ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Design Of High Performance Rc4 Stream Cipher For Secured Communication

R.Prabu[1]

ME-VLSI Design, Shreenivasa Engineering College, B.Pallipatti, Dharmapuri, Tamilnadu, India[1]

Abstract: The main feature of cryptography is to work out with problems, which are associated with secrecy, authentication and integrity. Cryptography also, is related with the meaning of protocol. A protocol is sequences of actions, which are concern two or more sides, designed to fulfill a goal. Thus, a cryptographic protocol is a protocol that uses cryptography. This protocol uses a cryptographic algorithm and its intention is to prevent attempts of thefts and invasions.RC4 is the most popular stream cipher in the domain of cryptology, In this project we propose the fastest known architecture for the cipher with the help of loop unrolling and hardware pipeline which is produce the one RC4 key stream byte per clock cycle. We have optimized and implemented our proposed design using VHDL description Synthesized with 65nm fabrication technology at clock frequency of 1.37 GHz to be obtained a final key stream throughput 30.72Gbps.

Key Words: Cryptography, hardware accelerator, high throughput, loop unrolling, pipelining, RC4, stream cipher

I. INTRODUCTION

RC4 encryption is the encryption used in most software applications today (to include HTTP and SSL), which makes it arguably the most commonly used encryption method in the world. No wonder then, when the architects of wireless network security sat down to design their protocol in 1997, they incorporated RC4 into their design; in the form of WEP (Wired Equivalent Privacy).

Wireless network traffic will always have an inherent liability of interception. It is very difficult to bind where wireless communication radiates. An office that has a wireless infrastructure most likely "spills" its network traffic out into the street, the parking lot, and neighboring buildings. This liability necessitates some form of securing wireless traffic while it is traveling in the air. WEP is the first major attempt to do this.

The design concept (and also the origin of its name) of WEP was to provide the equivalent security of having a wire connecting each wireless node. It doesn't promise to "secure" the network from all forms of attack, it just attempts to secure the traffic from node to node and mitigate the interception liability. Unfortunately, the attempt failed.

Despite adopting the most widely used and trusted encryption package on the market, the implementation of RC4 in WEP is tragically flawed. RC4was invented in 1987 by Ron Rivest, the founder of RSA Security, and details on its implementation remained a trade secret until 1994 when an unauthorized detailed-design of the RC4 Key-Scheduling Algorithm (KSA) and Pseudo-Random Generation Algorithm (PRGA) was anonymously posted on the internet.

STREAM Ciphers are broadly classified into two parts depending on the platform most suited to their implementation, namely software stream ciphers and hardware stream ciphers. RC4 is one of the widely used stream ciphers that is mostly implemented in software. Efficiency in terms of "key stream throughput" has always been a benchmarking parameter for stream ciphers. The efficiency of the RC4 obviously depends on the efficiency of KSA and PRGA. While the KSA invokes a fixed cost for generating the initial pseudorandom states, the PRGA incurs a variable cost in terms of the number of key stream bytes to be generated. An efficient implementation of RC4 would aim to minimize the cost for per round of KSA and PRGA to provide better throughput. RC4 has two components, namely the Key Scheduling Algorithm (KSA) and the Pseudo-Random Generation Algorithm (PRGA). The KSA uses the key K to generate a pseudorandom permutation S of (0, 1, N – 1) and PRGA use this pseudorandom permutation to generate arbitrary number of pseudorandom key stream bytes.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

In this paper, focuses on efficient hardware implementation of the cipher. The main motivation is to test the limits to which RC4, the popular "software" stream cipher, can compete in hardware performance with the state-of-the-art hardware stream ciphers. If it can, we will have a stronger case in support of this time-tested cipher. Our proposed design is implementing the RC4 architecture that produces1 byte per cycle. The implementation of the design has been done using VHDL description.

II. EXISTING SYSTEM

In this system an efficient hardware implementation of the RC4 stream-cipher is designed, which support only fixed length key, the existing implementation integrates in the same hardware module an 8-bit up to 128-bit key length capability. Independently of the key length, this VLSI implementation achieves a data throughput up to 22 MBytes/sec in a maximum frequency of 64 MHz. The whole design was captured by using VHDL language and a FPGA device was used for the hardware implementation of the architecture.

