# Area Efficient Implementation Of 128-Bit Block Cipher, SEED

Young-Ho Seo, Jong-Hyeon Kim, Yong-Jin Jung and Dong-Wook Kim
Dept. of Electronic Materials Engineering, Kwangwoon University
447-1 Wolgye-Dong Nowon-Gu Seoul 139-050, Korea
Tel:+82-02-940-5167, Fax:+82-02-919-3940
E-mail:axl@explore.kwangwoon.ac.kr

**Abstract:** This paper presented a FPGA design of SEED, which is the Korea standard 128-bit block cipher. In this work, SEED was designed technology-independently for other applications such as ASIC or core-based designs. Hence in case of changing the target of design, it is not necessary to modify design or need only minor modification to reuse the design. Since SEED algorithm requires a lot of hardware resources, each unit was designed only once and used sequentially. So, the number of gates was minimized and SEED algorithm was fitted in FPGA without additional components. It was confirmed that the rate of resource usage is about 80% in ALTERA 10KE and the SEED design operates in a clock frequency of 131.57 MHz and an encryption rate of 29 Mbps.

## 1. Introduction

A lot of ciphers such as DES[1], FEAL[2], LOKI[3], SAFER[4], and IDEA[5] have been developed to guarantee confidentiality at electronic commerce using Internet. Also some algorithms were proposed for international standard and have been used in many fields.

Korean government had recognized an importance of cipher systems. So, SEED[6] was developed and proposed for a standard in Korea. In the future SEED with other ciphers will be applied to many commercial applications.

Ciphering algorithms are easily implemented in software. Typically software cryptosystems rapidly manage small size of data. But when many users simultaneously require data access to a cryptosystem, real-time data access is impossible because of system overload. Also if cryptosystems are software, the problem in vulnerability occurs such as trap-door and hacking. Thus cipher algorithm must map into hardware for higher throughput and security.

Since components such as S-box that have a large number of gates are used for each round, it may be impossible that the design fits in FPGA. Therefore in this paper SEED algorithm is designed to have a special structure to reduce the number of gates to successfully map into a FPGA without supplementary factors

## 2. SEED Cipher Algorithm

### 2.1 Global Structure

Korea block cipher SEED has Feistel structure with 16 rounds and the input block size of plain text is 128-bit. Also the block size of cipher key is 128-bit. The input, received 128-bit is separated into two 64-bit blocks. Encryption is performed through round keys extracted by cipher key and F-function. After 16 rounds, the final cipher text is created. The global structure of SEED is shown in Fig. 1.

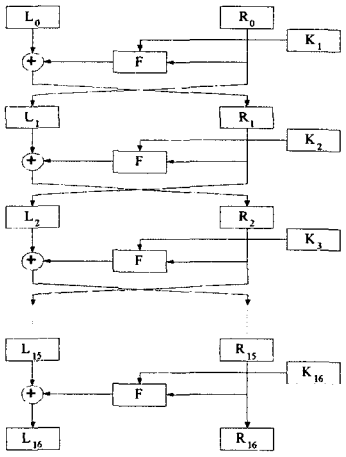


Fig. 1. Global structure of block cipher SEED

### 2.2 F-Function

As shown in Fig. 2, F-function of SEED is 64-bit Feistel structure and is an essential factor which characterizes a block cipher. F-function consists of XORs, G-functions, and Add/Modulo-$2^{32}$ modules.

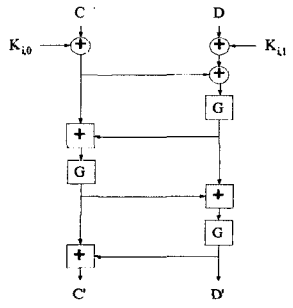


Fig. 2. Structure of F-function

### 2.3 G-Function

G-function is used in both F-function and round key generation. A 32-bit input is separated into four 8-bit blocks. Each separated block passes through S-box and executes bit-wise AND operation with values from $m_0$ to $m_3$. As a result, Input block of 32-bit in G-function is expanded to 128-bit block. Expanded block performs XOR operation with each other to form output block of 32-bit. G-function is shown in Fig. 3.

