© 2017 IJSRST | Volume 3 | Issue 8 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

A Comparative Study on the Implementation of Block Cipher Algorithms on FPGA

Lavanya R*[1], Karpagam M[2], Jaikumar R[3]

*[1]Department of Electronics and Communication Engineering, Sri Ramakrishna Institute of Technology, Coimbatore, Tamilnadu, India
[2]Department of Electrical and Electronics Engineering, Hindusthan College of Engineering and Technology, Coimbatore,Tamilnadu, India
[3]Department of Electronics and Communication Engineering, R V S College of Engineering and Technology, Coimbatore,Tamilnadu, India

ABSTRACT

Advanced Encryption Standard(Rijindael) is a widely used algorithm for data security, but there are many notable block cipher algorithms which have not been widely adopted for usage. The algorithms such as Blowfish, Serpent, Twofish and RC6 were the top finalist of the AES contest but are not widely used. They have strong security features compared to that of AES and cryptanalysis of these algorithms are still difficult. These algorithms are realized on FPGA to carry out a comparative study on the timing, functional blocks and area requirements. This study helps understand different aspects of the algorithms and identify suitable applications with given limitations where they can be applied.

Keywords: Cryptography, Symmetric Encryption, Block Ciphers, FPGA

I. INTRODUCTION

Internet is now an essential platform where personal databases are stored, bank transactions are carried out, social interaction happens and exchange of large important information takes place, and hence security becomes an extremely important issue for all users. One essential way of delivering security to end users is through cryptography. Cryptography is a technique, which applies complex mathematical transformations to hide secret information that needs to be communicated through unsafe channels. Cryptography provides privacy/confidentiality for the information/message intended for the recipient whereby it helps in concealing data from the third party. In cryptography the original information is called the plaintext and the process of converting the plaintext to unintelligible information is called encryption. The resultant information is called the cipher text. The conversion is done by using an essential component called the Key. At the receiver, the process of inverse conversion from cipher text to plaintext is called as decryption. There are many encryption algorithms formulated for the purpose of secure information transfer. The cryptographic algorithms can be classified based on the type of key used as secret key cryptography and public key cryptography. Secret key cryptography is primarily used for privacy and confidentiality it uses a single key for both encryption and decryption. Public key cryptography is suited for authentication, non-repudiation, and key exchange it uses one key for encryption and another key for decryption.

II. SECRET KEY CRYPTOGRAPHY

Secret key cryptography (Symmetric Encryption) uses a common key for encryption and decryption. The plaintext is converted into the ciphertext at the transmitting end using a key K. The same key K is used at the receiver to decrypt the information. Hence the key acts as a shared entity between the two parties of communication. The symmetric cryptography is depicted in Figure 1.Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. In this system, it is necessary that the key must be known to both the sender and receiver prior to transmission. The distribution of key plays an important factor in this type. Symmetric key Cryptography are generally classified as stream ciphers and Block ciphers.

Stream ciphers encrypts data as a bit or as byte or as a bit stream as applicable to the algorithm used whereas block ciphers encrypts data in terms of blocks, one block at a time, each block is processed by the encryption algorithm using the same key.

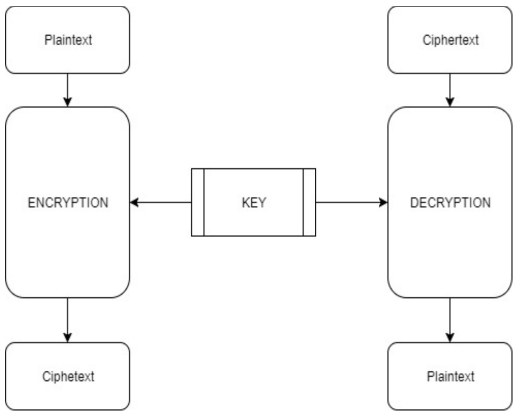

Figure 1 : Symmetic Key Cryptography

In the case of stream ciphers the streams of data processed by the algorithm use a different key value for each encryption. The keys are generated through different forms of feedback mechanism.

