT block cipher. The architecture works for an 80-bit key length that provides a sufficient level of security for IoT applications. In order to compare the architecture with existing 16-bit architectures, the architecture has been implemented on a set of FPGA devices that include Xilinx Virtex-4 xc4vlx25-12ff668, Virtex-5 xc5vlx50t-3ff11336, Spartan-3 xc3s200-5ff256 and xc6slx16-3csg324. The following section provides details of our contributions.

2.1 Our Contribution

The Proposed architecture with 80-bit key size and 16-bit data-path has latency of 37 clock cycles and 23.46 Mbps throughput at 13.56 MHz frequency which are the best in available 16-bit architectures of the PRESENT block cipher in our knowledge. Moreover, the best results in terms of area and performance are obtained on Xilinx Spartan-6 xc6slx16-3csg324. In addition to the number of utilized slices, maximum operating frequency, throughput at maximum frequency and *throughput-per-slice* at 13.56 MHz frequency is also been improved in comparison to [6]. On the other three FPGA devices better timing parameters are obtained like latency, maximum frequency, throughput

at maximum frequency, throughput at frequency 13.56 MHz and *throughput-per-slice* than other existing 16-bit architectures of the PRESENT block cipher.

It has been observed that *throughput-per-slice* at the RFID frequency(13.56 MHz) is much improved. It is due to the fact that the combined effect of area and performance parameters have been improved in compared to a set of existing 16-bit architectures. In addition, it has also been observed that the performance, resource utilization, energy, and *energy-per-bit* are better in Xilinx Virtex-5 xc5vlx50t-3ff11336 and Xilinx Spartan-6 xc6slx16-3csg324 which uses LUT-6. Thus, the proposed architecture performs better in LUT-6 based FPGA device as compared to LUT-4 based FPGA devices. Proposed architecture consumes lesser energy and *energy-per-bit* than [7].

By experimental results, it has been observed that in comparison to the architecture of [7], the proposed architecture with 80-bit key consumes 31.34% more FPGA slices and there is a gain of 237.70% in throughput on Xilinx Virtex-5 xc5vlx50t whereas it consumes equal number of slices and gain of 333.26% in throughput on Spartan-6 xc6slx16-3csg324. Similarly, in comparison to [6] it consumes 30.43% less slices with a gain of 270.04% in throughput on device Xilinx Spartan-6 xc6slx16-3csg324. In our architecture, gain of around 257% than [7], [6] has been observed in throughput at 13.56 MHz. Proposed architecture consumes around 72% less energy than [7] on all four FPGA devices. Some of the key aspects of our work are as follows.

(i) The selected 16-bit data-path provides an optimal trade-off between used hardware/number of IO pins/area and latency/speed.

(ii) An implementation of pipelining for input and user key is explained in the latter part of this paper.

(iii) The user has the flexibility to either fix the input key or change it for each subsequent blocks of data. However, effective latency of 37 clock cycles has been obtained, irrespective of fix key or changed key.

Proposed architecture as shown in Fig. 2.1 consists of three main components: encryption engine, *key scheduling* and controller. Controller further consists of a counter and some minor parts for generating different keys for each round and to encrypt the

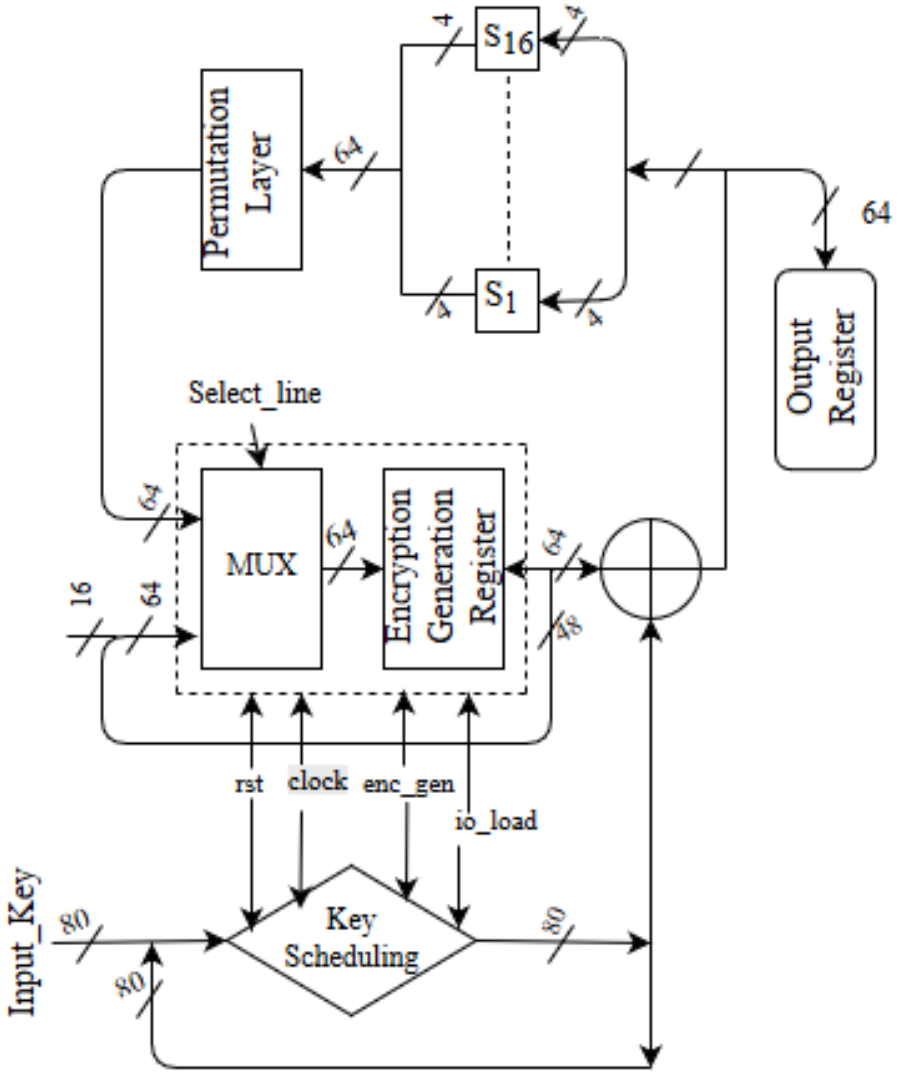

Figure 2.1: Proposed architecture for PRESENT cipher.

data-block. The data-block contains 64-bit and 80-bit input-key. The architecture is based on 16-bit data-path.

The data-path of the given architecture consists of a set of flip-flops, registers, *XOR* gates and multiplexers. *s-box* layer is simply a standard mapping between input and output data of *s-box*. In this work, 16 4×4 *s-box*es have been used in order to process 64-bit data-block. The *permutation layer* is a simple operation of bit-transposition, which requires only simple wiring. So *permutation layer* occupies 0 GE. The major building blocks of the architecture have been explained in subsequent subsections.

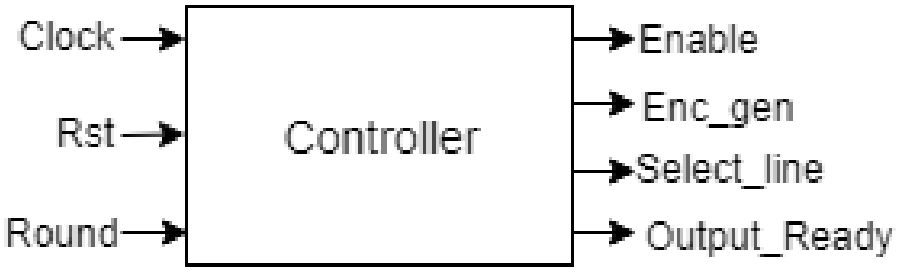

Figure 2.2: Controller

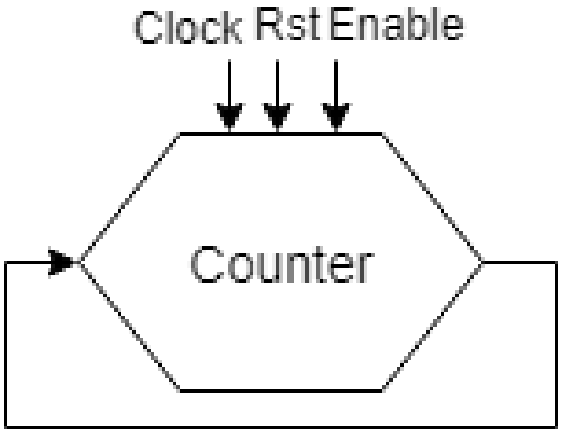

Figure 2.3: Counter for the Proposed Architecture

2.2 Datapath of the Proposed Architecture

The proposed architecture is based on 16-bit data-path constituents of a 64-bit *encryption register* which is competent to encrypt data with an 80-bit Key Register which is part of *key scheduling* block. An 80-bit key register is used to store keys of intermediate rounds. There are two phases in our architecture which are loading phase and computation phase. In order to switch data between these phases, one 64-bit multiplexer along with one 80-bit multiplexer are used. Our data-path consists of (16 *s-box*es) and one *s-box* for the *key scheduling* process. One *permutation layer* is also used in the process of encryption. Along with these components, some additional components, one 2-bit up counter, one 5-bit up counter, One 64-bit *XOR* gate, and 5-bit *XOR* gates are also used. The plaintext is loaded after reset clock cycle (in second clock cycle) and in next clock cycles multiplexer is used to switch the data. According to FSM as shown in Fig. 2.5, multiplexers get the control signal and accordingly gets

switched in loading or computation phase. So effectively data is loaded in four clock cycles which that is also shown in the state diagram of Fig. 2.5. Afterward, in the next 31 clock cycles, all intermediate states are computed. Data is available at the *encryption register* which is *XORed* with key of intermediate round. Then, this mixed State is passed to the *s-box* layer(which consists of 16, 4×4 multiplexers) then this 64-bit data is concurrently provided to the *permutation layer*. Subsequently, data is passed to the *encryption register* through a multiplexer as shown in Fig. 2.1.

In the last clock cycle, the encrypted text is available at the output register. It takes a total of 42 clock cycles to encrypt the first block of data and subsequently takes 37 clock cycles for subsequent blocks of data which is explained in Fig. 2.5. The execution of the round keys performs on-the-fly mode.

2.3 Design Strategies for s-box Layer

To achieve good performance, low-cost and area-efficient design of *s-box* layer, various approaches can be used. Two major design approaches for *s-box* layer are: RAM-based approach [10], and combinational logic-based approach [9], [14]. In RAM-based design, 16×4 bit size of memory is required by single *s-box*. In order to get an adequate level of speed of the encryption operation, on an average 16 *s-box*es are required. So, the requirement of memory lifted to $(16 \times 4) \times 16$ bits, which equal to 1-Kb, which is a considerably large amount of memory. So an alternative is required in order to reduce memory requirement as well as to minimize the delay and area requirements. In order to achieve these requirements, the combinational logic technique has been selected, because it provides further simplification of factors by which the size of *s-box* can be minimized and also there is no memory requirement. Moreover, it is also synthesizable to ASIC implementations so it can also be used in ASIC implementation.

2.4 Architecture for Key Scheduling Process

The key processing unit performs *on-the-fly* key generation along with each round. In order to store the intermediate round keys, an 80-bit key register is used. The first

leftmost 64-bit of the key register is *XORed* with the intermediate state of the cipher. At the next clock after *reset*, the 16-bit input key is loaded at a time and it has been stored into an 80-bit key register, which is shown in Fig. 2.4. The following three steps are performed for *key scheduling* operation.

(a) The output of the 80-bit key register is rotated to the left by 61 bits.

$k_{79}k_{78}k_{77}.....k_2k_1k_0 \rightarrow k_{18}k_{17}k_{16}....k_{20}k_{19}$

(b) The First 4-bit is passed through *s-box*.

$k_{79}k_{78}k_{77}k_{76} \rightarrow \mathrm{S}[k_{79}k_{78}k_{77}k_{76}]$

(c) Counter value is *XORed* with 5-bit of key.

$k_{19}k_{18}k_{17}k_{16}k_{15} \rightarrow k_{19}k_{18}k_{17}k_{16}k_{15} \oplus \text{round\_counter}$

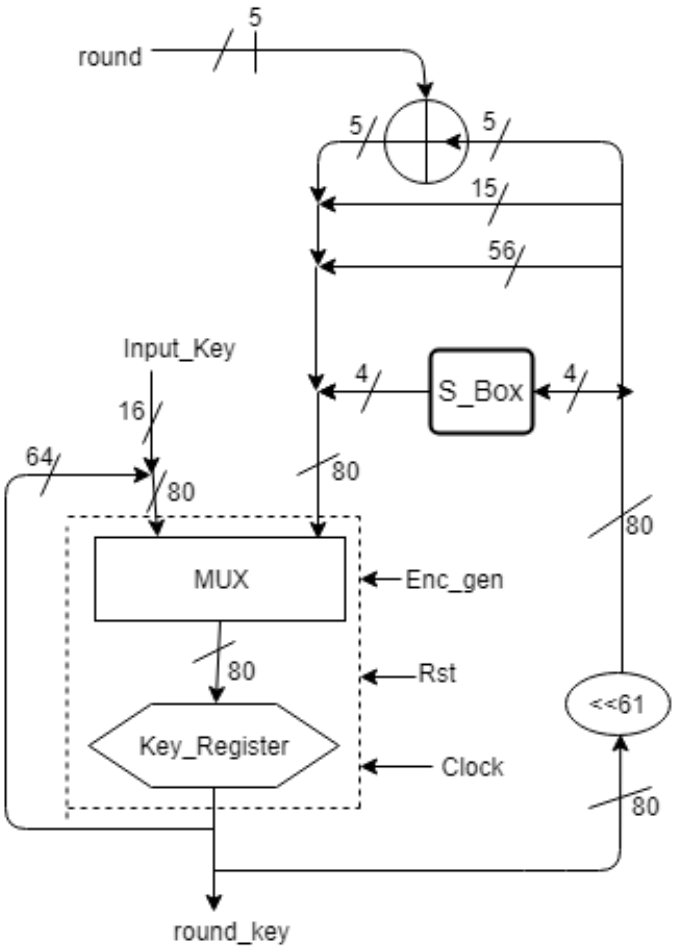

Figure 2.4: The key-scheduling process in the PRESENT cipher with 80-bit input key.

2.5 Finite State Diagram of The Proposed Architecture for Encryption of Data

A controller, as shown in Fig. 2.5 is designed to generate various control signals which are required to implement key generator and encryption generation block. There are six stages of FSM in order to implement the algorithm which works according to control signals. Controller generates four control signals which are *io_load*, *enc_gen*, *enable* and *out_ready*. Here the working of the controller is described. There are two phases of the controller. For the first block, it takes data-path as S0, S1, S2, S3, S4, S5. So for the first block of data, it uses six states. Input key can be loaded input key whenever encryption generation is inactive.input can be inserted when encryption generation is inactive and *io_load* is active. State S0 is reserved to reset the controller and as soon as *io_load* goes high, FSM goes in State S1. Whenever *io_load* is active, then it enables *encryption generation register* to store first 16-bit of data block. So it fetches 16-bit of input data as well as 16-bit of the input key in each clock cycle. It stays in state S1 for four clock cycles. By this time complete block of data(64-bit) is inserted but as input key is of length 80-bit, so one more clock cycle is required to insert remaining 16-bit of the input key. The remaining 16-bit of input key needs to be stored, therefore, by the end of state S2, input_key(80-bit) and input(64-bit) have been processed. For this, after state S2 the first block of data along with input key is already inserted and in order to enable 5-bit counter, *enable* signal is activated in the state S2.