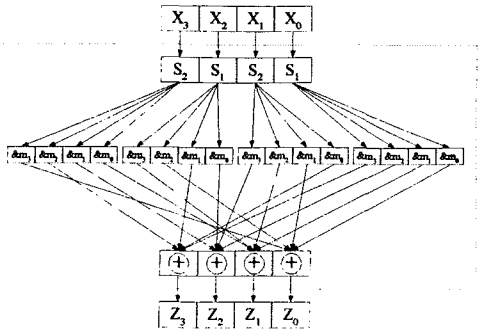

Fig. 3. Structure of G-function

### 2.4 Round Key Generation

128-bit cipher key is separated into left/right two 64-bit blocks and the separated blocks perform rotate-left shift in turn. After add/modulo and G-function operations, the round keys are generated. The process for round key is shown in Fig. 4.

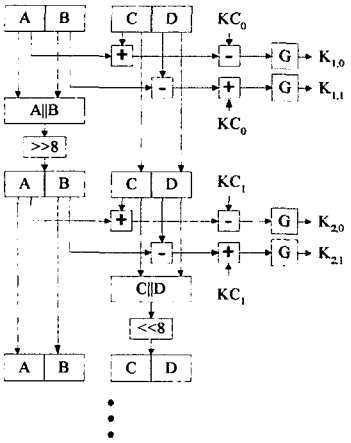

Fig. 4. Structure of Round Key Generation

## 3. Hardware Design

SEED requires large hardware resource due to characteristics of algorithm. During 16 rounds, 16 F-functions, 80 G-functions, and 160 S-boxes are used. So, SEED cipher algorithm must be designed into optimized hardware. All modules were designed at RT-level and were connected at structure-level. Since each module is repeatedly used, multiplexes and registers are used to deal with data feedback.

### 3.1 SEED Global Block Design

As shown in Fig. 5, global block of SEED is composed with Key Generator, F-function, Round Key Register, Controller and Round process part. SEED has block size of 128-bit primary input/output. But in consideration of interface with other systems, primary input/output is designed to 32-bit in block sizes and block size of 64-bit for internal data movement. Each module in data path is controlled by a control block, which consists of three finite state machines. Global operation sequence controlled by controller is depicted in Fig. 6. Round keys are generated during round key generation sequence and the resulting round keys are stored at Round Key Register in F-function block. During cipher text generation, the stored round keys are used in F-function operational sequence. After 16 round operations, 128-bit cipher text is created. Then new round key is generated by new cipher key or the cipher text is generated by next plain text.

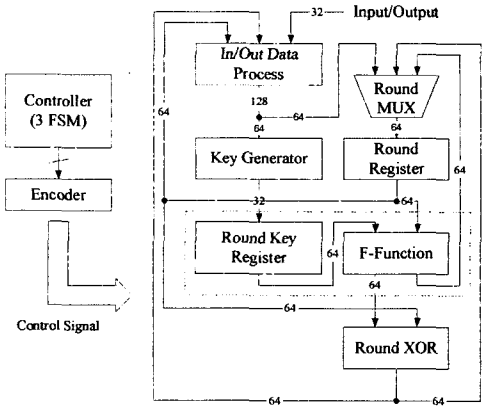

Fig. 5. Global block diagram of SEED

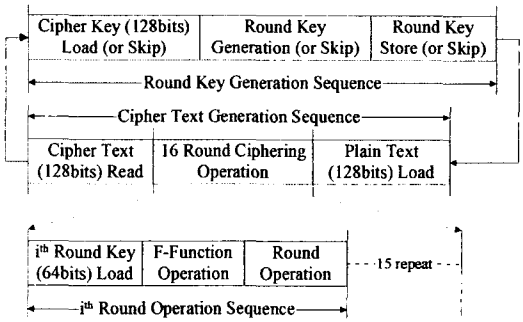

Fig. 6. Hardware operational sequence of SEED

### 3.2 Key Generation Block Design

Fig. 7 illustrates Key Generation block except G-function which is in F-function. Since round key is repeatedly generated by one structure, only one block is designed for one process of round key generation and is sequentially used. When two 32-bit data and the round key constants are added or subtracted, carry save adder and carry-look-ahead adder are used.

