Chapter 1: Introduction

A cyclic redundancy check (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to digital data. Blocks of data entering these systems get a short check value attached, based on the remainder of a polynomial division of their contents. On retrieval, the calculation is repeated and, in the event the check values do not match, corrective action can be taken against data corruption. CRCs can be used for error correction.

CRCs are so called because the check (data verification) value is a redundancy (it expands the message without adding information) and the algorithm is based on cyclic codes. CRCs are popular because they are simple to implement in binary hardware, easy to analyze mathematically, and particularly good at detecting common errors caused by noise in transmission channels. Because the check value has a fixed length, the function that generates it is occasionally used as a hash function.

Cryptography is technique of securing information and communications through use of codes so that only those people for whom the information is intended can understand it and process it. Thus, preventing unauthorized access to information. The prefix "crypt" means "hidden" and suffix "graphs" means "writing." In Cryptography the techniques which are used to protect information are obtained from mathematical concepts and a set of rule-based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on internet and to protect confidential transactions such as credit card and debit card transactions.

Techniques used For Cryptography: In today's age of computers cryptography is often associated with the process where an ordinary plain text is converted to cipher text which is the text made such that intended receiver of the text can only decode it and hence this process is known as encryption. The process of conversion of cipher text to plain text this is known as decryption.

The AES encryption algorithm is a block cipher that uses an encryption key and a several rounds of encryption. A block cipher is an encryption algorithm that works on a single block of data at a time. In the case of standard AES encryption, the block is 128 bits, or 16 bytes, in length. The term "rounds" refers to the way in which the encryption algorithm mixes the data re-encrypting it ten to fourteen times depending on the length of the key. This is described in the Wikipedia article on AES encryption. The AES algorithm itself is not a computer program or computer source code. It is a mathematical description of a process of obscuring data. Several people have created source code implementations of AES encryption, including the original authors.

AES encryption uses a single key as a part of the encryption process. The key can be 128 bits (16 bytes), 192 bits (24 bytes), or 256 bits (32 bytes) in length. The term 128-bit encryption refers to

the use of a 128-bit encryption key. With AES both the encryption and the decryption are performed using the same key. This is called a symmetric encryption algorithm. Encryption algorithms that use two different keys, a public and a private key, are called asymmetric encryption algorithms. An encryption key is simply a binary string of data used in the encryption process. Because the same encryption key is used to encrypt and decrypt data, it is important to keep the encryption key a secret and to use keys that are hard to guess. Some keys are generated by software used for this specific task. Another method is to derive a key from a pass phrase. Good encryption systems never use a pass phrase alone as an encryption key.

This approach adds an extra layer of error detection to ensure the integrity of the transmitted data, making it useful in scenarios where data transmission or storage is unreliable, such as over noisy channels or in wireless communications. However, it's important to note that CRC does not provide any additional security beyond what AES already provides. The primary means of security in this scenario is provided by AES, while CRC serves as an additional layer of error detection.

Chapter 2: Literature Survey

[1] Cryptosystem designers frequently assume that secret parameters will be manipulated in closed reliable computing environments, Kocher et al. reported that microchips leak information correlated with the data handled and introduced a new kind of attacks which were radically different from software and algorithmic attacks. These attacks use leaking or side-channel information, like power consumption data, electromagnetic emanations or computing time to recover the secret key. While FPGAs are becoming increasingly popular for cryptographic applications, there are only a few articles that assess their vulnerability to such attacks. This paper describes the principles of differential power analysis (DPA) attack and also illustrates a practical and successful implementation of this attack against an FPGA implementation of the advanced encryption standard (AES) algorithm. The results obtained in this work clearly demonstrate that DPA is a serious threat against realization of encryption algorithms on SRAM-based FPGAs without effective countermeasures.

[2] This article examines vulnerabilities to power analysis attacks between software and hardware implementations of cryptographic algorithms. A simulation-based experimental environment is built to acquire power data, and single-bit differential power analysis (DPA), and multi-bit DPA and correlation power analysis (CPA) attacks are conducted based on two implementations respectively. The experimental results show that the hardware implementation has less data-dependent power leakages to resist power attacks. Furthermore, an improved DPA approach is proposed. It adopts hamming distance of intermediate results as power model and arranges plaintext inputs to differentiate power traces to the maximal probability. Compared with the original power attacks, our improved DPA performs a successful attack on AES hardware implementations with acceptable power measurements and fewer computations.

[3] Differential Power Analysis (DPA) is a powerful and practical technique used to attack a cryptographic im plementation in a resource limited application environment. In this paper, they show that some intermediate values from the inner rounds can be exploited by deploying techniques such as fixing certain plaintext/cipher text bytes. We give five general principles on DPA vulnerability of unprotected AES implementations, and give several general principles on DPA vulnerability of protected AES implementations. These principles specify which operations of AES are vulnerable to first and second-order DPA. To illustrate the principles, we attack the two AES implementations [1, 2] that use two kinds of countermeasures to achieve a high resistance against power analysis, and demonstrate that they are even vulnerable to DPA. Finally, we conclude that at least the first two and a half rounds and the last three rounds of AES should be secured for an AES software implementation to be resistant against first and second-order DPA in practice.

[4] This paper presents novel high-speed architectures for the hardware implementation of the Advanced Encryption Standard (AES) algorithm. Unlike previous works which rely on look-up tables to implement the Sub Bytes and Inv Sub Bytes transformations of the AES algorithm, the proposed design employs combinational logic only. As a direct consequence, the unbreakable delay incurred by look-up tables in the conventional approaches is eliminated, and the advantage of sub pipelining can be further explored. Furthermore, composite field arithmetic is employed to reduce the area requirements, and different implementations for the inversion in subfield GF(2/sup 4/) are compared. In addition, efficient key expansion architecture suitable for the sub pipelined round units is also presented.

[5] A practical and efficient method to estimate the entropy rate of a TRNG based on free running oscillators that does not require outputting and analysing the clock signals with external equipment. Rather it relies on very simple computations that can be embedded in any logic device such as FPGA or ASIC. The method can be used for the calibration of an oscillator based TRNG or for online certification of its entropy rate. Our approach, which is inspired by the coherent sampling method, works under the general assumption that the period jitter is small compared to the period of the generated clock signal. We show that, in this case, it is possible to measure the relative phase between clocks of two oscillators with far higher precision than the time resolution given by the period of any internal clock signal. We use this observation to recover, under some reasonable heuristics, the distribution of the random walk component of the jitter, from which it is possible to obtain a lower bound on the entropy rate of the TRNG. Our method was thoroughly tested in simulations and in hardware. At the end of the paper, we draw some conclusions and make recommendations for a reliable implementation of TRNGs in cryptographic applications. In this paper, we presented a simple method to recover the statistical parameters of the random walk component of the jitter of an elementary TRNG.

[6] This paper presents a source-follower-delay-cell, multiloop ring oscillator that provides power-supply isolation. The main contributions of this work are a source-follower-based delay cell with a multiloop ring structure achieving improved supply rejection, a design-oriented analysis of the proposed structure to facilitate its use, and a layout technique allowing straightforward mask design for the multiloop oscillator. The oscillator also features differential control voltages to allow rejection of common-mode control and supply noise. The oscillator was fabricated in a UMC 90-nm CMOS pure logic process with no Analog components (regular VT), and the minimum measured incremental supply sensitivity is 0.003 [%-change %-change], which is more than 20 dB better than that of a conventional CMOS-delay-cell quadrature oscillator fabricated on the same test chip. The oscillator's measured tuning range is 0.63–8.1 GHz. Over the tuning range, the phase noise varies from 106 to 88 dBc/Hz at 10-MHz offset, and the power consumption ranges from 7 to 26 Mwfrom a 1-V supply. The measured mean quadrature accuracy performance is within to error including board parasitic without any trimming/tuning across the oscillator's frequency range. This paper has discussed the design, measurement, and

analysis of a source-follower-based MRO achieving 20-dB improvement in supply rejection over a conventional CMOS-inverter- based quadrature-oscillator delay-cell structure. The source-follower arrangement isolates the supply through the output resistance of the transistor, and furthermore, a "sweet-spot" bias exists that allows even further reduction of supply sensitivity ideally to zero incremental sensitivity.

[7] This brief covers the design and fabrication of a ring oscillator-based truly random number generator (TRNG), which was fabricated in 0.13-μm CMOS technology. The randomness originates from the phase noise in a ring oscillator. Timing jitter resulting from crossing the threshold multiple times, i.e., last passage time (LPT), is exploited. Previously, the jitter model was developed and applied to the core delay cell of the slow VCO, part of the ring oscillator, where a slow slew rate phase was introduced to greatly increase phase noise. In this brief, the successful design of the entire TRNG was performed. This includes designing the circuit to avoid introducing correlation in the TRNG. Toward this end, novel timing circuitry is designed to properly control both the beginning and termination of this slow slew rate phase by tapping into the previous stage's output. 1/f noise also has to be minimized. Furthermore, the entire TRNG is now designed/implemented and fabricated, and experimental results are shown. The fabricated ring oscillator was shown to possess a timing jitter of 1.5 ns. Simulation under PVT variations of the entire cell shows that jitter variations are within 30%, showing that the designed control circuit was able to perform under such PVT variations. Entropy simulation with power supply variations applied to the TRNG was also run to assess its effectiveness as the biasing condition is changing. The randomness of the entire TRNG was assessed by applying the National Institute of Standards and Technology (NIST) tests.

[8] Cryptography is fundamentally important for securing data, whether that data is in transit over the Internet or at rest on a storage device. In the past, cryptography relied on secret algorithms and codes. This, however, proved impractical when parties who didn't know each other in advance wanted to communicate securely. In contrast, modern encryption and authentication algorithms are public and have been extensively analysed. Today, cryptographic security depends primarily on having strong keys and keeping them secret. Deterministic random bit generators (DRBGs), also known as pseudorandom number generators (PRNGs), are used to generate keys, but the sequence of numbers they generate can be traced predictably to the seed (initial value) supplied to the generator. In other words, by knowing the seed, you can reconstruct the sequence of numbers a particular DRBG produces. Thus, DRBGs must be seeded with sufficient entropy from a reliable source. Entropy sources that provide true randomness are usually based on nondeterministic physical processes, such as ring oscillators, or unpredictable events, such as human-driven mouse movements and keyboard stroke timings. The importance of obtaining and using highly unpredictable keys isn't just an academic question. There are many practical examples showing that failure to obtain sufficient entropy compromises any security provided by long keys and sound algorithms. Even the best algorithms can't compensate for

weak keys generated using insufficient entropy. Such systems are vulnerable to attackers, with potentially disastrous results.

[9] This paper presents novel high-speed architectures for the hardware implementation of the Advanced Encryption Standard (AES) algorithm. Unlike previous works which rely on look-up tables to implement the Sub Bytes and Inv Sub Bytes transformations of the AES algorithm, the proposed design employs combinational logic only. As a direct consequence, the unbreakable delay incurred by look-up tables in the conventional approaches is eliminated, and the advantage of sub pipelining can be further explored. Furthermore, composite field arithmetic is employed to reduce the area requirements, and different implementations for the inversion in subfield GF(2/sup 4/) are compared. In addition, efficient key expansion architecture suitable for the sub pipelined round units is also presented.

[10] In this paper, we present a hardware implementation of an Advanced Encryption Standard (AES) Rijndael (128-bit block and 128-bit key) using Xilinx development tools and Spartan FPGA circuits. All the modules in this core are described by using VHDL language. The developed Rijndael core is aimed at providing sufficient performance with good area efficiency. In fact, the encryption/decryption data path operates at 29,45MHz resulting in a throughput of 289,98Mbits per second for the encryption and 157,1Mbits per second for decryption. Encryption/decryption circuit will fit in one Xilinx Spartan XC2S600E circuit taking approximately 87% of the area (6068 Slices).

Chapter 3: Problem Analysis & Solution

3.1 Problem Definition:

This approach adds an extra layer of error detection to ensure the integrity of the transmitted data, making it useful in scenarios where data transmission or storage is unreliable, such as over noisy channels or in wireless communications. However, it's important to note that CRC does not provide any additional security beyond what AES already provides. The primary means of security in this scenario is provided by AES, while CRC serves as an additional layer of error detection.

The problem at hand is to design and implement CRC-based cryptography using FPGA (Field-Programmable Gate Array) for secure data communication. The objective is to develop a system that can ensure the confidentiality and integrity of data during transmission over a communication channel. This entails designing a cryptographic algorithm based on CRC and incorporating encryption and decryption mechanisms. The algorithm will be implemented on an FPGA platform to leverage its reconfigurable nature for performance optimization. The system should establish a secure communication channel, encrypting data at the sender's end and decrypting it at the receiver's end. Performance optimization techniques such as parallel processing and hardware acceleration will be explored to achieve high-speed data processing. The functionality, security, and robustness of the cryptographic system will be thoroughly verified and tested. The design, implementation, and evaluation will be documented, providing clear instructions for integration into existing communication infrastructure. The system's performance, security features, and resource utilization will be evaluated to ensure a reliable and efficient solution for secure data communication.

3.2 Proposed Solution:

The data is first encrypted using AES, generating cipher text. Then a CRC checksum is calculated over the cipher text, producing a small value that is appended to the end of the cipher text. The entire message, including the checksum, is then transmitted, or stored.

When the receiver gets the message, they first perform the decryption process using the same AES key to generate the original plain text. Next, they calculate the CRC checksum over the decrypted message, and compare it to the checksum that was transmitted with the message. If the two checksums match, the receiver can be confident that the message was transmitted without errors. If the checksums do not match, the receiver knows that an error has occurred during transmission or storage of the message.

This approach adds an extra layer of error detection to ensure the integrity of the transmitted data, making it useful in scenarios where data transmission or storage is unreliable, such as over

noisy channels or in wireless communications. However, it's important to note that CRC does not provide any additional security beyond what AES already provides. The primary means of security in this scenario is provided by AES, while CRC serves as an additional layer of error detection.

In AES (Advanced Encryption Standard), the Cipher Feedback (CFB) mode of operation can be used with both segmented and non-segmented CRC (Cyclic Redundancy Check) design.

CRC is a type of error detection code that is commonly used in digital communication systems to detect errors in data transmission. In AES, the CRC is used to ensure the integrity of the data being transmitted.

In segmented CRC design, the message is divided into fixed-length segments and a CRC is calculated for each segment separately. This allows for easier error detection and correction since errors can be detected and corrected within each segment separately.

In non-segmented CRC design, the CRC is calculated over the entire message as a single block. This design is simpler and requires less computation, but it can be less effective at detecting errors since errors that occur in one part of the message can affect the entire CRC.

Both segmented and non-segmented CRC design can be used in AES-CFB mode, but the choice depends on the specific requirements of the application. Segmented CRC design may be preferred in applications where data is transmitted in fixed-length segments, while non-segmented CRC design may be preferred in applications where the data is transmitted as a single block.

Proposed System Technique: AES Cryptography with CRC Design

Proposed System Advantages: Low-energy high-throughput hardware AES encryption module providing multiple levels and High security with Best Attacks Prevention scheme

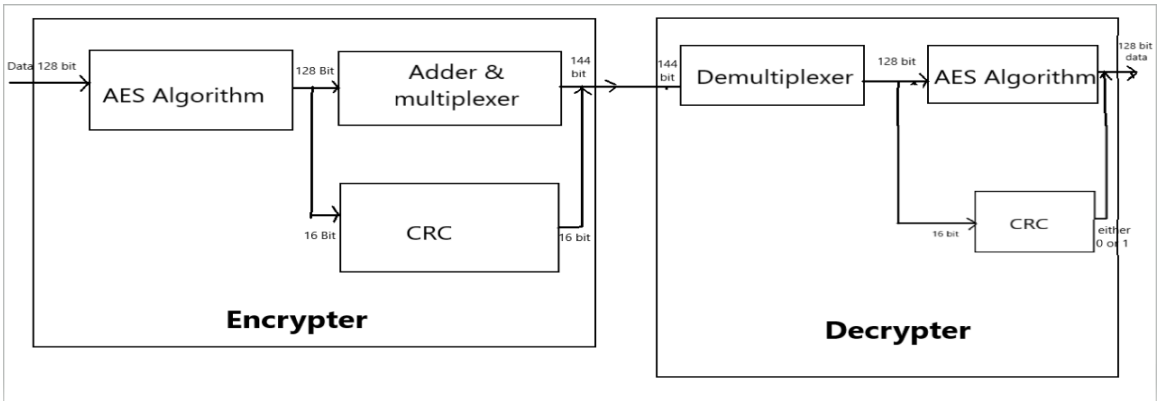


Fig:3.2.1. General Block Diagram

Chapter 4: Methodology and Implementation

4.1 Block Diagram:

4.1.1. Segmented CRC:

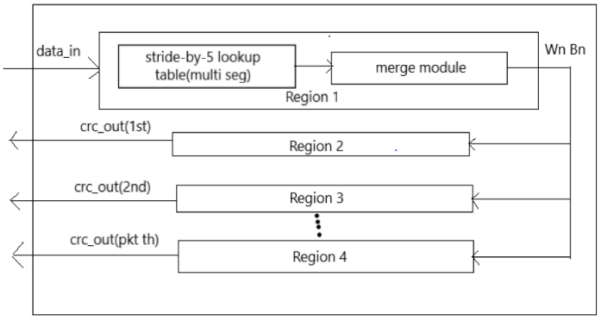


Fig 4.1.1. Segmented CRC

The non-segmented system architecture cannot process multiple frames in one word (clock), which reduces the throughput of short or misaligned frames. This is called the bus efficiency problem. A segmented system architecture is proposed to solve the problem. The bus format is the same as that, and the block is another name for the segment. For example, a 4096-bit bus can process eight complete frames set the same time; hence, the bus can be divided into eight regions. The number of regions depends only on the bus width. Different segment widths are feasible, and if a64-bit segment width is chosen, one region can be divided into eight segments (blocks). The proposed segmented system architecture is shown in above figure. Compared with the proposed non-segmented system architecture, the proposed segmented system architecture has a slightly more complex Region 1 and Region 2 and multiple duplicates of Region3 and Region 4.

Segmented CRC, also known as Cyclic Redundancy Check, is an error-detection algorithm commonly used in digital communication systems. It ensures the integrity of data transmitted over unreliable channels by adding a checksum to the data, which allows the receiver to detect any errors that may have occurred during transmission.

The Segmented CRC algorithm works by dividing the data into fixed-size segments and calculating a CRC value for each segment individually. These CRC values are then combined to form a final checksum for the entire data stream. When the data is received, the receiver performs the same calculation and compares the calculated checksum with the received checksum. If they match, it indicates that the data was transmitted without any errors. If they don't match, it suggests that errors might have occurred during transmission.