2.1 SYSTEM ARCHITECTURE

The following Fig.1shows the block diagram of the existing architecture. It consists of a control and a storage unit. The storage unit is responsible for the key setup and keystream generation.

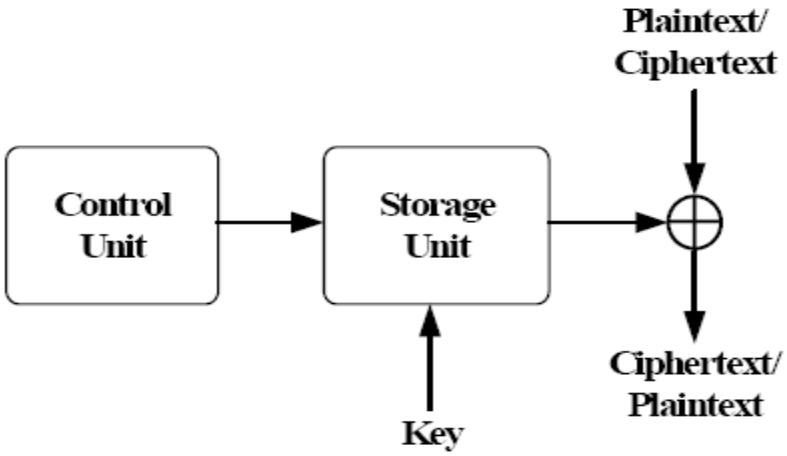


Fig 1: Existing system architecture

2.2 STORAGE UNIT IMPLEMENTATION

Storage unit is used to generate the keystream; the implementation of the storage unit is shown in Fig.2. The storage unit contains memory elements for the S-Box and K-Box, along with 8-bit registers, adders and one multiplexer.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

