IP Design of Corrected Block TEA Cipher with Variable-Length Message for Smart IoT

Hyeopgoo Yeo[1], Seungil Sonh[1*], Mingoo Kang[2]
[1] Division of Information & Telecommunications, Hanshin University
Osan-si, Gyeonggi-do, 18101 - Korea
[e-mail: hgyeo@hs.ac.kr]
[e-mail: saisonh@hs.ac.kr]
[2] Dept. of IT Contents, Hanshin University
Osan-si, Gyeonggi-do, 18101 - Korea
[e-mail: kangmg@ hs.ac.kr]
*Corresponding author:Seungil Sonh

Received August 14, 2019; revised October 17, 2019; accepted November 12, 2019; published February 29, 2020

Abstract

Corrected Block TEA(or XXTEA) is a block cipher designed to correct security weakness in the original block TEA in 1998. In this paper, XXTEA cipher hardware which can encrypt or decrypt between 64-bit and 256-bit messages using 128-bit master key is implemented. Minimum message block size is 64-bit wide and maximal message block size is 256-bit wide. The designed XXTEA can encrypt and decrypt variable-length message blocks which are some arbitrary multiple of 32 bits in message block sizes. XXTEA core of this paper is described using Verilog-HDL and downloaded on Vertex4. The operation frequency is 177MHz. The maximum throughput for 64-bit message blocks is 174Mbps and that of 256-bit message blocks is 467Mbps. The cryptographic IP of this paper is applicable as security module of the mobile areas such as smart card, internet banking, e-commerce and IoT.

Keywords: Corrected Block TEA(XXTEA), Symmetric Block Cipher, Encryption, Decryption

This research was supported by a research grant from Hanshin Unersity.

http://doi.org/10.3837/tiis.2020.02.014 **ISSN : 1976-7277**

1. Introduction

With present network system, it is possible to transmit various formats of data including e-mail, picture, videos and voices[1]. In a world where the personal portable mobile devices such as PDAs(Personal Data Assistant) and cell phones are evolving rapidly, information security has become inarguably the most important issue than ever[3]. It is expected that the development of IoT service will inflate the chance of potential security threats, as the IoT service utilizes applications and heterogeneous network through smart devices or sensors[3,4]. The Symmetric cryptography system to be discussed is a system which the key for encryption and decryption is identical. This system has been published in various systems such as DES(Data Encryption Standard), IDEA(International Data Encryption Algorithm), SKIPJACK, MISTY, Camellia, SEED, LEA(Lightweight Encryption Algorithm) and AES(Advanced Encryption Standard) algorithm[5]. In particular, LEA(Lightweight Encryption Algorithm) is the lightweight symetric block cipher proposed by NSRI(National Security Research Institute) in 2013[3]. It has 128-bit message block size and 128, 192, or 256-bit master key sizes. Also it only uses ARX(modular Addition, bitwise Rotation, and bitwise XOR) operations for 32-bit words as round operations. It is known that LEA is optimized for 32-bit software platform and can operate fast in 32-bit platform.

Almost cipher algorithms adopt a fixed-length input message block, but the length of master keys can be changed to enhance the security intensity for encrypting and decrypting the fixed message block. On the other hand, XXTEA encryption algorithm design introduced in this paper employs a 128-bit Master Key that embodies the encryption and decryption according to the variable-length message inputs adopting minimum input of 64-bit to multiples of 32-bit, which are 96/128/160/192/224/256–bit.

Table 1 shows the classification of the symmetric block ciphers according to a variation of message block and master key sizes. In some block ciphers, such as DES, 3DES, and IDEA, message block and key sizes are fixed. Also, in some block ciphers such as AES, Blowfish, RC6, and MARS, key size can be changed . In XXTEA block cipher of this paper, only message block size can be changed.

TEA was proposed by David Wheeler and Roger Needham of the Cambridge Computer Laboratory in 1994. TEA uses a 128-bit key and 64-bit message block. Also it has a Feistel structure of 64-round. It can also be vulnerable to a related-key attack against the block cipher[6]. XTEA(eXtended TEA) is a symmetric block cipher proposed to correct the security weakness in TEA by David Wheeler and Roger Needham in 1997. However, it is vulnerable to a related-key differential attack[4] . Finally, XXTEA block cipher was released to compensate for the security weakness of TEA and XTEA symmetric block ciphers[7]. Of course, it has also been announced that security attacks on XXTEA cryptographic algorithms may be partially vulnerable[8].