After one clock cycle, the state switches to S3 and *enc_gen* goes to high. So it starts the encryption operation. In S3 still, the signal *enable*='1' which keeps counter active for the next 31 clock cycles in order to perform encryption for that duration. Then state then goes to S4 where *out_ready* is activated. After one clock cycle the state switches to S5 where *io_load* and *out_ready* are high here for four clock cycles so complete 64-bit of output is available through the *Output Register*. Here, the concept of pipelining has been used in state S5. The next block of 64-bit data and 64-bit of 80-bit *input_key* have been taken in four clock cycles when output is generated and

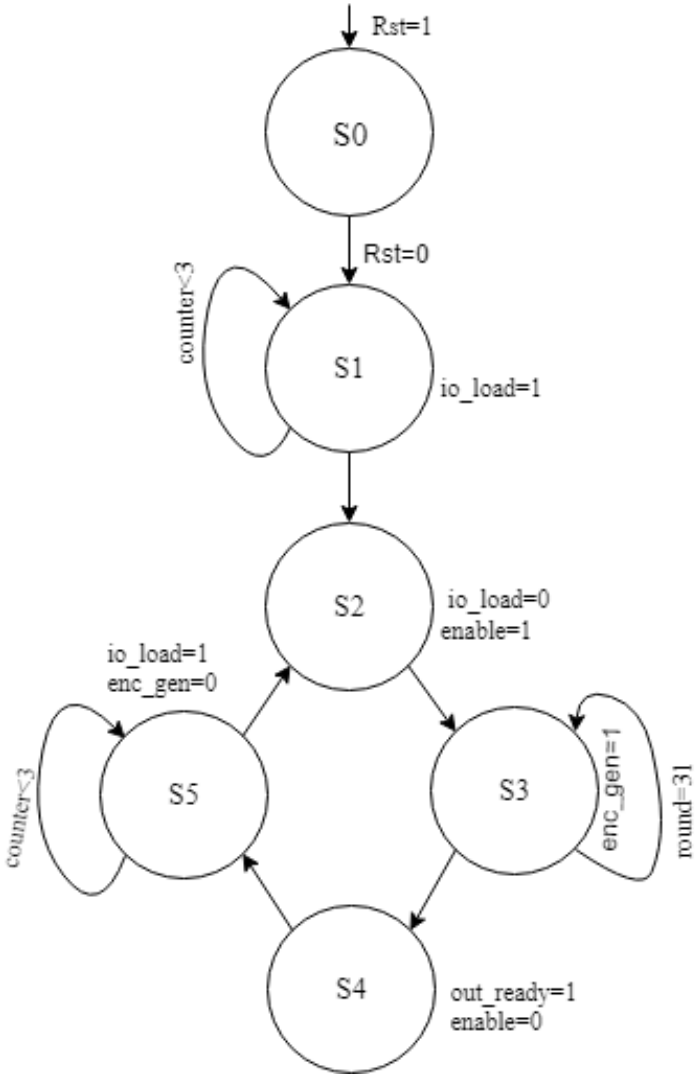

Figure 2.5: Finite state machine of the proposed controller.

then instead of moving to state S0, the state switches to S2 and the remaining 16-bit of the input key is stored in state S2. As shown in the FSM of the proposed architecture in Fig. 2.5, it follows same path (S5, S2, S3, S4, S5) for successive data-blocks. From that, 5 clock cycles of state S0 and S1 are saved. So the proposed architecture requires 42 clock cycles to generate an output of the first block of data and 37 clock cycles for the successive data-blocks.

2.6 Summary

This chapter is the heart of our work. Basically, a novel architecture has proposed in this chapter. In the section named our contribution, we have discussed some crucial advancements and results in the existing set of 16-bit architectures of the PRESENT block cipher. Further, we tried to understand datapath in order to understand dataflow in the proposed architecture. *s-box* is one of the crucial blocks in the architecture. So we discussed the strategies which we used during the implementation of the cipher. Which is basically need less memory and less area has been occupied by the architecture with the help of these strategies. After that architecture for the *key scheduling* process has been discussed. In the last section of this chapter, the most important part of this project has discussed in which we actually plan the flow of our architecture by keeping timing in the mind. Which is represented in terms of FSM which describes switching between the states according to the control signals. Pipelining concept is basically introduced in FSM.

Chapter 3

Used Tools and Devices

We used many tools for the many steps of design and analysis. So in this section, we have introduced those tools with there exact use in our project work.

3.1 Used Tools

(i) Xilinx Vivado

(ii) Xilinx Integrated Synthesis Environment (ISE)

(iii) Mentor Questasim

(iV) Xilinx Power Analyzer (XPA)

3.1.1 Xilinx Vivado 14.7

Xilinx Vivado is a tool provided by Xilinx which provides us a text editor to write code in hardware descriptive language (HDL). Simulation can also be done using this tool. Further, it also facilitates us to observe the resource utilization and performance parameters of our design for a particular FPGA Device.

In our project, Xilinx Vivado 14.7 is used for the purpose to write code using VHDL and Verilog HDL in a text editor. Our code is written in a mixed environment of VHDL and Verilog code. A hierarchy is followed by this code. All the components are written in Verilog HDL whereas the code of main design and test bench are written in VHDL.

Components have further instantiated in the main design. After that, Simulation is also done using the same tool.

3.1.2 Xilinx Integrated Synthesis Environment

This tool is also a product of Xilinx. Xilinx ISE is very much similar to Vivado. Basically, we used this tool to obtained resource utilization and performance parameters of the RTL code of proposed architecture. As synthesis and implementation results may differ because after synthesis also there are a lot of steps in the implementation. So we capture the results after synthesis as well as after implementation each of the FPGA devices which has used in this work. According to our knowledge, ISE is designed for the FPGAs of 6th series and older whereas Vivado is designed for 7th series and newer one.

3.1.3 Mentor Questasim

Questasim is a tool provided by Mentor Graphics which provides us a Text Editor to write code in Hardware Descriptive Language and Simulation can also be done using this tool. Moreover, it is useful to create a SAIF(Switching Activity Interchange format) and VCD(Value Change Dump) files. Which plays a crucial role in power analysis. these are Optional files to calculate power but required to enhance the accuracy of power analysis. These files have specific switching information (frequency information, toggle rates, and signal rates). This information is very helpful for accurate power estimation.

In order to see the effect of switching information in power estimation. We made a file off 100 input vectors then wrote an RTL code for test bench and obtained output file for the corresponding input vectors. To observe switching activity between this process a SAIF file was generated using the command line in Questasim.

During work, we observed that Vivado stops simulation after 2290 ns but as we need to generate output almost 100 vectors to get more accurate power estimation. So it consumes more time. So to get the output of 100 vectors we used Questasim.

3.1.4 Xilinx Power Analyzer

This is a very powerful tool which is provided by Xilinx. Xilinx power analyzer is a very important tool to estimate the power of the register-transfer level (RTL) code of our design for a particular board. We estimated power for RTL code on each board which is used in our project. After power estimation, static and dynamic powers are explicitly provided by this tool.

Along with SAIF file we have to NCD(Native Circuit Description) and PCF(Physical Constraint File) files have also be given as an input to the Xilinx power analyzer to get better power estimation. These files have been obtained from implementation.

NCD file is basically obtained from NGD(Native Generic Description) file from the mapping process which will be further modified after the place and route step of implementation. Mapping process basically maps the logic of NGD file into CLBs(Configurable logic blocks), buffers and IOBs. Clock information from the UCF (User Constraints File) are reported in the PCF(physical constrained file).

3.2 Devices

We have performed the resource utilization on the four FPGA devices from the different families of FPGAs including Spartan III, Spartan VI, Virtex IV, Virtex V. We analyzed our design on the different type of devices, different speed grades and different package types. In order to get a fairer comparison with an existing set of 16-bit architectures of the PRESENT block cipher, we performed this analysis on the same devices, with same speed grades and package types. We analyzed the proposed architecture on four different FPGA devices. Two of them are LUT-4 based devices and the other two are LUT-6 based devices. The devices from Spartan III and Virtex-IV family of FPGAs uses LUT-4 whereas devices from Virtex V and Spartan VI family are uses LUT-6 for implementation purpose.

Clock Management is a very important task in any circuit. To minimize these unwanted effects including clock skew and Jitter, good management of clock is needed. So there are two types of blocks used to manage the clock of an FPGA device.

Before the implementation of the design, it is really an important task to decide the device for implementation of the design. There are a lot of factors including the number of input pins, number of output pins, memory requirement, maximum frequency supported by that device for your design, the frequency of your interest according to application and power consumption, etc. plays a crucial role in order to decide the device on which design can be implemented
The FPGA devices under evaluation are as follows.

(i) xc3s200-5ff256

(ii) xc6slx16-3csg324

(ii) xc4vlx25-12ff668

(iii) xc5vlx50t-3ff11336

3.2.1 xc3s200-5ff256

This FPGA belongs to one of the basic family of FPGA named Spartan III. Unlike other three FPGA devices used in work. Spartan III has an only single platform for the device. In 4 input LUTs are used in this device. All of the LUTs used in this device are based on RAM-based which can use the latches and flip flops for the purpose of the implementation of any function of the design. Like other FPGA devices, CLBs of xc3s200-5ff256 are Reconfigurable.

To provide and to manage the clock, a DCM block embedded in the design.DCM is consists of Digital Frequency Synthesizer (DFS), Digital Frequency Synthesizer (DFS), status logic and phase shifter. DCM has basically reduced the clock skew which enhances the functionality of the design. Reduction in the clock skew may decrease the hold time requirements in the design. At the same time reduction in clock skew may lead to setup violation but that can be resolved by using low threshold cells at the time physical design which will reduce the data arrival time between to flip flops. DFS in the DCM provides a wide range of frequencies to the design by the multiply the frequency with some different factors. Phase shift in the clock is provided by Phase Shifter.

The basic details of the device are given in Table 3.1.

Table 3.1: Basic details of the device xc3s200-5ff256

Basic Details of the FPGA device xc3s200-5ff256	
FPGA Family	Spartan III
Technology Node	90 nm
Device Type	xc3s200
Package Type	ff
Total Number of Pins	256
Speed Grade	5
Type of used LUTs	LUT-4
Number of Slices in Each CLBs	4
Clock Management Device	DCM

Some important details of the device xc3s200-5ff256 are as follows.

(i) There is a total of 480 CLBs on FPGA device xc3s200-5ff256 device which are arranged in an array of size. 24×20.

(ii) Each of the 4 slices contains two LUTs, two storage elements, multiplexers (MUXs), arithmetic gates and carry logic.

(iii) Maximum 173 ports can be used as IO ports.

(iv) It contains 30 Kb distributed RAM and 216 Kb block RAM.

(iv) Each of the 4 DCMs, are used to control the clock used in the design.

(v) A logic cell contains LUT-4 and D flip flop.

(vi) 4320 Equivalent logic cells are present this FPGA device.

(vii) It contains 250k system gates.

3.2.2 xc4vlx25-12ff668

This FPGA belongs to one of the basic family of FPGA named Spartan IV. Devices of the Virtex IV FPGA family are available in three different platforms named LX, SX, and FX. SX is basically used for the Digital Signal Processing (DSP) systems whereas FX is effective for the design of embedded systems. The devices of LX families are

used for high-performance logic designs. That is a reason behind the selection of LX family FPGA design for our design. In 4 input LUTs are used in this device. All of the LUTs used in this device are based on RAM-based which can use the latches and flip flops for the purpose of the implementation of any function of the design. The basic details of the device are given in Table 3.2.

Table 3.2: Basic details of the device xc4vlx25-12ff668

Basic Details of the FPGA device xc3s200-5ff256	
FPGA Family	Virtex IV
Technology Node	90 nm
Device Type	xc4vlx25
Package Type	ff
Total Number of Pins	668
Speed Grade	12
Type of used LUTs	LUT-4
Number of Slices in Each CLBs	4
Clock Management Device	DCM

Before implementation of the design it is really an important task to decide on which our design should be implemented. There are a lot of factors including number of input pins, number of output pins, memory requirement etc. plays a crucial role in order to decide the device on which design can be implemented.
Some important details of the device xc5vlx25-12ff668 are as follows.

(i) There are total 2688 CLBs on FPGA device xc5vlx25-12ff668 device which are arranged in an array of size. 96×28.

(ii) Each of the 4 slices contains two LUTs, two storage elements, MUXs, arithmetic gates and carry logic.

(iii) There are 11 IO banks available in this device.

(iv) Maximum 448 ports can be used as IO ports.

(v) It contains 168 Kb distributed RAM in the CLB itself and 1296 Kb block RAM.

(vi) Each of the 8 DCMs, are used to control the clock used in the design.

3.2.3 Xc5vlx50t-3ff11336

This FPGA belongs to one of the basic family of FPGA named Spartan IV. Devices of the Virtex IV FPGA family are available in three different platforms named LX, SX, and FX. We chose LX series for the same reason which has been mentioned earlier in the section of the Virtex IV FPGA device. In 4 input LUTs are used in this device. We used the LX family as it is used to design logic for high-performance applications. The LUTs used in this device are based on RAM-based which can use the latches and flip flops for the purpose of the implementation of any function of the design. The basic details of the device are given in Table 3.3.

Table 3.3: Basic details of the device xc5vlx50t-3ff11336

Basic Details of the FPGA device xc5vlx50t-3ff11336	
FPGA Family	Virtex V
Technology Node	65 nm
Device Type	xc5vlx50t
Package Type	ff
Total Number of Pins	11336
Speed Grade	3
Type of used LUTs	LUT-6
Number of Slices in Each CLBs	2
Clock Management Device	CMT

Before implementation of the design it is really an important task to decide on which our design should be implemented. There are a lot of factors including number of input pins, number of output pins, memory requirement etc. plays a crucial role in order to decide the device on which design can be implemented.
Some important details of the device xc5vlx50t-3ff11336 are as follows.

(i) There are total 3600 CLBs on FPGA device xc5vlx50t-3ff11336 device which are arranged in an array of size 120×30.

(ii) Each of the 4 slices contains 4 LUTs and 4 flip flops.

(iii) There are 15 IO banks available in this device.

(iv) Maximum 480 ports can be used as IO ports.

(v) It contains 480 Kb distributed RAM in the CLB itself and 2160 Kb block RAM.

(vi) Each of the 6 clock management tiles (CMTs), are used to control the clock used in the design.

3.2.4 xc6slx16-3csg324

This FPGA belongs to one of the basic family of FPGA named Spartan VI. Devices of the Spartan VI FPGA family are available in three different platforms named LX and LXT. In 4 input LUTs are used in this device. The basic details of the device are given in Table 3.4.