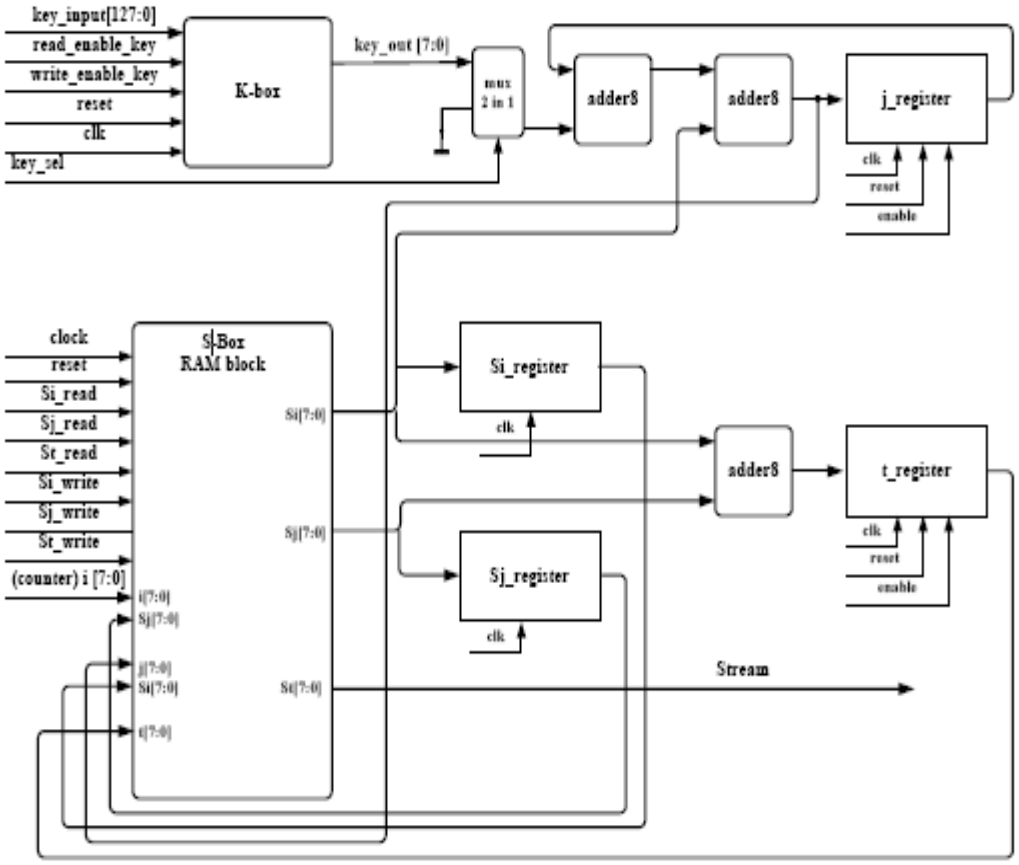


Fig 2: Storage unit

The block diagram of the S-box RAM is shown Fig It consists of three 256 bytes RAM blocks. Each RAM block has four inputs and one output. The two inputs are the read and write signals while the other two are the address and data signals. Also, all the three RAM boxes have the same signals of clock and reset. The operation of RAM blocks is quite simple. If the reset signal occurs the blocks are initialized linearly. For each block, if the write signal occurs new data are stored in the address position, or if the read signal occurs the data in the address position are available on the output of the block of step the S-box is filled. The S-Box is initialized linearly, such as S_0=0, S_1=1, S_2=2,…,S_{255}=255 when the reset state occurs.

In the second step of the key setup, the S-Box is randomizely filled. For the S-box, 3x256-bytes RAM memory is used as it's shown in Fig. The Si and Sj registers are used for the necessary by the algorithm swapping's. The jand tregisters are used in order to temporary store all the intermediate variables that are produced.

After the completion each byte in the keystream can be generated and used for encryption/decryption. The encryption/decryption is achieved by the bitwise XORing of the keystream with the plaintext/ciphertext. The total duration time of this phase is 3*n (n is the number of bytes of the plaintext or ciphertext) and the total duration time of the whole procedure of the two phases is 768 + 3*n. The operation of the storage unit is synchronized by the control unit. The control unit is responsible for the generation of the clock and control signals.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

DISADVANTAGES

- Low throughput.
- This implementation restricts the number of port accesses per cycle.
- It providesan output N bytes in 3N+768 clock cycles this implies existing design requires three cycles to produce one byte data.
- Long latency because of using a RAM based storage unit.
- Encrypts/decrypts one byte at every seven clock cycles and of course increases the total latency of the algorithm execution.

III. PROPOSED SYSTEM

I propose an RC4 architecture that produces 1 byte per cycle. The main contribution of our work is to take a new look at RC4 hardware design and introduce the idea of loop unrolling in this context. I combine consecutive pairs of cycles in a pipelined fashion, and read off the values of one state of the S-box from previous or later rounds of the cipher. To the best of our knowledge, the idea of loop unrolling in RC4 has never been exploited in designing an efficient hardware. I present a novel RC4 hardware that provides a keystream throughput of 1 byte-per cycle using the idea of loop unrolling and hardware pipelining.

ADVANTAGES

- The obvious advantage of this design is its compactness.
- It provides better throughput without resorting to loop unrolling.
- Time and area constraints are better.
- It achieves the claimed throughput by means of hardware pipeline, and instead of two iterations per cycle, one iteration per cycle in PRGA is performed.
- In terms of throughput, this design provides 1-cycle-per-byte output in PRGA and completes KSA in 256 cycles, with an initial lag of three cycles due to the 4-stage pipeline structure. Thus, N bytes of output is produced in approximately N+259 clock cycles, which is comparable to the performance of our Design.

3.1 CALCULATION OF i1 AND i2

Incrementing i0 by 1 and 2 can be done by the same clock pulse applied to two synchronous 8-bit counters. The counter for i1 is initially loaded with 00000001 and the counter for i2 is loaded with 00000010, the initial states of these two indices. This serves the purpose for the first two rounds of RC4, in both KSA and PRGA. Thereafter, in every other cycle, the clock pulse is applied to all the flip-flops except the ones at the LSB position for both the counters, as shown in Fig.3. This will result in proper increments of i1 that assumes only the odd values 1,3, 5, . . . , and that of i2 assuming only the even values 2,4,6, . . . , as required in RC4. This is assured as the LSB of i1 will always be 1 and that of i2 will always be 0, as shown in Fig.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

