CHAPTER 5

DESIGN AND DEVELOPMENT OF AES-128/192/256 ENCRYPTOR AND DECRYPTOR UNIT

5.1 INTRODUCTION

From the literature survey, it has been observed that encryption is also a crucial security issue during data transmission in forthcoming wireless devices. Encryption algorithms are used to ensure the security in the transmission channels. Similarly the area and power consumption is also another major thing to be viewed since most of the mobile terminals are battery operated. So a mobile terminal which has an encryption unit with less area and power consumption is appreciated.

Until recently, the most utilized and trusted encryption algorithm has been 3DES. It is based on DES developed in 1970’s. In 1997 NIST initiated an encryption development process for replacing over twenty year served DES, eventually 3DES. As result NIST received several proposals algorithms and finally, after great amount of debate and analysis, a Dutch algorithm called Rijndael was selected as AES. The design requirements for AES were that it should support key sizes of 128,192 and 256 bits. In addition the chosen algorithm was required to be highly resistant against all known cryptographic attacks.

As most of the block ciphers, AES consists of successive, similar iteration rounds. In addition to variable-sized keys AES also supports data
block size of 128, 192 and 256 bits. Depending on sizes of key and data the number of rounds is 10, 12 or 14. Each round contains a byte substitution, data shift and mixing and an encryption key addition. The algorithms most essential part is byte substitutions performed by non-linear s-box data mixing done by applying modular multiplication. The encryption key addition is exclusive XOR operation. Because of public standardization process and popularity of AES, as well as the generally increased interest in information security large numbers of hardware implementations have been published in AES in academic literature. AES implementation that have most significant results and implementation are considered to compare with this research results.

Polydoros et al (2000) described that flexibility of a communication terminal requires the system to be adaptive and reconfigurable. The adaptive system respond to the application changes by properly altering the numerical value of the set of parameters and reconfigurable system can rearrange at behavioral, structural or architectural level (Harada et al 2000).


Elbirt et al (2000) have developed a architecture with partially unrolled design. M.McLoone(2001) presented an architecture capable of performing both encryption and decryption operation for 128-bit key.

However, there are no known single-chip FPGA implementations of the Rijndael algorithm, which perform both encryption and decryption for all the three keys. Hence, in this research work the architecture has been developed for encryption and decryption unit and it has been implemented in
hardware using Xilinx virtex FPGA device. The schematic analysis, device utilization during the implementation, place and routing analysis and power analysis has been done and presented. The hardware has been validated for different schemes.

5.2 ENCRYPTOR/ DECRYPTOR ARCHITECTURE – AES-128/192/256

The basic techniques for implementing a block cipher with rounds are iterated, pipelined and loop-unrolled architecture. The more advanced technique includes partial pipelined and sub-pipelining combined with basic techniques. The developed architecture is designed to implement AES-128/192/256. The developed architecture has advantages such as all the three keys and, both the Encryptor and Decryptor in the same hardware with less power consumption and less area occupancy.

The iterated architecture leads to smallest implementation as it consists of one round component which is fed with its output until required number of round it completed. One round is executed per clock cycle and the implementation can process only one block of data at a time.

The pipelined architecture contains all rounds as separate component with registers in between. As a result it is fastest and largest of basic techniques. After pipeline is filled, an encrypted block is output at every clock cycle. The throughput depends on number of clock cycle and level of pipeline.

The loop unrolled architecture perform two or more rounds per clock cycle and execution of cipher is iterated. The unfolding factor affects area and register to register delay. Even though delay is larger than iterated mode throughput is higher.
In sub pipelining the registers are placed inside the round component in order to increase the maximum clock frequency. In partial pipelining scheme the pipeline contains half of the round with the registers in between. Encryption is performed by processing a data block twice through the pipeline. Due to the feedback loop of utilized mode pipeline in not beneficial in SDR security scheme. The larger size makes fully pipelined structure unattractive for low-cost and low power implementation. Hence iterated and/or loop unrolled architecture should be used. In some case additional registers can be decrease to reduce total resources consumption in FPGA implementation.

The developed architecture has module two modules,

(i) Encryption module

(ii) Decryption module.