This segmented approach offers several advantages. Firstly, it allows for the detection of errors within specific segments of the data, enabling localized error correction or retransmission. Secondly, it provides a more efficient way to handle large data streams since the CRC calculations can be performed in parallel on different segments. This improves the overall speed and efficiency of the error-detection process.

To calculate the CRC value for each segment, the Segmented CRC algorithm uses a polynomial division method. A generator polynomial is chosen, and the data segment is treated as a polynomial. The polynomial is divided by the generator polynomial using modulo-2 division, and the remainder is the CRC value for that segment. The generator polynomial used in segmented CRC can vary depending on the specific application or protocol. Commonly used generator polynomials include CRC-16, CRC-32, and CRC-CCITT. These polynomials are standardized and widely implemented in various communication protocols and data storage systems.

One of the main advantages of the segmented CRC algorithm is its ability to localize errors. Instead of treating the entire data stream as a single entity, errors can be pinpointed to specific segments, enabling targeted error correction or retransmission. This localization feature is particularly useful in scenarios where only certain parts of the data need to be retransmitted, minimizing the impact on overall system performance. Furthermore, the segmented CRC approach improves efficiency when dealing with large data streams. Since the CRC calculations can be performed in parallel on different segments, it significantly reduces the time required for error detection. This parallel processing capability makes segmented CRC ideal for high-speed data transmission applications where real-time error detection is crucial.

The calculation of CRC values for each segment involves polynomial division. A generator polynomial is chosen, and the data segment is treated as a polynomial. Modulo-2 division is applied between the segment polynomial and the generator polynomial, resulting in a remainder known as the CRC value for that segment. Different generator polynomials, such as CRC-16, CRC-32, and CRC-CCITT, can be used depending on the specific requirements of the application or protocol. Segmented CRC is widely implemented in various communication protocols, including Ethernet, USB, and SATA. It is also used in storage systems like hard drives and flash memory to ensure data integrity and reliability. The simplicity, efficiency, and error localization capabilities of segmented CRC make it a robust and widely adopted algorithm in the realm of digital data transmission and storage

In conclusion, Segmented CRC is an error-detection algorithm that divides data into segments and calculates a checksum for each segment using a generator polynomial. This approach allows for efficient error detection and correction in digital communication systems, providing increased reliability in data transmission.

4.1.2. Non Segmented CRC:

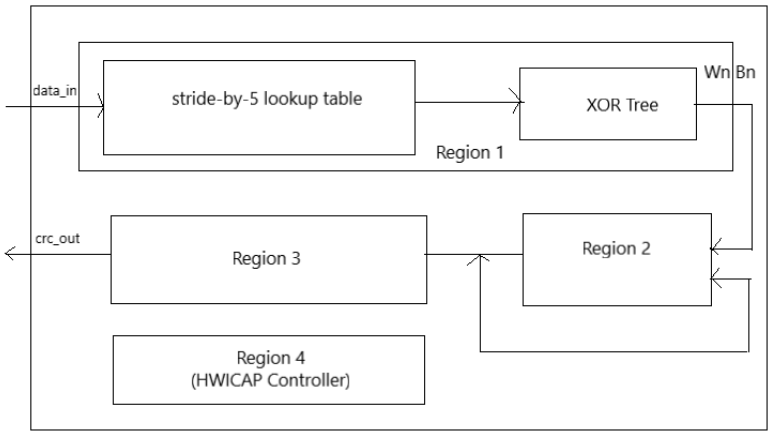


Fig.4.1.2. Non-Segmented CRC

The proposed non-segmented system architecture is shown in Fig. In a non-segmented system architecture, there should be one frame in a single word, and a segmented system architecture can process multiple frames at the same time. Regions 1 and 2 correspond to the computation of Wln Bn. Region 1 consumes most of the LUTs, and the number of consumed LUTs linearly depends on the size of Wln. The stride-by-5 algorithm is proposed to reduce the LUT consumption of region 1. Region2 is implemented by means of an XOR tree instead of a one stage XOR function to achieve higher performance. Region 3completes the computation of (1). It consumes few LUTs for the small size of Tn. The padding zeros problem is solved by region 4, and the pipelining go back algorithm, which results in an O (log2 n) resource utilization, is proposed. Region 5 is an HWICAP controller hat can modify the content of the LUTs dynamically.

Non-segmented CRC (Cyclic Redundancy Check) is an error-detection algorithm used in digital communication systems and data storage applications. Unlike segmented CRC, which divides data into segments for individual error detection, non-segmented CRC calculates a single checksum for the entire data stream. This checksum is then used by the receiver to detect and correct errors that may have occurred during transmission.

The non-segmented CRC algorithm operates by treating the entire data stream as a single entity and performing a polynomial division. A generator polynomial is selected, and the data stream is treated as a polynomial. The polynomial is divided by the generator polynomial using modulo-2 division, resulting in a remainder known as the CRC value. This CRC value is appended to the data stream and transmitted to the receiver.
Upon receiving the data, the receiver performs the same CRC calculation on the received data stream. If the calculated CRC value matches the received CRC value, it indicates that the data was transmitted without errors. However, if there is a mismatch between the calculated and

received CRC values, it signifies the presence of errors in the data stream. Non-segmented CRC offers simplicity in both implementation and calculation. It is relatively easy to integrate into hardware or software systems. The algorithm is efficient and does not require extensive computational resources.

The choice of the generator polynomial is crucial in non-segmented CRC. Commonly used generator polynomials include CRC-16, CRC-32, and CRC-CCITT, which have been standardized and widely implemented in various communication protocols and data storage systems. The specific generator polynomial chosen depends on factors such as the desired error-detection capability, data length, and the error characteristics of the transmission medium.

One of the advantages of non-segmented CRC is its simplicity. It requires a relatively straightforward calculation that can be efficiently implemented in hardware or software. Additionally, non-segmented CRC provides a good balance between error-detection capabilities and computational complexity.

However, non-segmented CRC has some limitations. Since it calculates a single checksum for the entire data stream, it does not provide the ability to localize errors to specific segments or blocks. This means that in the event of an error, the entire data stream may need to be retransmitted, which can impact overall system performance and efficiency.

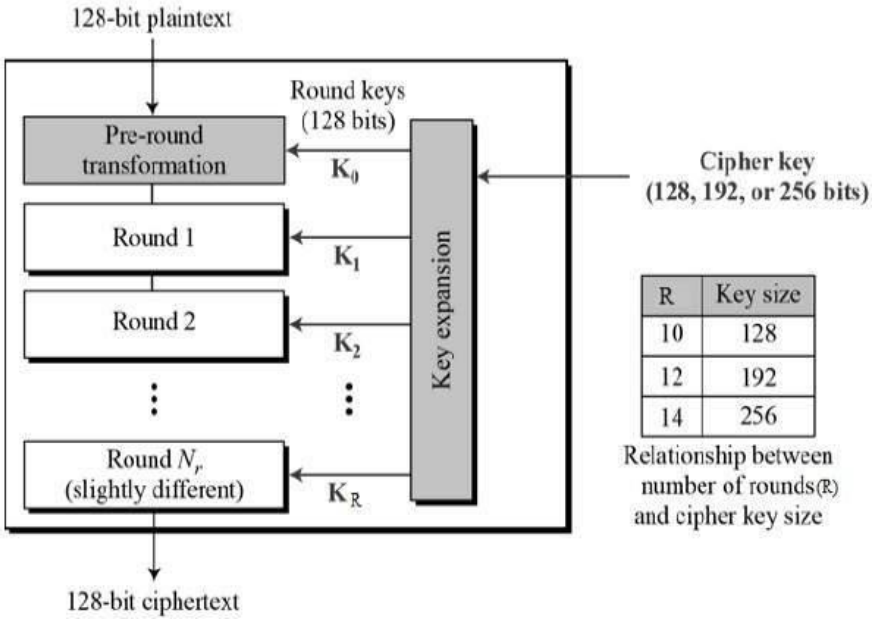
To mitigate the limitations of non-segmented CRC, other error-detection and error-correction techniques are often used in conjunction with it. For example, forward error correction (FEC) codes like Reed-Solomon or convolution codes can be employed to not only detect but also correct errors in the data stream.

The CRC value calculated for the entire data stream is appended to the end of the data before transmission or storage. Upon receiving the data, the receiver performs the same CRC calculation on the received data stream, including the appended CRC value. If the calculated CRC value matches the received CRC value, it indicates that the data was likely transmitted or stored without errors. However, if there is a mismatch between the calculated and received CRC values, it suggests the presence of errors in the data.

In conclusion, non-segmented CRC is an error-detection algorithm that calculates a single checksum for the entire data stream. It is relatively simple to implement and widely used in digital communication systems and data storage applications. While it lacks the error localization capabilities of segmented CRC, non-segmented CRC can be complemented by additional error-correction techniques to enhance the reliability and integrity of transmitted data.

4.1.3. AES Algorithm:

The schematic of AES structure is given in the following illustration



R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Fig.4.1.3. AES Algorithm

AES (Advanced Encryption Standard) is a widely used symmetric encryption algorithm that ensures the confidentiality and integrity of data. It is a symmetric key algorithm, meaning the same key is used for both encryption and decryption processes.

AES was selected by the U.S. National Institute of Standards and Technology (NIST) in 2001 as a replacement for the aging Data Encryption Standard (DES). It has since become the most commonly used encryption algorithm worldwide, adopted in various applications and industries, including government, finance, and communication. AES performs several rounds of substitution, permutation, and mixing operations on the data, providing strong confusion and diffusion properties. The algorithm has undergone extensive analysis and is considered secure against known attacks. AES can be efficiently implemented in both software and hardware, making it suitable for a wide range of applications

The AES algorithm operates on fixed-size blocks of data, typically 128 bits in length. It supports key lengths of 128, 192, and 256 bits, offering different levels of security. The algorithm consists of several rounds, each involving a series of mathematical transformations, including substitution, permutation, and mixing operations.

The AES Encryption and Decryption process is mentioned in above diagram. The algorithm begins with an Add round key stage followed by 9 rounds of four stages and a tenth round of

three stages. This applies for both encryption and decryption with the exception that each stage of a round the decryption algorithm is the inverse of it's counterpart in the encryption algorithm. The four stages are as follows:

1. Substitute bytes
2. Shift rows
3. Mix Columns
4. Add Round Key

The tenth round simply leaves out the Mix Columns stage. The first nine rounds of the decryption algorithm consist of the following:

1. Inverse Shift rows
2. Inverse Substitute bytes
3. Inverse Add Round Key
4. Inverse Mix Columns

AES is a symmetric block cipher. This means that it uses the same key for both encryption and decryption. However, AES is quite different from DES in a number of ways. The algorithm Rijindael allows for a variety of block and key sizes and not just the 64 and 56 bits of DES' block and key size. The block and key can in fact be chosen independently from 128, 160, 192, 224, 256 bits and need not be the same. However, the AES standard states that the algorithm can only accept a block size of 128 bits and a choice of three keys - 128, 192, 256 bits. Depending on which version is used, the name of the standard is modified to AES-128, AES-192 or AES- 256 respectively. As well as these differences AES differs from DES in that it is not a feistel structure. Recall that in a feistel structure, half of the data block is used to modify the other half of the data block and then the halves are swapped. In this case the entire data block is processed in parallel during each round using substitutions and permutations.

The Pipelined algorithm is a symmetric iterated block cipher. The block and key lengths can be 128, 192, or 256 bits. The NIST requested that the AES must implement a symmetric block cipher with a block size of 128 bits. Due to this requirement, variations of Pipelined that can operate on larger block sizes will not be included in the actual standard. Pipelined also has a variable number of iterations or rounds: 10, 12, and 14 when the key lengths are 128, 192, and 256 respectively. The transformations in Pipelined consider the data block as a four-column rectangular array of 4-byte vectors. The key is also considered to be a rectangular array of 4-byte vectors—the number of columns is dependent on key length.

Pipelined decryption comprises the inverse of the transformations that encryption uses, performed in reverse order. Decryption commences with the inverse of the final round, followed by the inverses of the rounds, and finishes with the initial data/key addition, which is its own inverse.

AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

A number of AES parameters depend on the key length. For example, if the key size used is 128 then the number of rounds is 10 whereas it is 12 and 14 for 192 and 256 bits respectively. At present the most common key size likely to be used is the 128 bit key. This description of the AES algorithm therefore describes this particular implementation.

During the encryption process, AES takes the plaintext and applies a series of transformations based on the selected key. These transformations include substitution using a substitution box (S-box), permutation through a process known as the Shift Rows operation, and mixing of data using the Mix Columns operation. The number of rounds performed depends on the key length, with more rounds being applied for longer keys.

The strength and security of AES lie in its ability to resist various attacks, including brute force, differential cryptanalysis, and linear cryptanalysis. AES has a well-defined mathematical structure and has undergone extensive analysis by cryptographers worldwide to validate its security.

One of the advantages of AES is its efficiency in software and hardware implementations. The algorithm is highly optimized, enabling fast encryption and decryption operations, even on resource-constrained devices. This efficiency has contributed to its widespread adoption in various applications.

AES has become the de facto standard for symmetric encryption due to its robust security, efficiency, and broad compatibility. It is supported by most modern operating systems, programming languages, and cryptographic libraries. AES is used to protect sensitive data in various scenarios, such as secure communication over the internet (e.g., HTTPS), data storage encryption, and securing wireless networks (e.g., WPA2).

Rijindael was designed to have the following characteristics:

- Resistance against all known attacks.
- Speed and code compactness on a wide range of platforms.
- Design Simplicity

In summary, AES is a widely adopted symmetric encryption algorithm that ensures data confidentiality and integrity. It operates on fixed-size blocks of data, supports multiple key lengths, and employs a series of mathematical transformations to provide strong encryption. AES's security, efficiency, and compatibility have made it the encryption algorithm of choice in numerous applications and industries.

4.1.4. Multiplier:

Cryptographic architectures provide protection for sensitive and smart infrastructures such as secure healthcare, smart grid, fabric, and home. Cryptography is closely related to the disciplines of cryptology and cryptanalysis. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as clear text) into cipher text (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers. Nonetheless, the use of cryptographic architectures does not guarantee immunity against faults occurring in these infrastructures. Defects in VLSI systems may cause smart usage models to malfunction. Extensive research has been done for detecting such faults in the cryptographic algorithms such as elliptic curve cryptography and the Advanced Encryption Standard. Design for reliability and fault immunity ensures that with the presence of faults, reliability is provided for the aforementioned sensitive cryptographic architectures. The Module description is given below:

4.1.4.1. Substitution Box (S- Box):

The Sub Bytes operation is a nonlinear byte substitution. Each byte from the input state is replaced by another byte according to the substitution box (called the S-box). The S-box is computed based on a multiplicative inverse in the finite field GF (2^8) and a bitwise affine transformation.

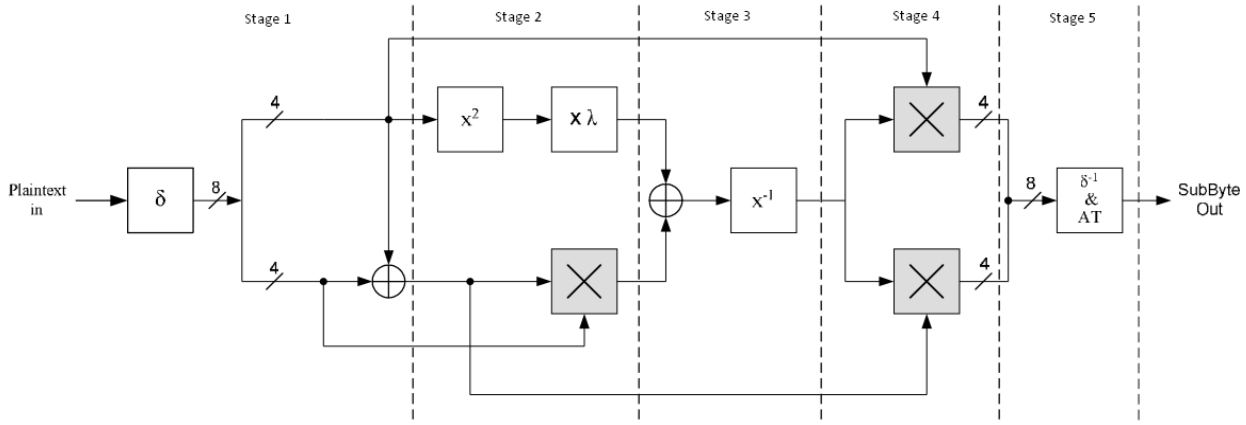


Fig.4.1.4.1. S box Block diagram

In this module the implementation of the composite field S-BOX is accomplished using combinational logic circuits rather than using pre-stored S-BOX values.

Table.4.1.4.1. Truth Table of S Box

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

4.1.5. Hardware Required:

Integrated Circuits:

The term integrated circuit is used to describe a wide variety of devices ranging from simple logic gates through to complex state-of- the-art microprocessors. Integrated circuits basically consist of a circuit, typically made up from a number of transistors and their interconnections, fabricated from a single semiconductor chip or die.

a) Analogue integrated circuits

Analogue integrated circuits include a wide range of applications, many of which are highly specific. Some examples are the simple operational amplifiers and timers, and the more complex FM stereo decoders and single-chip FM radios.

There has been a trend towards fabricating the more commonly used analogue circuits into single chip form. An example of this is FM radio receiver, which is a fairly complex circuit when fabricated from discrete components. A FM radio receiver can now be constructed from a FM radio chip, an audio amplifier chip and a few discrete passive components.

b) Digital integrated circuits

Digital integrated circuits are devices, which are functionally based on logic gates (AND & OR gates). They are commercially available in families of devices which take their name from the fabrication method used to manufacture the devices from different families are not readily compatible in the same circuit. The more common types of logic integrated circuits are typically represented in each family of devices like TTL, Schottky TTL, CMOS and the new high speed CMOS. The CMOS family devices have a very low power consumption that makes them very popular for many applications where very high speeds are not required.

c) Computer integrated circuits

Computer integrated circuits are devices, which form the active components of a computer system. They are often used in conjunction with digital integrated circuits, which provide a 'glue logic' function. Computer integrated circuits can be functionally divided into microprocessors, memory devices and peripheral control devices.

History Of Ic's:

The first silicon chip or integrated circuit consisting of many transistors fabricated on the same piece of silicon was made at Fairchild in 1959. More recent developments have been towards miniaturization-packing more and more active components or gates on to a single chip of silicon. The level of device complexity is usually referred to as a scale of integration. The evolution from scale integration (SSI), through large-scale integration (LSI), to very large-scale integration (VLSI) has already occurred, and the scale is running out of adjectives. The scale of integration is based on the number of logic elements that constitute a device.