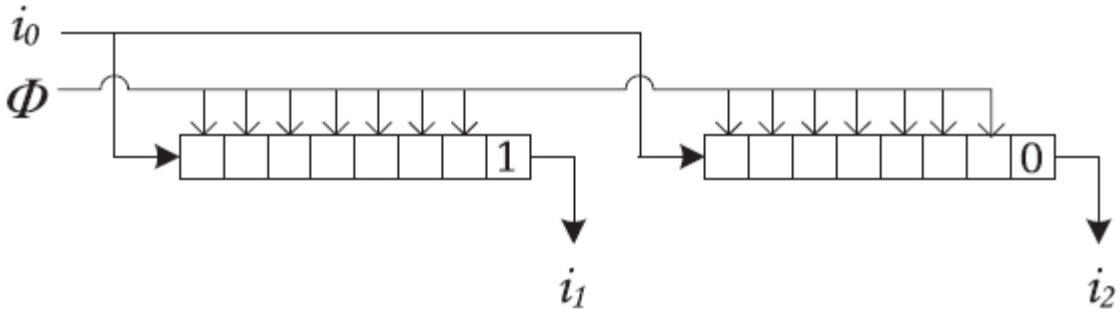


Fig 3: Circuit to compute i1 and i2

3.2 CALCULATION OF j1 AND j2

The values of j1 and j2 will be computed and stored in two 8-bit registers. To compute j1, we need a 2-input parallel adder unit. It may be one using a carry lookahead adder, or one using scan operation as proposed by Sinha and Srimani.or one using carry-lookahead-tree as proposed by Lynch and Swarzlander, Jr. For computing j2, there are two special cases:

$$j_2 = j_0 + S_0[i_1] + S_1[i_2] = \begin{cases} j_0 + S_0[i_1] + S_0[i_2] \text{ if } i_2 \neq j_1, \\ j_0 + S_0[i_1] + S_0[i_1] \text{ if } i_2 = j_1. \end{cases}$$

The only change from S0 to S1 is the swap S0[i1] <-> S0[j1],and hence we need to check if i2 is equal to either of i1 or j1. Now, i2 cannot be equal to i1 as they differ only by 1 modulo 256. Therefore, S1[i2] , S1[j1],S0[i1] if i2 , j1, and S1[i2],S0[i2] otherwise. In both the cases, three binary numbers are to be added. Let us denote the kth bit of j0,S0[i1] and S1[i2] by ak,bk, and ck respectively. We first construct two 9-bit vectors R and C, where the k bits of R and C are given by

$$R_k = \mathrm{XOR}(a_k, b_k, c_k) \text{ for } 0 \leq k \leq 7,\ R_8 = 0,\ C_0 = 0,$$

$$C_k = a_{k-1}b_{k-1} + b_{k-1}c_{k-1} + c_{k-1}a_{k-1} \text{ for } 1 \leq k \leq 8.$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

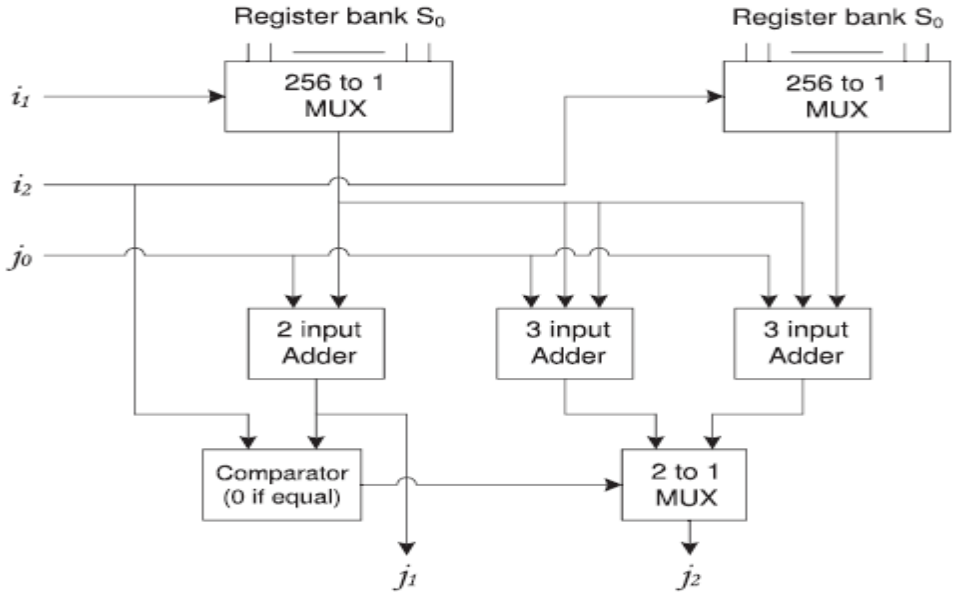


Fig 4: Circuit to compute j1 and j2.