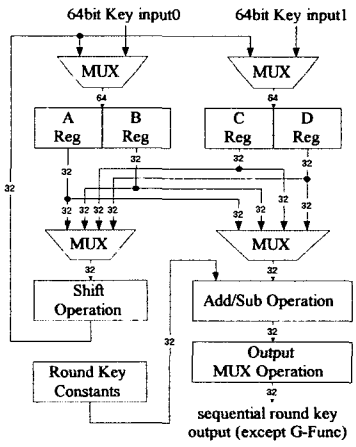

Fig. 7 Structure of Round Key Generation block diagram

### 3.3 F-Function Block Design

F-function consists of G-function, S-box and adder, which occupy large hardware area and are repeatedly used. Hence each unit is designed only once and hardware blocks are connected by means of data feedback structure and each module is sequentially operated. Add/Modulo-$2^{32}$ unit consists of a carry-look-ahead adder. During round key generation, internal G-function in F-function is used and round key is stored in Round Key Register in F-function. F-function block is depicted in Fig. 8.

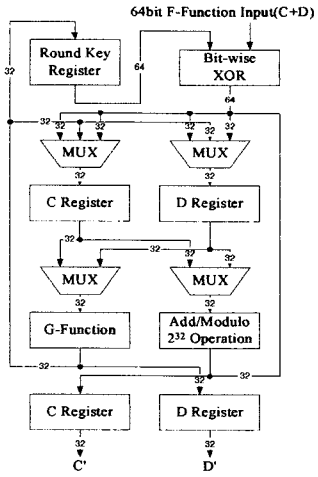


Fig. 8. Structure of F-function block diagram

### 3.4 G-Function Block Design

G-function is composed with register, S-box1/S-box2, Mask and XOR. Register receives input data of 32-bit block and extracts 16-bit data twice. So S-box1 and S-box2 are sequentially used. 16-bit data passed through S-box is expanded into 64-bit in Mask block. Two 64-bit data are reduced to one 32-bit block by Sequential Exclusive-OR operation. The structure of G-function is illustrated in Fig. 9.

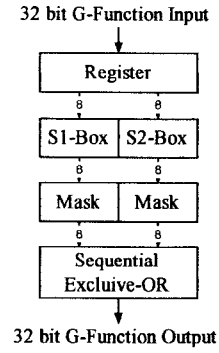


Fig. 9. Structure of G-function block diagram

### 3.5 Controller Design

Control part provides control signals to data path blocks to achieve the operation of SEED algorithm. As shown in Fig. 10, the controller consists of three FSM, one for round, one for Key Generation/ F-function, and one for G-function. The Controller is operated by signals of clock, reset, key_gen, and data_act. And conditions of data path operation are represented by signals of key_ready and data_ready. G-function is operated in both F-function and Key Generation. So the FSM controlling G-function is controlled by the one creating control signals for F-function and Key Generation. The FSM for F-function and Key Generation is controlled by the one for round process. Round FSM decides total operational order. Round FSM activated by key_gen signal drives the FSM for Key Generation(F-function). According to operational sequence, the FSM for Key Generation drives the one for G-function. So round key is generated and is stored at round key register in F-function. Also data_act signal drives the FSM for F-function(Key Generation). The FSM for F-function creates both control signal for F-function block and activation signal of the one for G-function. Then cipher text is obtained and stored output register in I/O Process block.

