IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering and Technology
Vol. 6, Issue 4, April 2021

Improving Security Using Modified S-Box For Aescryptographic Primitives

[1] S. Lakshmi,[2] B.Hemalatha,[3] Sadhana.D

[1] Head Of The Department, Department of Electronics and Communication Engineering, Thirumalai Engineering College, Kilambi, Kanchipuram- 631551, Anna University, Chennai, India.

[2][3] Assistant Professor, Department of Electronics and Communication Engineering, Thirumalai Engineering College, Kilambi, Kanchipuram- 631551, Anna University, Chennai, India.

Abstract: The success of AES encryption standard created challenges for the cryptographers to construct strong substitution-boxes using different underlying approaches. It is because they are solely responsible to decide the robustness of cryptosystem against linear and differential cryptanalysis. With an aim to full the mentioned requirement of robustness, a novel group theoretic and graphical method is proposed to construct S-box with optimal features. Firstly, a strong S-box is generated with the help of orbits of coset graphs and the action of proposed powerful permutation of symmetric group S256. In addition, a septic group is designed the action of whose pairs of permutations has the ability to generate as many as 462422016 strong S-boxes. Few of such proposed S-boxes are reported and assessed against standard performance parameters to validate the effectiveness of proposed Sending. The features of proposed S-boxes are compared with most of the recent boxes to validate the superior performance. Moreover, they are also applied for image encryption to demonstrate their suitability for multimedia security applications.

INTRODUCTION

Electronic systems have proliferated over the past few decades to the point that most aspects of daily life are aided or affected by the automation, control, monitoring, or computational power provided by Integrated Circuits (ICs). The ability to trust these ICs to perform their specified operation (and only their specified operation) has always been a security concern and has recently become a more active topic of research.

ICs are the building blocks for all modern electronic systems; they form the information backbone of many critical sectors including the financial, military, industrial, and transportation sectors. Without trust in these ICs, the systems they support cannot necessarily be trusted to perform as specified and may even be susceptible to attack by a malicious adversary.

A new disruptive threat has surfaced over the past five years1, a hardware-based security threat known as the Hardware Trojan. A Hardware Trojan is a malicious, undesired, intentional modification of an electronic circuit or design, resulting in the incorrect behaviour of an electronic device when in operation. Akin to a software Trojan Horse program (Thompson 1984), a Hardware Trojan is a back-door that can be inserted into hardware. The added advantages of a Hardware Trojan include residing at the lowest level of information processing, and being persistent – the threat is present as long as the infected hardware is in use. A Hardware Trojan may be able to defeat any and all security mechanisms (software or hardware-based) and subvert or augment the normal operation of an infected IC.

The modification can affect any type of IC, including processors, memory, Digital Signal Processors (DSPs), Application Specific Integrated Circuits (ASICs), and even configuration bit-streams for reconfigurable logic. The undesired behaviour can include modifications to the functionality or specification of the hardware, the leaking of sensitive information, or a Denial of Service (DoS) attack. Effects may range from a subtle degradation of service through to a complete and permanent shut-down of a system.

Hardware Trojans can affect a system by themselves alone, or provide a foothold for software based attacks (King et al. 2008), where colluding software is aware of the inserted Trojan. Hardware Trojans most often remain dormant and are only activated when triggered by, for example, the passing of time, some specific activity, or external events. The spectrum of Hardware Trojans – their capabilities, size, trigger mechanisms, and payloads – is enormous, and the state-space of IC development both physically and procedurally provides ample opportunity for concealment.

The ease with which Hardware Trojans can make their way into modern ICs and electronic designs is concerning. Modifications to hardware can occur at any stage during the design and manufacturing process, including the specification, design, verification and manufacturing stages. Hardware Trojans may even be retro-fitted to existing ICs post manufacture, as proposed by Abramovici & Bradley (2009). Maintaining tight control over the IC design life-cycle is costly; the current trend is towards separation of design from manufacture, relying on a handful of large CMOS fabrication facilities world-wide, mostly located within Asia.

This trend towards out-sourcing is not limited to manufacture; designers trust third-party Electronic Design Automation (EDA) tools, utilize third-party Intellectual Property (IP) cores – many of which are provided only as a binary net-list and outsource to contract-design houses. With so many entities and individuals involved, there is ample opportunity to insert a Trojan into the final hardware. A Hardware Trojan might be as simple as a paragraph change in the specification, an extra source-code line in the Hardware Description Language (HDL), a modification of the silicon die at the fabrication plant, or just a modulation of the CMOS geometries used.