A. Block Ciphers

Block ciphers consist of paired algorithms, one for encryption and the other for decryption. A block cipher handles a block of length n bits from a string of P bits, the key used is of size k bits. The Encryption function E yields a ciphertext C which of length n bits. In block ciphers plain text and ciphertext size are equal-n bits. The Decryption algorithm is the inverse for E,D = E^{-1} is a function taking a key K and a ciphertext C to return a plaintext value P.

$$C_n= E_k(P_n)$$
$$P_n = D_k(C_n)$$

Where

C - Ciphertext

E - Encryption function

P - Plaintext

D - Decrytion function

K - Key size

n - Block size

In general, the plaintext block will always encrypt to the same ciphertext when using the same key. The most commonly used encryption methodology in block ciphers is the combination of operations such as substitution, permutation and round function. A substitution function replaces the input data into another value of output data. The permutation function performs bit shuffling/ transposition operation on the processed bits from substitution function. The Round function is generally an XOR operation whereby the key is XORed with the processed plaintext. The most commonly used block cipher architecture is based on the Feistel structure. In this method the block is split into two equal halves, one half is processed using the round key function sing the sub-key derived from the master key. The output of round function is then XORed with the other half of the plain text. These two values are again swapped and the process is continued as an iterative structure for n number of rounds. The main advantage of feistel design is the invertibility, requiring only the reversal operation in decryption which reduces the system complexity in terms of hardware and software[1].

B. Modes of Operation of Block Cipher

Block ciphers process the plain text as blocks of n length, for p length messages the data needs to be split to m n-bit block, where m is the number of blocks that can be formed from the given plaintext. These m n bit blocks needs to be encrypted for which the modes of operation allows easier transformation. The simplest mode is the Electronic Code Book (ECB) where message is split into n-bit blocks and padding bits are appended to the last block. Here encryption and decryption of each block is performed independently. Next is the Cipher Block Chaining (CBC) mode in order to enhance the security initialization vector is XORed with the plaintext before encryption. The resulting ciphertext of the first block acts as an initialization vector for the second plaintext block and the process is repeated. The Initialization vector is assumed to be a pseudorandom number. The third mode is the Cipher Feedback mode (CFB) where the initialization vector is first encrypted and then XOred with the plain text. The output feedback (OFB) mode encrypts the initialization vector to generate the key for all the blocks. The counter uses unique random values, whereby the initialization vector is used as a block counter and this counter is encrypted for each block of data. The modes of operation are meant to semantic security, it means that with a known part of ciphertext with unknown key one should not be able to practically derive any information.

C. Block Cipher Algorithms

There are a variety of block cipher algorithms that were proposed with its own advantage and security perspectives. A few of them are Camellia, CAST-128, IDEA, RC2, RC5, RC6, SEED, ARIA, Skipjack, DES, SDES, AES, PRESENT, BLOWFISH, TWOFISH, SERPENT. A Few notable block ciphers are taken for study of implementation aspects of the algorithms on the Virtex-7 FPGA platform[2].

1. Blowfish

Blowfish uses a feistel structure which encrypts a block of length 64 bits, the key length is variable from 32 bits to 448 bits shown in figure 2. It uses a iterative structure with 16 rounds of processing and it utilizes large key dependent S boxes. The structure of blowfish is similar to CAST-128, in which fixed S boxes were used. Each round of blowfish consists of four operations, the plaintext is split into two equal halves. The operations are performed using five sub key arrays. Blowfish's key schedule starts by initializing the P-array and S-boxes with values derived from the hexadecimal digits of pi, which contain no obvious pattern. There is one 18 entry P-array and there are four 256 entry S boxes called S0, S1, S2 and S3. The First operation involves performing XOR operation of the left half of plaintext with the r^{th} P-array value, secondly this data is fed as input to the F-function of blowfish, next this value is XORed with the right half of the plaintext and lastly perform a swap operation of the two halves. The Function F take the 32 bit data as four 8-bit values, which are fed independently to the four S boxes. The outputs are added modulo 2 and then XORed to get the final 32 bit. Decryption is exactly the same as encryption, except that P1, P2,..., P18 are used in the reverse order. Blowfish is license-free and is available free for all uses and with use of P-array and S-array based on Pi it offers good protection against brute-force attacks[2].