Table 3.4: Basic details of the device xc6slx25-3csg324

Basic Details of the FPGA device xc6slx16-3csg324	
FPGA Family	Spartan VI
Technology Node	45 nm
Device Type	xc6slx16
Package Type	csg
Total Number of Pins	324
Speed Grade	3
Type of used LUTs	LUT-6
Number of Slices in Each CLBs	2
Clock Management Device	CMT

Before implementation of the design it is really an important task to decide on which our design should be implemented. There are a lot of factors including number of input pins, number of output pins, memory requirement etc. plays a crucial role in order to decide the device on which design can be implemented.
Some important details of the device xc6slx16-3csg324 are as follows.

(i) Each slice contains 4 LUTs and 8 flip flops.

(ii) There are 4 IO banks available in this device.

(iv) Maximum 232 ports can be used as IO ports.

(v) It contains 136 Kb distributed RAM in the CLB itself and 1296 Kb block RAM.

(vi) Each of the 2 CMTs, are used to control the clock used in the design.

We observed that Spartan 3 and Spartan 6 FPGAs have DCM while modern FPGAs have CMT. So it is very important to understand these two blocks in order to have a good knowledge of Clock Management in FPGAs.

3.3 Digital Clock Manager

DCM is a block which controls the clock in the circuit. Mainly it controls clock frequency, clock skew and phase shift in the clock. DCM has a Delayed Locked Loop (DLL), which is a digital control system, in which feedback is used to maintain the characteristics of the clock in spite of normal variations in the voltage and temperature. The major task of the DLL is to reduce clock skew. The major uses of DLLs are shown as follows.

3.3.1 Elimination of Clock Skew

Clock Skew is basically the problem in which the clock reaches on different parts of the circuit at different times. It may lead to a timing violation problem. It is completely undesirable. In DCM, an Output clock signal is aligned by the feedback clock signal to cancels out distribution delays that may lie in the path originating from the output clock of DCM to the feedback of DCM.

3.3.2 Frequency Synthesis

By dividing or multiplying the input clock frequency with some different factors a wide range of output clock frequencies can be generated by this block which is embedded in DCM.

3.3.3 Phase Shifting

There is a dedicated block present in the DCM named phase shifter, which is capable to shift all output clock signal of DCM with the input clock signal of DCM block.

3.4 Clock Management Tile

CMT has improved the management of DCM. As one itself contains two DCMs along with one PLL block.

A phase-locked loop (PLL) is an electronic circuit with a voltage-driven oscillator that continuously adjusts the output clock to match the input clock frequency. PLLs are immune to noise, which helps the CMT more immune to noise than DCM. CMT have following advantages over DCM.

(i) Provides flexible clocking

(iI) Reduce the noise.

(iII) provides clocking with low jitter.

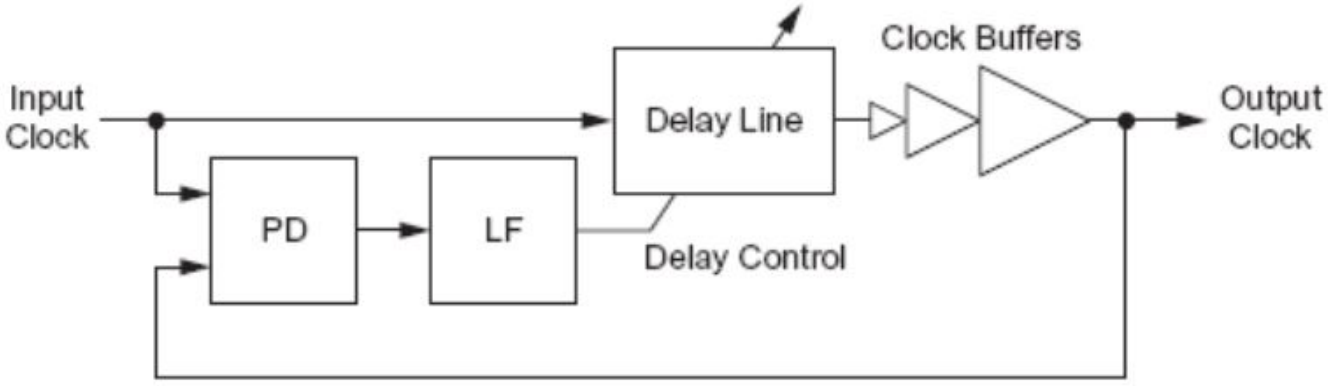

Figure 3.1: DLL

3.5 Summary

This chapter deals with the tools and devices used in our work. Moreover, it briefly, tells about the general uses of these tools. Some tasks can be completed by more than one tools, in that case, it emphasizes on the specific uses of these tools for which they are used. Uses of these tools vary from writing HDL description to the analysis of resource utilization, performance, and power. This chapter also gives an insight into the properties of the devices. The details of available different resources for a particular

FPGA device are also given in this chapter. This chapter develops an understanding of the used FPGA devices which may help us to decide the FPGA device on which our application will perform better according to requirements of the application. Sometimes design is mainly focused on a particular goal, with the knowledge of the device one can wisely choose a device for the implementation of design on that FPGA device.

Chapter 4

RTL Coding Style

RTL coding Style is adopted by us to write the code. Previously it means that the coding style in which data transfer occurs from register to register. But nowadays it's meaning changes. According to modern definitions, any code which is synthesizable is known as RTL style code or in simple words, one it can be defined as the code which can be implemented is known as RTL style code.

Here we will discuss some scenarios where the code is not synthesizable or simulation and synthesis results differs. In short, we will discuss it for Verilog. Similarly, it is also applicable for VHDL for equivalent blocks like process block in VHDL is similar to always block in Verilog HDL. The major scenarios are as follows:

4.1 Issue in Assignment in the Always block

Every signal inside an always block must be assigned in for each possible path. Otherwise, there is no issue in simulation but in real gates, this behavior is not possible unless latches used to retain values. For the better understanding of the scenario, a piece of code is presented here.

always @(p or q or r or s or condition)

if(condition)

f = p + q;

else

g = r + s;

end

here true condition g is not assigned and for false condition, f is not assigned which creates an issue in synthesis.

4.2 Partial Sensitivity Lists

If incomplete sensitive list used inside an always block then synthesis will assume that they are included and generates correct synthesized netlist whereas actual or partial sensitivity list is considered by the simulator. Which misleads the behavior of the simulation. This situation leads to a conflict between simulation and synthesis. For a better understanding of the scenario, a piece of code is presented here.

always @(p or q or r or s)

if(condition)

f = p + q;

else

f = r + s;

end

here the condition is not in the sensitivity list which creates the problem.

4.3 Issue of Multiple Drivers

This situation occurs when the same signal is driven by multiple always block. In this case, if any two always block assign a different value then ambiguity may be generated in the signal assignment. Which leads to a problem.

always @(posedge clock)

begin

if(rst)

value <= 1'b1;

end

```
    always @(posedge clock)
begin
if(flag)
value <= n;
end
```

4.4 Problem due to Default Constant Width

When constants are used without specifying any particular width during assignment then by default they expanded in 32 bit. It will lead to a problem when it is added with some other signal of different width. To resolve this issue we have to define its width during the assignment. for the purpose of better understanding. let's try to understand it with the following piece of code.

```
wire [5:0] bat;
wire [5:0] ball;
    assign bat = ball + 1;
```

4.5 Unsynthesizable Construct

The "initial" statements are not synthesizable, there is no netlist generated corresponding to initial statements. So we should avoid the initial statements in the RTL code. Let us have a real-time example. Majority of chips have reset signal. So we should use that signal to reset our design instead of resetting some values of the design inside an always block. In order to get a clear picture. A piece of code is given as follows:

```
always @(posedge clk) begin
if(reset) begin
// Reset all state
value <= 1'b0;
end
else
```

```
begin
// functionality of the design end
end
```

4.6 Summary

This is a very crucial chapter when it comes to the implementation of the circuit. It awares us about some situations in the code where either simulation or synthesis failed along with conflicts between netlist generated by these steps. We have to avoid these pieces of code in the design so that the successful implementation of the design is possible.

Chapter 5

Resource Utilization Results

We have performed the resource utilization on the four FPGA devices of the family Spartan III, Virtex IV, Spartan VI, Virtex V for a different type of devices, different speed grades and different package types. In order to get a fairer comparison with an existing set of 16-bit architectures of the PRESENT block cipher, we performed this analysis on the same devices, with same speed grades and package type as mentioned in the start of the chapter. We analyzed the system at the frequencies 13.56 MHz and 100 MHz. As it globally opts by my many research papers and other is it is used in the IoT transmitters and IoT devices are on our priority as per market demands. The reason behind the selection of 100 MHz is that it has been used in some of the research contents. Moreover, we performed an analysis on the two settings. In the first one, our target is to reduce the area on the cost of other parameters. While in the second setting is balanced. In which tool tries to optimize the design for all the parameters. Actually, balance is the default setting for the tool.

5.1 Results at 100 MHz

This section will show the results on both the settings at the frequency of 100 MHz on all four devices under evaluation. Here results obtained by synthesis and implementation has shown explicitly. Implementation involves many steps and there is a very high possibility that results will be altered after implementation.

5.1.1 Device utilization Results for area reduction setting

As the area is evolving as one of the major parameters with the growth of technology. It can be observed that our devices are scaled down rapidly and Moore's law also stating the same thing. So we chose the area as our design goal in the area reduction setting.

Device Utilization Reports for Spartan III

Post-synthesis and post-implementation resource utilization results of the design at 100 MHz on the FPGA device xc3s200-5ff256 under area reduction setting are as shown in Table 5.1 and Table 5.2 respectively.

Table 5.1: Post-Synthesis Report of the Architecture at 100 MHz on Spartan III under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	178	1920	9%
Number of Slice Flip Flops	202	3840	5%
Number of 4 input LUTs	278	3840	7%
Number of bonded IOBs	50	173	28%
Number of GCLKs	1	8	12%

Table 5.2: Post-Implementation Report of the Architecture at 100 MHz on Spartan III under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	202	3840	5%
Number of 4 input LUTs	276	3840	7%
Number of occupied Slices	171	1920	8%
Number of Slices Containing only related logic	171	171	100%
Number of Slices Containing unrelated logic	0	171	0%
Total Number of 4 input LUTs	340	3840	8%
Number Used as logic	276		
Number Used as a route-thru	64		
Number of bonded IOBs	50	173	28%
IOB Flip Flops	16		
Number of BUFGMUXs	1	8	12%
Average Fanout of Non-Clock Nets	4.34		

Device Utilization Reports for Spartan VI

For the analysis, we chose a device of type xc6slx16 from the ff type package on the speed grade of 3 from Spartan III family which have 324 number of pins embedded on it. A package type of device is csg. We obtain our results on the speed grade of 5 on this device. It contains LUTs of type LUT-4 and DCM for the controlling of the clock. Resource utilization results obtained after synthesized the design on the FPGA device xc6slx16-3csg324 at 100 MHz are as shown in Table 5.3. As per the timing parameter maximum frequency is obtained from the synthesis step of the design flow. We obtained a maximum frequency of 310.243 MHz for the balance setting and 210.102 MHz for the area reduction setting on the FPGA device xc6slx-3csg324. It is observed that maximum operating frequency for the FPGA device xc6slx-3csg324 on both of the frequencies is the same. Now we will consider some of the architectural parameters of the devices from the Spartan III family.

(i) There is an increase logic cell capability of the FPGA device from Spartan VI family due to new 6 input LUT architecture.

(ii) Size of block RAM is 18 Kb and it contains two blocks. So each block of 9 Kb can be used seperately.

(iii) One PLL and two DCMs are embedded in one CMT.

Post-synthesis resource utilization results of the design at 100 MHz on the FPGA device xc6slx16-3csg324 under area reduction setting are as shown in Table 5.3.

Table 5.3: Post-Synthesis Report of the Architecture at 100 MHz on Spartan VI under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	18224	1%
Number of Slice LUTs	222	9112	2%
Number of fully used LUT-FF pairs	202	222	90%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGCTRLs	1	16	6%

Table 5.4: Post-Implementation Report of the Architecture at 100 MHz on Spartan III under area reduction setting.

Device Utilization Summary			
Number of Slice Registers	202	18224	1%
Number used as Flip Flops	202		
Number used as Latch-thrus	0		
Number used as AND/OR logics	0		1%
Number of Slice LUTs	171	9112	1%
Number used as Logic	171	9112	
Number using O6 output only	120		
Number using O5 output only	0		
Number using O5 and O6	51		
Number used as ROM	0		
Number used as Memory	0	2176	0%
Number of Occupied Slices	47	2278	2%
Number of MUXCY used	0	4556	0%
Number of LUT Flip Flop pairs used	179		
Number with an unused Flip Flop	17	179	9%
Number with an unused LUT	8	179	4%
Number of fully used LUT-FF pairs	154	179	86%
Number of unique control sets	5		
Number of slice register sites lost to control set restrictions	14	18224	1%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGMUXs	1	16	6%
Avg. fanout of Non-clock nets	3.92		

The resource utilization results obtained after implementation of the design on the FPGA device xc6slx16-5ff256 at 100 MHz are as shown in Table 5.4.

Device Utilization Results For Virtex IV

It contains LUTs of type LUT-4 and DCM for the controlling of the clock. As per the timing parameter maximum frequency is obtained from the synthesis step of the design flow. We obtained a maximum frequency of 316.481 MHz for the area reduction setting on the FPGA device xc4vlx25-12ff668. It is observed that maximum operating frequency for the FPGA device xc4vlx25-12ff668 on both of the frequencies is the same. Now we will consider some of the architectural parameters of the devices from the Virtex IV family.

(i) CLBs on FPGA device xc4vlx25 device are arranged in a array of size. 96×28.

(ii) It contains 8 DCMs

(ii) It contains total 11 IO banks in it.

(iii) Maximum 448 IO ports are available in this FPGA device.

(iv) Maximum distributed RAM is 168 Kb.

The resource utilization results of synthesis on xc4vlx25-12ff668 FPGA device under area reduction setting at 100 MHz are represented in Table 5.5

Table 5.5: Post-Synthesis Report of the Architecture at 100 MHz on Virtex IV under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	178	10752	1%
Number of Slice Flip Flops	202	21504	0%
Number of 4 inputs	278	21504	1%
Number of bonded IOBs	50	448	11%
Number of GCLKs	1	32	3%

The resource utilization results of implementation xc4vlx25-12ff668 FPGA device under area reduction setting at 100 MHz are represented in Table 5.6

Table 5.6: Post-Implementation Report of the Architecture at 100 MHz on Virtex IV under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	202	21504	1%
Number of 4 input LUTs	276	21504	1%
Number of occupied Slices	171	10752	1%
Number of Slices Containing only related logic	171	171	100%
Number of Slices Containing unrelated logic	0	171	0%
Total Number of 4 input LUTs	340	21504	1%
Number Used as logic	276		
Number Used as a route-thru	64		
Number of bonded IOBs	50	448	11%
IOB Flip Flops	16		8%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	4.10		

Device Utilization Results For Virtex V

For the analysis, we chose a device of type xc5vlx50t from the ff type package on the speed grade of 3 from Spartan III family which have 324 number of pins embedded on it. A package type of device is csg. We obtain our results on the speed grade of 5 on this device. It contains LUTs of type LUT-4 and DCM for the controlling of the clock. As per the timing parameter maximum frequency is obtained from the synthesis step of the design flow. It is observed that maximum operating frequency for the FPGA device xc5vlx50t-3ff11336 on both of the frequencies is the same. Now we will consider some of the architectural parameters of the devices from the Virtex V family of FPGAs.