Most ASIC designs are too large for either exhaustive testing or formal verification, so it is likely that many Trojans will never be detected. The life-cycle stage at which a Trojan is inserted and its logical structure also affects the effectiveness of existing detection methods.

This report introduces the threats posed by Hardware Trojans and the variety of manners in which they might be activated. It also presents the state-of-the-art in prevention, detection, and countermeasure techniques currently being researched. Unfortunately, current methods are not suitable for providing adequate protection – at best statistical guarantees can be provided on preventing or detecting a Hardware Trojan. In the near future, if not now, it will become necessary for hardware systems to be able to operate securely in the presence of Hardware Trojans. Recently, researchers have looked at operating securely despite the potential presence of Hardware Trojans.

1.2 THREATS

Due to the large number of insertion vectors and potential low-level actions a Hardware Trojan could perform, the danger posed by this threat is grave. Hardware Trojans are persistent; once a system has been infected the threat remains whenever the system is powered on. Hardware Trojans have the potential ability to undermine confidence in all modern electronic systems, by infecting and changing the behaviour of any integrated circuit. The effects of a Hardware Trojan can range from simple targeted attacks through to sophisticated attacks that provide footholds for higher level software attacks. Example targeted attacks could include performing a bit-flip that changes the integrity of stored data, weakening the functionality of cryptographic cores, or leaking confidential information.

Systems may also be infected by multiple Hardware Trojans that can together subvert a supposedly secure system. Although the threat is ultimately bounded by the actions a Hardware Trojan can take, considering the insertion vectors and activation mechanisms at the same time gives a greater insight into the power and stealth of a Hardware Trojan. Hardware Trojan threats, insertion vectors, and activation mechanisms will continue to develop concurrently and will need to be countered by the best prevention, detection, and countermeasure techniques to maintain the security of deployed ICs.

When examining the threat posed by a Hardware Trojan, one must first consider its inherent attributes in order to determine its effect on an information system. There have been several Hardware Trojan taxonomies proposed to describe such attributes, with the aim to enable a systematic study of Hardware Trojan characteristics, to aid the development of detection and mitigation techniques for given classes, and to facilitate benchmarking for detection and mitigation strategies.

2. LITERATURE SURVEY

1. TITLE : Construction of Non-linear Component of Block Cipher by Means of Chaotic Dynamical System and Symmetric Group

AUTHOR : ADNAN JAVEED

YEAR : 2020

DESCRIPTION:

The interesting features of chaos theory are utilized now a day's in information security. The simplest chaotic dynamical system is the double pendulum. Here in this article, two double pendulums are used to enhance the chaotic behavior of a dynamical system. This system is sensitive to initial conditions and bears complex and chaotic trajectory. Moreover, being multi dimensional system it endures grander solution space for the generation of large number of S-boxes. Furthermore, a permutation comprising on only two cycles of symmetric group of order 256 is applied to generate integer values for the construction of desired substitution box. The algebraic analysis of suggested S-box emphasis on its application, thereafter, an image is encrypted with the help of this S-box, whose statistical analysis validates its efficacy.

ADVANTAGES:

Effectiveness and improved performance

DISADVANTAGES:

Energy and cost problem

2. TITLE : A NOVEL CONSTRUCTION OF EFFICIENT SUBSTITUTION-BOXES USING CUBIC FRACTIONAL TRANSFORMATION

AUTHOR : Amjad Hussain Zahid.

YEAR : 2019

DESCRIPTION:

A symmetric block cipher employing a substitution–permutation duo is an effective technique for the provision of information security. For substitution, modern block ciphers use one or more substitution boxes (S-Boxes). Certain criteria and design principles are fulfilled and followed for the construction of a good S-Box. In this paper, an innovative technique to construct substitution-boxes using our cubic fractional transformation (CFT) is presented. The cryptographic strength of the proposed S-box is critically evaluated against the state of the art performance criteria of strong S-boxes, including bijection, nonlinearity, bit independence criterion, strict avalanche effect, and linear and differential approximation probabilities.