5.3 IMPLEMENTATION OF– AES-128/192/256 ENCRYPTION ARCHITECTURE

The block diagram of the architecture developed for implementing symmetric key block cipher is shown in Figure 5.1 Two major functional blocks is encryption/decryption unit and key scheduling unit. The interpretation of the function to be performed by the encryption/decryption unit is not uniform among various design groups involved in the comparison of the AES candidates.
The architectures have been developed for the Rijndael algorithm to reduce the required hardware resources and to achieve high-speed performance. The architectures serve the encryption and decryption process in the same hardware device and use all the three keys of length 128-bit, 192-bit and 256-bit keys.

The architecture of the basic block round of the algorithm is composed of basic building blocks: Sub Bytes, Shift Row, Mix Column, and Key Addition.

1. **SubBytes**: This is the first step of the data transformation process, where each block is replaced by its substitution in an S-Box table. The implementation of the S-Box consists of two different mathematical...
functions: a) the multiplicative inverse of each byte of the State in the finite field GF($2^8$) and b) an affine mapping transformation over GF(2).

2. **Shift Row:** The Shift Row operation is achieved with simple use of four multiplexers.

3. **Mix Column:** This operation is applied over the State column. Every column $S$ of the State consists of four bytes $S = \{S_0, S_1, S_2, S_3\}$. In both encryption and decryption, the state column is multiplied by a different specified polynomial, and, finally, a transformed column $T = \{T_0, T_1, T_2, T_3\}$ is generated. The Mix Column component does not operate in the last round of the algorithm. An appropriate select signal determinates

4. **Key Addition:** The Key Addition component consists of eight 2-input XORs for every byte of the State column. Every bit of the round key is XORed with the appropriate bit of the transformed data byte.

5.3.1 **Encryption Module**

The processing steps of encryption module is shown in Figure 5.2. The input ports are data, three different keys, a select key to reconfigure the device and clock and reset signals. The output port is the encrypted cipher. The system is made reconfigurable by using the select key signal which selects the key which has to be used. If select key is 1 then the key is 128-bit key, if 2 then the key is 192-bit key, if 3 then the key is 256-bit key.
Figure 5.2 Encryption Process of AES

5.3.1.1 Results and Discussion

The developed encryption and decryption unit hardware architecture is implemented by using VHDL. After generating VHDL code, the synthesis, place and route has been done using Xilinx ISE 10.1i. The results obtained during each process are explained for clear understanding.

5.3.1.2 Substitution Bytes (S - Box)

The substitution bytes block gets an 8 bit (two words) data, clock and reset as input and produces an 8 bit (two words) data as output after doing the byte substitution function. The schematic diagram of the substitution bytes is shown in Figure 5.3.
The FDC in the Substitution byte block is a single D-type flip flop with data D which is a 8-bit data with a clock and an asynchronous clear input (CLR) and a data output (Q). The asynchronous CLR when high overrides all the other inputs and sets Q output low. The data on the D input is loaded into the flip flop when CLR is low on the Low-to-High clock transition. The block diagram of the FDC is shown in Figure 5.4.
5.3.1.3 Shift Rows

The shift row module shifts the given input. The module gets a 128-bit state as input and process the input into a 8-bit words and shift the words with respect to their position in a 4x4 array. The output of the module is also a 128-bit word generated using the shift row function. The schematic diagram of shift rows is shown in Figure 5.6.
Inside the shift row block it has an input buffer which gives a 8-bit word at a clock cycle to a FDC which shifts the data as per their position and returns the output data to the output buffer in that clock cycle. The data is obtained from the input buffer when the asynchronous clear (CLR) is high and when the clock transition is from low to high. The blocks inside the shift row module are given in Figure 5.7.

![Figure 5.7 Functional Blocks inside Shift Row Module](image)

### 5.3.1.4 Mix-Column

The mix-column module gets the 128-bit data as input along with a clock and reset command. The mix-column module does the operation of polynomial multiplication with a constant 4×4 matrix and gives an 128-bit output. The main block diagram of the mix-column operation is shown in Figure 5.8.