Introduction To VLSI:

VLSI stands for "Very Large-Scale Integrated Circuits". It's a classification of ICs. An IC of common VLSI includes about millions active devices. Typical functions of VLSI include Memories, computers, and signal processors, etc. A semiconductor process technology is a method by which working circuits can be manufactured from designed specifications. There are many such technologies, each of which creates a different environment or style of design. In integrated circuit design, the specification consists of polygons of conducting and semiconducting material that will be layered on top of each other to produce a working chip.

When a chip is custom-designed for a specific use, it is called an application-specific integrated circuit (ASIC). Printed-circuit (PC) design also results in precise positions of conducting materials, as they will appear on a circuit board; in addition, PC design aggregates the bulk of the electronic activity into standard IC packages, the position and interconnection of which are essential to the final circuit. Printed circuitry may be easier to debug than integrated circuitry is, but it is slower, less compact, more expensive, and unable to take advantage of specialized silicon layout structures that make VLSI systems so attractive. The design of these electronic circuits can be achieved at many different refinement levels from the most detailed layout to the most abstract architectures. Given the complexity that is demanded at all levels, computers are increasingly used to aid this design at each step. It is no longer reasonable to use manual design techniques, in which each layer is hand etched or composed by laying tape on film. Thus the term computer-aided design or CAD is a most accurate description of this modern way and seems more broad in its scope than the recently popular term computer-aided engineering (CAE)

Application Areas Of VLSI:

- **PLAs:**

Combinational circuit elements are an important part of any digital design. Three common methods of implementing a combinational block are random logic, read-only memory (ROM), and programmable logic array (PLA). In random-logic designs, the logic description of the circuit is directly translated into hardware structures such as AND and OR gates. The PLA occupies less area on the silicon due to reduced interconnection wire space; however, it may be slower than purely random logic. A PLA can also be used as a compact finite state machine by feeding back part of its outputs to the inputs and clocking both sides. Normally, for high-speed applications, the PLA is not implemented as two NOR arrays. The inputs and outputs are inverted to preserve the AND-OR structure.

- **Gate-Arrays:**

The gate-array is a popular technique used to design IC chips. Like the PLA, it contains a fixed mesh of unfinished layout that must be customized to yield the final circuit. Gate-arrays are more powerful, however, because the contents of the mesh are less structured so the interconnection options are more flexible. Gate-arrays exist in many forms with many names, eg: uncommitted logic arrays and master-slice. The disadvantage of gate-arrays is that they are not optimal for any task.

- **Gate Matrices:**

The gate matrix is the next step in the evolution of automatically generated layout from high-level specification. Like the PLA, this layout has no fixed size; a gate matrix grows according to its complexity. Like all regular forms of layout, this one has its fixed aspects and its customizable aspects. In gate matrix layout the fixed design consists of vertical columns of polysilicon gating material. The customizable part is the metal and diffusion wires that run horizontally to interconnect and form gates with the columns.

Applications Of VLSI:

Electronic systems now perform a wide variety of tasks in daily life. Electronic systems in some cases have replaced mechanisms that operated mechanically, hydraulically, or by other means; electronics are usually smaller, more flexible, and easier to service. In other cases electronic systems have created totally new applications. Electronic systems perform a variety of tasks, some of them visible, some more hidden:

- Personal entertainment systems such as portable MP3 players and DVD players perform sophisticated algorithms with remarkably little energy.
- Electronic systems in cars operate stereo systems and displays; they also control fuel injection systems, adjust suspensions to varying terrain, and perform the control functions required for anti-lock braking (ABS) systems.
- Digital electronics compress and decompress video, even at high definition data rates, on-the-fly in consumer electronics.
- Low-cost terminals for Web browsing still require sophisticated electronics, despite their dedicated function.
- Personal computers and workstations provide word-processing, financial analysis, and games. Computers include both central processing units (CPUs) and special-purpose hardware for disk access, faster screen display, etc.

Advantages Of VLSI:

While we will concentrate on integrated circuits in this book, the properties of integrated circuits what we can and cannot efficiently put in an integrated circuit—largely determine the architecture of the entire system. Integrated circuits improve system characteristics in several critical ways. ICs have three key advantages over digital circuits built from discrete components:

• **Size.** Integrated circuits are much smaller—both transistors and wires are shrunk to micrometre sizes, compared to the millimetre or centimetre scales of discrete components. Small size leads to advantages in speed and power consumption, since smaller components have smaller parasitic resistances, capacitances, and inductances.

• **Speed.** Signals can be switched between logic 0 and logic 1 much quicker within a chip than they can between chips. Communication within a chip can occur hundreds of times faster than communication between chips on a printed circuit board. The high speed of circuits on-chip is due to their small size—smaller components and wires have smaller parasitic capacitances to slow down the signal

• **Power consumption.** Logic operations within a chip also take much less power. Once again, lower power consumption is largely due to the small size of circuits on the chip—smaller parasitic capacitances and resistances require less power to drive them.

VLSI And Systems

These advantages of integrated circuits translate into advantages at the system level:

• **Smaller physical size**. Smallness is often an advantage in itself—considers portable televisions or handheld cellular telephones.

• **Lower power consumption.** Replacing a handful of standard parts with a single chip reduces total power consumption. Reducing power consumption has a ripple effect on the rest of the system: a smaller, cheaper power supply can be used; since less power consumption means less heat, a fan may no longer be necessary; a simpler cabinet with less shielding for electromagnetic shielding may be feasible, too.

• **Reduced cost.** Reducing the number of components, the power supply requirements, cabinet costs, and so on, will inevitably reduce system cost. The ripple effect of integration is such that the cost of a system built from custom ICs can be less, even though the individual ICs cost more than the standard parts they replace. Understanding why integrated circuit technology has such profound influence on the design of digital systems requires understanding both the technology of IC manufacturing and the economics of ICs and digital systems.

Introduction To Asics and Programmable Logic:

The last 15 years have witnessed the demise in the number of cell-based ASIC designs as a means for developing customized SoCs. Rising NREs, development times and risk have mostly restricted the use of cell-based ASICs to the highest volume applications; applications that can withstand the multi-million-dollar development costs associated with 1-2 design re-spins. Analysts estimate that the number of cell-based ASIC design starts per year is now only between 2000-3000 compared to ~10,000 in the late 1990s. The FPGA has emerged as a technology that fills some of the gap left by cell-based ASICs. Yet even after 20+ years of existence and 40X more design starts per year than cell-based ASICs, the size of the FPGA market in dollar terms remains only a fraction that of cell-based ASICs. This suggests that there are many FPGA designs that never make it into production and that for the most part; the FPGA is still seen by many as a vehicle for prototyping or college education and has perhaps even succeeded in actually stifling industry innovation. This paper introduces a new technology, the second

generation Structured ASIC that is tipped to reenergize the path to innovation within the electronics industry. It brings together some of the key advantages of FPGA technology (i.e. fast turnaround, no mask charges, no minimum order quantity) and of cell-based ASIC (i.e. low unit cost and power) to deliver a new platform for SoC design. This document defines requirements for development of Application Specific Integrated Circuits (ASICs). It is intended to be used as an appendix to a Statement of Work. The document complements the ESA ASIC Design and Assurance Requirements (AD1), which is a precursor to a future ESA PSS document on ASIC design.

Moore's Law

In the 1960s Gordon Moore predicted that the number of transistors that could be manufactured on a chip would grow exponentially. His prediction, now known as Moore's Law, was remarkably prescient. Moore's ultimate prediction was that transistor count would double every two years, an estimate that has held up remarkably well. Today, an industry group maintains the International Technology Roadmap for Semiconductors (ITRS), that maps out strategies to maintain the pace of Moore's Law.

Applications For Nextreme Structured Asics:

Embedded Processing

Nextreme Structured ASICs are ideally suited for embedded processing applications. The availability of a firm, 150MHz ARM926EJT™ processor and AMBA peripherals backed by industry standard development tools from ARM and its Connected Community™ partners, designers have the option to implement control circuits in software. A major benefit of using Nextreme for implementing embedded systems is that designers are able to make performance, area and feature tradeoffs using both hardware and software allowing for highly differentiated yet cost-optimized systems.

Signal, Video and Image Processing

Having to deal with programmable metal interconnect and its associated carry chain delays ultimately forced FPGA vendors to develop dedicated DSP blocks and slices to overcome performance bottlenecks. With Nextreme Structured ASICs, the elimination of massive amounts of metal interconnect means that these devices are not subject to unacceptable carry chain delays and many signal processing structured can be implemented, at speed, using logic fabric alone. Another capability with in Nextreme that makes them particularly suitable for signal processing is memories. eRAM blocks are particularly suited for distributed applications such as semi-parallel filters and video processing. As these blocks are located very close together, they can be connected to form larger blocks up to 4Kbits per eUnit.

Field-Programmable Gate Array(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. The ability to update the functionality after shipping, partial re-configuration of the portion of the design and the low non-recurring engineering costs relative to an ASIC design (notwithstanding the generally higher unit cost), offer advantages for many applications.

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.

In addition to digital functions, some FPGAs have analog features. The most common analog feature is programmable slew rate and drive strength on each output pin, allowing the engineer to set slow rates on lightly loaded pins that would otherwise ring unacceptably, and to set stronger, faster rates on heavily loaded pins on high-speed channels that would otherwise run too slow. Another relatively common analog feature is differential comparators on input pins designed to be connected to differential signaling channels. A few "mixed signal FPGAs" have integrated peripheral Analog-to-Digital Converters (ADCs) and Digital-to-Analog Converters (DACs) with analog signal conditioning blocks allowing them to operate as a system-on-a-chip. Such devices blur the line between an FPGA, which carries digital ones and zeros on its internal programmable interconnect fabric, and field-programmable analog array (FPAA), which carries analog values on its internal programmable interconnect fabric.

Definitions of Relevant Terminology

The most important terminology used below.

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. Programming of such a device often involves placing the chip into a special programming unit, but some chips can also be configured “in-system”. Another name for FPDs is programmable logic devices (PLDs); although PLDs encompass the same types of chips as FPDs, we prefer the term FPD because historically the word PLD has referred to relatively simple types of devices.

Programmable Logic Array (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 (note: although PLA structures are sometimes embedded into full-custom chips, we refer here only to those PLAs that are provided as separate integrated circuits and are user-programmable).

Programmable Array Logic (PAL)

A Programmable Array Logic (PAL) is a relatively small FPD that has a programmable AND-plane followed by a fixed OR-plane.

Simple PLD

Refers to any type of Simple PLD, usually either a PLA or PAL.

Complex PLD

A more Complex PLD that consists of an arrangement of multiple SPLD-like blocks on a single chip. Alternative names (that will not be used in this paper) sometimes adopted for this style of chip are Enhanced PLD (EPLD), Super PAL, Mega PAL, and others.

Field-Programmable Gate Array (FPGA)

A Field-Programmable Gate Array is an FPD featuring a general structure that allows very high logic capacity. Whereas CPLDs feature logic resources with a wide number of inputs (AND planes), FPGAs offer more narrow logic resources. FPGAs also offer a higher ratio of flip-flops to logic resources than do CPLDs.

High-Capacity PLDs (HCPLD):

high-capacity PLDs: a single acronym that refers to both CPLDs and FPGAs. This term has been coined in trade literature for providing an easy way to refer to both types of devices. PAL is a trademark of Advanced Micro Devices.

- **Interconnect -**The wiring resources in an FPD.
- **Programmable Switch-** a user-programmable switch that can connect a logic element to an interconnect wire, or one interconnect wire to another
- **Logic Block**- a relatively small circuit block that is replicated in an array in an FPD. When a circuit is implemented in an FPD, it is first decomposed into smaller sub-circuits that can each be mapped into a logic block. The term logic block is mostly used in the context of FPGAs, but it could also refer to a block of circuitry in a CPLD.
- **Logic Capacity**- the amount of digital logic that can be mapped into a single FPD. This is usually measured in units of "equivalent number of gates in a traditional gate array". In

other words, the capacity of an FPD is measured by the size of gate array that it is comparable to. In simpler terms, logic capacity can be thought of as "number of 2-input NAND gates".

- **Logic Density** - the amount of logic per unit area in an FPD.
- **Speed-Performance**- measures the maximum operable speed of a circuit when implemented in an FPD. For combinational circuits, it is set by the longest delay through any path, and for sequential circuits it is the maximum clock frequency for which the circuit functions properly. In the remainder of this section, to provide insight into FPD development the evolution of FPDs over the past two decades is described. Additional background information is also included on the semiconductor technologies used in the manufacture of FPDs.

Evolution of Programmable Logic Devices:

The first type of user-programmable chip that could implement logic circuits was the Programmable Read-Only Memory (PROM), in which address lines can be used as logic circuit inputs and data lines as outputs. Logic functions, however, rarely require more than a few product terms, and a PROM contains a full decoder for its address inputs. PROMS are thus an inefficient architecture for realizing logic circuits, and so are rarely used in practice for that purpose. The first device developed later specifically for implementing logic circuits was the Field-Programmable Logic Array (FPLA), or simply PLA for short. A PLA consists of two levels of logic gates: a programmable "wired" AND-plane followed by a programmable "wired" OR-plane. A PLA is structured so that any of its inputs (or their complements) can be AND'ed together in the AND-plane; each AND-plane output can thus correspond to any product term of the inputs. Similarly, each OR plane output can be configured to produce the logical sum of any of the AND-plane outputs. With this structure, PLAs are well-suited for implementing logic functions in sum-of-products form. They are also quite versatile, since both the AND terms and OR terms can have many inputs (this feature is often referred to as wide AND and OR gates). When PLAs were introduced in the early 1970s, by Philips, their main drawbacks were that they were expensive to manufacture and offered somewhat poor speed-performance.

Both disadvantages were due to the two levels of configurable logic, because programmable logic planes were difficult to manufacture and introduced significant propagation delays. To overcome these weaknesses, Programmable Array Logic (PAL) devices were developed. PALs feature only a single level of programmability, consisting of a programmable "wired" AND plane that feeds fixed OR-gates. To compensate for lack of generality incurred because the OR-Outputs plane is fixed, several variants of PALs are produced, with different numbers of inputs and outputs, and various sizes of OR-gates. PALs usually contain flip-flops connected to the OR-gate outputs so that sequential circuits can be realized.

4.1.5.1.Basys 3 Artix-7 FPGA Board

The Basys 3 is an entry-level FPGA development board designed exclusively for the Vivado Design Suite featuring the AMD Artix 7 FPGA architecture. Basys 3 is the newest addition to the popular Basys line of FPGA development boards for students or beginners just getting started with FPGA technology. The Basys 3 includes the standard features found on all Basys boards: complete ready-to-use hardware, a large collection of on-board I/O devices, all required FPGA support circuits, and a free version of development tools and at a student-level price point.

Fig 4.1.5.1.1. Basys 3 Artix-7 FPGA Board

The Basys 3 Artix-7 is a popular development board designed by Digilent Inc. It features the Xilinx Artix-7 FPGA (Field Programmable Gate Array) and offers a wide range of functionalities for digital design, prototyping, and educational purposes. With its powerful hardware and versatile features, the Basys 3 Artix-7 board is highly regarded in the field of electronics and FPGA-based projects.

The heart of the Basys 3 Artix-7 board is the Xilinx Artix-7 FPGA, which provides a programmable platform for implementing digital circuits. The Artix-7 FPGA belongs to Xilinx's 7-series FPGAs and offers a balance of performance, power efficiency, and cost-effectiveness. It consists of programmable logic blocks, DSP (Digital Signal Processing) slices, and memory resources, allowing users to implement a wide range of digital designs.

The Basys 3 Artix-7 board boasts a rich set of peripherals and interfaces, making it suitable for various applications. It includes 12-bit VGA output for video display, a 16-button keypad, a 4-digit seven-segment display, and multiple slide switches and LEDs for user

input and feedback. Additionally, the board features a USB-UART bridge, Ethernet port, and a microSD card slot for data transfer and storage.

To facilitate development and programming, the Basys 3 Artix-7 board supports multiple programming methods. It can be programmed using Xilinx's Vivado Design Suite, which offers a comprehensive development environment for FPGA design. The board also supports Digilent's Adept software, which provides an intuitive interface for configuring and programming the FPGA.

The Basys 3 Artix-7 board is widely used in educational settings for teaching digital design, FPGA programming, and embedded systems. Its user-friendly features and extensive documentation make it accessible to both beginners and advanced users. The board is often utilized for projects involving digital circuits, processor-based systems, real-time signal processing, and more.

Due to its powerful capabilities, the Basys 3 Artix-7 board serves as a versatile platform for rapid prototyping and experimentation. It enables users to implement custom digital designs, test algorithms, and create functional prototypes. The programmable nature of the Artix-7 FPGA allows for flexibility and iterative development, making it an ideal tool for design exploration and validation.

In conclusion, the Basys 3 Artix-7 board, powered by the Xilinx Artix-7 FPGA, offers a robust and feature-rich platform for digital design and prototyping. Its wide range of peripherals and interfaces, along with its support for various programming methods, make it highly versatile and suitable for both educational and professional applications. The Basys 3 Artix-7 board is a reliable choice for those looking to explore FPGA-based projects, develop digital systems, and gain hands-on experience in the field of digital design.

