FPGA Implementation of Pipelined Architecture for RC5

Rakesh Bhimannavar
Department of ECE, PESIT
Bangalore, Karnataka
India

Jayashree HV
Asst Professor, Department of ECE, PESIT
Bangalore, Karnataka
India

ABSTRACT
Transferring the data more securely plays a vital role in today's communication world. Ensuring that the data transferred is secure is a challenge. It is necessary to make sure that information is hidden from anyone for whom it is not intended. Cryptographic algorithms are extensively used to assure the privacy of the data by providing required security through encryption. Encryption is carried out by performing multiple iterations of complex mathematical operations on the given data using a secret key, such that original data is hard to retrieve unless the secret is known. This paper describes the RC5 algorithm, which is one of the methods to provide security to the data. It involves the use of encryption and decryption processes. The main goal of this paper is to show that the exchange of information is secure in a very strong sense. In this paper a reconfigurable RC5 algorithm for crypto system is implemented in Xilinx Spartan - 6 FPGA with a pipeline scheme. The design has been described in VHDL. This architecture reduces the required hardware resources compared to previous works [9] and achieves high-speed performance.

General Terms
Cryptography, RC5, pipelined architecture, FPGA.

1. INTRODUCTION
Communication plays a vital role in our life. There has been a rapid growth in communication systems in last few years, and also the use of internet in our daily activities has increased tremendously. The rapid developments in communication systems and the growth of the internet have resulted in the need for effective security and reliability of data communication, processing and storage. Today, it is important that information is sent confidentially over the network without fear of hackers or unauthorized access to it. Security of the data is important for applications like finance, defence services, wireless sensor networks, etc. that need privacy. Loss of data and hacking of data in these fields affect the organizations to a great extent.

Cryptography is one of the technological methods to provide security to data being transmitted on information and communications systems. Cryptography uses mathematical operations to encrypt and decrypt the data. Cryptography provides services such as authentication, confidentiality or secrecy, integrity, non-repudiation, service reliability and availability of the system. Encryption and decryption processes are carried out by performing multiple iterations of complex mathematical operations on the given data using a secret key, such that original data is hard to retrieve unless the secret key is known. The other side of encryption process is that the multiple iterations of complex mathematical operations require extra power, time and resources. The time overhead of the encryption process increases the response time of the overall system. Hence there is a need of improvements in the performance of existing encryption algorithms. In this paper it is shown that the exchange of information is secure in a very strong sense. This paper describes the pipelined architecture of RC5 to reduce the product of area and delay. Hardware Description Language is used to model the proposed architecture.

2. RELATED WORKS
Many authors have worked on different architectures of RC5 which are designed based on FPGA. Each of these architectures is used for variety of applications. In 2003, N. Sklavos, C. Machas and 0. Koufopavlou proposed an area optimized architecture for RC5 encryption with throughput nearly 2.1 Gb/sec and in 2004, they have implemented RC5 architecture for WAP (Wireless Application Protocol) based applications using alternative adder and subtracter designs to achieve high performance and low area resources. In 2008, Omar Elkeelany introduced pipeline technique to increase the encryption throughput. In this work, reconfigurable RC5 encryption and decryption is introduced using pipelined architecture to optimize the area and increase the throughput.

3. RC5 CIPHER
RC5 is a parameterized symmetric block cipher with a variable block size (w), a variable key size (b) and a variable number of rounds (r). RC5 cipher was designed by Professor Ronald L. Rivest of MIT (Massachusetts Institute of Technology).

The RC5 allows a range of parameter values so that they can be configured according to the requirements of the users, while providing an evolutionary path for adjusting their parameters as necessary in the future, which is one of the outstanding features of RC5. The RC5 cipher works in combination with a key to encrypt the plain text and decrypt the cipher text. The three parameters of RC5 are summarized below:

- **Word size (w)**
  The size of the word (w) in bits refers to a string of bits of a particular length. RC5 cipher accepts 16, 32, or 64 bits word sizes; the block size is twice the word size.
- **Rounds (r)**
  The number of rounds (r) is the number of times the operation employs the inner encryption function. The number of rounds allows trade-offs between speed and security. The greater the number of rounds, the greater the security, but slower is the execution. The number of rounds can range from 0 (zero) to 255.
- **Key Size (b)**
  The secret key is provided by the sender of the data to protect it from unauthorized access. The secret key can range from 0 bits to 2040 bits in size. For most applications, an 80 to 128-bit key is sufficient. This secret

key is used to generate an expanded key table during the initialization phase. The key table is then used during RC5 encryption or decryption process.