![Figure 5.8 Schematic Diagram of Mix-column](image)
The blocks inside the mix-column module Data (0) is the 4×4 state input (data) and Data(1) is a 4×4 matrix which is constant for encryption module and it is given to a polynomial multiplication module which does the operation of polynomial multiplication and gives an 128-bit data as input. The blocks inside mix-column module are shown in Figure 5.9.

![Figure 5.9 Functional Blocks inside Mix-Column Module](image)

The blocks inside the polynomial multiplication block are shown in Figure 5.10 the gates present are two input buffers, two 2-input and gates with one input inverted and a two-input or gate and an output buffer.

![Figure 5.10 Blocks inside Polynomial Multiplication Block](image)

5.3.1.5 Key expansion

The key expansion block gets key and the round number as input and gets a 128-bit data as output. It has many gates and buffers inside it which
does the operation of key expansion. The schematic diagram of key expansion is shown in Figure 5.11. The key expansion block is the main block which makes the system reconfigurable. It get all the three keys as input along with the round number and key. The key gets only three different inputs 1 or 2 or 3 and for 1 128-bit key expansion will be performed similarly for 2 and 3 192 and 256 bit key expansion is performed.

![Schematic Diagram for Key Expansion](image)

**Figure 5.11 Schematic Diagram for Key Expansion**

### 5.3.1.6 Add round key

The block diagram of the add round key takes data and round key as input along with clock and reset signals and gives a 128-bit output. The add round key block has an input buffer and an output buffer with a processing unit shown in Figure 5.12 which has 128 separate processing blocks which is shown in Figure 5.13.
Figure 5.12 Schematic Diagram of Add Round Key

Figure 5.13 Functional Blocks inside Add round key
Figure 5.14 Functional Blocks Inside Processing Block

The Figure 5.14 shows the gates inside block 122 in the processing block which does the operation of XOR for two data, data(250) and data(122) to produce a result 122. The schematic diagram of gates inside the blocks shown in Figure 5.15.

Figure 5.15 Gates Inside Block 122
The encryption module which performs the encryption process using all the three possible keys of AES-128/192 and 256 is designed using VHDL. The schematic diagram of encryption module of AES is shown in Figure 5.16.

5.4 IMPLEMENTATION OF AES-128/192/256 DECRYPTION ARCHITECTURE

The decryption process has the same structure as the encryption architecture. The only main difference is that for every function that is used in the basic round, the mathematical inverse is taken. The key expansion unit performs almost the same operation with the encryption process. The only difference is that the decryption of the round keys is obtained by applying the inverse MixColumn to the corresponding round keys. The decryption process of AES is shown in Figure 5.17. The initial value of the key for the decryption operation is changed. The appropriate basic decryption key (Daemen and
Rijmen 2001) must be loaded in the key buffer before the decryption beginning.

![Figure 5.17 Decryption Process of AES](image)

### 5.4.1 Inverse Shift Rows

The block diagram of the inverse shift row module is shown in Figure 5.18. The shift row operation is shown in algorithm 5.6. It takes 128-bit data as input along with clock and reset and gives a 128-bit data as output.
5.4.2 Inverse Byte Substitution

The block diagram of inverse byte substitution function is shown in Figure 5.19. The inverse byte substitution uses an substitution box which is the inverse of the substitution box which is used in encryption process. The module gets an 128-bit data as input and stores it in the input buffer and gives a 8-bit data at one cycle and gets the output and stores it in the output buffer which is shown in Figure 5.20.
Figure 5.20  Functional Blocks Inside Inverse Byte Substitution Module

5.4.3 Inverse Mix Column

The block diagram of inverse mix column transformation is shown on Figure 5.21.
5.4.4 Decryption Architecture

The decryption implementation results are similar to the encryption implementation. The key expansion module is modified in the reverse order. In which last round key is treated as the first round and decreasing order follows. The complete decryption process is shown in Figure 5.22.
5.5 IMPLEMENTATION OF AES ENCRYPTOR AND DECRYPTOR

The architecture designed to implement encryption and decryption are combined and AES encryptor/decryptor has been developed. The schematic diagram of the hardware containing both encryptor and decryptor is given in Figure 5.23.