(i) CLBs on FPGA device xc5vlx50t device are arranged in an array of size. 120×30.

(ii) Maximum 480 IO ports are available in this FPGA device.

(iii) Four LUTs and eight flip flops are embedded in each slice.

(iv) It contains 4-input LUTs.

(v) Size of block RAM is 18 Kb and it contains two blocks. So each block of 9 Kb can be used seperately.

(vi) One PLL and two DCMs are embedded in one .

Resource utilization results obtained after synthesized the design on the FPGA device xc5vlx25-1ff11336 under area reduction setting are as shown in Table 5.7.

Table 5.7: Post-Synthesis Report of the Architecture at 100 MHz on Virtex V under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	28800	0%
Number of Slice LUTs	222	28800	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%

Resource utilization results obtained after implementation of the the design on the FPGA device xc5vlx25-1ff11336 for area reduction setting are as shown in Table 5.8.

Table 5.8: Post-Implementation Report of the Architecture at 100 MHz on Virtex V under area reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	28800	1%
Number used as Flip Flops	202		
Number of Slice LUTs	222	28800	1%
Number used as logic	222	28800	1%
Number using O6 output only	222		
Number of occupied Slices	59	7200	1%
Number of LUT Flip Flop pairs Used	222		
Number with an unused Flip Flop	20	222	9%
Number with an unused LUT	0	222	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of unique Control Sets	5		
Number of Slices register sites lost to control set restrictions	6	28800	1%
Number of bonded IOBs	50	480	10%
IOB Flip Flops	16		
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	4.19		

5.1.2 Device utilization Results at Default setting

If nothing is mentioned then by default it uses balance as a device setting As most of the time, an optimized design is needed so this setting is designed to optimized design w.r.t. all parameters.

Device Utilization Results For Spartan III

The least number of resources are available in the xc3s200-5ff256 FPGA device among the set of FPGA devices under evaluation. Post-synthesis and post-implementation resource utilization results of the design at 100 MHz on the FPGA device xc3s200-5ff256 under default setting are as shown in Table 5.9 and Table 5.10 respectively.

Table 5.9: Post-Synthesis Report of the Architecture at 100 MHz on Spartan III under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	182	1920	9%
Number of Slice Flip Flops	227	3840	5%
Number of 4 input LUTs	294	3840	7%
Number of bonded IOBs	50	173	28%
Number of GCLKs	1	8	12%

Table 5.10: Post-Implementation Report of the Architecture at 100 MHz on Spartan III under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	227	3840	5%
Number of 4 input LUTs	292	3840	7%
Number of occupied Slices	179	1920	9%
Number of Slices Containing only related logic	179	179	100%
Number of Slices Containing unrelated logic	0	179	0%
Total Number of 4 input LUTs	340	3840	8%
Number Used as logic	292		
Number Used as a route-thru	48		
Number of bonded IOBs	50	173	28%
Number of BUFGMUXs	1	8	12%
Average Fanout of Non-Clock Nets	3.70		

Device Utilization Results For Spartan VI

xc6slx16-3csg324 is the latest FPGA device among the set of the device under evaluations. Better resource utilization results have been obtained on this device in comparison to other devices for a particular architecture at a particular frequency and particular setting.

The resource utilization results after synthesis under balance setting for FPGA device xc6slx16-3csg at 100 MHz are represented in Table 5.11 and post-implementation parameters are as shown in Table 5.12.

Table 5.11: Post-Synthesis Report of the Architecture at 100 MHz on Spartan VI under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	226	18224	1%
Number of Slice LUTs	224	9112	2%
Number of fully used LUT-FF pairs	220	230	95%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGCTRLs	1	16	6%

Table 5.12: Post-Implementation Report of the Architecture at 100 MHz on Spartan VI under Default setting.

Device Utilization Summary			
Number of Slice Registers	226	18224	1%
Number used as Flip Flops	226		
Number used as Latch-thrus	0		
Number used as AND/OR logics	0		1%
Number of Slice LUTs	185	9112	2%
Number used as Logic	184	9112	2%
Number using O6 output only	114		
Number using O5 output only	0		
Number using O5 and O6	40		
Number used as ROM	0		
Number used as Memory	0	2176	0%
Number of Occupied Slices	48	2278	2%
Number of MUXCY used	0	4556	0%
Number of LUT Flip Flop pairs used	187		
Number with an unused Flip Flop	2	187	1%
Number with an unused LUT	2	187	1%
Number of fully used LUT-FF pairs	183	187	97%
Number of unique control sets	7		
Number of slice register sites lost to control set restrictions	14	18224	1%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGMUXs	1	16	6%

Device Utilization Results For Virtex IV

In this section, device utilization reports of the design for the Virtex IV FPGA device is present. The device we chose for results is from the LX platform.

The resource utilization results after synthesis for Virtex IV device with balance

setting at 100 MHz are represented in Table 5.13.

Table 5.13: Post-Synthesis Report of the Architecture at 100 MHz on Virtex IV under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	178	10752	1%
Number of Slice Flip Flops	221	21504	1%
Number of 4 inputs	294	21504	1%
Number of bonded IOBs	50	448	11%
Number of GCLKs	1	32	3%

The resource utilization results after implementation for Virtex IV device with balance setting at 100 MHz are represented in Table 5.14.

Table 5.14: Post-Implementation Report of the Architecture at 100 MHz on Virtex IV under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	221	21504	1%
Number of 4 input LUTs	292	21504	1%
Number of occupied Slices	173	10752	1%
Number of Slices Containing only related logic	173	173	100%
Number of Slices Containing unrelated logic	0	173	0%
Total Number of 4 input LUTs	340	21504	1%
Number Used as logic	292		
Number Used as a route-thru	48		
Number of bonded IOBs	50	448	11%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	3.68		

Device Utilization Results For Virtex V

This device has a maximum number of slice registers, LUTs and flip flops among existing architectures. Results may or may not alter in the implementation step. Some of the parameters in post-synthesis reports are changed in post-implementation report whereas some remain the same. This stuff can be observed by the comparison between the data of Table 5.15 and Table 5.16. Post-synthesis full utilization of LUT-FF pairs is improved and it obtained 90% in post-implementation results. The resource

utilization results after synthesis for Virtex V device with balance setting at 100 MHz for the FPGA device xc5vlx50t-3ff11336 are as shown in Table 5.15.

Table 5.15: Post-Synthesis Report of the Architecture at 100 MHz on Virtex V under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	218	28800	0%
Number of Slice LUTs	286	28800	0%
Number of fully used LUT-FF pairs	218	286	76%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%

The resource utilization results after implementation for Virtex V device with balance setting at 100 MHz are represented in Table 5.16.

Table 5.16: Post-Implementation Report of the Architecture at 100 MHz on Virtex V under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	218	28800	1%
Number used as Flip Flops	218		
Number of Slice LUTs	286	28800	1%
Number used as logic	286	28800	1%
Number using O6 output only	286		
Number of occupied Slices	89	7200	1%
Number of LUT Flip Flop pairs Used	286		
Number with an unused Flip Flop	68	286	23%
Number with an unused LUT	0	286	0%
Number of fully used LUT-FF pairs	286	76	90%
Number of unique Control Sets	4		
Number of Slices register sites lost to control set restrictions	6	28800	1%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	3.99		

5.2 Results at 13.56 MHz

5.2.1 Device utilization Results for area reduction setting

Device Utilization Results For Spartan III

The resource utilization results after synthesis for Spartan III FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.17

Table 5.17: Post-Synthesis Report of the Architecture at 13.56 MHz on Spartan III under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	28800	0%
Number of Slice LUTs	222	28800	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of bonded IOBs	50	480	10%
Number of GCLKs	1	32	3%

The resource utilization results after implementation of the design for Spartan III FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.18

Table 5.18: Post-Implementation Report of the Architecture at 13.56 MHz on Spartan III under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	202	3840	5%
Number of 4 input LUTs	276	3840	7%
Number of occupied Slices	171	1920	8%
Number of Slices Containing only related logic	171	171	100%
Number of Slices Containing unrelated logic	0	171	0%
Total Number of 4 input LUTs	340	3840	8%
Number Used as logic	276		
Number Used as a route-thru	64		
Number of bonded IOBs	50	173	28%
IOB Flip Flops	16		28%
Number of BUFGMUXs	1	8	12%
Average Fanout of Non-Clock Nets	4.34		

Device Utilization Results For Spartan VI

The resource utilization results after synthesis of the design for Spartan VI FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.19.

Table 5.19: Post-Synthesis Report of the Architecture at 13.56 MHz on Spartan VI under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	18224	1%
Number of Slice LUTs	222	9112	2%
Number of fully used LUT-FF pairs	202	222	90%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGCTRLs	1	16	6%

The resource utilization results after implementation of the design for Spartan VI FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.20.

Table 5.20: Post-Implementation Report of the Architecture at 13.56 MHz on Spartan VI under Area Reduction setting.

Device Utilization Summary			
Number of Slice Registers	202	18224	1%
Number used as Flip Flops	202		
Number used as Latch-thrus	0		
Number used as AND/OR logics	0		1%
Number of Slice LUTs	222	9112	2%
Number used as Logic	222	9112	2%
Number using O6 output only	222		
Number using O5 output only	0		
Number using O5 and O6	0		
Number used as ROM	0		
Number used as Memory	0	2176	0%
Number of Occupied Slices	46	2278	2%
Number of MUXCY used	0	4556	0%
Number of LUT Flip Flop pairs used	222		
Number with an unused Flip Flop	20	222	9%
Number with an unused LUT	0	222	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of unique control sets	5		
Number of slice register sites lost to control set restrictions	14	18224	1%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGMUXs	1	16	6%
Avg. fanout of Non-clock nets	4.19		

Device Utilization Results For Virtex IV

The resource utilization results after synthesis of the design for Virtex IV FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.21.

Table 5.21: Post-Synthesis Report of the Architecture at 13.56 MHz on Virtex IV under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	178	10752	1%
Number of Slice Flip Flops	202	21504	0%
Number of 4 inputs	278	21504	1%
Number of bonded IOBs	50	448	11%
Number of GCLKs	1	32	3%

The resource utilization results after implementation of the design for Virtex IV FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.22.

Table 5.22: Post-Implementation Report of the Architecture at 13.56 MHz on Virtex IV under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	202	21504	1%
Number of 4 input LUTs	276	21504	1%
Number of occupied Slices	171	10752	1%
Number of Slices Containing only related logic	171	171	100%
Number of Slices Containing unrelated logic	0	171	0%
Total Number of 4 input LUTs	340	21504	1%
Number Used as logic	276		
Number Used as a route-thru	64		
Number of bonded IOBs	50	448	11%
IOB Flip Flops	16		8%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	4.10		

Synthesis for Virtex V at 13.56 MHz

The resource utilization results after synthesis for Virtex V FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.23.

Table 5.23: Post-Synthesis Report of the Architecture at 13.56 MHz on Virtex V under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	28800	0%
Number of Slice LUTs	222	28800	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%

The resource utilization results after implementation of the design for Virtex V FPGA device with area reduction setting at the frequency of 13.56 MHz are represented in Table 5.24.

Table 5.24: Post-Implementation Report of the Architecture at 13.56 MHz on Virtex V under Area Reduction setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	202	28800	1%
Number used as Flip Flops	202		
Number of Slice LUTs	222	28800	1%
Number used as logic	222	28800	1%
Number using O6 output only	222		
Number of occupied Slices	59	7200	1%
Number of LUT Flip Flop pairs Used	222		
Number with an unused Flip Flop	20	222	9%
Number with an unused LUT	0	222	0%
Number of fully used LUT-FF pairs	202	222	90%
Number of unique Control Sets	5		
Number of Slices register sites lost to control set restrictions	6	28800	1%
Number of bonded IOBs	50	480	10%
IOB Flip Flops	16		
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	4.19		

5.2.2 Device utilization Results for Balanced setting

Device Utilization Results For Spartan III

The resource utilization results after synthesis for Spartan III FPGA device with balance setting at the frequency of 13.56 MHz are represented in Table 5.25.

Table 5.25: Post-Synthesis Report of the Architecture at 13.56 MHz on Spartan III under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	182	1920	9%
Number of Slice Flip Flops	227	3840	5%
Number of 4 input LUTs	294	3840	7%
Number of bonded IOBs	50	173	28%
Number of GCLKs	1	8	12%

The resource utilization results after synthesis for Spartan III FPGA device with balance setting at the frequency of 13.56 MHz are represented in Table 5.26.

Table 5.26: Post-Implementation Report of the Architecture at 13.56 MHz on Spartan III under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	227	3840	5%
Number of 4 input LUTs	292	3840	7%
Number of occupied Slices	179	1920	9%
Number of Slices Containing only related logic	179	179	100%
Number of Slices Containing unrelated logic	0	179	0%
Total Number of 4 input LUTs	340	3840	8%
Number Used as logic	292		
Number Used as a route-thru	48		
Number of bonded IOBs	50	173	28%
Number of BUFGMUXs	1	8	12%
Average Fanout of Non-Clock Nets	3.70		

Device Utilization Results For Spartan VI

Results of the resource utilization as well as for the power consumption are best for the Spartan VI among device under evaluation. It also have more type of resources

than other three FPGA devices which makes it different. It uses LUT-6 which helps us to reduce the area of the design. On this device LUT-FF pair utlization on the FPGA device is 97%, which is 7% better then other setting on the same device which shows that it is an optimized design. Post-synthesis and post-implementation resource utilization results of the design at 13.56 MHz on the FPGA device xc6slx16-3csg324 under default setting are as shown in Table 5.27 and Table 5.28 respectively.

Table 5.27: Post-Synthesis Report of the Architecture at 13.56 MHz on Spartan VI under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	226	18224	1%
Number of Slice LUTs	224	9112	2%
Number of fully used LUT-FF pairs	220	230	95%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGCTRLs	1	16	6%

Table 5.28: Post-Implementation Report of the Architecture at 13.56 MHz on Spartan VI under Default setting.

Device Utilization Summary			
Number of Slice Registers	226	18224	1%
Number used as Flip Flops	226		
Number used as Latch-thrus	0		
Number used as AND/OR logics	0		1%
Number of Slice LUTs	185	9112	2%
Number used as Logic	184	9112	2%
Number using O6 output only	144		
Number using O5 output only	0		
Number using O5 and O6	40		
Number used as ROM	0		
Number used as Memory	0	2176	0%
Number of Occupied Slices	48	2278	2%
Number of MUXCY used	0	4556	0%
Number of LUT Flip Flop pairs used	187		
Number with an unused Flip Flop	2	187	1%
Number with an unused LUT	2	187	1%
Number of fully used LUT-FF pairs	183	187	97%
Number of unique control sets	7		
Number of slice register sites lost to control set restrictions	14	18224	1%
Number of bonded IOBs	50	232	21%
Number of BUFG/BUFGMUXs	1	16	6%

Device Utilization Results For Virtex IV

The resource utilization results after synthesis for the FPGA device xc4vlx25-12ff668 under balance setting at 13.56 MHz are represented in Table 5.29.