ADVANTAGES:

Lower power consumption

DISADVANTAGES:

To minimize the energy cost

3. PROPOSED BLOCK DIAGRAM

INTRODUCTION

ModelSim is a verification and simulation tool for VHDL, Verilog, SystemVerilog, and mixedlanguage designs. This lesson provides a brief conceptual overview of the ModelSim simulation environment. It is divided into fourtopics, which you will learn more about in subsequent lessons.

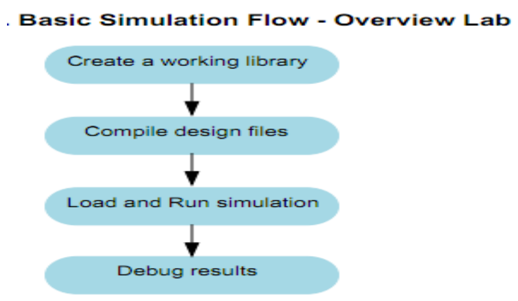
• Basic simulation flow

• Project flow

• Multiple library flow

• Debugging tools

Basic Simulation Flow:



1. Creating the Working Library

In ModelSim, all designs are compiled into a library. You typically start a new simulation in ModelSim by creating a working library called "work," which is the default lvibrary name used by the compiler as the default destination for compiled design units.

2.Compilng Your Design

After creating the working library, you compile your design units into it. The ModelSim library format is compatible across all supported platforms. You can simulate your design on any platform without having to recompile your design.

Loading the Simulator with Your Design and Running the Simulation With the design compiled, you load the simulator with your design by invoking the simulator on a top-level module (Verilog) or a configuration or entity/architecture pair (VHDL). Assuming the design loads successfully, the simulation time is set to zero, and you enter a run command to begin simulation.

3.Debugging Your Results

If you don’t get the results you expect, you can use Model Sim’s robust debugging environment to track down the cause of the problem.

INTRODUCTION TO EDA:

The Altera Quartus II design software provides a complete, multiplatform design environment that easily adapts to your specific design needs. It is a comprehensive environment for system-on-a-programmable-chip (SOPC) design. The Quartus II software includes solutions for all phases of FPGA and CPLD design (Figure 1).

Graphical User Interface Design Flow

You can use the Quartus II software graphical user interface (GUI) to perform all stages of the design flow. Figure 2 shows the Quartus II GUI as it appears when you first start the software.

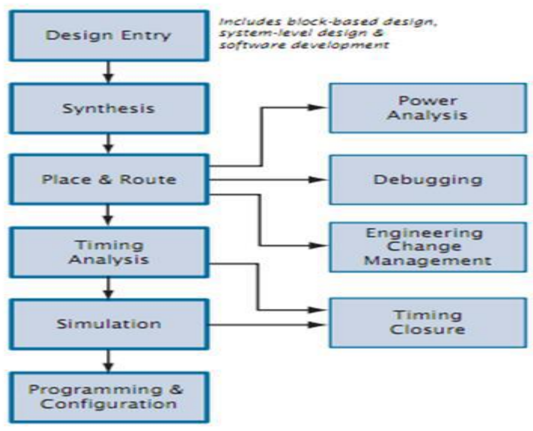
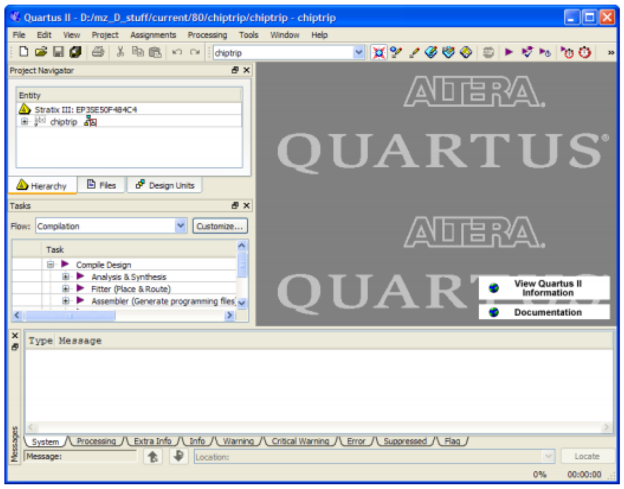

Design Entry
Includes block-based design, system-level design & software development
Synthesis
Place & Route
Timing Analysis
Simulation
Programming & Configuration
Power Analysis
Debugging
Engineering Change Management
Timing Closure