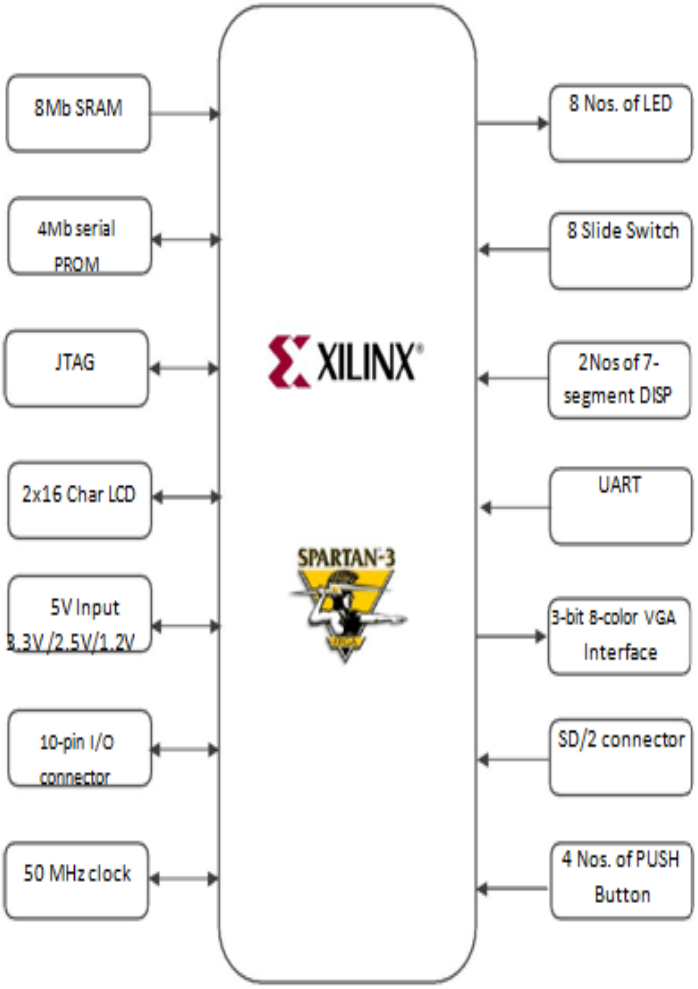


Fig 4.1.5.1.2 Basys 3 Artix-7 FPGA Board Generalised Block Diagram

4.1.5.2 On-board Peripherals:

The Basys 3 Artix-7FPGA Lab Kit comes with many interfacing options

- 4 Pmod ports: 3 Standard 12-pin Pmod ports, 1 dual purpose XADC signal / standard Pmod port
- 4-digit 7-segment display
- 5 user pushbuttons
- 16 user LEDs
- 16 user switches
- USB HID Host for mice, keyboards and memory sticks
- 12-bit VGA output
- USB-UART Bridge
- Serial Flash

- Digilent USB-JTAG port for FPGA programming and communication
- On-chip analog-to-digital converter (XADC)
- Internal clock speeds exceeding 450 MHz
- 90 DSP slices
- Five clock management tiles, each with a phase-locked loop (PLL)
- 1,800 Kbits of fast block RAM
- 33,280 logic cells in 5200 slices (each slice contains four 6-input LUTs and 8 flip-flops)
- Features the AMD Artix-7 FPGA: XC7A35T-1CPG236C

4.1.5.2.1. Seven Segment Display:

The Spartan3 FPGA Kit has a two-character, seven-segment LED display controlled by FPGA user-I/O pins. Each digit shares eight common control signals to light individual LED segments. Each individual character has a separate anode control input. The pin number for each FPGA pin connected to the LED display is shown in Table 1. To light an individual signal, drive the individual segment control signal Low along with the associated anode control signal for the individual character.

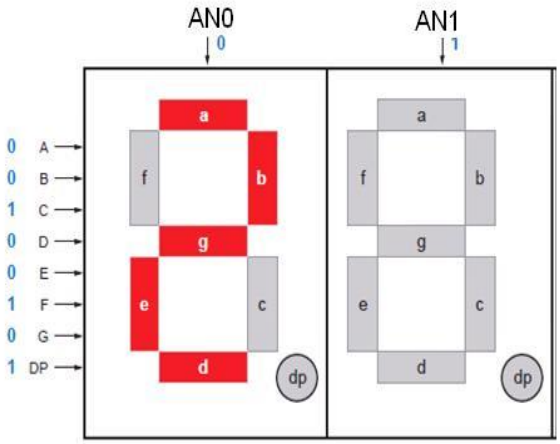


Fig 4.1.5.2.1. Seven Segment Display

Table 4.1.5.2.1. Seven-segment display connections to the FPGA pins

Segment	FPGA PIN
A	P82
B	P83
C	P84
D	P85

E	P86
F	P87
G	P89
DP	P90

Table 4.1.5.2.3. Digit Enable (Anode Control) Signals (Active Low)

Anode Control	**FPGA PIN**
AN1	**P76**
AN0	**P77**

The LED control signals are time-multiplexed to display data on two characters. Present the value to be displayed on the segment control inputs and select the specified character by driving the associated anode control signal Low. Through persistence of vision, the human brain perceives that all four characters appear simultaneously, similar to the way the brain perceives a TV display.

This "scanning" technique reduces the number of I/O pins required for the four characters. In case an FPGA pin were dedicated for each individual segment, then 32 pins are required to drive four 7-segment LED characters. The scanning technique reduces the required I/O down to 12 pins. The drawback to this approach is that the FPGA logic must continuously scan data out to the displays—a small price to save 20 additional I/O pins.

When working with the Basys 3's seven-segment display, the user typically needs to write logic or software code to determine the segments that need to be activated for each desired digit or character. Different encodings or lookup tables can be used to map specific numbers or characters to the corresponding segment patterns.

In summary, the Basys 3 development board's seven-segment display is a versatile peripheral that allows users to display numeric values and certain characters. It is a common and straightforward interface used in many electronic projects and serves as a valuable tool for conveying information in a clear and concise manner.

4.1.5.2.2. Digital Inputs Toggle Switch:

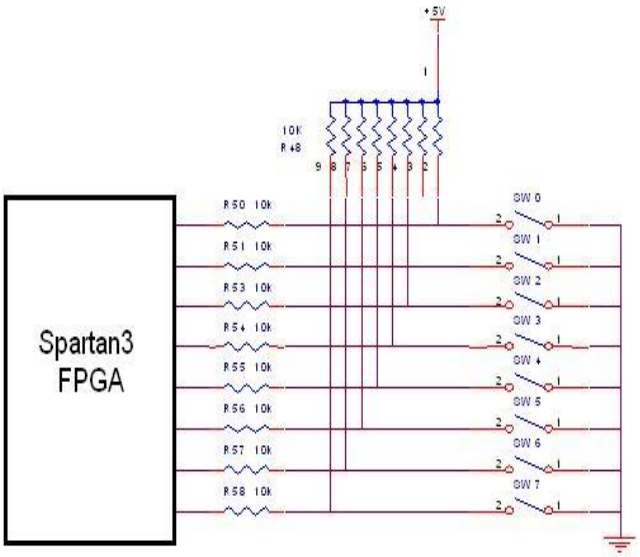


Fig 4.1.5.2.2.1. Digital Inputs Toggle Switch

The Basys 3 development board provides multiple digital input options, including toggle switches. Toggle switches are hardware components on the board that allow users to manually toggle between two states: ON and OFF. These switches provide a simple and tactile way to input binary values into the FPGA.

The Basys 3 board features several slide switches that can be used as toggle switches. Each slide switch represents a single digital input, typically associated with a specific signal or functionality in the FPGA design.

To utilize the toggle switches on the Basys 3 board, users can connect the switches to specific input pins on the FPGA. By configuring the FPGA design to monitor the state of these input pins, users can read the positions of the toggle switches and use that information to control the behavior of their digital circuits or applications.

The Basys 3 board often includes additional circuitry, such as debouncing mechanisms, to ensure stable and reliable readings from the toggle switches. Debouncing prevents erratic or false readings that can occur when a switch's mechanical contacts make rapid contact or break contact during switching.

Toggle switches are commonly used in various FPGA-based projects for tasks such as selecting modes, enabling or disabling features, setting configuration options, or controlling logic operations. They provide a user-friendly interface for manual input and can be used to trigger specific actions or initiate different modes of operation within a digital design.

In summary, the Basys 3 development board offers toggle switches in the form of slide switches as digital input options. These switches provide a straightforward way to input binary values into the FPGA, allowing users to control the behaviour of their digital circuits. By connecting the switches to specific input pins and incorporating them into the FPGA design, users can utilize the toggle switches for tasks such as mode selection, configuration, or enabling/disabling features.

Pin Mapping: The Basys 3 board provides a pin mapping guide that specifies the mapping between the slide switches and the corresponding FPGA pins. This mapping helps users identify the specific input pins associated with each toggle switch.

Input Voltage Levels: The slide switches on the Basys 3 board typically provide a logical HIGH or logical LOW voltage level when toggled. The specific voltage levels depend on the voltage reference used by the FPGA and the associated input buffer circuitry. It's important to refer to the board's documentation or user guide for details regarding voltage levels and compatibility with the FPGA's input voltage requirements.

Design Considerations: When using toggle switches as digital inputs, it is important to consider the design requirements and potential issues. This includes debouncing the input signal to prevent false readings, incorporating proper error checking or synchronization mechanisms, and ensuring that the FPGA design is responsive to switch changes in a timely manner.

In summary, the Basys 3 board's toggle switches, typically implemented as slide switches, serve as digital input devices. They can be connected to specific FPGA pins and utilized within the FPGA design to read the state of the switches. Toggle switches are versatile input devices and can be used in various applications. Debouncing circuitry and proper design considerations ensure stable and reliable readings from the toggle switches. Toggle switches offer a user-friendly and manual input method for controlling and interacting with digital circuits and applications on the Basys 3 board.

4.1.5.2.3 USB-UART Bridge:

The Basys 3 Artix-7 is a popular development board designed by Digilent Inc. It features the Xilinx Artix-7 FPGA (Field Programmable Gate Array) and offers a wide range of functionalities for digital design, prototyping, and educational purposes. With its powerful hardware and versatile features, the Basys 3 Artix-7 board is highly regarded in the field of electronics and FPGA-based projects.

The heart of the Basys 3 Artix-7 board is the Xilinx Artix-7 FPGA, which provides a programmable platform for implementing digital circuits. The Artix-7 FPGA belongs to Xilinx's 7-series FPGAs and offers a balance of performance, power efficiency, and cost-effectiveness. It consists of programmable logic blocks, DSP (Digital Signal Processing) slices, and memory resources, allowing users to implement a wide range of digital designs.

The Basys 3 Artix-7 board boasts a rich set of peripherals and interfaces, making it suitable for various applications. It includes 12-bit VGA output for video display, a 16-button keypad, a 4-digit seven-segment display, and multiple slide switches and LEDs for user input and feedback. Additionally, the board features a USB-UART bridge, Ethernet port, and a microSD card slot for data transfer and storage.

To facilitate development and programming, the Basys 3 Artix-7 board supports multiple programming methods. It can be programmed using Xilinx's Vivado Design Suite, which offers a comprehensive development environment for FPGA design. The board also supports Digilent's Adept software, which provides an intuitive interface for configuring and programming the FPGA.

The Basys 3 Artix-7 board is widely used in educational settings for teaching digital design, FPGA programming, and embedded systems. Its user-friendly features and extensive documentation make it accessible to both beginners and advanced users. The board is often utilized for projects involving digital circuits, processor-based systems, real-time signal processing, and more.

Due to its powerful capabilities, the Basys 3 Artix-7 board serves as a versatile platform for rapid prototyping and experimentation. It enables users to implement custom digital designs, test algorithms, and create functional prototypes. The programmable nature of the Artix-7 FPGA allows for flexibility and iterative development, making it an ideal tool for design exploration and validation.

Users can establish a serial communication link between the Basys 3 board and the computer, enabling data exchange and control. For example, users can send commands or instructions from the computer to the FPGA, and the FPGA can transmit data or status information back to the computer for further processing or analysis.

The USB-UART bridge on the Basys 3 board simplifies the communication process and provides a standardized and widely supported method for serial communication. It enhances the functionality and versatility of the Basys 3 board, allowing users to integrate the board with external systems, software applications, and development environments.

The USB-UART bridge on the Basys 3 board facilitates bidirectional data transfer, allowing users to exchange data between the FPGA (Field-Programmable Gate Array) on the board and the host computer. It serves as an interface that converts the UART serial data format used by the FPGA to USB data format, and vice versa

The USB-UART bridge establishes a virtual serial communication link between the Basys 3 board and the computer. On the computer's operating system, the bridge typically appears as a virtual COM port, which can be accessed by serial communication software or custom programs to enable data transfer and communication with the FPGA.

In conclusion, the Basys 3 Artix-7 board, powered by the Xilinx Artix-7 FPGA, offers a robust and feature-rich platform for digital design and prototyping. Its wide range of peripherals and interfaces, along with its support for various programming methods, make it highly versatile and suitable for both educational and professional applications. The Basys 3 Artix-7 board is a reliable choice for those looking to explore FPGA-based projects, develop digital systems, and gain hands-on experience in the field of digital design. It converts UART serial data to USB data format and appears as a virtual COM port on the computer's operating system. The USB-UART bridge provides a convenient and standardized method for bidirectional data transfer, enabling various applications and interactions with the Basys 3 board.

4.5.2. Software Requirement:

4.5.2.1. Verification Tool

1. Modelsim 6.4a

4.5.2.2. Synthesis Tool

2. Xilinx ISE 9.1/ Xilinx 13.2

Verification Tool :Modelsim

Modelsim SE - High Performance Simulation and Debug

ModelSim SE is our UNIX, Linux, and Windows-based simulation and debug environment, combining high performance with the most powerful and intuitive GUI in the industry.

What's New in ModelSim SE?

- Improved FSM debug options including control of basic information, transition table and warning messages. Added support of FSM Multi-state transitions coverage (i.e. coverage for all possible FSM state sequences).
- Improved debugging with hyperlinked navigation between objects and their declaration, and between visited source files.
- The dataflow window can now compute and display all paths from one net to another.
- Enhanced code coverage data management with fine grain control of information in the source window.
- Toggle coverage has been enhanced to support SystemVerilog types: structures, packed unions, fixed-size multi-dimensional arrays and real.
- Some IEEE VHDL 2008 features are supported including source code encryption. Added support of new VPI types, including packed arrays of struct nets and variables.

ModelSim SE Features:

- Multi-language, high performance simulation engine
- Verilog, VHDL, SystemVerilog Design
- Code Coverage
- SystemVerilog for Design
- Integrated debug
- JobSpy Regression Monitor
- Mixed HDL simulation option
- SystemC Option
- TCL/tk
- Solaris and Linux 32 & 64-bit
- Windows 32-bit

ModelSim SE Benefits:

- High performance HDL simulation solution for FPGA & ASIC design teams
- The best mixed-language environment and performance in the industry
- Intuitive GUI for efficient interactive or post-simulation debug of RTL and gate-level designs
- Merging, ranking and reporting of code coverage for tracking verification progress
- Sign-off support for popular ASIC libraries
- All Model ----------Sim products are 100% standards based. This means your investment is protected, risk is lowered, reuse is enabled, and productivity is enhanced
- Award-winning technical support

High-Performance, Scalable Simulation Environment:

ModelSim provides seamless, scalable performance and capabilities. Through the use of a single compiler and library system for all ModelSim configurations, employing the right ModelSim configuration for project needs is as simple as pointing your environment to the appropriate installation directory.

ModelSim also supports very fast time-tenet-simulation turnarounds while maintaining high performance with its new black box use model, known as bbox. With bbox, non-changing elements can be compiled and optimized once and reused when running a modified version of the test bench. bbox delivers dramatic throughput improvements of up to 3X when running a large suite of test cases.

Easy-to-Use Simulation Environment:

An intelligently engineered graphical user interface (GUI) efficiently displays design data for analysis and debug. The default configuration of windows and information is designed to meet the needs of most users. However, the flexibility of the ModelSim SE GUI allows users to easily customize it to their preferences. The result is a feature-rich GUI that is easy to use and quickly mastered.

A message viewer enables simulation messages to be logged to the ModelSim results file in addition to the standard transcript file. The GUI's organizational and filtering capabilities allow design and simulation information to be quickly reduced to focus on areas of interest, such as possible causes of design bugs.

ModelSim SE allows many debug and analysis capabilities to be employed post-simulation on saved results, as well as during live simulation runs. For example, the coverage viewer analyzes and annotates source code with code coverage results, including FSM state and transition, statement, expression, branch, and toggle coverage. Signal values can be annotated in the source window and viewed in the waveform viewer. Race conditions, delta, and event activity can be analyzed in the list and wave windows. User-defined enumeration values can be easily defined for quicker understanding of simulation results. For improved debug productivity, ModelSim also has graphical and textual dataflow capabilities. The memory window identifies memories in the design and accommodates flexible viewing and modification of the memory contents. Powerful search, fill, load, and save functionalities are supported. The memory window allows memories to be pre-loaded with specific or randomly generated values, saving the time-consuming step of initializing sections of the simulation merely to load memories. All functions are available via the command line, so they can be used in scripting.

Advanced Code Coverage:

The ModelSim advanced code coverage capabilities deliver high performance with ease of use. Most simulation optimizations remain enabled with code coverage. Code coverage metrics can be reported by-instance or by-design unit, providing flexibility in managing coverage data. All coverage information is now stored in the Unified Coverage Database (UCDB), which is used to collect and manage all coverage information in one highly efficient database. Coverage utilities that analyze code coverage data, such as merging and test ranking, are available.

The coverage types supported include:

- Statement coverage: number of statements executed during a run
- Branch coverage: expressions and case statements that affect the control flow of the HDL execution

- Condition coverage: breaks down the condition on a branch into elements that make the result true or false
- Expression coverage: the same as condition coverage, but covers concurrent signal assignments instead of branch decisions
- Focused expression coverage: presents expression coverage data in a manner that accounts for each independent input to the expression in determining coverage results
- Enhanced toggle coverage: in default mode, counts low-to-high and high-to-low transitions; in extended mode, counts transitions to and from X
- Finite State Machine coverage: state and state transition coverage

Synthesis Tool: Xilinx Ise

Introduction

For two-and-a-half decades, Xilinx has been at the forefront of the programmable logic revolution, with the invention and continued migration of FPGA platform technology. During that time, the role of the FPGA has evolved from a vehicle for prototyping and glue-logic to a highly flexible alternative to ASICs and ASSPs for a host of applications and markets. Today, Xilinx FPGAs have become strategically essential to world-class system companies that are hoping to survive and compete in these times of extreme global economic instability, turning what was once the programmable revolution into the "programmable imperative" for both Xilinx and our customers.

Programmable Imperative:

When viewed from the customer's perspective, the programmable imperative is the necessity to do more with less, to remove risk wherever possible, and to differentiate in order to survive. In essence, it is the quest to simultaneously satisfy the conflicting demands created by ever-evolving product requirements (i.e., cost, power, performance, and density) and mounting business challenges (i.e., shrinking market windows, fickle market demands, capped engineering budgets, escalating ASIC and ASSP non-recurring engineering costs, spiralling complexity, and increased risk). To Xilinx, the programmable imperative represents a two-fold commitment. The first is to continue developing programmable silicon innovations at every process node that deliver industry-leading value for every key figure of merit against which FPGAs are measured: price, power, performance, density, features, and programmability. The second commitment is to provide customers with simpler, smarter, and more strategically viable design platforms for the creation of world-class FPGA-based solutions in a wide variety of industries—what Xilinx calls targeted design platforms.

Base Platform:

The base platform is both the delivery vehicle for all new silicon offerings from Xilinx and the foundation upon which all Xilinx targeted design platforms are built. As such, it is the most fundamental platform used to develop and run customer-specific software applications and hardware designs as production system solutions. Released at launch, the base platform comprises a robust set of well-integrated, tested, and targeted elements that enable customers to immediately start a design. These elements include:

• FPGA silicon

• ISE Design Suite design environment

• Third-party synthesis, simulation, and signal integrity tools

•Reference designs common to many applications, such as memory interface and configuration designs.

• Development boards that run the reference designs

• A host of widely used IP, such as GigE, Ethernet, memory controllers, and PCIe.