![Schematic Diagram Of Encryptor And Decryptor](image)

Figure 5.23 Schematic Diagram Of Encryptor And Decryptor

5.5.1 Synthesis Output

The first stage of the synthesis is to analyze the generated code to check the compatibility for synthesizing. After analyzing the source code, the target device has been synthesized and the net list has been created. The device utilization summary for the target architecture of 2V4000BF957-6 has been provided in the Table 5.1.
Table 5.1 Device Utilization Summary for Synthesizing Target Device

<table>
<thead>
<tr>
<th>Allocated Area</th>
<th>Used /available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>Virtex 2v 4000</td>
</tr>
<tr>
<td>Slices</td>
<td>2943/23040</td>
</tr>
<tr>
<td>LUTs</td>
<td>5802</td>
</tr>
<tr>
<td>IOs</td>
<td>836/</td>
</tr>
<tr>
<td>Bonded IOs</td>
<td>384/684</td>
</tr>
<tr>
<td>Delay</td>
<td>15.899ns</td>
</tr>
</tbody>
</table>

It is seen from the Table 5.1 that the developed model for encryption and decryption utilizes 3.6% of bonded IOBs. The delay in developed architecture is around 16ns. The total lookup table utilized is 5802 out of available 23040. The operating speed of the developed architecture is 62.5 MHz.

5.5.2 Testing and Verification

The developed architectures is implemented by using VHDL, with structural description logic and simulated for the correct encryption and decryption operation using the test vectors provided by the AES submission package (NIST 2000). The VHDL codes of the designs are synthesized, placed, and routed using FPGA devices of Xilinx Virtex device. The RTL schematic of developed architecture is shown in Figure 5.24. The throughput reaches the value of 16099 Mbits/s for both encryption and decryption process.
Figure 5.24 RTL Schematic of Encryptor and Decryptor
5.6 VERIFICATION AND VALIDATION OF THE RECONFIGURABLE HARDWARE

5.6.1 Verification of the Hardware

The implemented reconfigurable encryption and decryption architecture has been verified and validated for different inputs. For example, the generated model has been verified for given input and the output is shown in Figures 5.25 to 5.27 respectively. The simulation result generates 128 bit output for three different key and correctness verified with test vectors.

Figure 5.25 Simulated Output of 256 bit key
Figure 5.26 Simulated Output of 192-bit key
Figure 5.27  Simulated Output of 128-bit key
5.7 PERFORMANCE ANALYSIS

The developed architectures support both encryption and decryption in the same dedicated hardware device. The developed architectures performances has been compared with existing works and is tabulated in Table 5.2

Table 5.2 Comparison of Hardware Performance of the AES Candidates Using Different Architecture

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Process</th>
<th>Device</th>
<th>CLB Slices</th>
<th>Frequency MHZ</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elbirt2000</td>
<td>Encryption</td>
<td>XCV1000BG560</td>
<td>5302/10992</td>
<td>141/318</td>
<td>300/1940</td>
</tr>
<tr>
<td>Mroczkowski(2001)</td>
<td>Decryption</td>
<td>ALTERA</td>
<td>2885</td>
<td>41.5</td>
<td>248</td>
</tr>
<tr>
<td>Gaj2000</td>
<td>Encryption/Decryption</td>
<td>XILINX</td>
<td>2902</td>
<td>25.9</td>
<td>331</td>
</tr>
<tr>
<td>Kuo2001</td>
<td>Encryption/</td>
<td>ASIC APPROACH</td>
<td>3.96 µm²</td>
<td>100</td>
<td>910</td>
</tr>
<tr>
<td>Skalvos 2002</td>
<td>Encryption/Decryption</td>
<td>XCV3000BG432</td>
<td>2358</td>
<td>22</td>
<td>259</td>
</tr>
<tr>
<td>Skalvos 2002</td>
<td>Encryption/Decryption</td>
<td>XCV1000BG560</td>
<td>17314</td>
<td>28.5</td>
<td>3650</td>
</tr>
<tr>
<td>Yibo fan et al (2007)</td>
<td>Encryption</td>
<td>ASIC APPROACH</td>
<td>0.18µm²</td>
<td>182</td>
<td>360</td>
</tr>
<tr>
<td><strong>Developed Architecture</strong></td>
<td><strong>Encryption/Decryption</strong></td>
<td><strong>XCV2V4000BF957-6</strong></td>
<td><strong>2973</strong></td>
<td><strong>62.896</strong></td>
<td><strong>16099</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Virtex 2v 6000</td>
<td>14509</td>
<td>22.25</td>
<td>5696</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Spartan 3e 1600</td>
<td>15425</td>
<td>12.74</td>
<td>3261</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Virtex E 3200</td>
<td>16467</td>
<td>9.41</td>
<td>2408</td>
</tr>
</tbody>
</table>