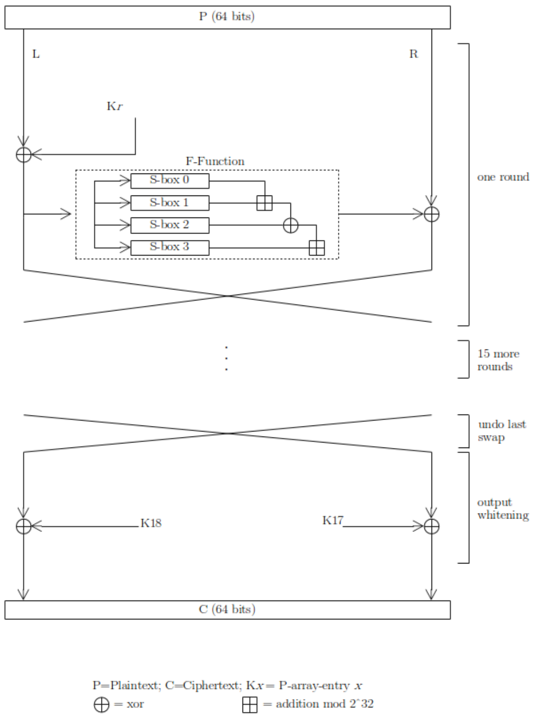

Figure 2. Blowfish Structure

2. Serpent

Serpent is a symmetric block cipher that was in the Advanced Encryption Standard contest and was ranked second to selected finalist Rijndael. It encrypts plaintext of block size 128 bits and supports three variable key sizes of 128,192 and 256 bits. It uses 32 rounds of substitution and permutation network . The Network operates on a block of four 32 bit words. Each round applies one of eight 4-bit to 4-bit S-boxes 32 times in parallel shown in Figure 3. Serpent was designed so that all operations can be executed in parallel, using 32 bit slices. The 32 rounds of Serpent offers higher security margin than Rijndael[3].

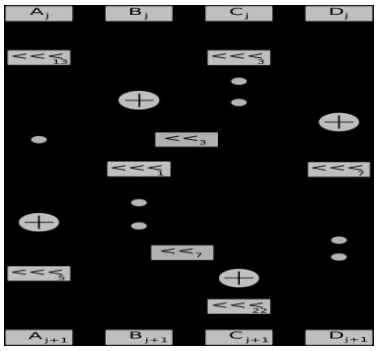

Figure 3. Serpent Structure

3. Twofish

Twofish is s a symmetric block cipher with a block size of 128 bits and variable key sizes of 128,192 and 256 bits shown in Figure.4. Twofish was one among the AES finalists. Twofish is a derivative of Blowfish in addition it uses pre-computed key dependent s boxes and a complex key schedule. In Twofish one half the key is used as the Key for the encryption process whereas the other half is used to produce the key dependent S-boxes, it utilizes pseudo Hadamard Transform in its process[4].