Table 5.29: Post-Synthesis Report of the Architecture at 13.56 MHz on Virtex IV under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slices	178	10752	1%
Number of Slice Flip Flops	221	21504	1%
Number of 4 inputs	294	21504	1%
Number of bonded IOBs	50	448	11%
Number of GCLKs	1	32	3%

The resource utilization results after implementation for the FPGA device xc4vlx25-12ff668 under balance setting at 13.56 MHz are represented in Table 5.30.

Table 5.30: Post-Implementation Report of the Architecture at 13.56 MHz on Virtex IV under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	221	21504	1%
Number of 4 input LUTs	292	21504	1%
Number of occupied Slices	173	10752	1%
Number of Slices Containing only related logic	173	171	100%
Number of Slices Containing unrelated logic	0	171	0%
Total Number of 4 input LUTs	340	21504	1%
Number Used as logic	292		
Number Used as a route-thru	48		
Number of bonded IOBs	50	448	11%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	3.68		

Device Utilization Results For Virtex V

In this particular setting at 13.56 MHz frequency, our design the values of the parameters in the synthesis report are in the implementation report also but in general, that is not the case. The resource utilization results after synthesis for the FPGA device

xc5vlx50t-1ff11336 under balance setting at 13.56 MHz are represented in Table 5.31.

Table 5.31: Post-Synthesis Report of the Architecture at 13.56 MHz on Virtex V under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	218	28800	0%
Number of Slice LUTs	286	28800	0%
Number of fully used LUT-FF pairs	218	286	76%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%

The resource utilization results after implementation under balance setting at 13.56 MHz are represented in Table 5.32.

Table 5.32: Post-Implementation Report of the Architecture at 13.56 MHz on Virtex V under Default setting.

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	218	28800	1%
Number used as Flip Flops	218		
Number of Slice LUTs	286	28800	1%
Number used as logic	286	28800	1%
Number using O6 output only	286		
Number of occupied Slices	89	7200	1%
Number of LUT Flip Flop pairs Used	286		
Number with an unused Flip Flop	68	286	23%
Number with an unused LUT	0	286	0%
Number of fully used LUT-FF pairs	218	286	76%
Number of unique Control Sets	4		
Number of Slices register sites lost to control set restrictions	6	28800	1%
Number of bonded IOBs	50	480	10%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	3.99		

Chapter 6

Result Analysis and Discussion

As described in the previous chapter that we have analyzed device utilization and timing analysis for the four FPGA devices Spartan III, Virtex IV, Spartan VI, Virtex V on different type of devices, different speed grades and different package types. To get a fairer comparison with an existing set of 16-bit architectures of the PRESENT block cipher, we performed this analysis on the same devices, with same speed grades and package type. As our architecture is supported by key_size of 80-bit so we compare with other architectures for the version which supports the same length of key_size. The major resource parameters are a number of occupied slices, flip flops and LUTs. In which slices are most important parameter because LUTs and Flip Flops are embedded inside it. that is the reason behind the importance of the number of occupied slices in the area analysis. Timing parameters like latency, maximum frequency supported by a particular FPGA board for the are also important in which latency is purely decided by architecture and it is independent of the type of FPGA boards whereas maximum frequency is dependent on both architecture and type of FPGA device. But the most common thing in latency and maximum frequency parameters is that both of them are primary parameters then there are some parameters which are derived from these primary parameters like throughput is dependent on the operational frequency and *throughput-per-slice* is derived parameter which is actually derived from throughput and number of occupied slices by the architecture. *Throughput-per-slice* shows the combined effect of area and timing parameters so this can be a very crucial parameter

for an optimized architecture in terms of area as well as timing.

To optimize the circuit, the intent of the application plays a crucial role to decide design goal for the application. Like in the defense system performance can't be compromised while cost can be ignored. When it comes to providing security to the IoT devices which has made for the purpose of daily uses at home, then an adequate level of security is sufficient and performance is required but at the same time cost reduction is very much needed. In other words for the daily purpose IoT applications, an optimized architecture in almost all parameters is desirable. So one can say design goal should be decided according to the intent of the application. So we decided to analysis our design of the encryption algorithm for the multiple design goals so that one can see the results according to the intent of their application and if the results of that particular goal are desirable then accordingly can decide incisively that whether they should use our architecture of algorithm or not. After knowing that much importance of design goals we decided to implement our design for multiple design goals and show results of those in this chapter.

In Timing parameters, latency is the most crucial parameter. Because it also affects the energy consumption on the particular device at a particular frequency. Basically reduction in latency will speed up our design. Throughput at a particular frequency is inversely proportional to latency so a reduction in latency will improve Throughput to a great extent. Energy consumption has a proportional relation to latency. So energy consumption is also reduced with a decrease in latency. We can say that reduction in latency will improve our design in terms of timing parameters as well as energy parameters.

Table 6.1 represents resource utilization and performance results of the proposed architecture under evaluation on the four different FPGA devices. The area is represented in terms of the number of occupied slices, flip flops and LUTs. The results are consistent with both LUT-4 FPGAs. Here performance is analyzed on two different frequencies. First is the maximum frequency of that architecture on particular FPGAs and second is at 13.56 MHz. This particular frequency is an operational frequency for RF applications, like IoT transmitters [1].

Table 6.1: Resource Utilization and Performance Computations of the Proposed Architecture on Different FPGAs.

Device	Xc4vlx25 -12FF668	Xc5vlx50t -3ff11336	Xc3s200 -5ff256	Xc6slx16 -3csg324
State(bit)	64	64	64	64
Key_Size(bit)	80	80	80	80
Flip Flop	221	218	227	226
LUT	292	286	292	185
SLC	173	88	179	48
Fmax(MHz)	339.80	509.476	179.420	310.243
Latency(cycles)	37	37	37	37
Thr(Mbps)	587.80	881.28	310.35	536.64
Thr*(Mbps)	23.46	23.46	23.46	23.46
Thr*/SLC(Kbps/slice)	135.61	266.59	131.06	488.75

6.1 Comparison of Resource Utilization and Timing Parameters With Existing 16-bit Architectures

This section provides us a brief comparison of resource utilization and timing Parameters between the proposed 16-bit architecture with the set of existing 16-bit architectures [6] and [7] for a key size of 80-bit. Here some of the major performance matrices of the proposed architecture have been compared with the set of existing 16-bit architectures in terms of area utilization and performance. There are two parameters in performance namely latency and throughput. According to us, in these parameters, latency is a more appropriate measure as it is device independent whereas throughput is device dependent. Because throughput is a function of the maximum operating frequency of a particular FPGA device. So in order to make throughput a device independent measure of performance, throughput at a 13.56 MHz frequency (thr*) has also been computed.

Resource utilization, performance and other design metrics on Xilinx xc6slx16-3csg324 FPGA device have been used for comparison [6]. Therefore, the results of the proposed architecture have been compared with [6] for the Xilinx xc6slx16-3csg324

FPGA device. Table 6.2 shows the comparison of resource utilization and performance parameters for the proposed architecture with [6]. Resource utilization and performance measures on all four FPGA devices as considered by [7] has also been considered. A comparison of the proposed architecture with [7] is shown in Table 6.3.

Table 6.2: Comparison of Resource Utilization and Performance Parameters between both the Architectures on Xilinx xc6slx16-3csg324 FPGA Device.

Work	W2	Proposed Work
State (bit)	64	64
Key_Size (bit)	80	80
Flip Flop	89	226
LUT	226	185
SLC	69	48
Fmax(Mbps)	172.92	310.243
Latency	132	37
Thr (Mbps)	83.84	536.64
Thr*(Mbps)	6.57	23.46
Thr*/SLC (Kbps/slice)	95.28	488.75

Table 6.3: Comparison of Area and Performance Parameters between both the Architectures at 13.56 MHz under default setting.

Device	Work	state (bit)	key (bit)	Flip Flop	LUT	Slice	Fmax (MHz)	Latency (cycles)	Thr (Mbps)	Thr* (Mbps)	Thr*/slice (Kbps/SLC)
Xc4vlx25-12FF668	W1	64	80	**153**	**215**	**124**	**375.66**	133	180.77	6.53	52.62
	Proposed Work	64	80	221	292	173	339.80	**37**	**587.80**	**23.46**	**135.61**
Xc5vlx50t-ff11336	W1	64	80	**153**	**190**	**67**	**542.30**	133	260.96	6.53	97.39
	Proposed Work	64	80	218	286	88	509.476	**37**	**881.28**	**23.46**	**266.59**
Xc3s200-5ff256	W1	64	80	**153**	**215**	**124**	**213.81**	133	102.89	6.53	52.62
	Proposed Work	64	80	227	292	179	179.420	**37**	**310.35**	**23.46**	**131.06**
Xc6slx16-3csg324	W1	64	80	**153**	**170**	**48**	257.40	133	123.86	6.53	135.94
	Proposed Work	64	80	226	185	**48**	**310.243**	**37**	**536.64**	**23.46**	**488.75**

The comparision of resource utilization and timing parameters of our work with [6] and [7] is shown below with bar diagrams. Bar diagrams has been used to see a clear picture of variation in parameters.

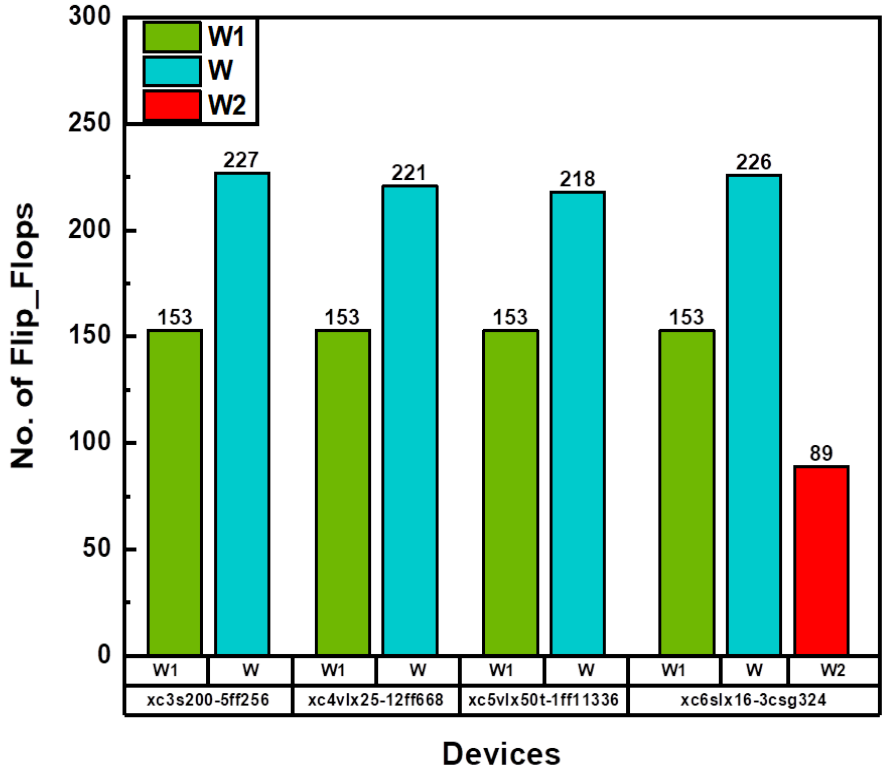

Figure 6.1: Comparison of Number of Flip Flops between both the Architectures at 13.56 MHz under default setting.

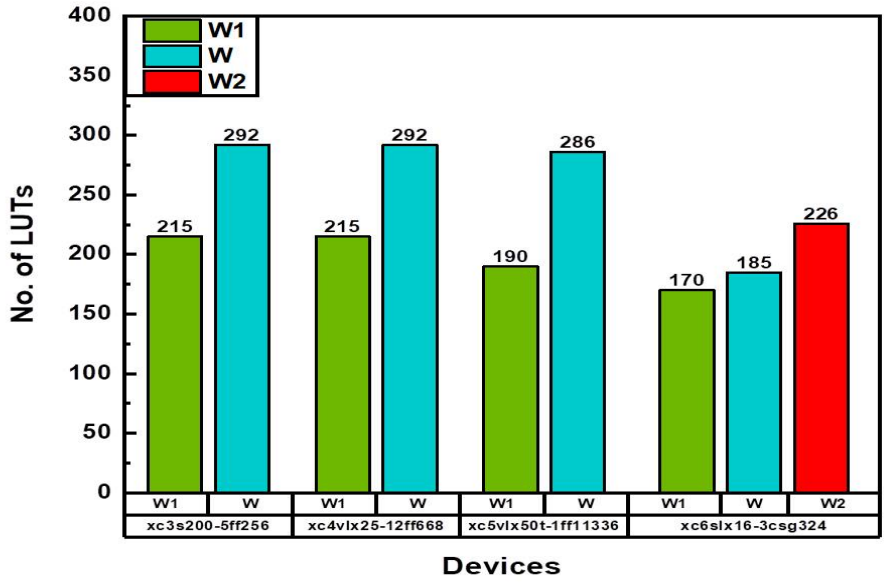

Figure 6.2: Comparison of Number of LUTs between both the Architectures at 13.56 MHz under default setting.

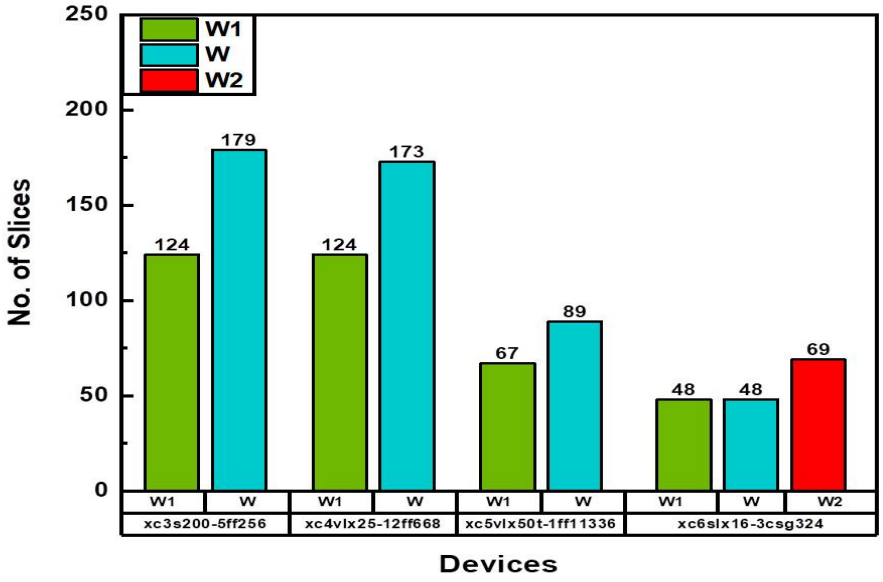

Figure 6.3: Comparison of Number of Slices between both the Architectures at 13.56 MHz under default setting.