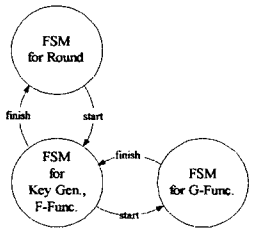


Fig. 10. Structure of Controller FSM

### 3.6 I/O Process Design

The block size of Input/Output of SEED is 128-bit, but it may be modified to interface external systems. Hence Input/Output process block is separately designed as shown in Fig. 11. In this paper, plain text, cipher key and cipher text use 32-bit bi-directional ports. Also I/O Data Process block is designed to change easily into other structures with input and output of 8, 16, 64 or 128-bit.

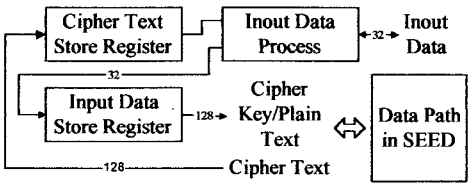


Fig. 11. Structure of I/O Data Process block diagram

## 4. Results

In the paper, SEED was designed and mapped in FPGA without additional hardware and software. All modules are designed using a design methodology with VHDL top-down design to allow technology-independent design and only IEEE standard library is used. Each unit is designed by RTL description and is connected by structure-level description.

Fig. 12 and Fig. 13 show the operation during one whole encryption sequence and control signals, respectively. Fig. 14 and Fig. 15 illustrate 128-bit input of cipher key and plain text respectively. Finally 128-bit block of cipher text is shown in Fig. 16. The result value of cipher text corresponds to the implemented value offered by Korea Information Security Agency.

To verify the design, SEED is synthesized in FPGA

Complier of SYNOPSYS with ALTERA 10K library and simulation is performed in ALTERA MAX+PLUS II. The design occupies 80% chip area of ALTERA device family of FLEX 10KE and used logic cell is about 4300 in FPGA. Key is generated in a clock frequency of 111.11 MHz and a key generation rate of 74 Mbps. The plain text is encrypted in a clock frequency of 131.57 MHz and an encryption rate of 29 Mbps. This encryption rate is about 10 times faster than a software cipher and about 2 times faster than a hardware cipher which was previously implemented in another research.

If the design is synthesized with ASIC library, data is encrypted in rate of about 50Mbps. And if modules which were sequentially used can be designed into parallel structures, performance would be progressed further. In case of inserting one S-box1 and S-box2, encryption rate is 61.9Mbps and the design uses the logic cell of 6100 in FPGA. In case of inserting two additional G-function which has internally parallel structure, encryption rate of 168Mbps is predicted in FPGA implementation.

## 5. Conclusion

In this paper SEED was designed and mapped in FPGA without additional hardware and software. Considering non-specific target, SEED had technology-independent design structure. Since SEED algorithm requires many hardware resources, SEED was implemented into structures with minimal gate size. In consideration of application into other systems, Primary I/O port with 32-bit bi-directional ports was separately designed in this paper. If cipher key is completely protected by physical means such as smart card, SEED which was implemented into hardware has enormous security. And high performance of SEED enables real time access of data. Therefore SEED that is a Korean standard symmetric block cipher can be practically applied in a lot of field, especially core-based design.

## References


[1] National Bureau of Standards. FIPS PUB 46: Data Encryption Standard, January 1997.

[2] Akihiro Shimizu and Shoji Miyaguchi. "Fast data encipherment algorithm FEAL." In David Chaum and Wyn L. Price, editors, Advances in Cryptology-Eurocrypt'87, vol.304 of Lecture Notes in Computer Science, pp.267-280, Springer-Verlag, Berlin, 1998.

[3] Lawewnce Brown, Josef Pieprzyk, and Jennifer Seberry."LOKI-a cryptographic primitive for authentica- tion and secrecy applications". In Jennifer Seberry and osef Pieprzyk, editors, Advances in Cryptology Auscry- pt'90, vol.453 of Lecture Notes in Computer Science, pp. 229-236, Springer-Verlag, Berlin, 1990.229-236, Springer-Verlag, Berlin, 1990.