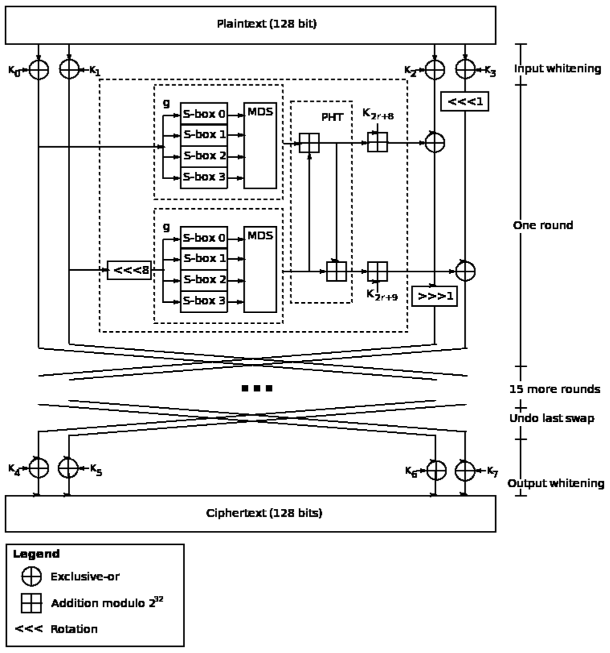

Figure 4. Twofish Structure

4. RC6

RC6 (Rivest cipher 6) is a symmetric key block cipher it operates on block length of 128 bits and allows the usage of key of length128, 192, and 256 bits up to 2040-bits.RC6 is parameterized to enable variable block length, key length and rounds. RC6 is similar to RC5 in structure. RC6 can be considered as two RC5 encryption processes running in parallel. RC6 encryption process involves sub process such as rotation, modular addition and XOR operation.RC6 uses data dependent rotations and an additional multiplication operation to increase the degree of confusion[5].

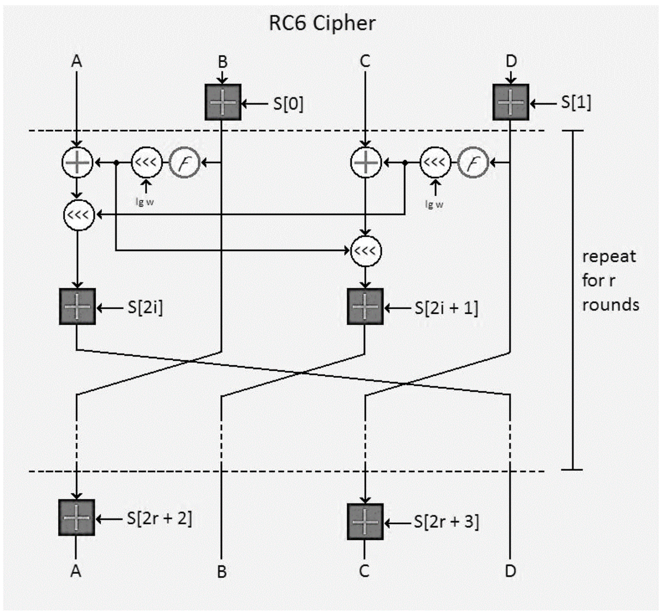

Figure 5. Twofish Structure

III. IMPLEMENTATION ENVIRONMENT AND RESULTS

In this section we present the main contribution of this paper where the above mentioned block cipher algorithms which were the finalists in the Advanced Encryption Standard(AES)[6]contest are implemented and tested for the purpose of comparative study to be carried on their timing requirements and architectural essentials on a Field Programmable Gate Array(FPGA). These algorithms were described in VHDL and synthesized on Xilinx Virtex-7 FPGA family, XC7V2000T device, package FLG1925 with speed grade of -2. The functionality of the algorithms are verified and then synthesized, the reports are tabulated below to give a summary of the details in the synthesized FPGA architecture.

TABLE I
TIMING RESULTS

Parameters	Algorithm			
	BLOWFISH	SERPENT	TWOFIS-H	RC6
Input Plaintext size(bits)	64	128	128	128
Key size(bits)	32-448	128,192 and 256	128,192 and 256	128,192 and 256
Minimum period(ns)	6.842	21.196	10.838	608.299
Maximum Frequency (MHz)	146.149	47.179	92.271	1.644
Minimum input arrival time before clock(ns)	1.583	1.439	1.055	1.808
Maximum output required time after clock(ns)	6.173	0.575	0.575	0.575

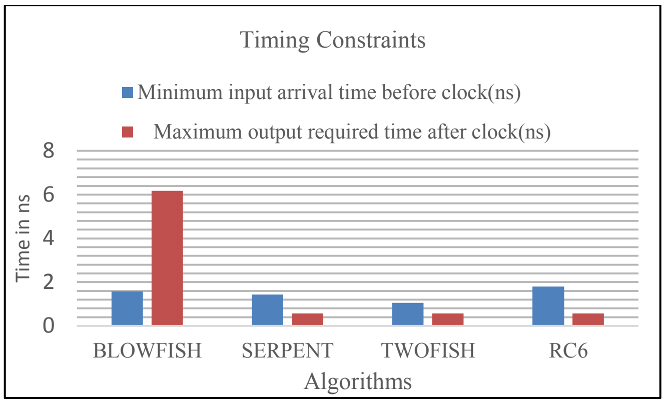

Figure 6. Timing constraints

The results above shown in Table I and Figure 6 indicate the maximum output required time after clock for blowfish has large deviation from the maximum input arrival time before clock. This increase in delay is attributed to the computation of the P-array and four S-array at run time. Whereas the blowfish variant Twofish which uses pre-computed arrays for keys shows a much lesser delay in the same comparison. The other two algorithms on the average have a smaller and acceptable delay.