Domain-Specific Platform:

The next layer in the targeted design platform hierarchy is the domain-specific platform. Released from three to six months after the base platform, each domain specific platform targets one of the three primary Xilinx FPGA user profiles (domains):the embedded processing developer, the digital signal processing (DSP) developer, or the logic/connectivity developer. This is where the real power and intent of the targeted design platform begins to emerge. Domain-specific platforms augment the base platform with a predictable, reliable, and intelligently targeted set of integrated technologies, including:

• Higher-level design methodologies and tools

• Domain-specific embedded, DSP, and connectivity IP

• Domain-specific development hardware and daughter cards

• Reference designs optimized for embedded processing, connectivity, and DSP

• Operating systems (required for embedded processing) and software

Every element in these platforms is tested, targeted, and supported by Xilinx and/or our ecosystem partners. Starting a design with the appropriate domain-specific platform can cut weeks, if not months, off of the user's development time.

Market-Specific Platform

A market-specific platform is an integrated combination of technologies that enables software or hardware developers to quickly build and then run their specific application or solution. Built for use in specific markets such as Automotive, Consumer, Mil/Aero, Communications, AVB, or ISM, market-specific platforms integrate both the base and domain-specific platforms and provide higher level elements that can be leveraged by customer-specific software and hardware designs. The market-specific platform can rely more heavily on third-party targeted IP than the base or domain-specific platforms. The market-specific platform includes: the base and domain-specific platforms, reference designs, and boards (or daughter cards) to run reference designs that are optimized for a particular market (e.g., lane departure early-warning systems, analytics, and display processing).Xilinx will begin releasing market-specific platforms three to six months after the domain-specific platforms, augmenting the domain-specific platforms with reference designs, IP, and software aimed at key growth markets. Initially, Xilinx will target markets such as Communications, Automotive, Video, and Displays with platform elements that abstract away the more mundane portions of the design, thereby further reducing the customer's development effort so they can focus their attention on creating differentiated value in their end solution. This systematic platform development and release strategy provides the framework for the consistent and efficient fulfillment of the programmable imperative—both by Xilinx and by its customers.

Platform Enablers:

Xilinx has instituted a number of changes and enhancements that have contributed substantially to the feasibility and viability of the targeted design platform. These platform-enabling changes cover six primary areas:

• Design environment enhancements

• Socket able IP creation

• New targeted reference designs

• Scalable unified board and kit strategy

• Ecosystem expansion

• Design services supporting the targeted design platform approach

Design Environment Enhancements:

With the breadth of advances and capabilities that the Virtex-6 and Spartan®-6 programmable devices deliver coupled with the access provided by the associated targeted design platforms, it is no longer feasible for one design flow or environment to fit every

designer's needs. System designers, algorithm designers, SW coders, and logic designers each represent a different user-profile, with unique requirements for a design methodology and associated design environment. Instead of addressing the problem in terms of individual fixed tools, Xilinx targets the required or preferred methodology for each user, to address their specific needs with the appropriate design flow. At this level, the design language changes from HDL (VHDL/Verilog) to C, C++, MATLAB® software, and other higher level languages which are more widely used by these designers, and the design abstraction moves up from the block or component to the system level. The result is a methodology and complete design flow tailored to each user profile that provides design creation, design implementation, and design verification. Indicative of the complexity of the problem, to fully understand the user profile of a "logic designer," one must consider the various levels of expertise represented by this demographic. The most basic category in this profile is the "push-button user" who wants to complete a design with minimum work or knowledge.

The push-button user just needs "good-enough" results. Contrastingly, more advanced users want some level of interactive capabilities to squeeze more value into their design, and the "power user" (the expert) wants full control over a vast array of variables. Add the traditional ASIC designers, tasked with migrating their designs to an FPGA (a growing trend, given the intolerable costs and risks posed by ASIC development these days), and clearly the imperative facing Xilinx is to offer targeted flows and tools that support each user's requirements and capabilities, on their terms. The most recent release of the ISE Design Suite includes numerous changes that fulfil requirements specifically pertinent to the targeted design platform. The new release features a complete tool chain for each top-level user profile (the domain-specific personas: the embedded, DSP, and logic/connectivity designers), including specific accommodations for everyone from the push-button user to the ASIC designer.

The tighter integration of embedded and DSP flows enables more seamless integration of designs that contain embedded, DSP, IP, and user blocks in one system. To further enhance productivity and help customers better manage the complexity of their designs, the new ISE Design Suite enables designers to target area, performance, or power by simply selecting a design goal in the setup. The tools then apply specific optimizations to help meet the design goal. In addition, the ISE Design Suite boasts substantially faster place-and-route and simulation run times, providing users with 2X faster compile times. Finally, Xilinx has adopted the FLEX-net Licensing strategy that provides a floating license to track and monitor usage.

XILINX ISE Design Tools:

Xilinx-ISE is the design tool provided by Xilinx. Xilinx would be virtually identical for our purposes.

There are four fundamental steps in all digital logic design. These consist of:

1. Design – The schematic or code that describes the circuit.

2. Synthesis – The intermediate conversion of human readable circuit description to FPGA code (EDIF) format. It involves syntax checking and combining of all the separate design files into a single file.
3. Place & Route– Where the layout of the circuit is finalized. This is the translation of the EDIF into logic gates on the FPGA.
4. Program – The FPGA is updated to reflect the design through the use of programming (.bit) files.

Test bench simulation is in the second step. As its name implies, it is used for testing the design by simulating the result of driving the inputs and observing the outputs to verify your design.

4.2. Flow Chart:

4.2.1. Proposed System Flow:

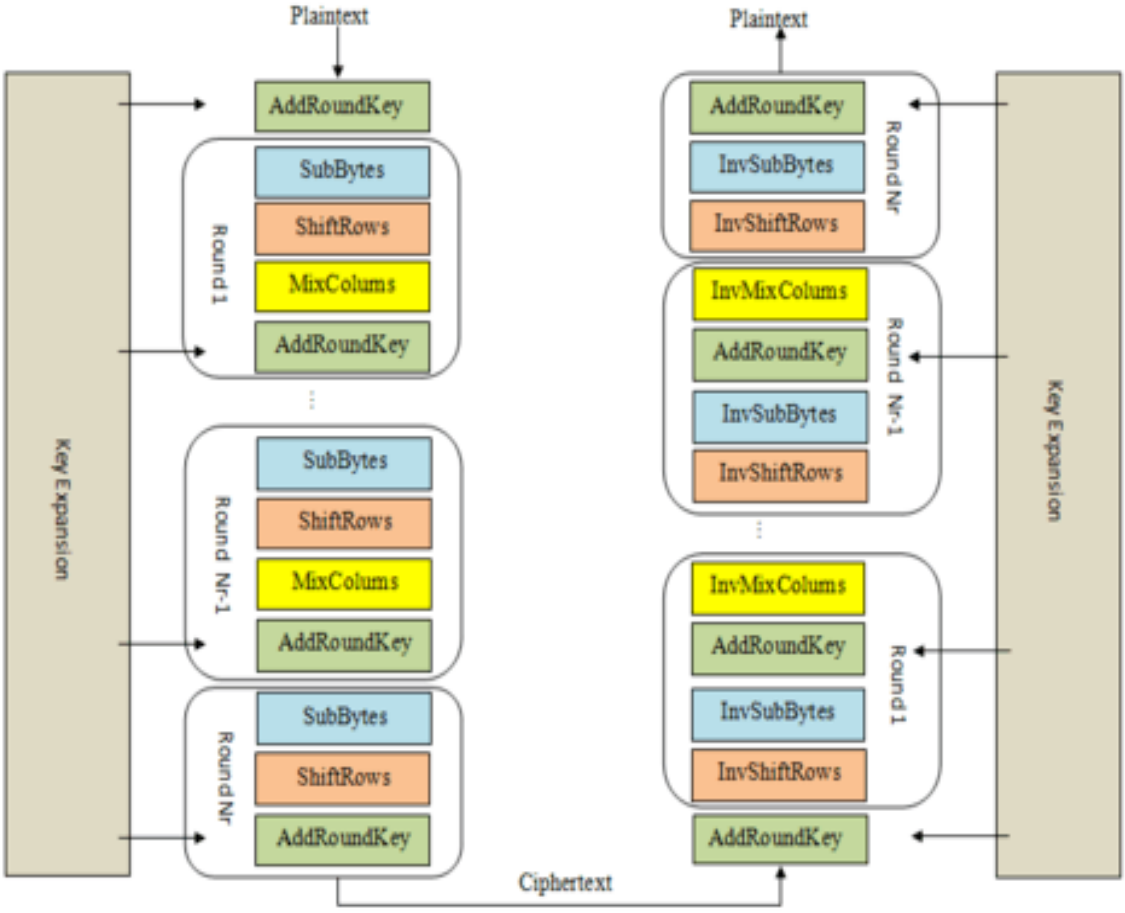


Fig 4.2.1. Proposed System Flow

CRC is a type of error detection code that is commonly used in digital communication systems to detect errors in data transmission. In AES, the CRC is used to ensure the integrity of the data being transmitted.In segmented CRC design, the message is divided into fixed-length segments and a CRC is calculated for each segment separately. This allows for easier error detection and correction since errors can be detected and corrected within each segment separately.In non-segmented CRC design, the CRC is calculated over the entire message as a single block. This design is simpler and requires less computation, but it can be less effective at detecting errors since errors that occur in one part of the message can affect the entire CRC.

4.2.2. Proposed System Technique:

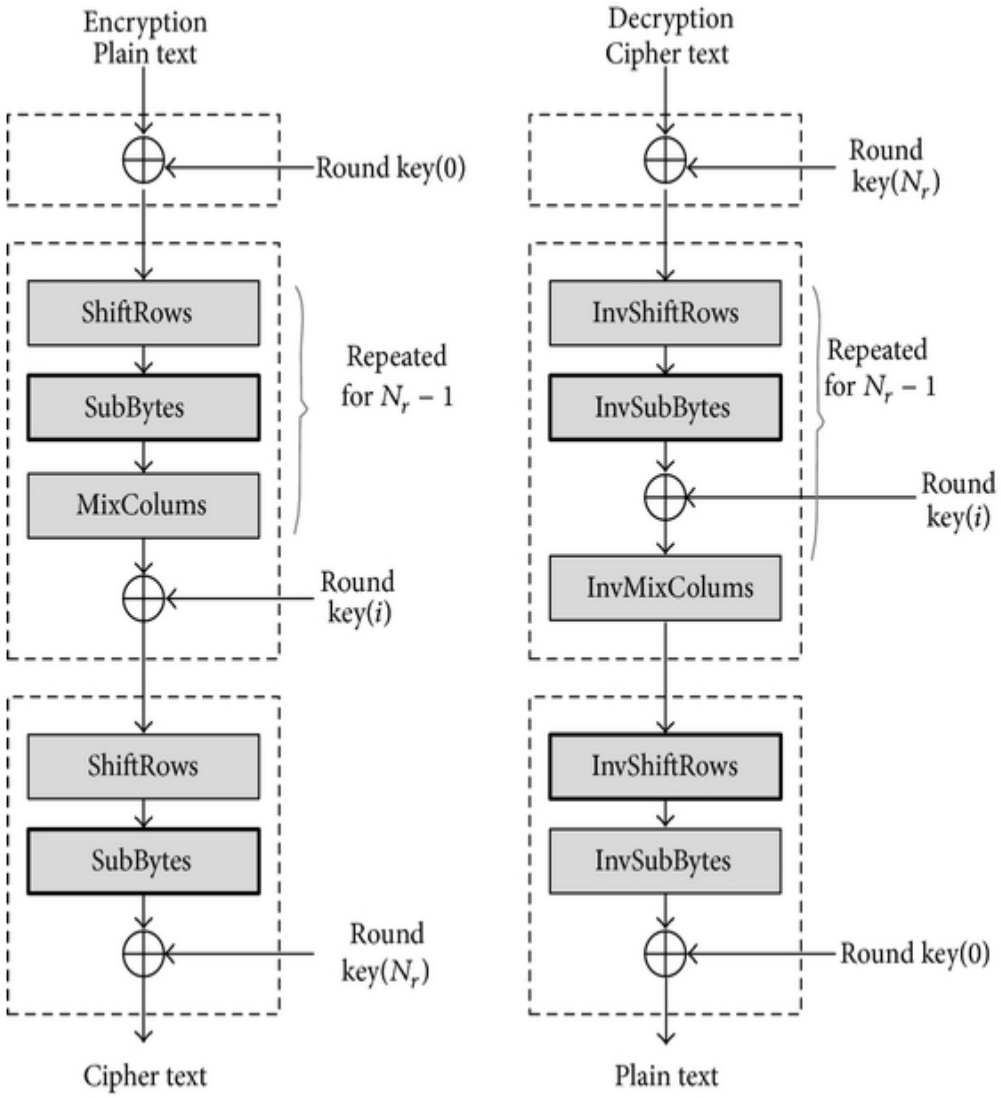


Fig 4.2.2. Proposed System Technique

The AES Encryption and Decryption process is mentioned in bellow Diagram. The process of decryption of an AES cipher text is similar to the encryption process in the reverse order. Each round consists of the four processes

- Add round key
- Mix columns
- Shift rows
- Byte substitution

4.2.3. Stride-By-5 Algorithm:

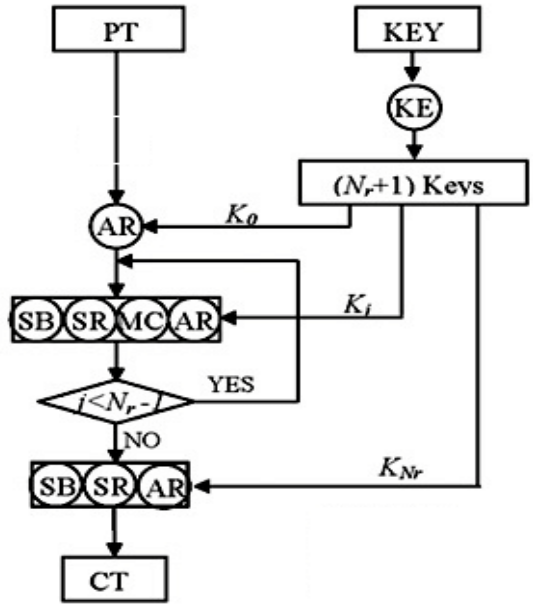


Fig 4.2.3. Stride-By-5 Algorithm

The model of the resource utilization is established, the stride-by-5 is proved to be the best stride for various bus widths, and the stride-by-5 algorithm is described in Algorithm Stride, as its name implies, refers to the number of bits processed by a single logical table. The logical table can be realized using FPGA LUTs, and it can load the truth table of a function.

$$\begin{cases} y_1 = x_1 + x_2 + x_3 + x_4 \\ y_2 = x_5 + x_6 + x_7 + x_8 \\ y = y_1 + y_2 \end{cases}$$

Chapter 5: Results & Discussion

5.1. Results:

Verilog HDL is a Hardware Description Language (HDL). A Hardware Description Language is a language used to describe a digital system, for example, a computer or a component of a computer. One may describe a digital system at several levels. For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) chip, i.e., the switch level. Or, it might describe the logical gates and flip flops in a digital system, i.e., the gate level. An even higher level describes the registers and the transfers of vectors of information between registers. This is called the Register Transfer Level (RTL). Verilog supports all of these levels. However, this handout focuses on only the portions of Verilog which support the RTL level. Verilog is one of the two major Hardware Description Languages (HDL) used by hardware designers in industry and academia. VHDL is the other one. The industry is currently split on which is better. Many feel that Verilog is easier to learn and use than VHDL. As one hardware designer puts it, VHDL is very most engineers have no experience. Verilog was introduced in 1985 by Gateway Design System Corporation, now a part of Cadence Design Systems, Inc.'s Systems Division. Cadence realized that Verilog HDL users wanted other software and service companies to embrace the language and develop Verilog-supported design tools.

Verilog HDL is a Hardware Description Language (HDL). A Hardware Description Language is a language used to describe a digital system, for example, a computer or a component of a computer. One may describe a digital system at several levels. For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) chip, i. e., the switch level. Or, it might describe the logical gates and flip flops in a digital system, i. e., the gate level. An even higher level describes the registers and the transfers of vectors of information between registers. This is called the Register Transfer Level (RTL). Verilog supports all of these levels. However, this handout focuses on only the portions of Verilog which support the RTL level.