Figure 2. Quartus II Graphical User Interface



The Quartus II software includes a modular Compiler. The Compiler includes the following modules (modules marked with an asterisk are optional during a compilation, depending on your settings):

■ Analysis & Synthesis

■ Partition Merge*

■ Fitter

■ Assembler*

■ TimeQuest Timing Analyzer*

■ Design Assistant*

■ EDA Netlist Writer*

■ HardCopy®

Netlist Writer*

To run all Compiler modules as part of a full compilation, on the Processing menu, click Start Compilation. You can also run each module individually by pointing to Start on the Processing menu, and then clicking the command for the module you want to start. In addition, you can use the Tasks window to start Compiler modules individually The Tasks window also allows you to change settings or view the report file for the module, or to start other tools related to each stage in a flow.

Summary:

Hardware Trojans are a present and ongoing threat to the security of electronic systems world-wide. The Australian Military has particular concern due to the outsourcing of design and manufacture of integrated electronic components and our reliance on COTS components to maintain capability. Hardware Trojans threaten to compromise the integrity of data and operations performed by any system containing integrated electronic components. The threats include functional and specification modifications, leaking of sensitive information and Denial of Service attacks.

Hardware Trojan payloads and their activation mechanisms can take advantage of the massive state-space formed by the combination of parallel logic, internal routing, and input and output, that exists within a modern IC. The resulting Hardware Trojans can remain hidden deep within the design of an IC, having very poor observability. Efforts to prevent Hardware Trojans being designed or manufactured into ICs are still in their infancy. Much current research is focused on post-fabrication detection

SOFTWARE DESCRIPTION

FPGA

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by the customer or designer after manufacturing—hence "field-programmable".

The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit (ASIC) (circuit diagrams were previously used to specify the configuration, as they were for ASICs, but this is increasingly rare). FPGAs can be used to implement any logical function that an ASIC could perform.

FPGAs contain programmable logic components called "logic blocks", and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together"—somewhat like a one-chip programmable breadboard.

Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

Introduction

The area of field programmable gate array (FPGA) design is evolving at a rapid pace. The increase in the complexity of the FPGA's architecture means that it can now be used in far more applications than before. The newer FPGAs are steering away from the plain vanilla type "logic only" architecture to one with embedded dedicated blocks for specialized applications.

Definitions of Relevant Terminology are Field-programmable Device (FPD) — a general term that refers to any type of integrated circuit used for implementing digital hardware, where the chip can be configured by the end user to realize different designs.

PLA — a Programmable Logic Array (PLA) is a relatively small FPD that contains two levels of logic, an AND-plane and an OR-plane, where both levels are programmable.

PAL— a Programmable Array Logic (PAL) is a relatively small FPD that has a programmable AND-plane followed by a fixed OR-plane. SPLD — refers to any type of Simple PLD, usually either a PLA or PAL. CPLD — a more Complex PLD that consists of an arrangement of multiple SPLD-like blocks on a single chip.

The FPGA Landscape

In the semiconductor industry, the programmable logic segment is the best indicator of the progress of technology. No other segment has such varied offerings as field programmable gate arrays. It is no wonder that FPGAs were among the first semiconductor products to move to the 0.13µm technology, and again recently to 90nm technology

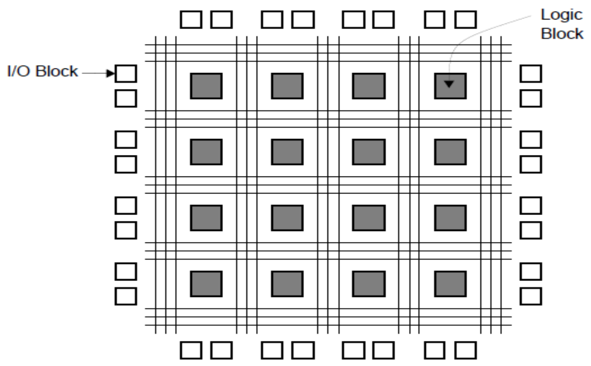


Figure.4.1 Structure of an FPGA

4.2 Altera DE0 Board

The DE0 board has many features that allow the user to implement a wide range of designed circuits, from simple circuits to various multimedia projects. The following hardware is provided on the DE0 board:

- Altera Cyclone® III 3C16 FPGA device
- Altera Serial Configuration device – EPCS4
- USB Blaster (on board) for programming and user API control; both JTAG and Active Serial
- (AS) programming modes are supported
- 8-Mbyte SDRAM

IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 6, Issue 4, April 2021

• 4-Mbyte Flash memory
• SD Card socket
• 3 pushbutton switches
• 10 toggle switches
• 10 green user LEDs
• 50-MHz oscillator for clock sources
VGA DAC (4-bit resistor network) with VGA-out
• RS-232 transceiver
• PS/2 mouse/keyboard connector
Two 40-pin Expansion Headers

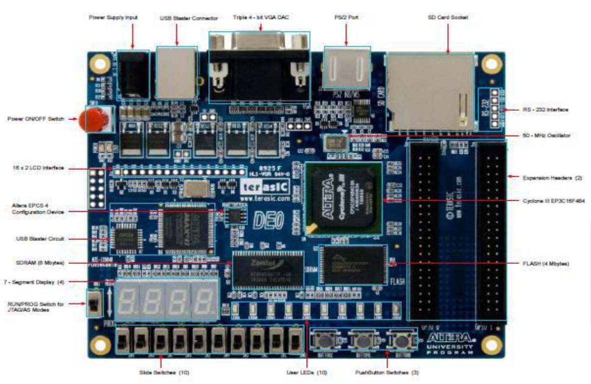
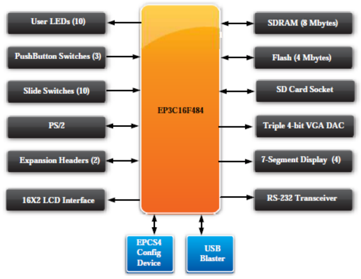


Figure 4.2 The DE0 board.

Block Diagram of the DE0 Board:

Figure 4.2 gives the block diagram of the DE0 board. To provide maximum flexibility for the user, all connections are made through the Cyclone IIII FPGA device. Thus, the user can configure the FPGA to implement any system design.



HARDWARE AND SOFTWARE USED

5.1 SIMPLE ASIC DESIGN FLOW

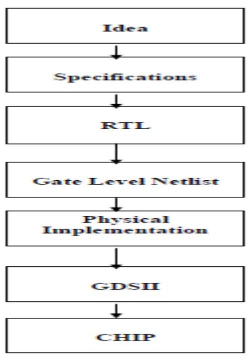


Figure 5.1: SIMPLE ASIC DESIGN FLOW

For any design to work at a specific speed, timing analysis has to be performed. We need to check whether the design is meeting the speed requirement mentioned in the specification. This is done by Static Timing Analysis Tool, for example Primetime.

RESULTS AND OUTPUTS

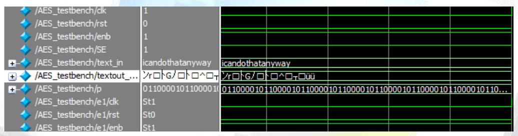
OUTPUTS

SYNTHESIS AREA RESULT:

The synthesis area report shows the total number of cells and nets in the netlist. It also uses the area parameter associated with each cell in the LSI_10K library file, to calculate the total combinational and sequential area of the netlist. The total area of the gate level netlist is unknown since it depends on total area of the interconnects, which itself is a function of the wiring load model used in physical design. The total cell area in the netlist is reported as 22978 units, which is the sum of combinational and sequential areas. The synthesis area report is shown below:

SYNTHESIS CONSTRAINT VIOLATORS RESULT:

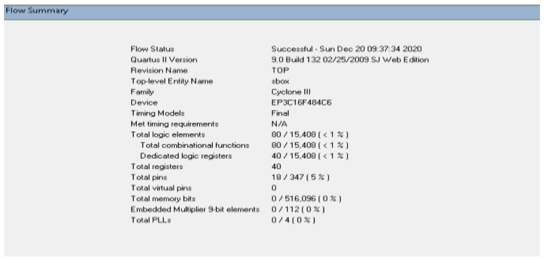
To enforce the synthesis tool to create the most compact netlist, the area of the gate level netlist was constrained to zero during the synthesis process. As a result, the only constraint violation, which is expected, is related to the area as shown bellow:



```
/AES_testbench/clk          1
/AES_testbench/rst          0
/AES_testbench/enb          1
/AES_testbench/SE           1
/AES_testbench/text_in      icandothatanyway     icandothatanyway
/AES_testbench/textout_...  ﾝr□ﾄG/□ﾄ□^□┬         ﾝr□ﾄG/□ﾄ□^□┬□üü
/AES_testbench/p            0110000101100001(    0110000101100001011000010110000101100001011000010110000101110...
/AES_testbench/e1/clk       St1
/AES_testbench/e1/rst       St0
/AES_testbench/e1/enb       St1
```