An important feature of RC5 is its heavy use of data dependent rotations i.e. the amount of rotation performed is dependent on the input data and is not predetermined. The security of RC5 is provided by the mixture of different operations and the use of data-dependent rotations. The rotations are the non linear operator in RC5.

There are three processes in RC5:

a) Key expansion.
b) Encryption.
c) Decryption.

The following three operations (and their inverses) are used to perform the above processes.

a) Modulo 2^w Addition of words, denoted by "+".
b) Bit-wise XOR of words.
c) Circular rotation.

3.1 Key expansion of RC5

The user-provided secret key is expanded in the key-expansion process, to fill a key table whose size depends on the number of rounds. This key table is then used in both encryption and decryption processes. Two magic constants, Pw and Qw are used in key expansion algorithm.

3.1.1 Magic constants

Two word length constants Pw and Qw are used in the key-expansion algorithm. They are defined for arbitrary 'w' as follows:

P_w = Odd [(e-2)2^w]
Q_w =Odd [(Ø-1)2^w]
where e = 2.718281828459 (base of natural logarithms)
Ø =1.618033988749 (golden ratio).

The constants P_w and Q_w for word size w = 16, 32 and 64 are given below in binary and in hexadecimal.

P16 = 1011011111100001 = b7e1
Q16= 1001111000110111 = 9e37
P32= 10110111111000010101000101100011 = b7e15163
Q32= 10011110001101110111100110111001 = 9e3779b9
P64=1011011111100001010100010110001010000101011101
1010010101001101011=b7e151628aed2a6b
Q64=1001111000110111011110011011100101111111101001
0100111110000010101=9e3779b97f4a7c15

3.1.2 Array S initialization

Using the magic constants Pw and Qw, the array *S* is initialized to a particular fixed (key-independent) pseudo-random bit pattern. This is the first algorithmic step of key expansion process. The size of table S is t = 2(r+1), where r is the number of rounds. In the following code, array *S* is initialized by *Pw* and then all of the entries of the array are found by adding *Qw* to the previous entry.

```
S[0] = Pw;
for i = 1 to t-1 do
    S[i] = S[i-1] + Qw;
```

3.1.3 Converting the secret key from bytes to words

The second algorithmic step of key expansion is to convert the b-byte key, K [0.....b-1] into a c-word array L [0....c-1], where *c* is *b/u*, *u* is given by *w/8* (number of bytes/word). This conversion is performed by zeroing out the array L and copying the string K directly into the memory positions represented by L.

3.1.4 Mixing in the secret key

To produce a final array S of sub keys, the initialized array S is mixed with the key array L. For this purpose, we use the following pseudo code.

```
i = j = X = Y = 0
do 3* max (t , c) times:
    S[ i ] = ( S[ i ] + X + Y) <<< 3;
    X = S[ i ];
    i = (i+1) mod (t);
    L[ j ] = (L[ j ] + X + Y) <<< ( X+Y );
    Y = L[ j ];
    j = (j+1) mod (c);
```

3.2 Encryption of RC5

Encryption is the process of converting plain text to a cryptic text to secure it against data thieves. The encryption process takes a plaintext as input and produces a cipher-text as the output. The key-expansion process must have already been performed before this process. In general, the same plaintext block will always encrypt to the same cipher-text when using the same key in a block cipher. The block diagram of RC5 encryption is shown in figure 1.

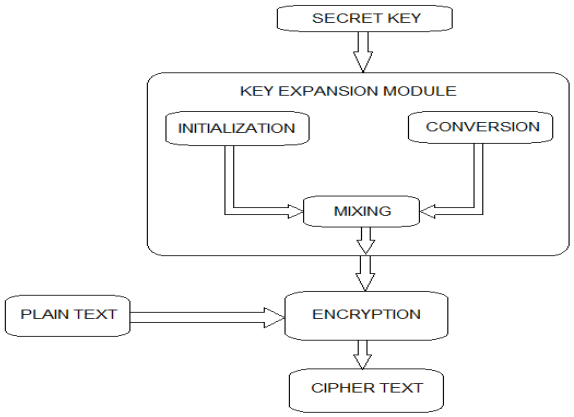

Fig 1: RC5 Encryption block diagram

The RC5 encryption algorithm is defined by the following pseudo code

```
LE0 = A + S [ 0 ]
RE0 = B + S [ 1 ]
For i= 1 to r do
    LEi = ((LEi-1 ⊕ REi-1) <<< REi-1) + S[2 * i];
    REi = ((REi-1 ⊕ LEi) <<< LEi) + S[2 * i + 1];
```

The resulting cipher text is available at the output of the r^{th} round contained in the two variables LEr and REr.