5.1.1.1. Sub Bytes:

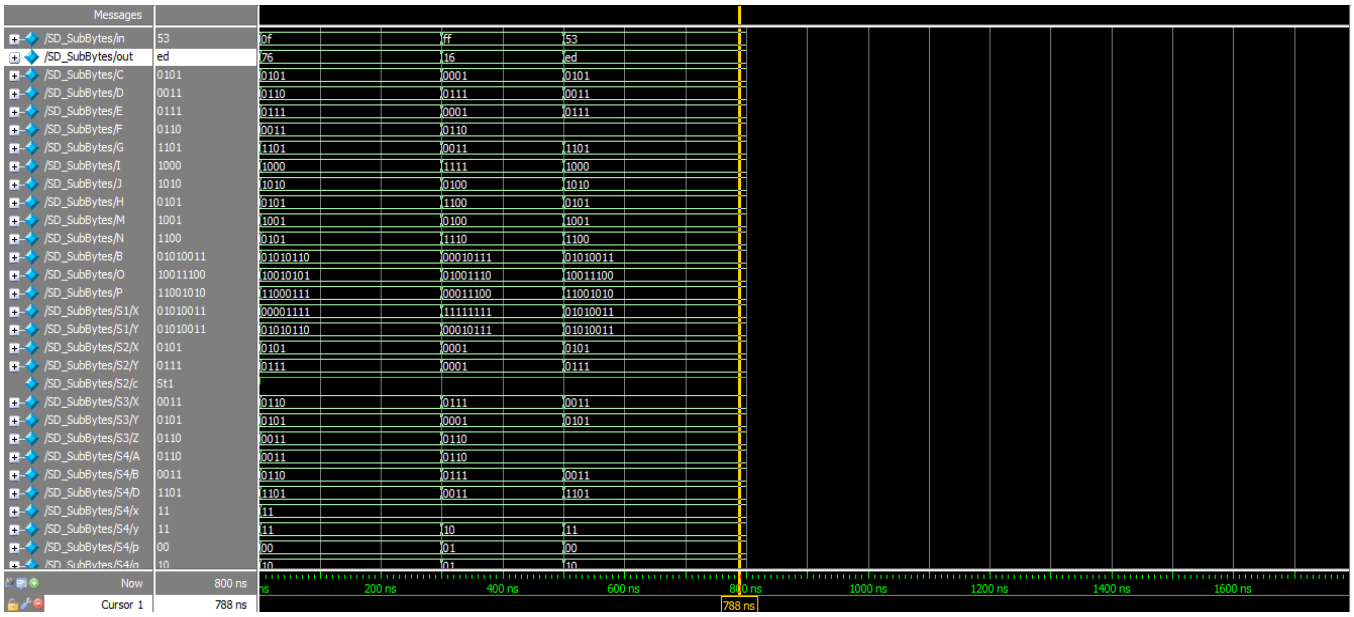


Fig 5.1.1.1. Sub Bytes

5.1.1.2. Shift Rows:

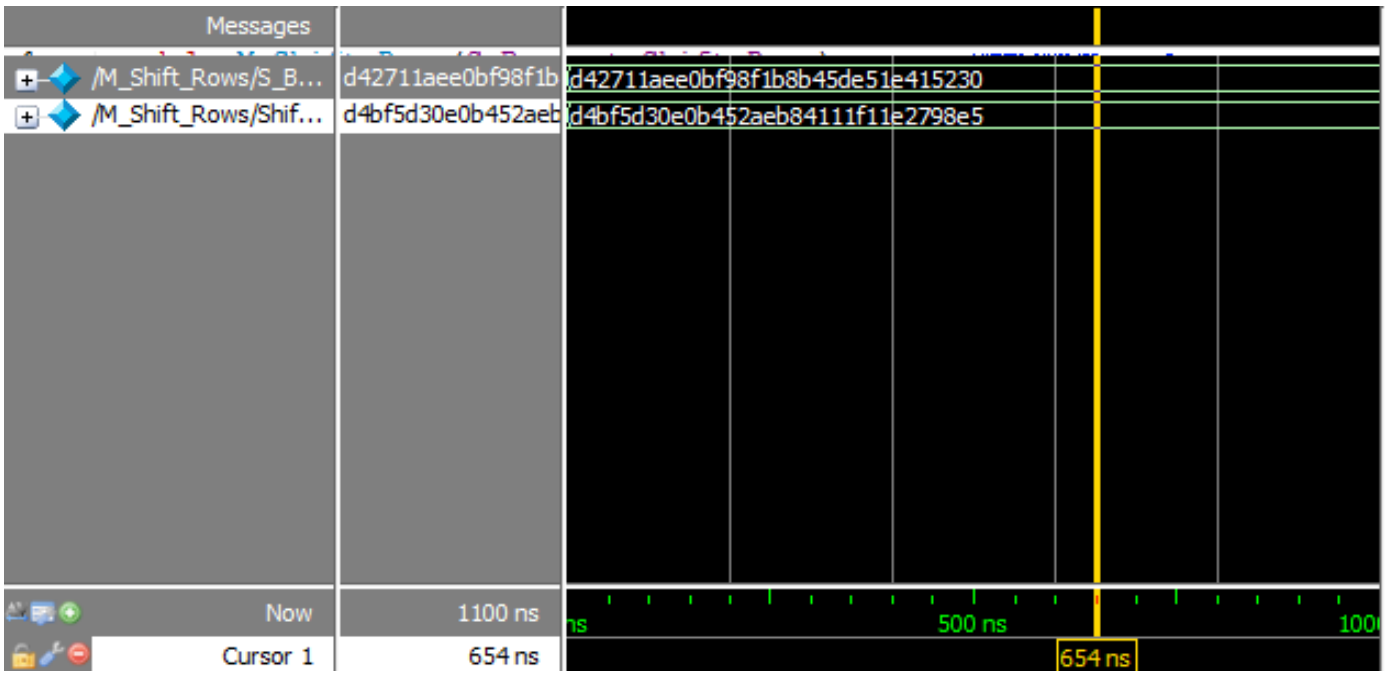


Fig 5.1.1.2. Shift Rows

5.1.1.3. Mixed Column:

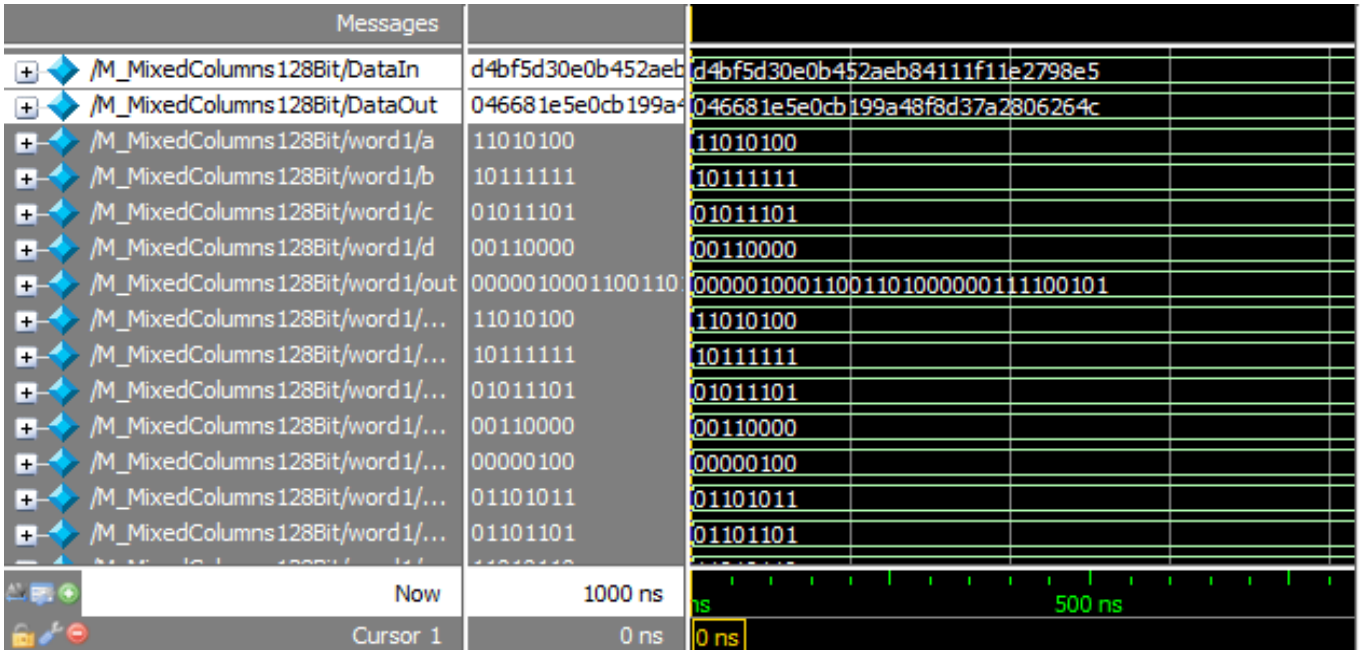


Fig 5.1.1.3. Mixed Column

5.1.1.4. Add Round Key:

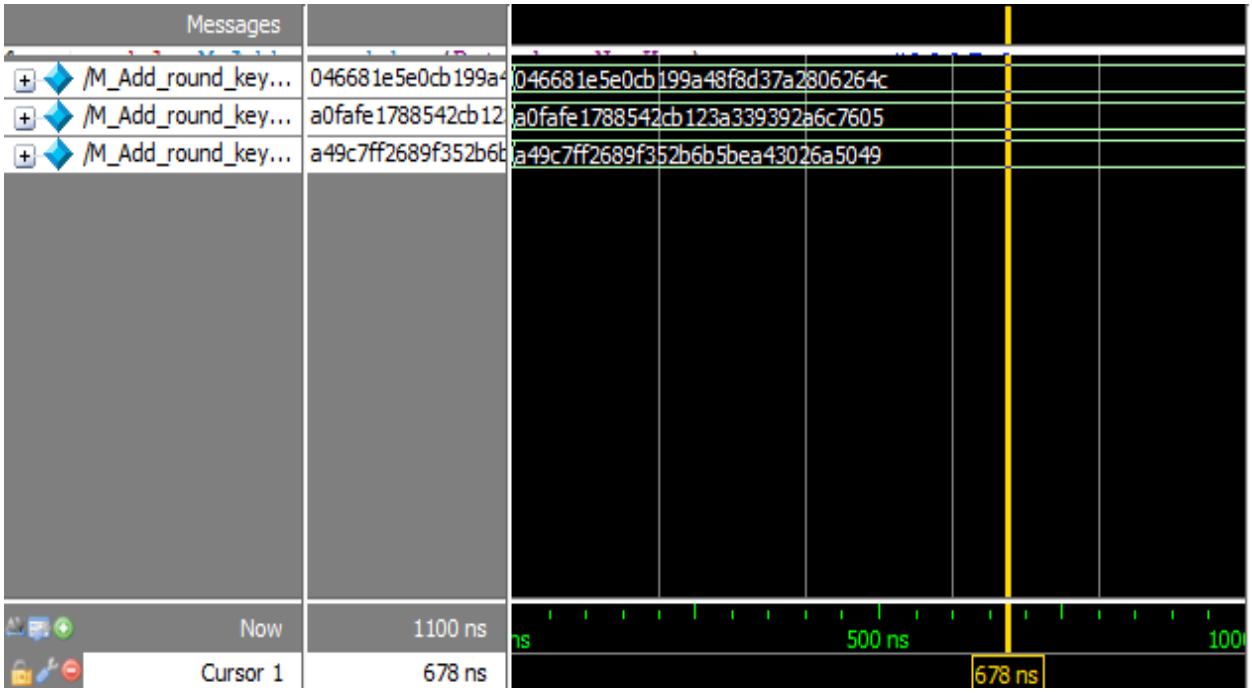


Fig 5.1.1.4. Add Round Key

5.1.1.5. Encryption:

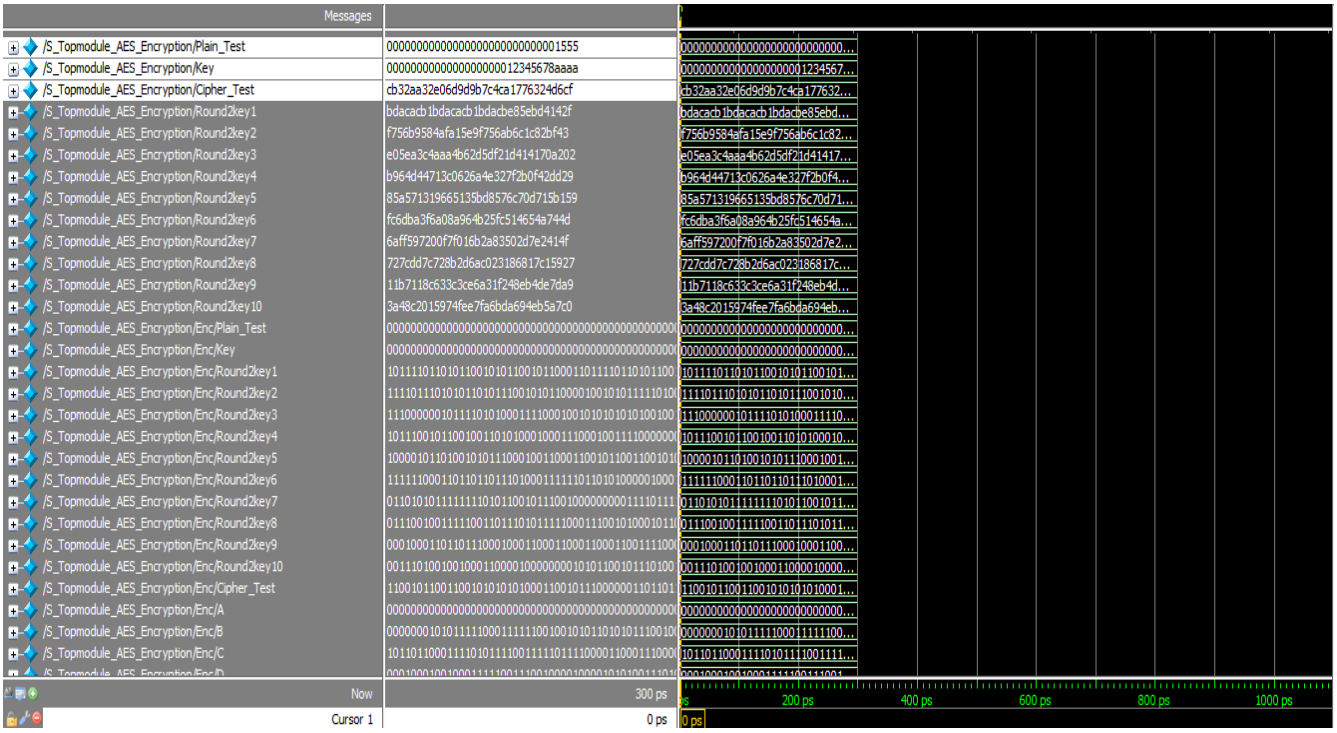


Fig 5.1.1.5. Encryption

5.1.1.6. Single Round Operation:

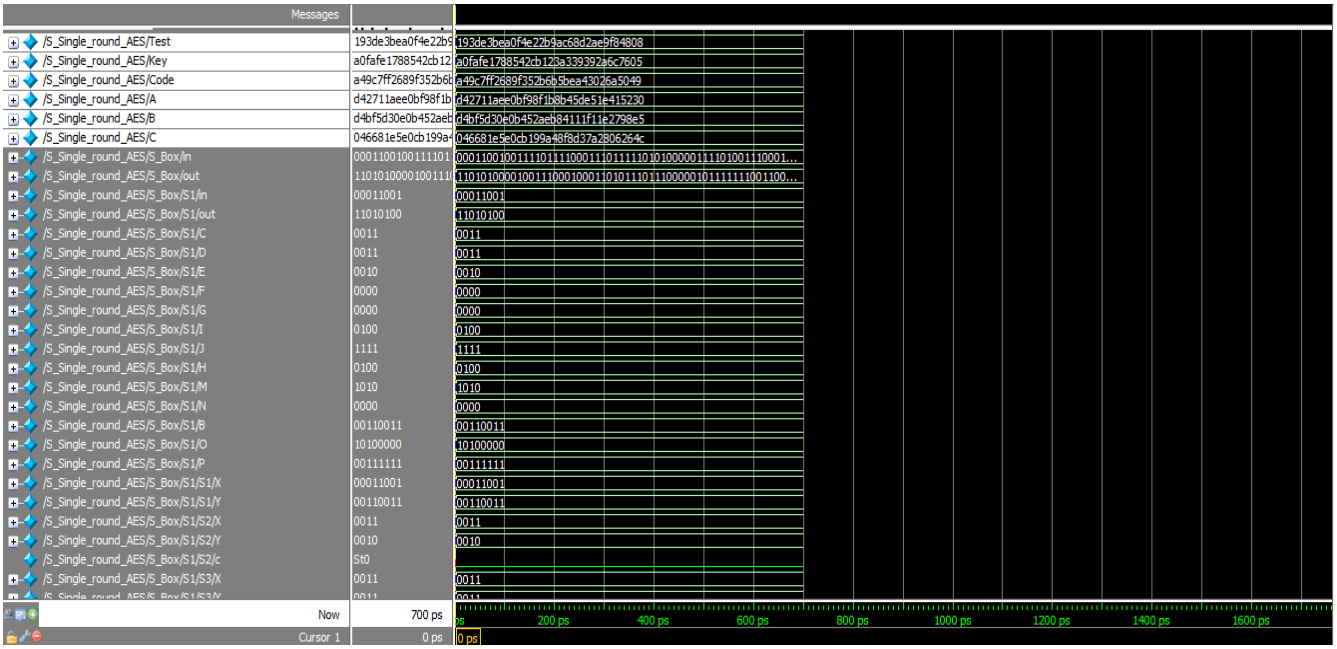


Fig 5.1.1.6. Single Round Operation

5.1.1.7. AES Encryption:

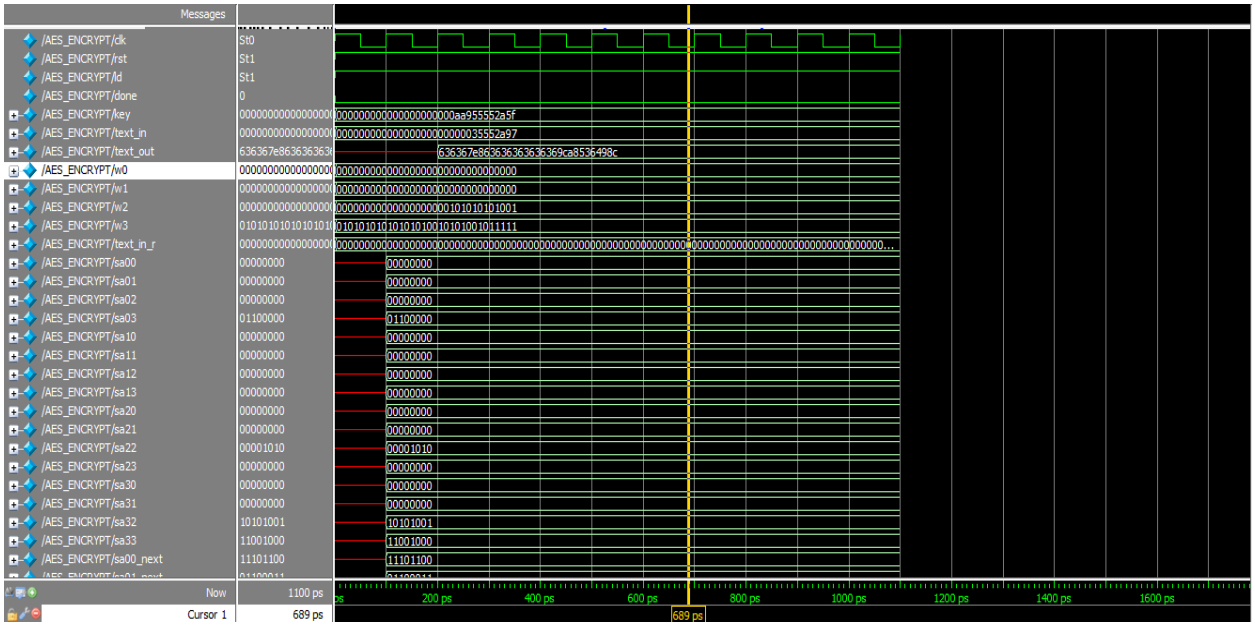


Fig 5.1.1.7. AES Encryption

5.1.1.8. AES Encryption & Decryption:

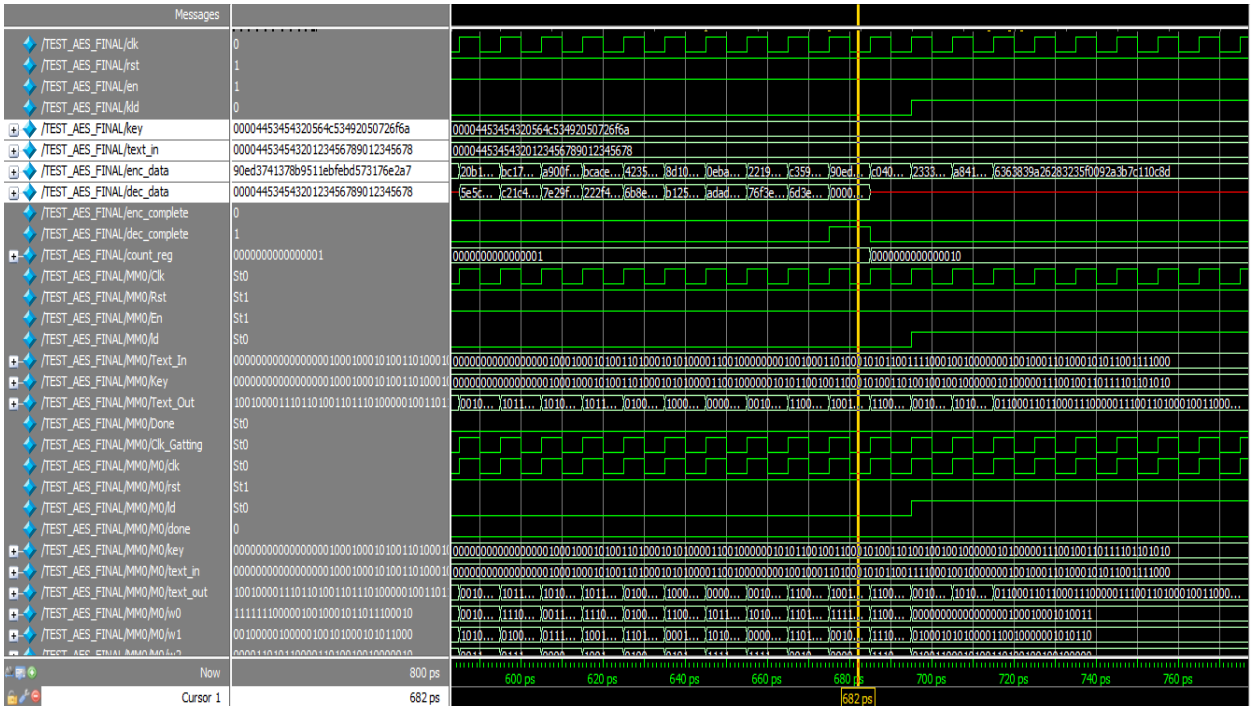


Fig 5.1.1.8. AES Encryption & Decryption

5.1.1.9. Segmented:

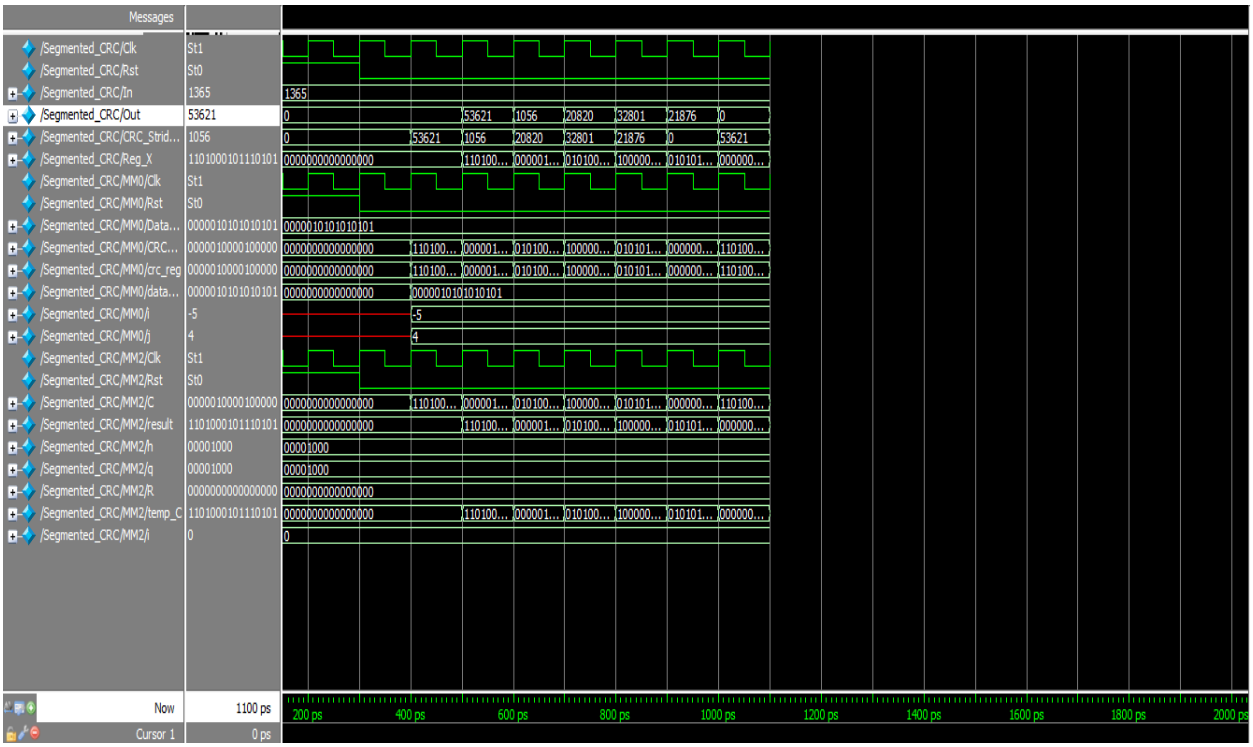


Fig 5.1.1.9. Segmented

5.1.1.10. Non Segmented:

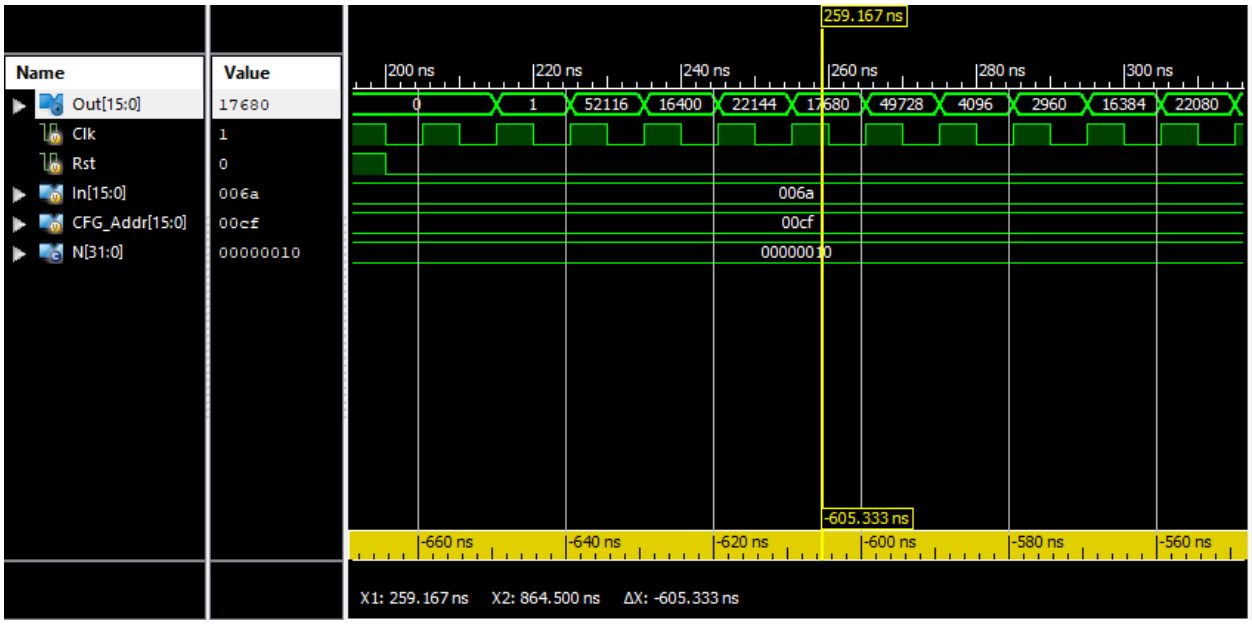


Fig 5.1.1.10. Non Segmented

5.1.1.11. Main Application Simulation:

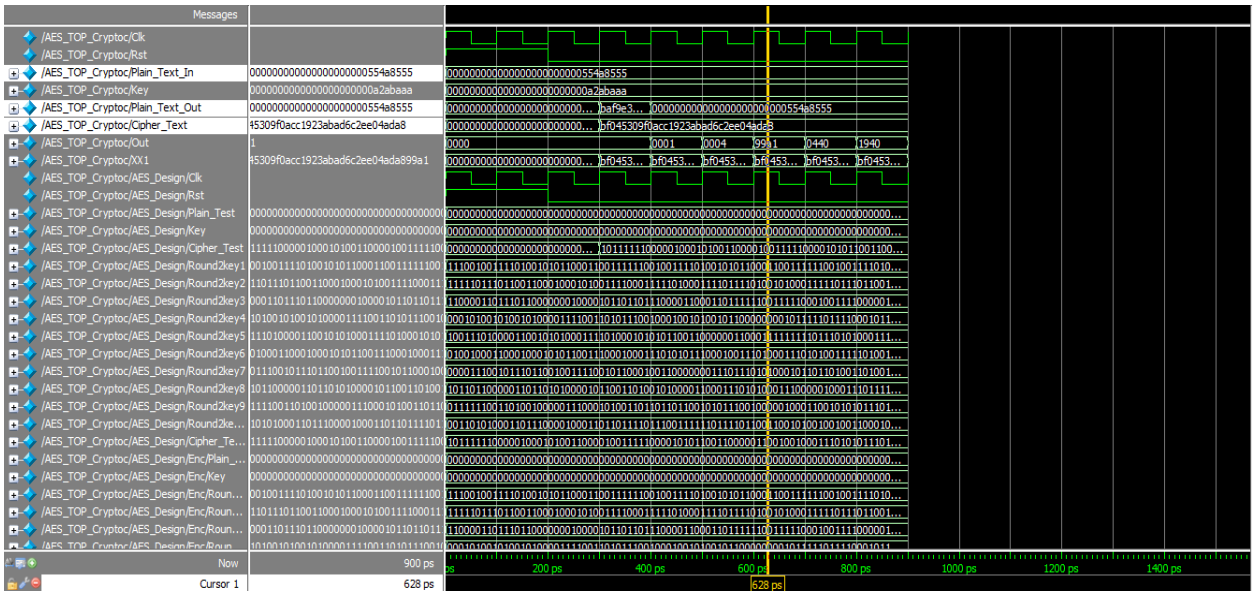


Fig 5.1.1.11. Main Application Simulation

5.1.1.12. Synthesis Report

Table.5.1.1.12. Device Utilization Summary

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	44	66,560	1%	
Number of 4 input LUTs	53	66,560	1%	
Logic Distribution				
Number of occupied Slices	36	33,280	1%	
Number of Slices containing only related logic	36	36	100%	
Number of Slices containing unrelated logic	0	36	0%	
Total Number of 4 input LUTs	53	66,560	1%	
Number of bonded IOBs	30	633	4%	
IOB Flip Flops	16			
Number of MULT18X18s	2	104	1%	
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	8,867			
Additional JTAG gate count for IOBs	1,440			

5.1.1.13. Synthesis Report:

Table.5.1.1.13. Device Utilization Summary