Figure 6.1 cipher transformation
Figure 6.1: Model Sim simulation report

AREA UTILIZATION REPORT:

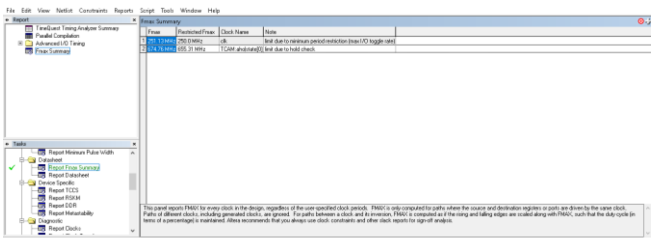


Flow Summary

Flow Status	Successful - Sun Dec 20 09:37:34 2020
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	TOP
Top-level Entity Name	sbox
Family	Cyclone III
Device	EP3C16F484C6
Timing Models	Final
Met timing requirements	N/A
Total logic elements	80 / 15,408 (< 1 %)
Total combinational functions	80 / 15,408 (< 1 %)
Dedicated logic registers	40 / 15,408 (< 1 %)
Total registers	40
Total pins	18 / 347 (5 %)
Total virtual pins	0
Total memory bits	0 / 516,096 (0 %)
Embedded Multiplier 9-bit elements	0 / 112 (0 %)
Total PLLs	0 / 4 (0 %)

Figure.6.2: Flow summary report.

PERFORMANCE REPORT

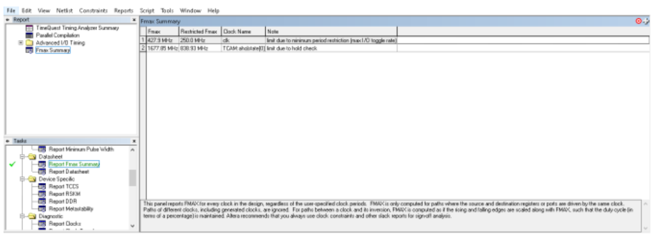


Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	251.13 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)
2	674.76 MHz	655.31 MHz	TCAM:aho|state[0]	limit due to hold check

This panel reports FMAX for every clock in the design, regardless of the user-specified clock periods. FMAX is only computed for paths where the source and destination registers or ports are driven by the same clock. Paths of different clocks, including generated clocks, are ignored. For paths between a clock and its inversion, FMAX is computed as if the rising and falling edges are scaled along with FMAX, such that the duty cycle (in terms of a percentage) is maintained. Altera recommends that you always use clock constraints and other slack reports for sign-off analysis.

Figure. 6.3(a) Fmax summary report (slow corner)



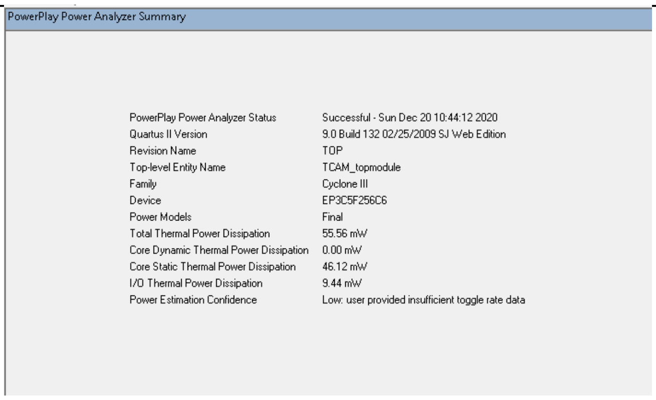
Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	427.9 MHz	250.0 MHz	clk	limit due to minimum period restriction (max I/O toggle rate)
2	1677.85 MHz	838.93 MHz	TCAM:aho|state[0]	limit due to hold check

This panel reports FMAX for every clock in the design, regardless of the user-specified clock periods. FMAX is only computed for paths where the source and destination registers or ports are driven by the same clock. Paths of different clocks, including generated clocks, are ignored. For paths between a clock and its inversion, FMAX is computed as if the rising and falling edges are scaled along with FMAX, such that the duty cycle (in terms of a percentage) is maintained. Altera recommends that you always use clock constraints and other slack reports for sign-off analysis.