The most of the publication have been targeted at comparing the final candidate algorithm AES standardization process for enabling fair comparison. Of these earliest publication presents a comprehensive study of AES encryption data path with different architectural choice and lacks
decryption and key generation. AES data path implementation along with decryption (Fischer et al (2001) Weeks (2000) Gaj 2000 Kuo 2001 Skalvos 2002 Shuenn-Shyang Wang et al (2004)) are published. However they are large implementation. In this research work AES with all the three keys 128 bits, 192 bits and 256 bits are implemented in same architecture which is well suited for SDR. In addition it is the first to utilize all the three keys and with both the encryption and decryption. Compared with other implementations it is found that total resources consumption is larger. Though the size is larger throughput comparison with other implementation show that the developed architecture is higher. The developed architecture is also simulated in different Xilinx devices and their performances are evaluated and shown in Table 5.3. The synthesis results shows that the proposed design performs better in all devices than the architectures available previously. Among various FPGA devices available the virtex2v4000 is better for the developed architecture with high speed and maximum throughput. Hardware utility of developed architecture in Virtex 2v4000 is found to be less when compared with other devices. Similarly uses less number of bonded IOs and also less number of slices. Efficiency of this device is also found to be better with others.

Table 5.3 Comparison of Performance of Developed Architecture with Different Devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Virtex 2v 6000</th>
<th>Virtex 2v 4000</th>
<th>Spartan 3e 1600e</th>
<th>Virtex E 3200E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slices</td>
<td>14509</td>
<td>2943</td>
<td>15425</td>
<td>16467</td>
</tr>
<tr>
<td>LUTs</td>
<td>28488</td>
<td>5802</td>
<td>30365</td>
<td>32379</td>
</tr>
<tr>
<td>IOs</td>
<td>836</td>
<td>836</td>
<td>1282</td>
<td>836</td>
</tr>
<tr>
<td>Bonded IOs</td>
<td>642</td>
<td>384</td>
<td>1280</td>
<td>642</td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>49.906</td>
<td>15.899</td>
<td>78.456</td>
<td>106.222</td>
</tr>
<tr>
<td>Throughput (Mbps)</td>
<td>5696</td>
<td>16099</td>
<td>3261</td>
<td>2408</td>
</tr>
<tr>
<td>Throughput/slice</td>
<td>0.411</td>
<td>5.47</td>
<td>0.211</td>
<td>0.14</td>
</tr>
<tr>
<td>Clock frequency MHz</td>
<td>22.25</td>
<td>62.89</td>
<td>12.74</td>
<td>9.41</td>
</tr>
</tbody>
</table>
5.8 SUMMARY

The Advanced Encryption Standard-Rijndael algorithm is an iterative private key symmetric block cipher that can process data blocks of 128 bits through the use of cipher keys with lengths of 128, 192, and 256 bits. An efficient FPGA implementation of 128 bit block and processed with 128/192/256 bit key AES-Rijndael cryptosystem has been proposed in this work. Optimized and Synthesizable VHDL code is developed for the implementation of both 128 or 192 bit or 256 bit data encryption and decryption process and description is verified using ISE 10.1 functional simulator from Xilinx. All the transformations of algorithm are simulated using an iterative design approach in order to minimize the hardware consumption. Each program is tested with some of the sample vectors provided by NIST. The throughput reaches the value of 16090Mbit/sec for both encryption and decryption process with Device XC2V4000 series of Xilinx Virtex Family. Thus a reconfigurable encryption and decryption unit is designed implemented and the simulation output is got by using the xilinx ISE simulator with virtex 2 device family and Virtex E, Spartan 3E devices and all the outputs are compared. It is proved that the modified architecture performs better in improving the throughput and optimized the hardware utilization.