[4] James L. Massey. SAFER K-64: "A byte-oriented block-ciphering algorithm". In Ross Anderson, editor, Fast Software Encryption, Cambridge Security Workshop, vol.809 of Lecture NOtes in Computer Science, pp.1-17, Springer-Verlag, Berlin, 1994.

[5] Xuejia Lai and James L. Massey. " A proposal for a now block encryption standard. in Ivan bjerre Damgar-d", editor, Advances in Cryptology Eurocrypt'90, vol.473 of Lecture Notes in Computer Science, pp.389-404, Springer-Verla, Berlin, 1990.

[6] Korea Information Security Agency, A Design and Analysis of 128-bit Symmetric Block Cipher(SEED), 1999. 4.


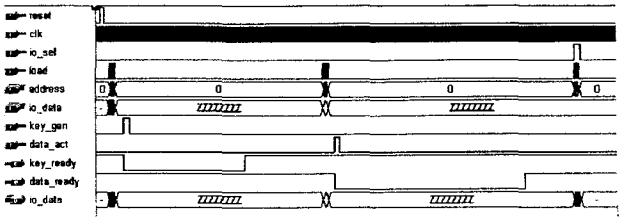


Fig. 12. Whole encryption

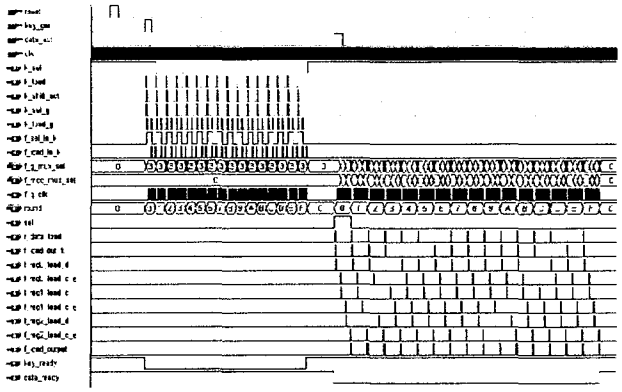

Fig. 13. Control signals

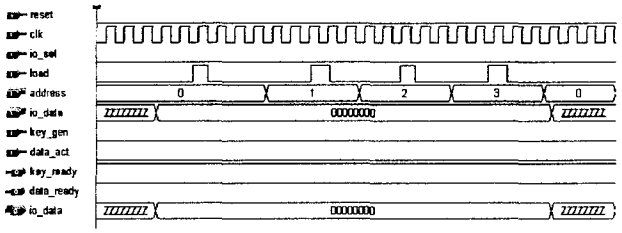


Fig. 14. Cipher key Input

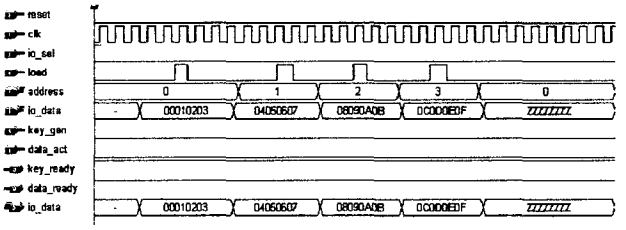


Fig. 15. Plain text input

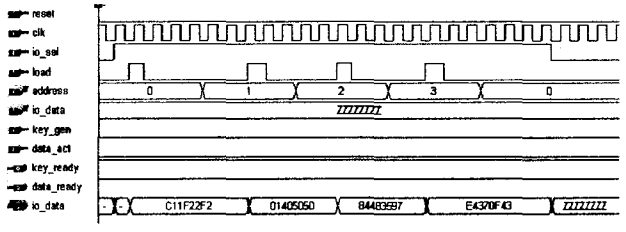


Fig. 16. Cipher text output