Figure.6.3 (b) Fmax summary report (fast corner)

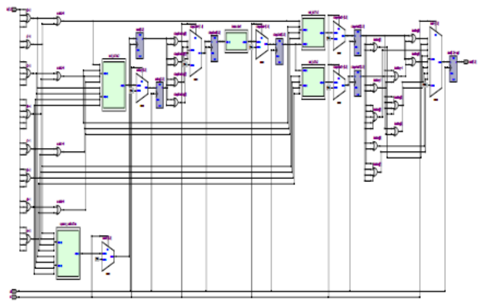
IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 6, Issue 4, April 2021


PowerPlay Power Analyzer Summary

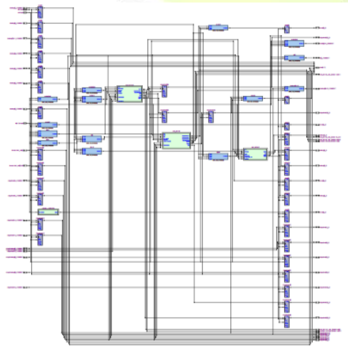
```
PowerPlay Power Analyzer Status           Successful - Sun Dec 20 10:44:12 2020
Quartus II Version                        9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name                             TOP
Top-level Entity Name                     TCAM_topmodule
Family                                    Cyclone III
Device                                    EP3C5F256C6
Power Models                              Final
Total Thermal Power Dissipation           55.56 mW
Core Dynamic Thermal Power Dissipation    0.00 mW
Core Static Thermal Power Dissipation     46.12 mW
I/O Thermal Power Dissipation             9.44 mW
Power Estimation Confidence               Low: user provided insufficient toggle rate data
```

Figure.6.4: power dissipation report

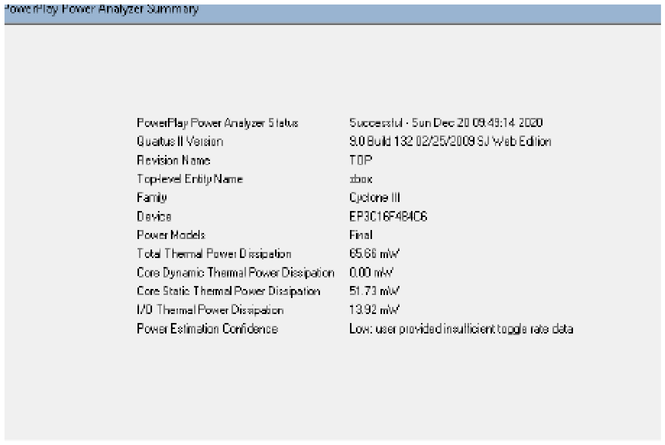
SYNTHESIS REPORT:



MAP VIEWER



POWER ANALYZES:


PowerPlay Power Analyzer Summary

```
PowerPlay Power Analyzer Status           Successful - Sun Dec 20 09:49:14 2020
Quartus II Version                        9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name                             TOP
Top-level Entity Name                     xbox
Family                                    Cyclone III
Device                                    EP3C16F484C6
Power Models                              Final
Total Thermal Power Dissipation           65.66 mW
Core Dynamic Thermal Power Dissipation    0.00 mW
Core Static Thermal Power Dissipation     51.73 mW
I/O Thermal Power Dissipation             13.92 mW
Power Estimation Confidence               Low: user provided insufficient toggle rate data
```

Figure.6.6 Power dissipation report

IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 6, Issue 4, April 2021

TABLE I logical depth analyzes of various S_BOX types with QUARTUS II

SBOX type	AREA	SPEED
S box –case 1	92	170.59 MHz
S box –case 2	83	431.22 MHz
S box –case 3	80	525.21 MHz

hardware synthesis using CYCLONE III family (EP3C16F484C6)

CONCLUSION

Here in this we carried out implementation of AES cryptographic algorithms with logical depth analyzes. It has been previously demonstrated that s-box always introduced most cases delay and key expansion module open a back door to potential attacks. In our project we studied in detail the mathematical foundations for LUT based S-box AES systems, basically the concepts of rings, fields, groups, Galois finite fields and their properties. The various algorithms for the computation of the scalar product of a point were studied and their complexity were analyzed.
The advantage of this over the other type of AES systems are proved by parameters .The key strength of this systems in comparison to other S-BOX type combosite fields can be used in all levels of algorithm implementation and this will increase reliability.