	Non_Segmented_CRC.ngr
Top Level Output File Name	Non_Segmented_CRC
Output Format	NGC
Optimization Goal	Speed
Keep Hierarchy	NO
Design Statistics	
# IOs	50
Cell Usage	
# BELS	76
# GND	1
# LUT2	49
# LUT3	2
# LUT4	2
# MUXCY	11
# XORCY	11
# FlipFlops/Latches	60
# FD	16
# FDC	15
# FDP	1
# FDR	28
# Clock Buffers	1
# BUFGP	1
# IO Buffers	29
# IBUF	13
# OBUF	16
# MULTs	2
# MULT18X18	2

5.1.1.14. Device Utilization summary:

Table.5.1.1.14. Device Utilization Summary

Selected Device	3s5000fg900-4			
Number of Slices	34	out of	33280	0%
Number of Slice Flip Flops	60	out of	66560	0%
Number of 4 input LUTs	53	out of	66560	0%
Number of IOs	50			
Number of bonded IOBs	30	outof	633	4%
Number of MULT18X18s	2	outof	104	1%
Number of GCLKs	1	out of	8	12%

5.1.1.15. Partition Resource Summary: No Partitions were found in this design.

5.1.1.16. Timing Report:

Note: these timing numbers are only a synthesis estimate for accurate timing information please refer to the trace report generated after place-and-route.

Clock Signal | Clock buffer(FF name) | Load |

Clk | BUFGP | 60 |

Asynchronous Control Signals Information:

Control Signal | Buffer(FF name) | Load |

Rst | IBUF | 16 |

5.1.1.16. Timing Summary:

Speed Grade: -4

Minimum period: 9.831ns (Maximum Frequency: 101.723MHz)

Minimum input arrival time before clock: 11.241ns

Maximum output required time after clock: 7.165ns

Maximum combinational path delay: No path found

Timing Detail:

[All values displayed in nanoseconds (ns)]

Timing constraint: Default period analysis for Clock 'Clk'

Clock period: 9.831ns (frequency: 101.723MHz)

Total number of paths / destination ports: 3247 / 60

Delay: 9.831ns (Levels of Logic = 5)

Source: MM_0/lfsr_state_13 (FF)

Destination: MM1/Result_15 (FF)