In RC4, all additions are done at modulo 256. Hence, we candiscard the 9th bit (k = 8) of the vectors R, Cwhile adding them together, and carry out normal 8-bit parallel addition. Therefore, one may add R and C by a parallel full adder as used for j1. The circuit to compute j1 and j2 is as shown in Fig.4.

3.3 THE COMPLETE CIRCUIT

The PRGA architecture of Design , as shown in Fig.5, produces 2N bytes of output stream in N iterations, over 2N clock cycles. Note that the initial clock pulse Φ0 is an extra one, and the production of the output bytes lag the feedback cycle by one clock pulse in every iteration Therefore, our model practically produces 2N output bytes in 2N clock cycles, that is "one byte per clock," after an initial lag of two clock cycles.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

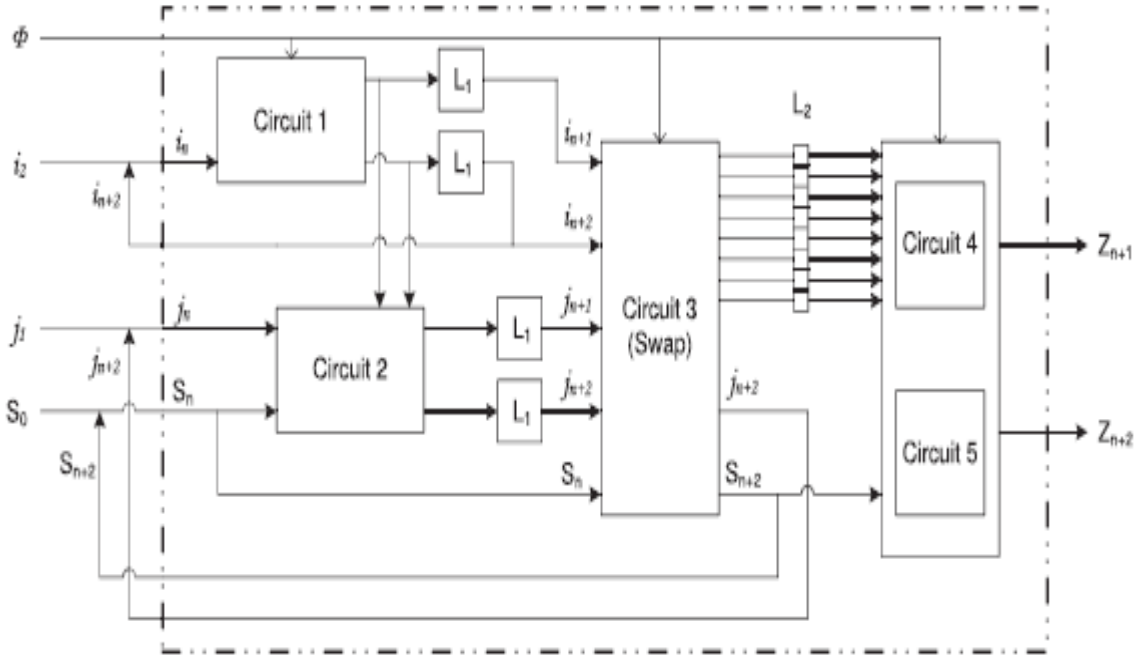
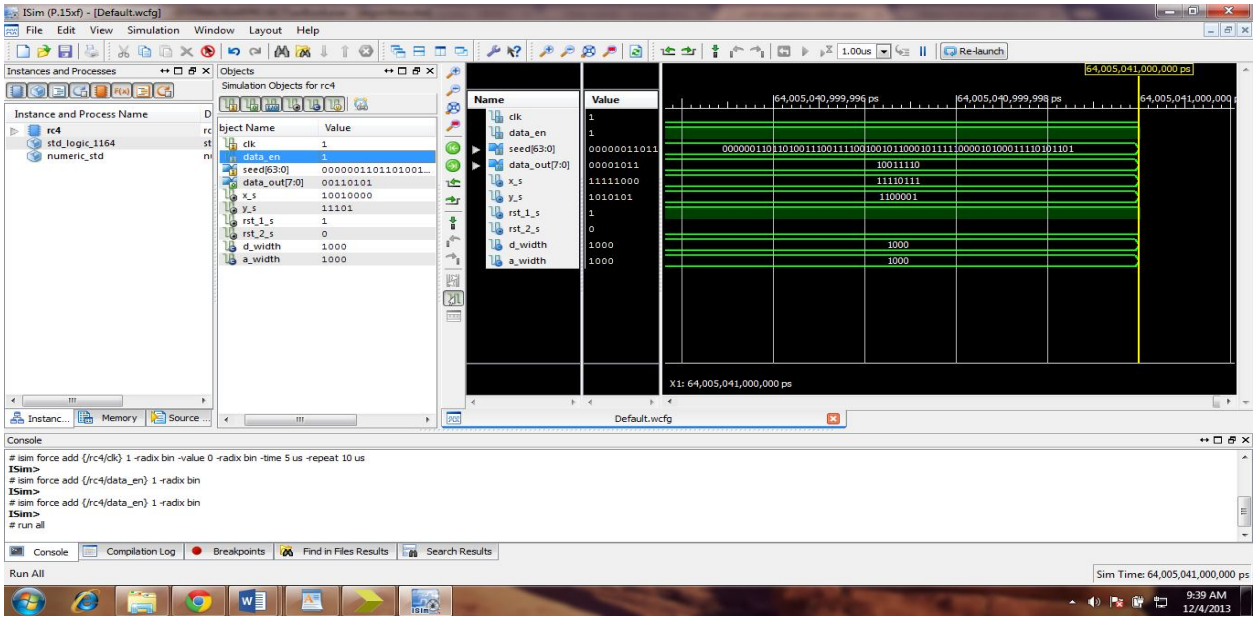