FUTURE ENHANCEMENT

Public-key infrastructures are secure, but only to the extent that private keys of individuals are maintained secret. Here are going to describe retinal based cryptogrphy which involves securing the private key(s) In order to provide a higher degree of security for embedded computing devices to propose hybrid pseudorandom bit sequences generator to overcome the issue of key management in any symmetric block ciphers.. Further, with various level of rounds and random memory location selection based key extraction from biometrics can secure keys with greater confidence and with a higher level of security will be compared to other security mechanisms

REFERENCES

1. M. Akkar and C. Giraud, “An Implementation of DES and AES, Secure against Some Attacks,” In Proc. of the Workshop on Cryptographic Hardware and Embedded Systems (CHES2001), Paris, France, pp. 315-325, May 2001.
2. R. Anderson, E. Biham, and L. Knudsen, “Serpent: A Proposal for the Advanced Encryption Standard,” AES algorithm submission, June 1998.
3. G. Bertoni, L. Breveglieri, I. Koren, P. Maistri, and V. Piuri, “Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard,” IEEE Trans. on Computers, vol. 52, no. 4, pp. 492-505, April 2003.
4. G. Bertoni, L. Breveglieri, I. Koren, and P. Maistri, “An efficient hardwarebased fault diagnosis scheme for AES: performances and cost,” In Proc. of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT2004), Cannes, France, pp. 130-138, Oct. 2004.
5. D. Boneh, R. A. DeMillo, and R. J. Lipton, “On the Importance of Eliminating Errors in Cryptographic Computations,” Journal of Cryptology, vol. 14, no. 2, pp. 101-119, 2001.
6. L. Breveglieri, I. Koren, and P. Maistri, “Incorporating Error Detection and Online Reconfiguration into a Regular Architecture for the Advanced Encryption Standard,” In Proc. of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT2005), Monterey, CA, USA, pp. 72-80, Oct. 2005.
7. C. Burwick et al., “MARS-A Candidate Cipher for AES,” AES algorithm submission, August 1999, available at http://www.nist.gov/.
8. D. Canright, “A Very Compact Rijndael S-box,” Naval Postgraduate School Technical Report: NPS-MA-05-001, May 2005.
9. G. C. Cardarilli, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, “Fault localization, error correction, and graceful degradation in radix 2 signed digit- based adders,” IEEE Trans. on Computers, vol. 55, no. 5, pp. 534-540, May 2006.
10. G. C. Cardarilli, S. Pontarelli, M. Re, and A. Salsano, “A self checking Reed Solomon encoder: design and analysis,” In Proc. of the IEEE International
Symposium on Defect and Fault Tolerance in VLSI Systems (DFT2005), Monterey, CA, USA, pp. 111-119, Oct. 2005.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 6, Issue 4, April 2021

11. A. Elbirt, W. Yip, B. Chetwynd, and C. Paar, “An FPGA-based performance evaluation of the AES block cipher candidate algorithm finalists,” IEEE Trans. of VLSI Systems, pp. 545-557, August 2001.
12. S. Fenn, M. Gossel, M. Benaissa, and D. Taylor, “On-Line Error Detection for Bit-Serial Multipliers in GF(2^m),” Journal of Electronic Testing: Theory and Applications, vol. 13, no. 1, August 1998.
13. A. Hodjat and I. Verbauwhede, “Area-Throughput Trade-Offs for Fully Pipelined30 to 70 Gbits/s AES Processors,” IEEE Trans. on Computers, vol, no. 4,pp. 366-372, April 2006.
14.T. Ichikawa et al, “Hardware Evaluation of the AES Finalists,” In Proc. 3th AES Candidate Conference, New York, April 2000.
15. R. Karri, W. Kaijie, P. Mishra, and K. Yongkook, “Fault-based Side-Channel ryptanalysis Tolerant Rijndael Symmetric Block Cipher Architecture.