III
SYNTHESIZED MACROSTATISTICS

Elements	Algorithm			
	BLOWFISH	SERPENT	TWOFIS-H	RC6
No. of RAMs	7	1056	-	-
No. of Adders/ Sub tractors	6	-	124	785
No. of Registers	25	17060	216	27054
No. of Multiplexers	82	363	201	909
No. of Multipliers	-	-	-	40

No. of Logic Shifters	-	-	-	211
No. of FSMs	2	-	-	-
No. of XORs	24	1058	2	80

TABLE IIIII
SLICE LOGIC UTILIZATION

Elements	Algorithm			
	BLOWFISH	SERPENT	TWOFIS-H	RC6
Number of Slice Registers out of 2443200	794	17145	6947	27054
Number of Slice LUTs out of 1221600	801	89034	31968	46600

The Macro-statistics and slice logic utilization shown in table II and III gives an overall idea about the different elements that are required when synthesized onto a FPGA. The RC6 algorithm uses a complex operation such as the multiplier, which contributes to large delay and increase in area and computation time. As the number of registers depend on the code where variables are used to store signals and value, therefore it might vary hence it is not considered for comparison of the architectural aspects of the algorithm. Without the count of registers RC6 is still found to occupy larger area as seen from table 2 and 3. Even though serpent algorithm has it strength in terms of security it comes next to RC6 in terms of area overhead. A cryptographic algorithm is an additional block meant to improve the security of the data processed in the functional block. Hence, there has to be minimum space requirements for cryptographic blocks on FPGA so as to allow us to accommodate larger functional block. Thus Blowfish tends to utilize lowest number of elements with less complex operations as compared to other algorithms. The security of blowfish is also good with no known practical cryptanalytic attacks. But a point to keep in mind is the data processed by the blowfish algorithm is only 64 bits and 16 rounds thus resulting in lower area. Twofish algorithms area requirement are mainly contributed by the pre-processed data.

IV. CONCLUSION

In this paper, blowfish, serpent, Twofish and RC6 encryption architecture are studied. These algorithms are simulated in Xilinx IDE and synthesized in Virtex 7 family FPGA, the results have been tabulated. The statistics of timing and slice utilization results help us in understanding the requirements of the algorithm and provide an idea in choosing suitable block cipher architecture for applications of choice for a given constraint. From the study Blowfish uses the lowest area but there it has a large delay whereas twofish has lesser delay and a nominal resource usage. Serpent and RC6 utilize large number of LUTs, but RC6 uses complex mathematical operations which may tend to increase the overall delay.

V. REFERENCES

[1] William Stallings, “Cryptography and Network Security: Principles and Practice”, 2nd edition, Prentice-Hall, Inc., 1999 pp 23-50.

[2] Bruce Schneier ,”Description of a new variable-length key, 64-bit block cipher (Blowfish)”,Part of the Lecture Notes in Computer Science book series (LNCS, volume 809),2005.

[3] E Biham, R Anderson, L Knudsen ,”Serpent: A new block cipher proposal”,Fast Software Encryption, 1998, Springer

[4] R.L. Rivest, M.J.B. Robshaw, R. Sidney, Y.L. Yin, “The RC6 block cipher,” Proc. of the 1st AES candidate conference, CD-1: Documentation, August 20– 22, 1998, Ventura.

[5] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, Niels Ferguson “On the

Twofish Key Schedule", Lecture Notes in Computer Science book series (LNCS, volume 1556).

[6] Joan Daemen, Vincent Rijmen,"The Design of Rijndael AES — The Advanced Encryption Standard", Springer-Verlag, November 26, 2001.