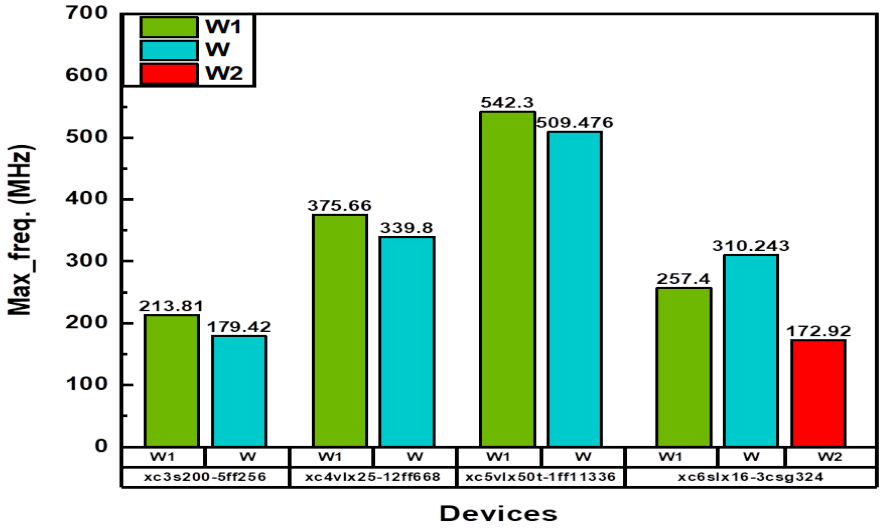

Figure 6.4: Comparison of Maximum Frequency between both the Architectures at 13.56 MHz under default setting.

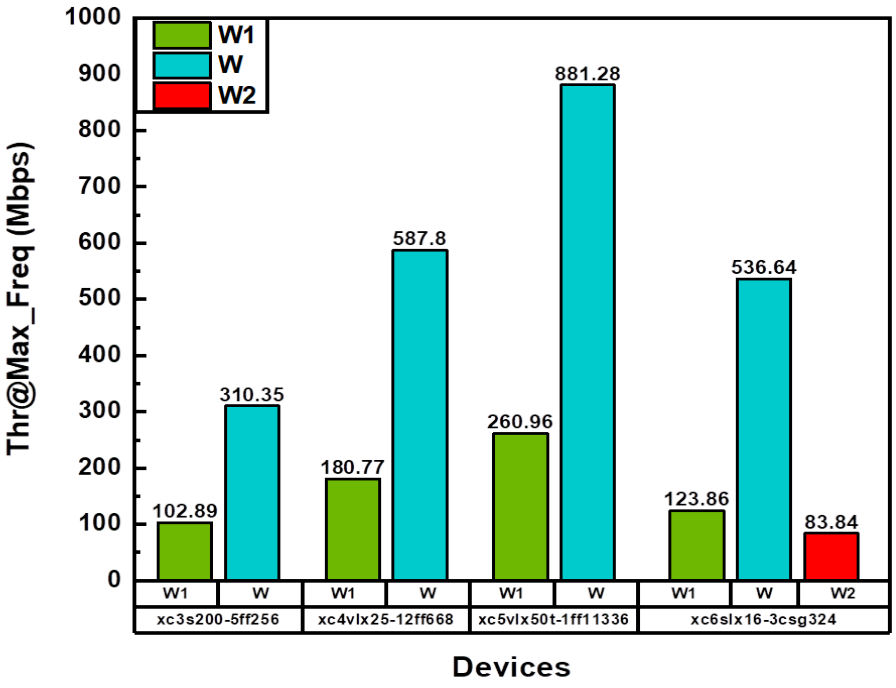

Figure 6.5: Comparison of Throughput at Maximum Frequency between both the Architectures at 13.56 MHz under default setting.

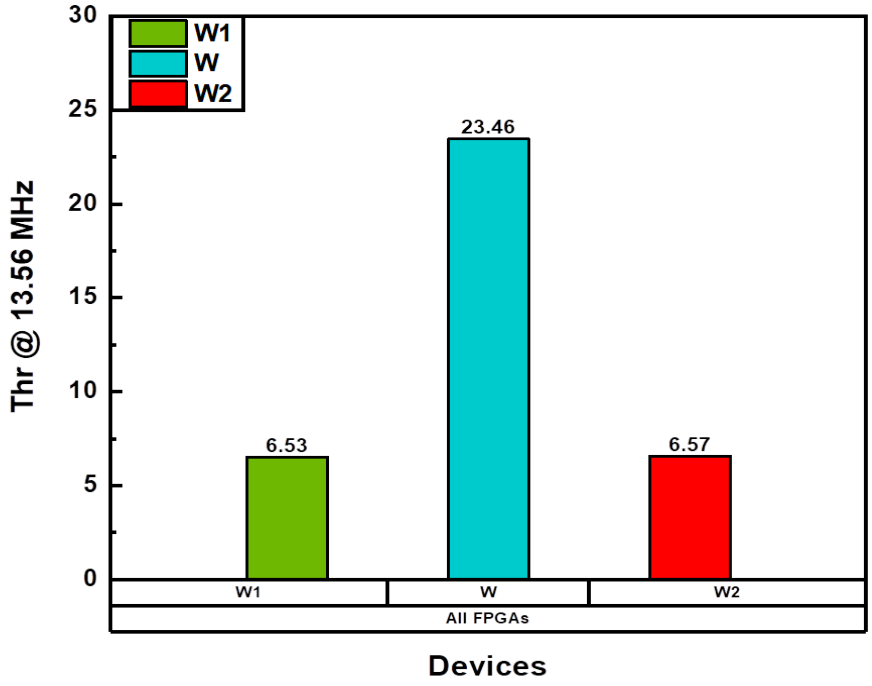

Figure 6.6: Comparison of Throughput at 13.56 MHz between both the Architectures under default setting.

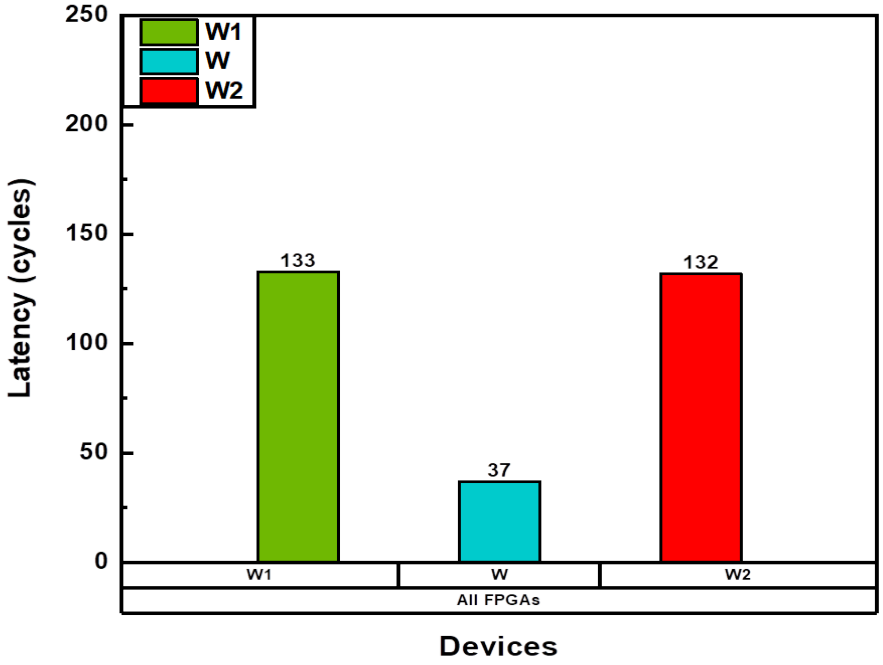

Figure 6.7: Latency.

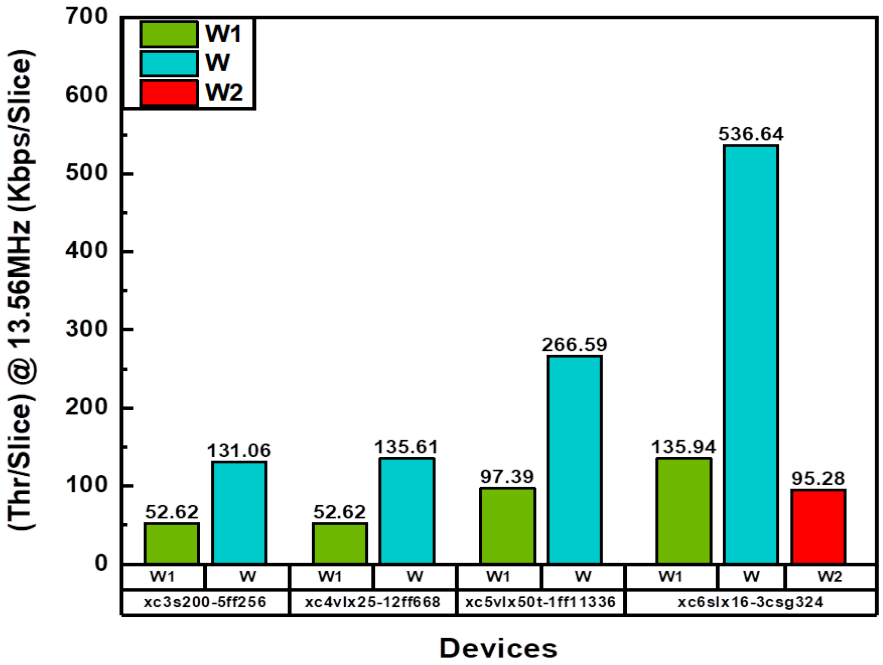

Figure 6.8: Comparison of Throughput-per-slice at 13.56 MHz between both the Architectures under default setting.

Table 6.4: Comparison of Area and Performance Parameters between both the Architectures at 13.56 MHz under area reduction setting.

Device	Work	state (bit)	key (bit)	Flip Flop	LUT	Slice	Fmax (MHz)	Latency (cycles)	Thr (Mbps)	Thr* (Mbps)	Thr*/slice (Kbps/SLC)
Xc4vlx25-12FF668	W1	64	80	**153**	**215**	**124**	**375.66**	133	180.77	6.53	52.62
	Proposed Work	64	80	202	276	171	316.481	**37**	**547.42**	**23.46**	**137.19**
Xc5vlx50t-ff11336	W1	64	80	**153**	**190**	67	**542.30**	133	260.96	6.53	97.39
	Proposed Work	64	80	202	222	**59**	409.333	**37**	**708.15**	**23.46**	**397.63**
Xc3s200-5ff256	W1	64	80	**153**	**215**	**124**	**213.81**	133	102.89	6.53	52.62
	Proposed Work	64	80	202	292	173	149.412	**37**	**258.48**	**23.46**	**135.60**
Xc6slx16-3csg324	W1	64	80	**153**	**170**	48	**257.40**	133	123.86	6.53	135.94
	Proposed Work	64	80	202	222	46	214.102	**37**	**370.40**	**23.46**	**510**

The comparision of resource utilization and timing parameters of our work, on the area reduction setting with [6] and [7] is shown below with bar diagrams.

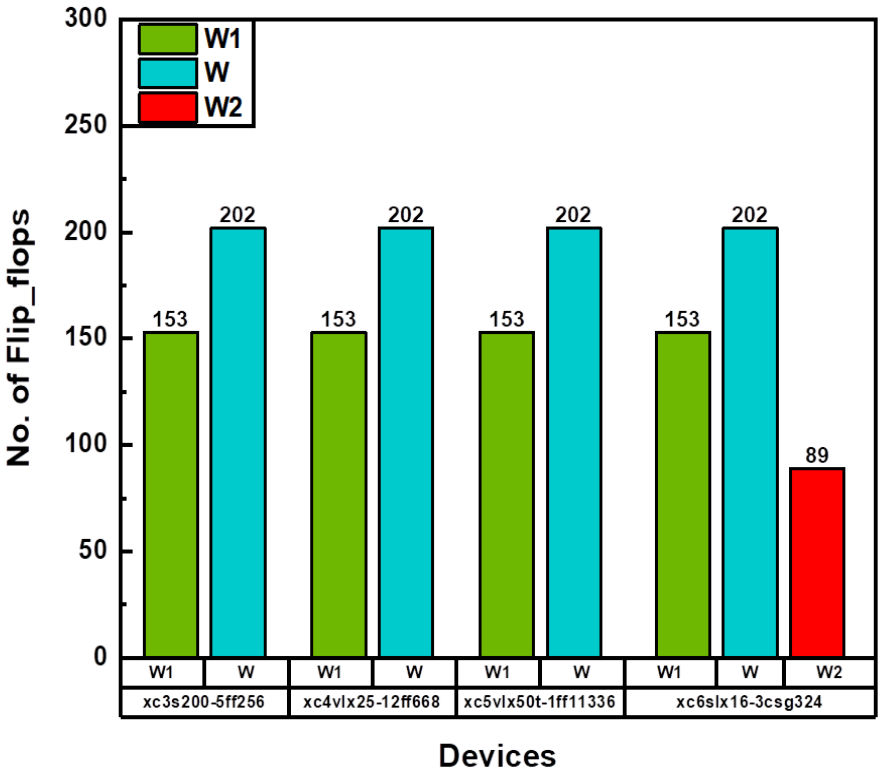

Figure 6.9: Comparison of Number of Flip Flops between both the Architectures at 13.56 MHz under area reduction setting.

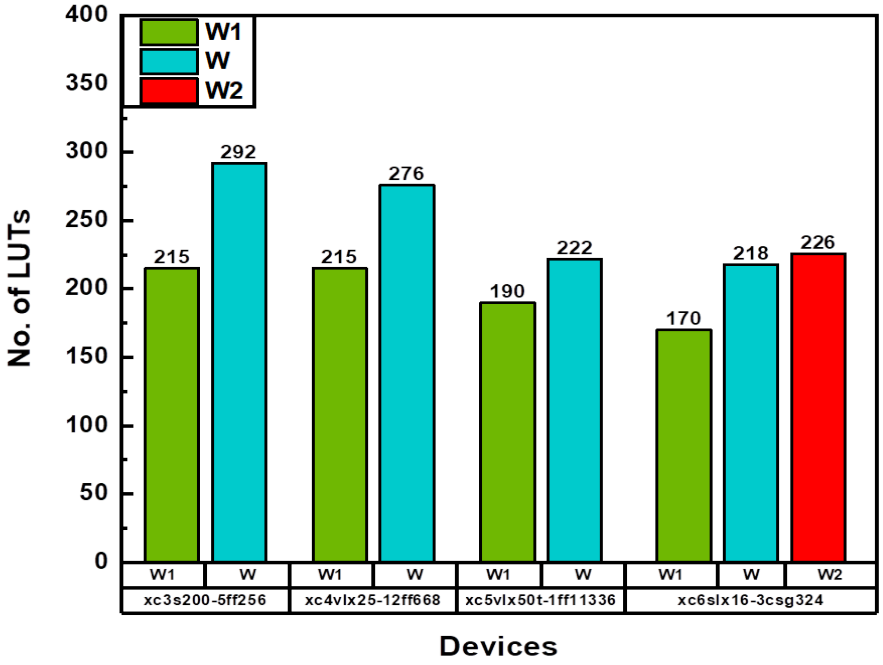

Figure 6.10: Comparison of Number of LUTs between both the Architectures at 13.56 MHz under area reduction setting.