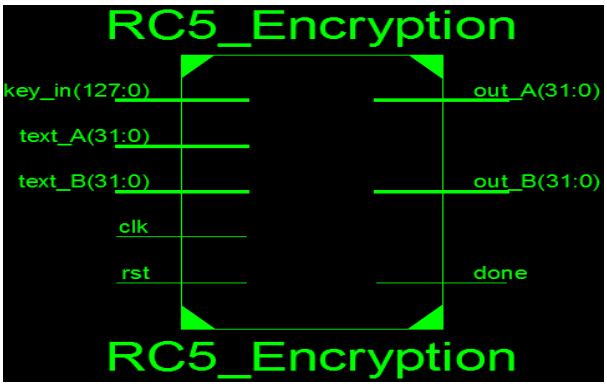

Fig 2: RTL schematic of RC5 encryption

The state machine for the proposed pipeline architecture of RC5 encryption is shown in figure 3.

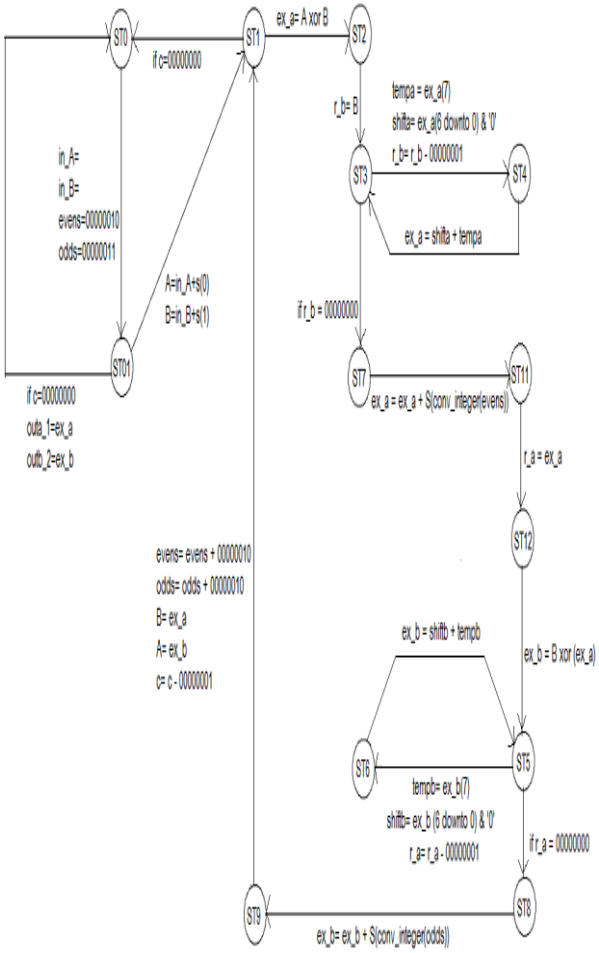

Fig 3: RC5 encryption state machine

3.3 Decryption of RC5

The decryption process takes a cipher-text as the input and produces a plaintext as the output. The decryption routine is easily derived from the encryption routine. In this routine the inverse formulation of encryption routine is processed. The decryption algorithm is defined by the following pseudo code.

```
for i = r downto 1 do
    RDi-1= ((RDi – S[2*i+1] >>> LDi) ⊕ LDi);
    LDi-1 = ((LDi -  S[2*i]>>>RDi-1) ⊕ RDi-1);
B= RD0 – S[1];
A= LD0 – S[0];
```

4. PIPELINED ARCHITECTURE

To improve the performance and to increase throughput, the RC5 encryption and decryption processes can be implemented in pipeline technique. Pipelining is an implementation technique in which numerous instructions are overlapped in execution, where different data will be executed parallel.

For a data to be encrypted, it should undergo multiple iterations of different complex arithmetic operations. The data is manipulated in each round at several stages using arithmetic and logical operations. The execution of each round depends on the intermediate output of previous round. The dependency of operation of each round on the output of previous round makes it a data dependent structure. The security of RC5 is provided by the heavy use of data-dependent rotations and the mixture of different operations. In a pipeline model of RC5 as shown in figure 4, there are 'n' rounds to be performed and operation of each round depends on the output of previous round. Finally the cipher text is taken from final unit after all rounds are completed. The resulting cipher text is available at the output of the n^{th} round.