This paper consists as such introduction of XXTEA encryption algorithm, the design of XXTEA encryption algorithm, verification and performance analysis of the designed XXTEA encryption algorithm, and conclusion.

Table 1. Classification of block ciphers according to a variation of message block and master key sizes

Classification	algorithm	Message length(bits)	Key size (bits)	# of rounds	# of S boxes	Algorithm structure
Fixed message block and key size	DES	64	56	16	8	Feistel
	3-DES	64	168	48	8	Feistel
	IDEA	64	128	8	-	Substitution & permutation
Variable key size	AES	128	128,192,256	10 ~ 14	-	Feistel
	Blowfish	64	128-448	16	4	Feistel
	RC6	128	128,192,256	20	-	Feistel
	Twofish	128	128,192,256	32	8	Feistel
	MARS	126	128-448	32	1	Feistel
Variable message size	XXTEA	64 – 256 Multiples of 32-bit	128	Under 32	-	Feistel

2. Definition of XXTEA Encryption Algorithm[8]

XXTEA encryption algorithm, like most of the other block cypher algorithms, can be distinguished into having two main functions: i) key scheduling block and ii) round block that can randomize data. First, **Fig. 1** shows Pseudo C code for encryption and decryption of XXTEA cipher. The MX function, which is the core of data randomization of this algorithm, consists of shift, xor and modulo addition. The y, z, sum, and round key are used as the inputs of the MX function. Also the key used in the MX function is selected using the lower 2 bits of the sub-round iteration counter p and the lower 2 bits of the 2 bits right shifted sum. q-value defines number of rounds performed, as can be noticed from equation, (q=6+52/length), the number of rounds of the length-value that informs of the number of 32-bit message inputs is a variable. That is, the minimum value of q is 6 and the maximum value of q is when length is the smallest, 2(=64-bit), which results in 32 rounds. Therefore, the XXTEA is able to perform round operations between rounds 6 and 32. Also, 32-round is when the input message is 64-bit and when the input message is 256-bit(32-bit*8), (q=6+52/8) results in 12 rounds operation in total.

In case of encryption, the sum-value is used in the MX function employing the delta constant as the initial value, and is added with the previous sum after the completion of the round. In case of decryption, sum(=q*DELTA) is used as the initial value, and the sum subtracted from DELTA-value is used in the MX function operation after the termination of every round. The minimum value of each input message can be used every 32-bit from v[0] to v[7] depending on the input bit mode. IF the input message is the minimum 64-bit, only v[0] and v[1] are used. For 96-bit, v[0], v[1], v[2] is used and for 256-bit, v[0] to v[7] is used. One round is variable depending on the number of valid message block v[i]. If i=2(64-bit), p=2 and performs 2 partial rounds. If i=8(256-bit), p=8 and requires to perform 8 partial rounds to conclude a round. Through this process, every v[i]-value that is valid after performing each round gets updated. v[i]-value that updates when encrypting gets updated after being added with the MX function value, and v[i]-value that updates when decrypting gets updated after

being subtracted by the MX function value. Y-value(in case of encryption) or z-value(in case of decryption) that participates in the update of v[i]-value employs a nearby v-value of register circulation system. In other words, when encrypting, v[0] and v[1] are used to update v[0], v[1] and v[2] are used to update v[1], and when decrypting, v[n-1](n=32-bit input number) and v[n-2] are used to update v[n-1], and v[n-2] and v[n-3] are used to update v[n-2]. Therefore, it can be seen that in the process of performing the partial rounds for encryption and decryption, the update sequence of the v-value is the exact opposite.

Also, as can be seen from Pseudo C code, y-value and z-value function as the opposite not only for the varying sequences for the input message, but also when it is used for the MX function input. In other words, in the calculation of MX function value, the arbitrary v-value is used as y-value for encryption, but it is used as z-value for decryption. Inversely, if the v-value is used as the z-value for encryption, it gets used as the y-value for decryption.

```
//MX function
#define MX ((z>>5)^(y<<2))+((y>>3)^(z<<4)))^ ((sum^y)+(key[(p&3)^e]^z))
```

```
//Encryption
void xxtea_encrypt(uint32_t *v, uint32_t length, uint32_t *key)
{
uint32_t z=v[length-1], y=v[0], sum=0, e;
uint32_t  DELTA=0x9e3779b9;
uint32_t p, q ;
z=v[length-1];
q = 6 + 52/length;
while (q-- > 0) {
sum += DELTA;
e = (sum >> 2) & 3;
for (p=0; p<length-1; p++) y = v[p+1], z = v[p] += MX;
y = v[0];
z = v[length-1] += MX;
   }
}
```

```
//Decryption
void xxtea_decrypt(uint32_t *v, uint32_t length, uint32_t *key)
{
uint32_t  z, y=v[0], sum=0, e;
uint32_t DELTA=0x9e3779b9;
uint32_t  p, q ;
q = 6 + 52/length;
sum = q*DELTA ;
while (sum != 0) {
e = (sum >> 2) & 3;
for (p=length-1; p>0; p--) z = v[p-1], y = v[p] -= MX;
z = v[length-1];
y = v[0] -= MX;
sum -= DELTA;
  }
}
```