Source Clock: Clk rising

Destination Clock: Clk rising

5.1.2. Technology Schematics

5.1.2.1. CRC Design:

Fig 5.1.2.1.1. CRC Design

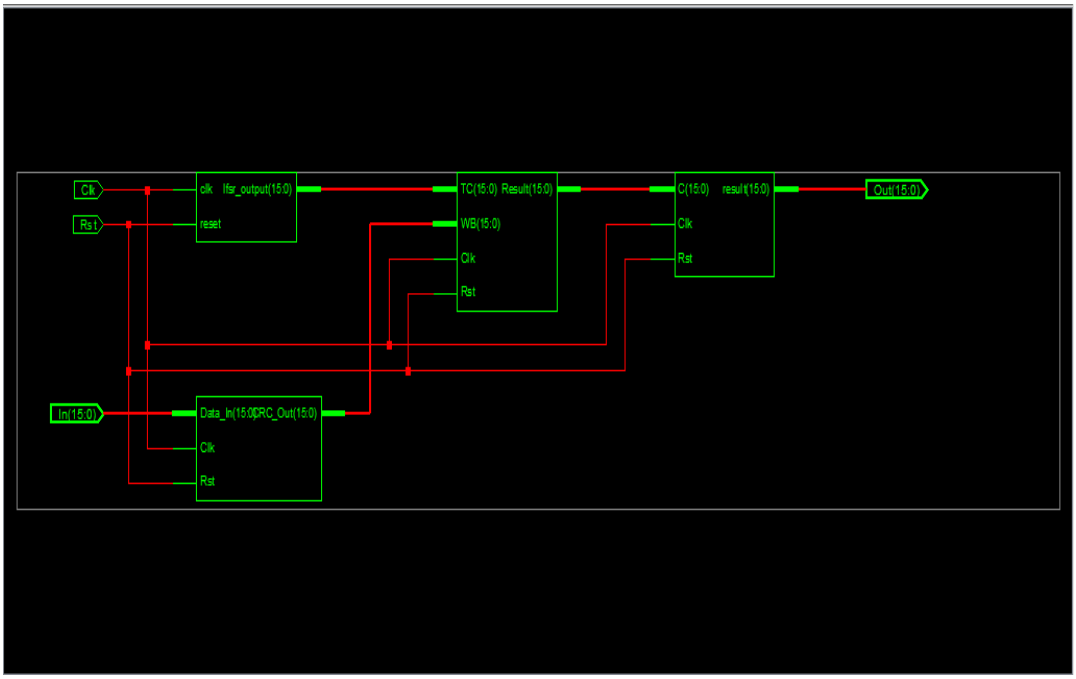


Fig 5.1.2.1.2. Internal Blocks of CRC Design

5.1.2.2. RTL View of Main AES Block

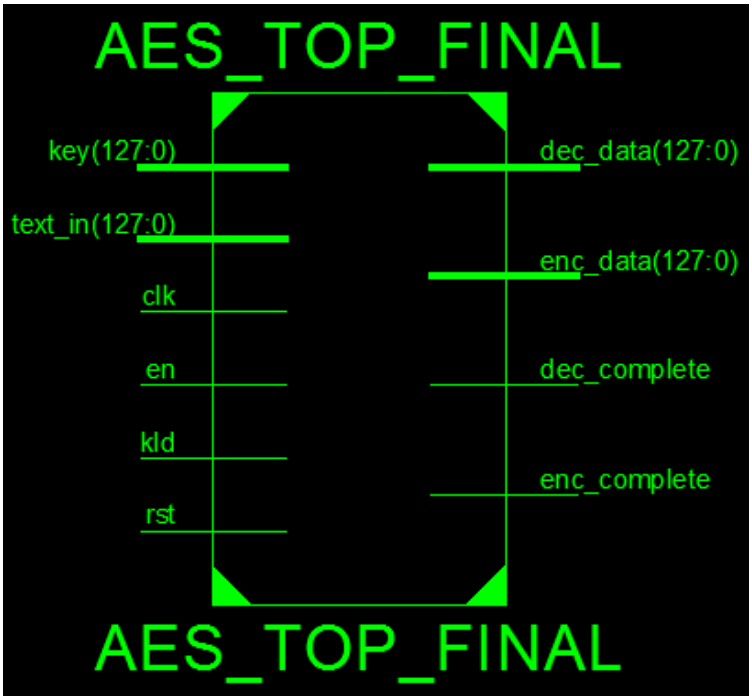


Fig 5.1.2.2. AES Main block

5.1.2.3. Inner View of AES Main Module

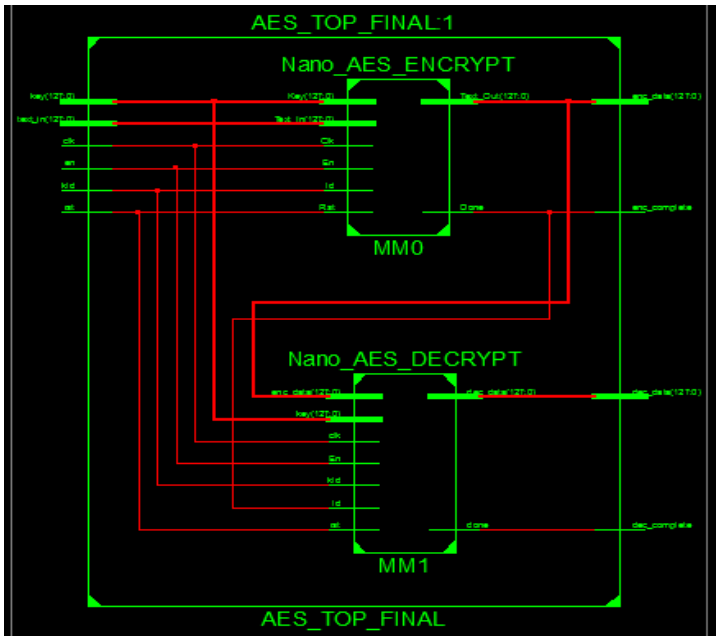


Fig 5.1.2.3. Inner View of AES Module

5.1.2.4. Encryption and Decryption Module

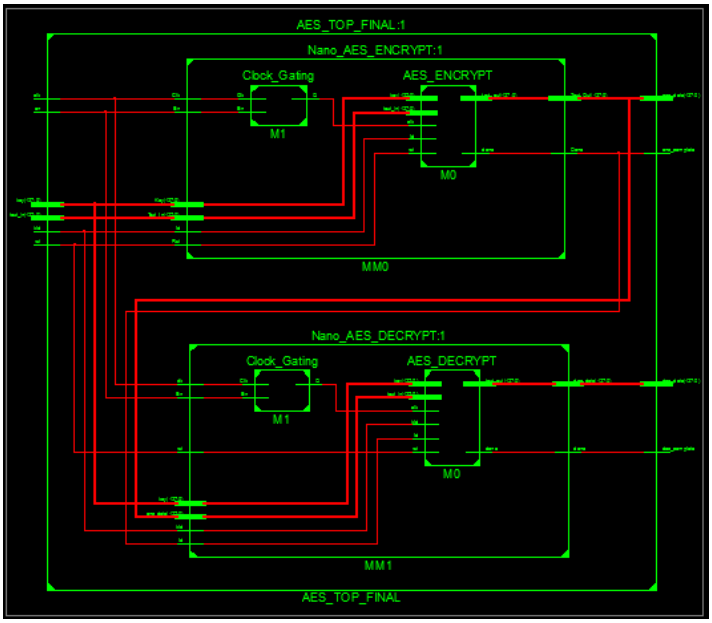


Fig 5.1.2.4.1. Encryption and Decryption Module

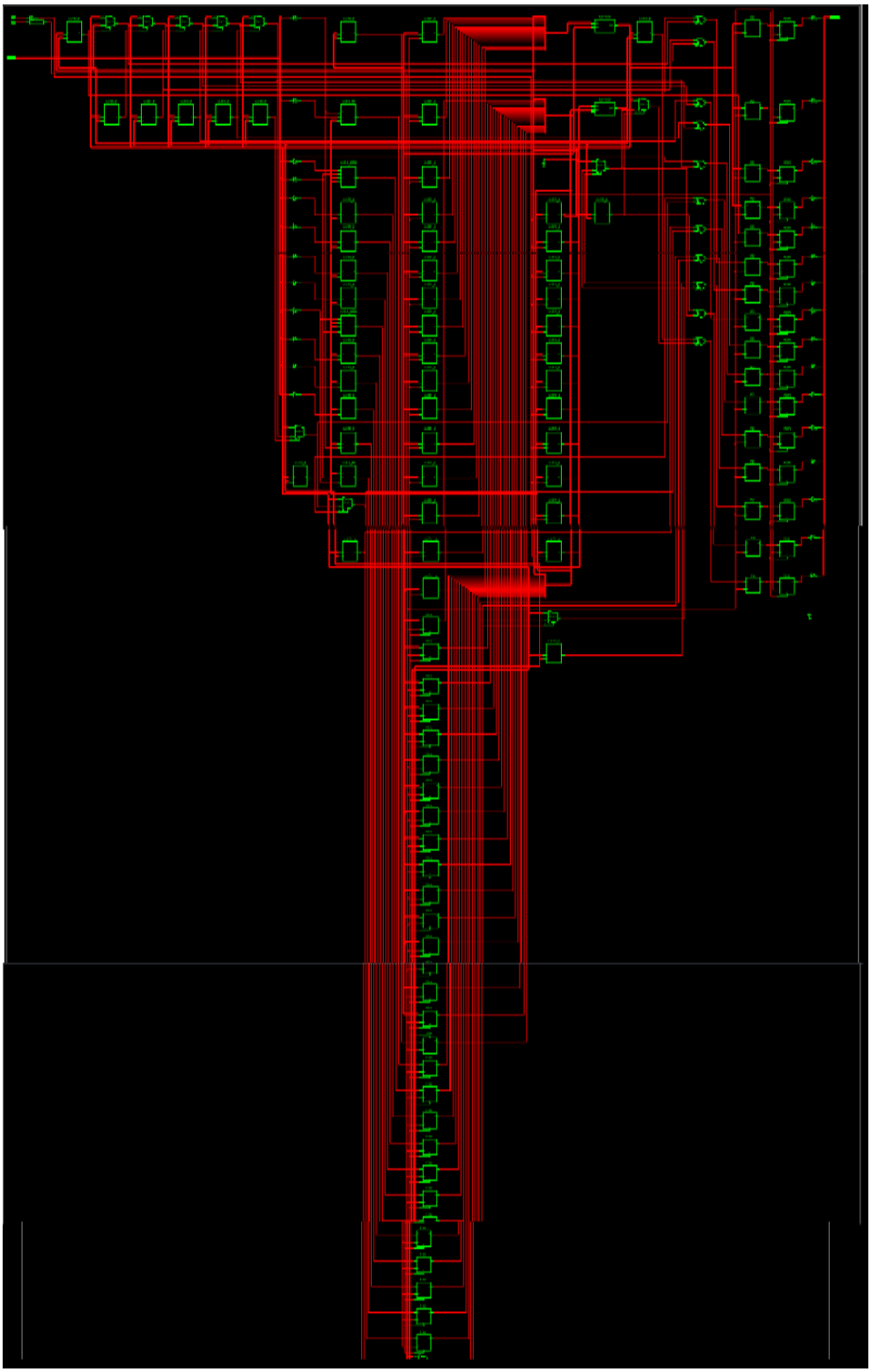


Fig 5.1.2.4.2.Detailed View of Encryption and Decryption Module

5.1.2.5. Technological Schematic of Nano AES Module

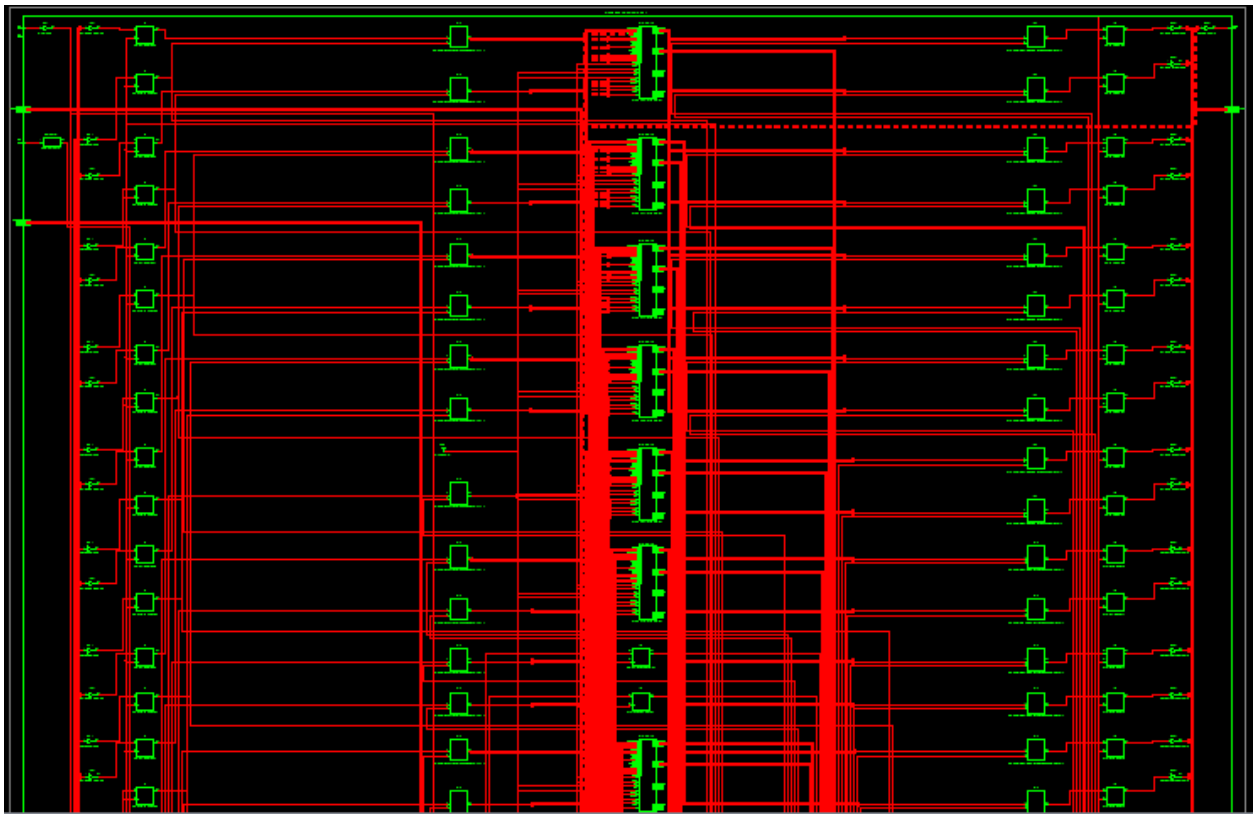


Fig 5.1.2.5. Technological Schematic of Nano AES Module

5.1.3. Comparison Table:

Table 5.1.3. Comparison Table

	Area			Delay		
	LUT	Slices	Gate Counts	Overall Delay	Gate Delay	Path Delay
Non Segmented CRC Design	53	36	8867	7.165ns	6.364ns	0.801ns
Segmented CRC Design	66	40	9001	7.165ns	6.364ns	0.801ns

5.1.3.1. Area Comparison:

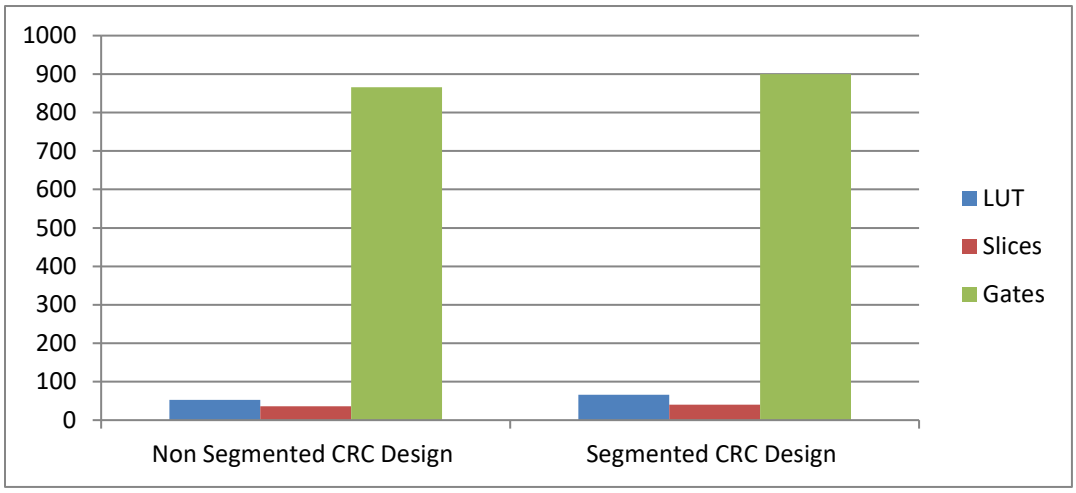


Fig5.1.3.1. Area Comparison

5.1.4.2. Delay Comparison:

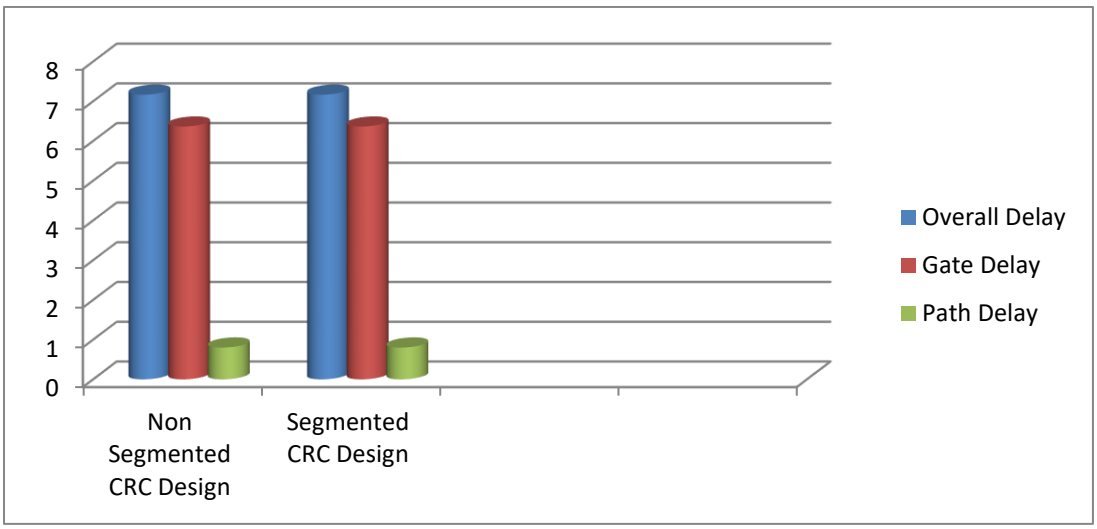


Fig 5.1.3.2. Delay Comparison

Chapter 6: Future Trends

6.1. Conclusion:

In cryptography, CRC is used as means of ensuring the integrity of encrypted data during transmission. We are combining AES encryption with CRC error detection to ensure accuracy of transmitted data and a comparison is made between segmented and non-segmented CRC design and the best amongst the two is consider for TRNG generation. The overall proposed system increases the speed, reduces the area consumption while maintaining the actual algorithm intact.

6.2. Future Trends:

In the future, the design and implementation of CRC-based cryptography using FPGA for secure data communication, along with the AES algorithm, may see several trends. Here are some possibilities.

Increased FPGA Performance: As FPGA technology advances, we can expect higher performance and larger capacity FPGAs to become more readily available. This would enable the implementation of more complex and computationally intensive cryptographic algorithms such as AES, along with CRC-based error detection codes.

Enhanced Security Features: In response to evolving security threats, future designs may incorporate additional security features in CRC-based cryptography using FPGA. These could include hardware-based random number generators, side-channel attack countermeasures, and tamper-resistant modules to protect cryptographic keys.

Improved Efficiency: Efforts will likely focus on optimizing the design and implementation of CRC-based cryptography to achieve better efficiency in terms of power consumption, area utilization, and throughput. Techniques like parallel processing, pipelining, and algorithmic optimizations may be employed to enhance overall system performance.

Integration with Network Protocols: To facilitate secure data communication over networks, future designs may integrate CRC-based cryptography with standard network protocols such as Ethernet, IP, or wireless communication protocols. This integration would enable seamless encryption and decryption of data at the network layer, providing end-to-end security.

FPGA-based Cryptographic Accelerators: FPGA-based cryptographic accelerators could become more prevalent in the future, offering dedicated hardware modules for cryptographic operations. These accelerators could be designed to support multiple cryptographic algorithms, including CRC and AES, and provide high-speed encryption and decryption capabilities.

Emphasis on Post-Quantum Cryptography: As the development of quantum computers progresses, there will be a growing need for post-quantum cryptographic algorithms. Future trends may involve the integration of post-quantum cryptographic algorithms alongside

traditional ones like AES, ensuring compatibility with both classical and quantum-resistant encryption methods

Hardware/Software Co-design: To achieve a balance between flexibility and performance, future designs might explore hardware/software co-design methodologies. This approach would involve implementing critical parts of the CRC-based cryptography system in hardware (FPGA) while utilizing software-based techniques for control, configuration, and higher-level operations.

Standardization and Interoperability: As CRC-based cryptography gains wider adoption, future trends may involve the establishment of standards and protocols for interoperability among different FPGA-based implementations. This would enable secure data communication across various systems and devices, promoting compatibility and ease of integration

It's important to note that these trends are speculative and based on potential future developments. The actual direction of CRC-based cryptography using FPGA, along with the AES algorithm, may vary depending on technological advancements, security requirements, and industry demands.

References:

[1]. Raghunath B H;Aravind H. S., "An Efficient FPGA-Based Dynamic Partial Reconfigurable Implementation", IJISAE,2023

[2] Aprilia Putri Dewanty; Bheta Agus Wardijono, "Analysis and Design of CRC-32 IEEE 802.3 Generator for 8 Bit Data Using VHDL", KILAT ,2022

[3] Noor Munir, Majid Khan, Tariq Shah, Ammar S. Alanazi ,Iqtadar Hussain, "Cryptanalysis of nonlinear confusion component based encryption algorithm", Science direct, 2021

[4] Huan Liu, Zhiliang Qiu, Weitao Pan, Member, IEEE, Jun Li, Ling Zheng, and Ya Gao, "Low-Cost and Programmable CRC Implementation Based on FPGA", IEEE Transactions On Circuits And Systems, 2021

[5] Favin Fernandes, GauraviDungarwal, Aishwariya Gaikwad, Ishan Kareliya, Swati Shilaskar, "VLSI Implementation of Cryptographic Algorithms & Techniques: A Literature Review", IEEE, 2019

[6] Behrouz Zolfaghari, Mehdi Sedighi and Mehran S. Falah, "Designing programmable parallel LFSR using parallel prefix trees", Journal of Engg. Research, 2019

[7] Peter Orosz, Tamas Tothfalusi, Pal Varga, "FPGA-Assisted DPI Systems: 100 Gbit/s and Beyond " , IEEE Communication Surveys and Tutorials, 2019

[8]. R. Roman, P. Najera, and J. Lopez, "Securing the Internet of things," Computer, Sep. 2018.

[9]. KizzepattaVigin, Nazarbayev University, Kazakhstan Suhaib A. Fahba, University of Warwick, United Kingdom, FPGA Dynamic and Partial Reconfiguration: A Survey of Architectures, Methods, and Applications, ACM Computing Surveys, Vol. 51, No. 4, Article 72.,2018

[10] M. Mozaffari-Kermani, A. Reyhani-Masoleh, "Concurrent structure independent fault detection schemes for the Advanced Encryption Standard," IEEE Trans. Comput., May 2017.