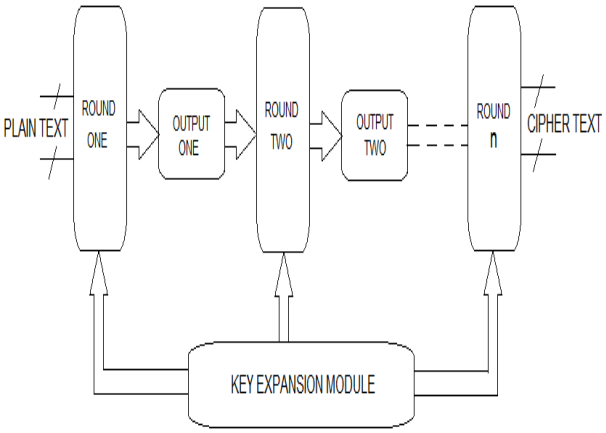

Fig 4: Pipeline model

5. RESULTS

The proposed pipeline model for RC5 is implemented in Spartan-6 FPGA. The simulation results of RC5 encryption and decryption processes for 128 bit secret key with 15 rounds are shown in figure 5 and 6.

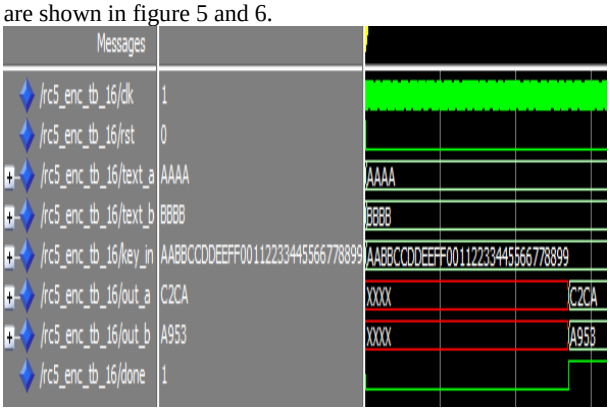

Fig 5: RC5 encryption simulation

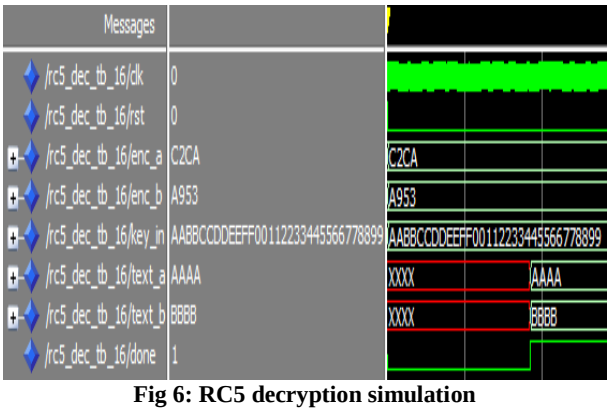

Fig 6: RC5 decryption simulation

Figure 7 shows the RC5 decryption simulation with wrong key which yields wrong decrypted data. This shows that the exchange of information is secure in a very strong sense. For comparisons, the proposed architecture is synthesized to FPGA device similar to the family of related work for 64-bit data, 16-byte key, with 15 rounds encryption.

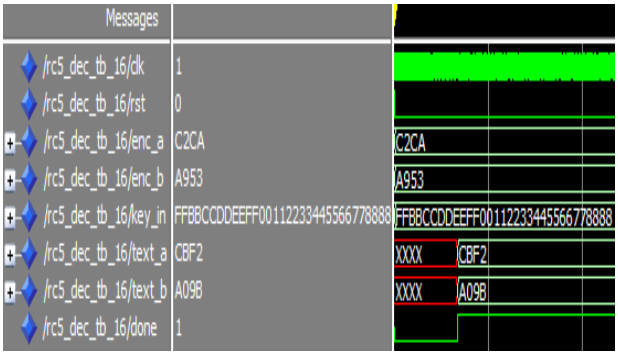

Fig 7: RC5 decryption simulation with wrong key

The synthesized summary for encryption is shown in table 1 for different architectures.

Table 1. Synthesized summary

Architecture	Slices	Delay (nsec)	Area * Delay	Frequency (MHz)
Our work	1492	4.882	7284	204.842
Ruhan Bevi, S.S.V. Sheshu, S. Malarvizhi [9]	1698	8.62	14636	116
M.Yoshikawa [7]	2488	9.92	24680	100
Elkeelany [5]	3618	66.303	239884	24
N.Sklavos [2]	998	N/A	N/A	71

This pipelined architecture for encryption resulted in reduced device utilization with improved response time. The product of area and delay achieves a reduction of nearly 50% when compared with the reference [9] RC5 design. Figure 8 shows the graph of area delay product for different architectures. Figure 9 shows schematic for RC5 encryption.