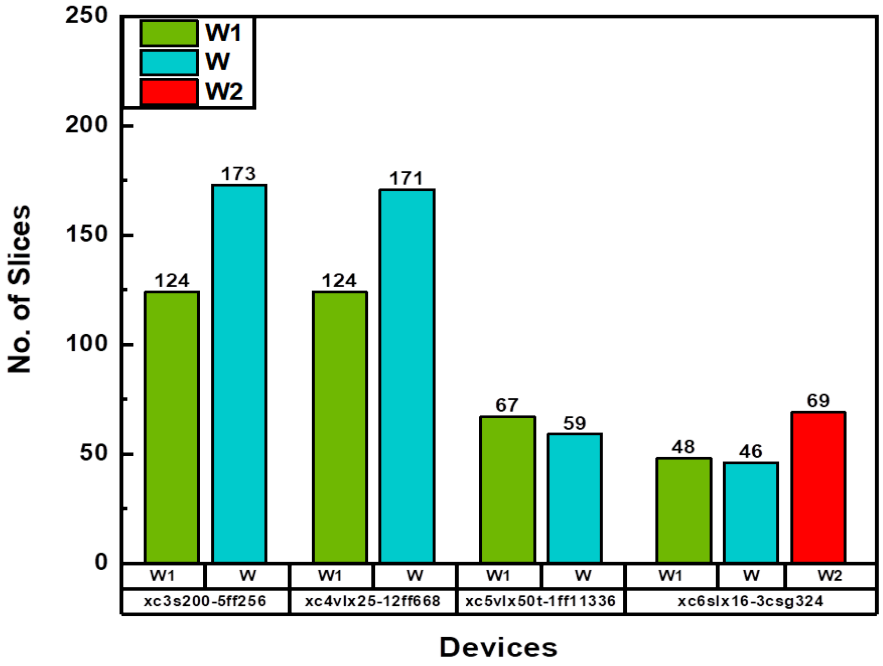

Figure 6.11: Comparison of Number of Slices between both the Architectures at 13.56 MHz under area reduction setting.

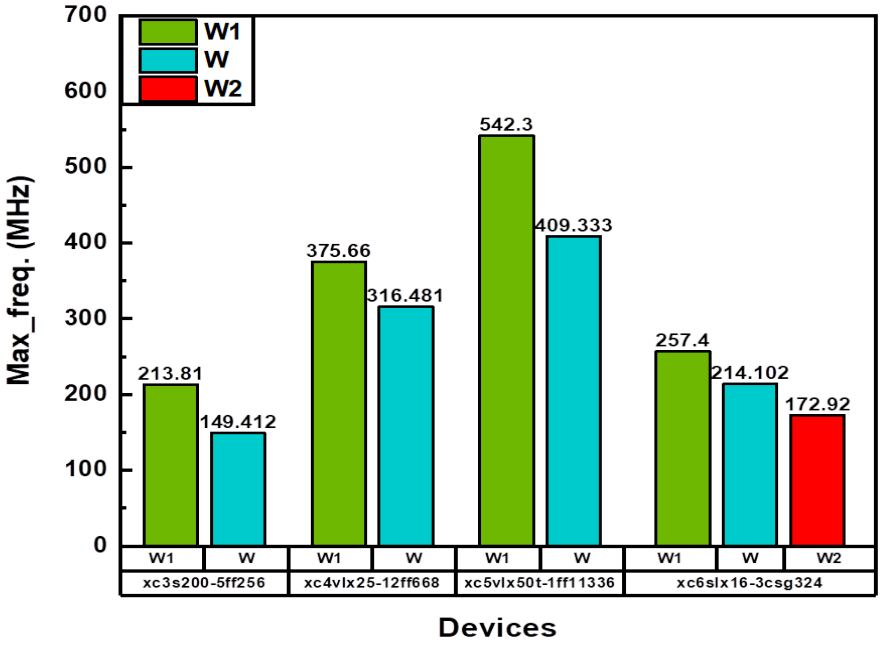

Figure 6.12: Comparison of Maximum Frequency between both the Architectures at 13.56 MHz under area reduction setting.

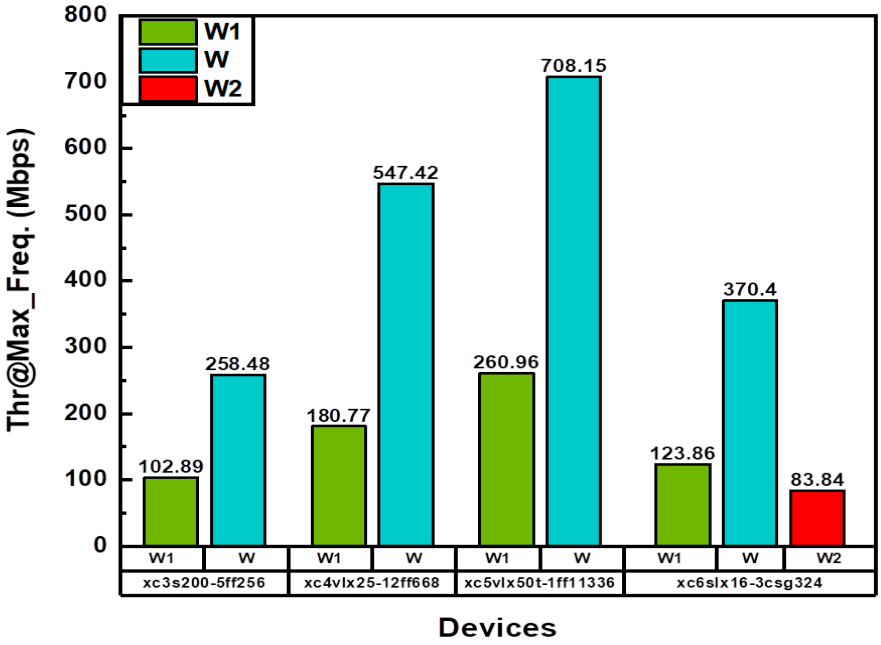

Figure 6.13: Comparison of Throughput at Maximum between both the Architectures at 13.56 MHz under area reduction setting.

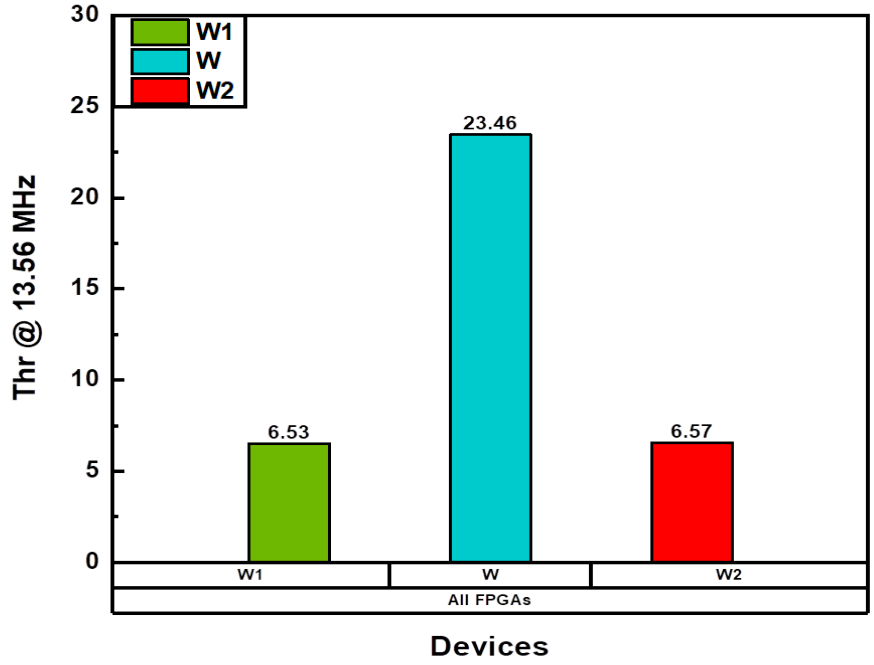

Figure 6.14: Comparison of Throughput at 13.56 MHz between both the Architectures under area reduction setting.

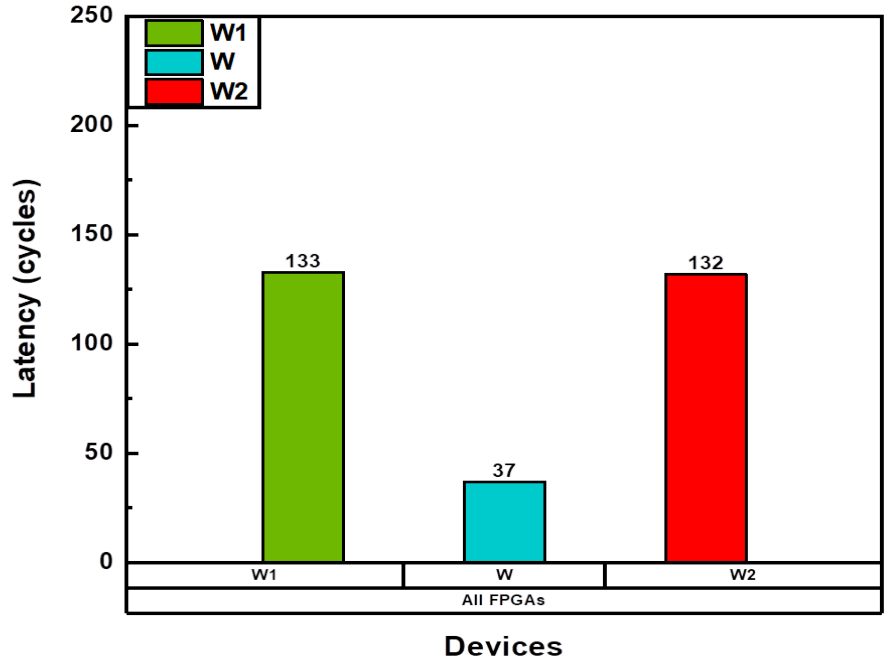

Figure 6.15: Latency.

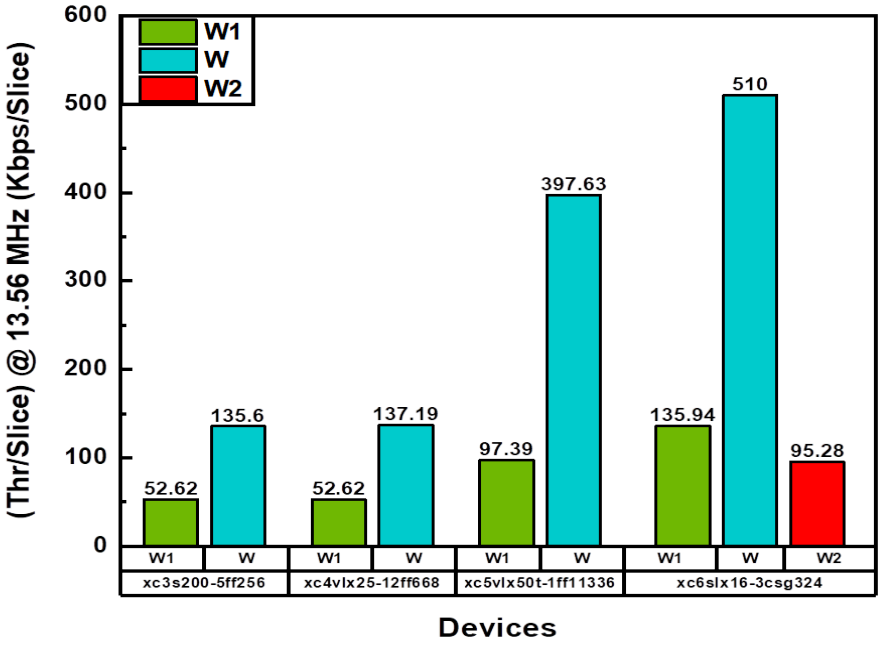

Figure 6.16: Comparison of Throughput at 13.56 MHz between both the Architectures under area reduction setting.

6.2 Comparison of Power and Energy Consumption With Existing 16-bit Architectures

With the advancement in technology, power and energy consumption evolves as a bigger challenge for the design of any system. Power consumption broadly can be classified in two categiories which are as follows.

(i) Static Power Consumption

(ii) Dynamic Power Consumption

With the results shown in Table 6.5 it can be easily understand that major part in the the total power consumption is contributed by static power consumption. Interestingly, for all the existing architectures including proposed architecture, it can also be observed that it is almost same for all the existing architectures including proposed

architecture rather it mostly depends on the device for which device has been analyzed. That is the reason that total power consumption on a particular FPGA device is nearly same for both architecture But energy consumption is related to latency so it has been improved around 72% in our architecture than [7]. The parameter energy-per-bit is also improved much which shows that it is better also optimized designed than [7] in terms of area and energy.

When it comes to power analysis, tool requires some information to estimate power. This information is provided to tool in the form of some files. Different files provides the different information and all those information are considered then power is estimated accordingly. There are mainly four types files which are primarily used for power analysis are NCD,PCF,setting f, SAIF and VCD files. NCD is obtained by place and route step of the circuit. So whole details related to placement of components are reside in this file. NCD is actually provides native circuit description. Settings for power analysis. are provided by setting file. Designer made a UCF file to get the implementation details of the design for a particular FPGA device. UCF contains details related to the clocking of circuit. Those clock related reports alongwith some other details are feed into PCF file which helps to get a better results for power analysis. VCD or SAIF files are generated by user which provides information like how data is switches between components with time. Basically it provides switching and activity rates for the design. These both are alternatives of each other. All these files has been feed into system for accurate power consumption analysis. VCD, SAIF and PCF files contributes a larger part in the dynamic power consumption.

Static power consumption mainly consists of leakage power consumption which is due to capacitor discharging property. There are always some unwanted capacitors associated with the MOS circuit. Most of the circuits are implemented in CMOS technology which consists of NMOS and PMOS. So these capacitors creates a problem of leakage power consumption. Technology node has scale down with the advancement in technology. In Spartan II and Virtex IV technology node is 90 nm whereas in Virtex V and Spartan VI technology node scaled down to 65 nm and 45 nm respectively. Less technology node is basically related with channel length. Electric field increases with

the decrement in channel length which results in a increment in power consumption in the form of leakage. So there should be more static power consumption in Virtex V and Spartan VI in compare to Spartan III and Virtex V. Table 6.5 same type of behaviour is observed for Virtex V FPGA device but static power consumption is exceptionally less in Spaton VI. The reason behind that there some arrangements are made in Spartan VI FPGA device to reduce static as well as dynamic power consumption. The major reasons are as follows.

(i) There is a hibernate power mode in this device .

(ii) There is a suspend mode in the devices of Spartan III family which maintain state and configuration with control enhancement and multi-pin wake-up.

(iii) Core voltage used is 1 V to 1.2 V only which vary with the platform and speed of the device that results in reduction of power consumption sufficiently.

Table 6.5 shows comparison of energy and power consumption for proposed architecture with [7].