Fig 5: Circuit for PRGA stage of the proposed RC4 architecture

SIMULATION RESULT



ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

IV. CONCLUSION

The present invention provides a system and method for encrypting and decrypting the files using a fast hardware implementation of the RC4 algorithm to enable secure access to information resources in a computer network. Here we analysis the various design techniques of RC4 stream cipher, for fast operation the proposed design will give one RC4 key stream byte per clock cycle. In this paper, we have presented a thorough study of RC4 hardware designing problem from the point of view of throughput, measured in gigabits per second output of the RC4 keystream. We have discussed the issues of loop unrolling and hardware pipelining in RC4 architecture to obtain better throughput, and have experimented extensively to optimize area and performance. Our design tops the existing by combining the idea of loop unrolling with that of efficient hardware pipelining to obtain two keystream bytes per cycle. This is the fastest known RC4 hardware architecture till date, providing a 30.72 Gbps in its optimized form, using 65 nm technology.

REFERENCES

[1] IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 4, "High-Performance Hardware Implementation for RC4 Stream Cipher" APRIL 2013.

[2] Software Performance Results from the eSTREAM Project,eSTREAM, the ECRYPT Stream Cipher Project, http://www. ecrypt.eu.org/stream/perf/#results, 2012.

[3] T. Good and M. Benaissa, "Hardware Results for Selected Stream Cipher Candidates," eSTREAM, ECRYPT Stream Cipher Project, SASC, Report 2007/023, 2007.

[4] I. Mantin, "Predicting and Distinguishing Attacks on RC4 KeystreamGenerator," Proc. 24th Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT '05), vol. 3494, pp. 491-506, 2005.

[5] P. Kitsos, G. Kostopoulos, N. Sklavos, and O. Koufopavlou,"Hardware Implementation of the RC4 Stream Cipher," Proc. IEEE 46th Midwest Symp. Circuits and Systems, http:// dsmc.eap.gr/en/members/pkitsos/papers/Kitsos_c14.pdf, 2003.

[6] I. Mironov, "(Not So) Random Shuffles of RC4," Proc. 22nd Ann.Int'l Cryptology Conf. Advances in Cryptology, pp. 304-319, 2002.

[7] P. Hamalainen, M. Hannikainen, T. Hamalainen, and J. Saarinen,"Hardware Implementation of the Improved WEP and RC4 Encryption Algorithms for Wireless Terminals," Proc. European Signal Processing Conf., pp. 2289-2292, 2000.

[8] M.D. Galanis, P. Kitsos, G. Kostopoulos, N. Sklavos, and C.E. Goutis, "Comparison of the Hardware Implementation of Stream Ciphers".