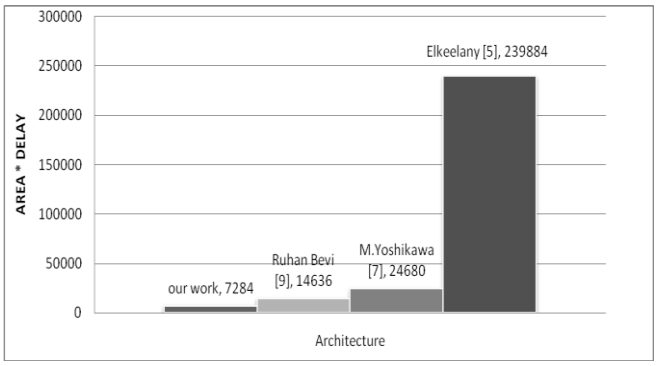

Fig 8: Area * delay comparison

6. CONCLUSION

RC5 encryption and decryption algorithm for cryptosystem is tested and implemented successfully. The code for RC5 encryption and decryption algorithm is written in VHDL. The design was tested for 32 bit and 64 bit input data, 15 rounds with 128 bit secret key. The code is synthesizable and is implemented in SPARTAN-6 FPGA board. This pipelined architecture for RC5 improves the response time of the system with less resource compared to previous architectures [9] [7] [5] and achieves a maximum encryption rate of 6.5 Gbps.

The design has been implemented for stepwise execution of RC5 encryption and decryption, which takes comparatively more time. The same design can be implemented using recursive iteration for high speed execution.

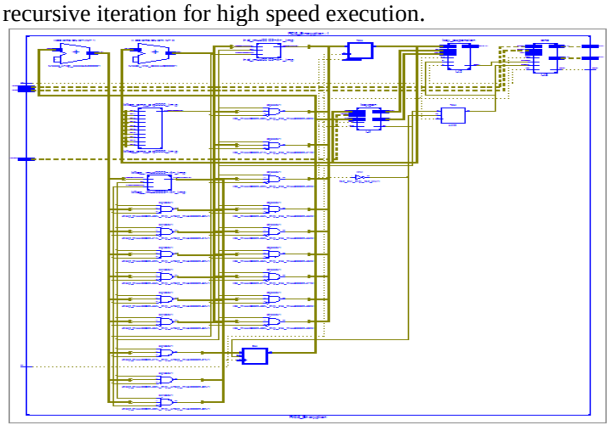

Fig 9: Schematic for encryption

7. REFERENCES

[1] R. L. Rivest. "The RC5 Encryption Algorithm," in Proc. of the 2nd Workshop on Fast Software Encryption, pages 86-96, Springer, 1995.

[2] N. Sklavos, C. Machas, O. Koufopavlou, "Area Optimized Architecture and VLSI Implementation of RC5 Encryption Algorithm," in Proc. of 10th IEEE International Conference on Electronics, Circuits and Systems (IEEE ICECS'03), vol. 1, pp.172-175, United Arab Emirates, Dec. 2003.

[3] S. Nimmagadda, O. Elkeelany, "Performance evaluation of different hardware models of RC5 algorithm," In the Proceeding of The IEEE 39th Southeastern Symposium on System Theory, 2007.

[4] N.Sklavos, A.P. Fournaris, O. Koufopavlou, "WAP Security: Implementation Cost and Performance Evaluation of a Scalable Architecture for RC5 Parameterized Block Cipher," in Proc. of 12th IEEE Mediterranean, vol. 2, pp. 795-798, May. 2004.

[5] O. Elkeelany, "Performance Comparisons, Design, and Implementation of RC5 Symmetric Encryption Core using Reconfigurable Hardware," Journal of Computers, vol. 3, no.3, pp. 48-55, Mar. 2008.

[6] L. Hua, L. Jianzhou and Y.Jing, "An efficient and reconfigurable architecture for RC5," Canadian Conference on Electrical and Computer Engineering, pp. 1648-1651, May 2005.

[7] M. Yoshikawa, K. Sakaue, "Dedicated Hardware for RC5 Cryptography and its Implementation," World Congress in Computer Science, Computer Engineering, and Applied Computing (WORLDCOM'11), Las Vegas Nevada, USA, Jul. 2011.

[8] W. Stallings, Cryptography and Network Security: Principles and Practice, Third Edition. New Jersey: Prentice hall, 2003.

[9] Ruhan Bevi, S.S.V. Sheshu, S. Malarvizhi, "FPGA based Pipelined Architecture for RC5 Encryption", IEEE, Digital information and communication technology and its application (DICTAP), pages 214-219, MAY. 2012.