Table 6.5: Comparison of Energy and Power Consumption between both the Architectures.

Device	Work	State (bit)	Key_Size (bit)	Latency (cycles)	Static Power (mw)	Dynamic Power (mw)	Total Power (mw)	Energy (µJ)	Energy/bit (nj/bit)
Xc4vlx25-12ff668	W1	64	80	133	232.97	12.81	245.78	2.411	37.667
	Proposed Work	64	80	**37**	**232.90**	**9.10**	**242**	**0.660**	**10.312**
Xc5vlx50-1ff11336	W1	64	80	133	**560.04**	**2.71**	**562.75**	5.520	86.244
	Proposed Work	64	80	**37**	560.05	4.11	564.16	**1.539**	**24.047**
Xc3s200-5ff256	W1	64	80	133	**40.99**	1.09	42.08	0.413	6.449
	Proposed Work	64	80	37	**40.99**	**0.78**	**41.76**	**0.115**	**1.797**
Xc6slx16-3csg324	W1	64	80	133	**19.91**	**1.70**	**21.61**	0.212	3.312
	Proposed Work	64	80	**37**	19.91	1.92	21.83	**0.059**	**0.922**

There some bar diagrams have drawn to see a clear picture of picture of the power and energy consumption analysis.

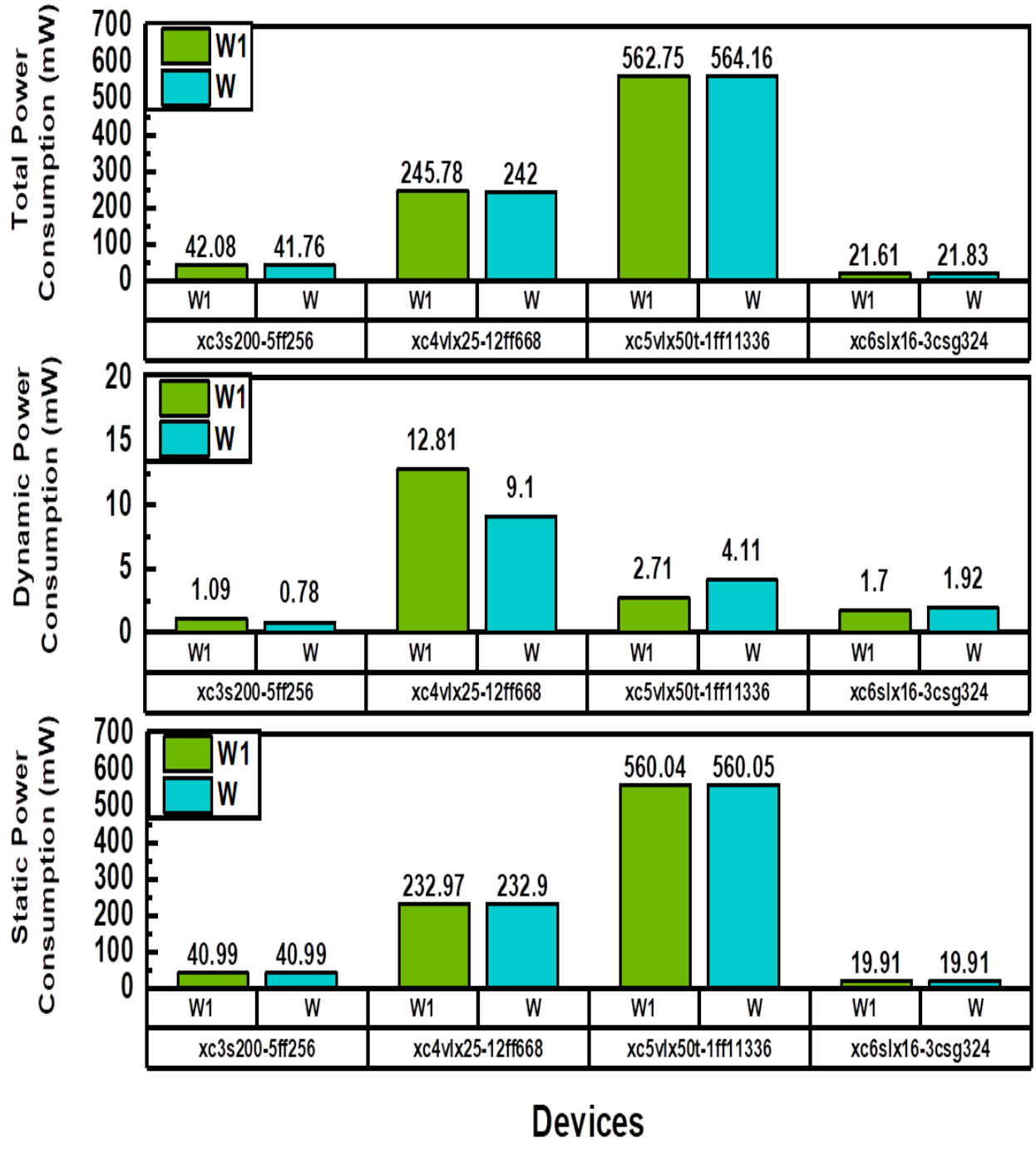

Figure 6.17: Comparison of Power Consumption at 13.56 MHz between both the Architectures.

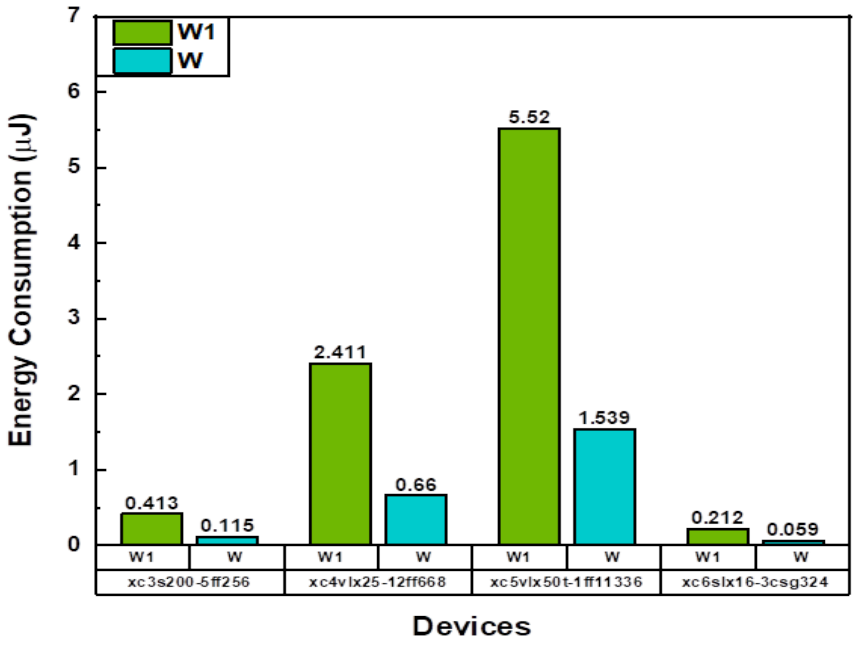

Figure 6.18: Comparison of Energy Consumption at 13.56 MHz between both the Architectures.

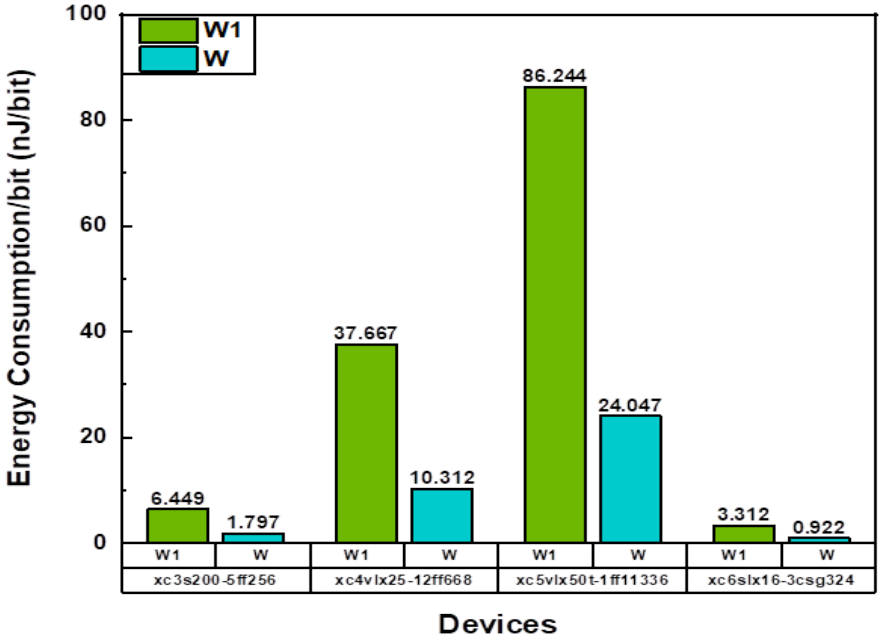

Figure 6.19: Comparison of Energy consumption per bit at 13.56 MHz between both the Architectures.

Chapter 7

Conclusion and Scope of Future Work

7.1 Conclusion

In this work, a high-performance and energy-efficient 16-bit architecture with 80-bit key_size for PRESENT block cipher and its FPGA implementation has been presented. The architecture has been synthesized on LUT-4 based FPGAs like Virtex-4 xc4vlx25-12ff668 and Spartan-3 xc3s200-5ff256 as well as on LUT-6 based modern FPGAs like Virtex-5 xc5vlx50t-3ff11336 and Spartan-6 xc6slx16-3csg324. Proposed architecture have the flexibility that either key can be fixed or changed for each 64-bit data block. The proposed architecture provides a throughput of 23.46 Mbps at 13.56 MHz and its latency is 37 clock cycles which are best among existing 16-bit architectures of PRESENT algorithm. Throughput at maximum frequency for different FPGA devices is also best among existing 16-bit PRESENT architectures. The *throughput-per-slice* at a particular frequency is one crucial parameter as it combines the effect of area and performance. Which actually shows the speed of operation for a particular amount of hardware. By reducing this parameter one can make a better trade-off between area and performance.

In the power analysis, static power for all the architectures which are used to compare our results including proposed architecture are same for a particular board.

So it is concluded that static power is independent of architecture rather it is only dependent on the device on which it is implemented. Dynamic power contributes negligible in total power as it is very less than the static power. Moreover, it is also nearly same for set of 16-bit PRESENT architectures. So power consumption is nearly same.

Now coming to energy analysis, Energy parametes acquired by analysis are expceptionally well. As energy is a derived parameter from latency and total power consumption. Because power consumption is nearly same and latency is reduced to a great extent which actually reduces energy consumption reduces around 72% on set of four FPGA devices. Now in order to make an optimize device, some derived parameters are needed, which can shows a combined effect of more than one parameter.In our work, the *throughput-per-slice*(Thr*/slice) and *energy-per-bit*(Energy/bit) at the frequency of 13.56 MHz are considered as such parameters.

The *throughput-per-slice*(Thr*/slice) at 13.56 MHz is also improved very much as compared to existing 16-bit architecture. While resource utilization is nearly same and also it also improved in LUT-6 based FPGA devices. Which shows that the proposed architecture performs better on modern devices. The parameter,*energy-per-bit*(Energy/bit) at the frequency of 13.56 MHz is also improved to a great extent. It reduces around 72% than the existing set of 16-bit architectures. It shows that the our architecture consumes around 72% lesser energy than the existing 16-bit architectures of PRESENT block cipher.

7.2 Scope of Future Work

As increase in the number of CPS and IoT devices demands of security related algorithms will be rapidly increasing continuously and to provide this security at cheaper cost will also be a challenging task and designer also is not supposed to compromise with it's performance either in terms of area, power consumption and energy consumption etc. So the demand of constraint based security algorithms will be increasing rapidly.

There is a vast scope of improvement this field but according to us there are two important levels of abstraction which are algorithmic level and architectural level. At algorithmic level, these algorithms can be modified and also can me made from scratch according to industry requirements. It is also a great idea to combine properties of two different algorithms to make a new algorithm for the market. Improvement is also possible in architectural level. Architecture is improved in our work also. Architecture decides the data flow and it's management with time. FSM plays an important role to finish this task efficiently.

There should be implemented a decryption algorithm for a particular encryption algorithm. So scope of implementation decryption is also very much needed. At the same time improvement in decryption algorithm is also possible on both of the levels which we disscused for the improvement of Encryption algorithms. The metrics of improvement are also quiet similar to the encryption algorithms.

Bibliography

[1] M. Alioto, *Enabling the Internet of Things: From Integrated Circuits to Integrated Systems*. Springer, 2017.

[2] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of iot systems: Design challenges and opportunities," in *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2014, pp. 417–423.

[3] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2007, pp. 450–466.

[4] J. G. Pandey, T. Goel, M. Nayak, C. Mitharwal, S. Khan, S. K. Vishvakarma, A. Karmakar, and R. Singh, "A vlsi architecture for the present block cipher with fpga and asic implementations," in *International Symposium on VLSI Design and Test*. Springer, 2018, pp. 210–220.

[5] X. Liu, M. Zhao, S. Li, F. Zhang, and W. Trappe, "A security framework for the internet of things in the future internet architecture," *Future Internet*, vol. 9, no. 3, p. 27, 2017.

[6] C. A. Lara-Nino, M. Morales-Sandoval, and A. Diaz-Perez, "Novel fpga-based low-cost hardware architecture for the present block cipher," in *2016 Euromicro Conference on Digital System Design (DSD)*. IEEE, 2016, pp. 646–650.

[7] C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval, "Lightweight hardware architectures for the present cipher in fpga," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2544–2555, 2017.

[8] C. Rolfes, A. Poschmann, G. Leander, and C. Paar, "Ultra-lightweight implementations for smart devices–security for 1000 gate equivalents," in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2008, pp. 89–103.

[9] M. Sbeiti, M. Silbermann, A. Poschmann, and C. Paar, "Design space exploration of present implementations for fpgas," in *2009 5th Southern Conference on Programmable Logic (SPL)*. IEEE, 2009, pp. 141–145.

[10] E. B. Kavun and T. Yalcin, "Ram-based ultra-lightweight fpga implementation of present," in *2011 International Conference on Reconfigurable Computing and FPGAs*. IEEE, 2011, pp. 280–285.

[11] N. Hanley and M. ONeill, "Hardware comparison of the iso/iec 29192-2 block ciphers," in *2012 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2012, pp. 57–62.

[12] J. G. Pandey, T. Goel, and A. Karmakar, "An efficient vlsi architecture for present block cipher and its fpga implementation," in *International Symposium on VLSI Design and Test*. Springer, 2017, pp. 270–278.

[13] P. Yalla and J.-P. Kaps, "Lightweight cryptography for fpgas," in *2009 International Conference on Reconfigurable Computing and FPGAs*. IEEE, 2009, pp. 225–230.

[14] J. Tay, M. Wong, M. Wong, C. Zhang, and I. Hijazin, "Compact fpga implementation of present with boolean s-box," in *2015 6th Asia Symposium on Quality Electronic Design (ASQED)*. IEEE, 2015, pp. 144–148.