Fig. 1. Pseudo code for encryption and decryption for XXTEA cipher

3. IP design of the XXTEA cipher

Fig. 2 shows the XXTEA block cipher processor designed in this paper. As the length of the master key is fixed to 128-bit, Key for encryption/decryption is saved when key buffer incorporates key value over 4 times. Also, through bus, the setup for encryption/decryption mode can be determined. The XXTEA encryption processor designed in this paper supports encryption and decryption for message blocks varying lengths of multiple of 32-bit from 64-bit to 256-bit, and the control value that informs this is transmitted to num_of_words register through an external bus. After the setup for control information and master key value is completed, it sends data to message buffer to be encrypted/decrypted.

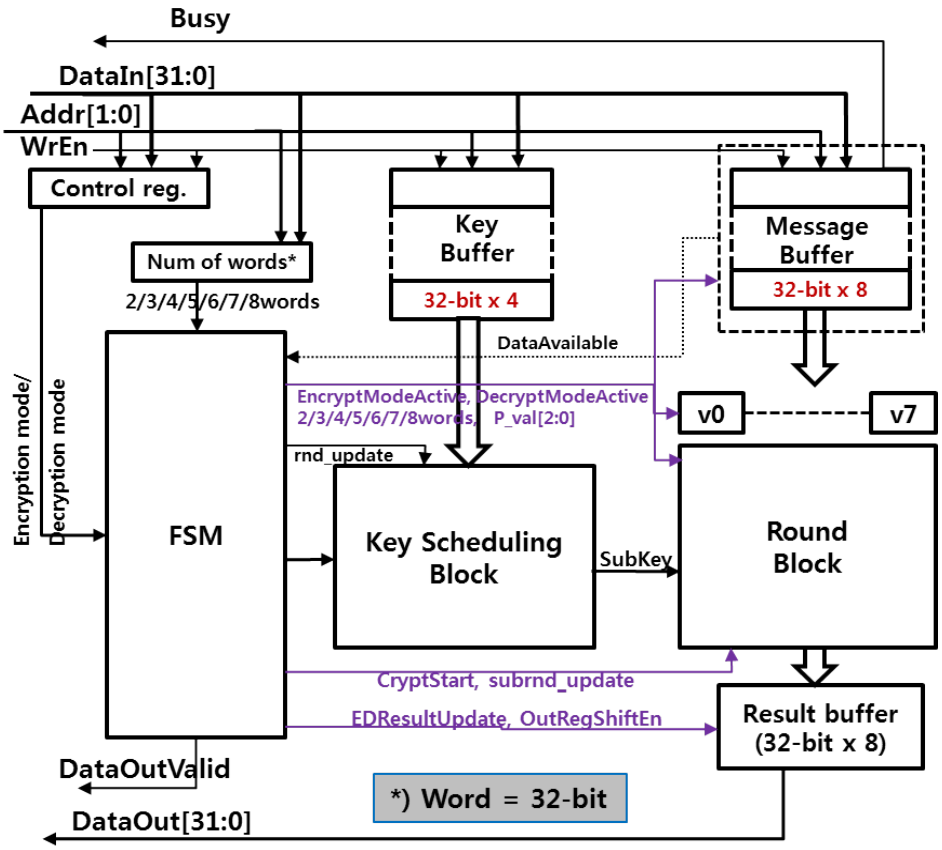

Fig. 2. The proposed XXTEA block cipher processor

Message buffer refers to 2/3/4/5/6/7/8 words signal which informs the minimum processing word size, and if the stipulated amount of data is put into the message buffer, immediately produces DataAvailable signal and notifies FSM(Finite State Machine). FSM activates CryptoStart and saves messages from Message Buffer to v-register to perform round function. After that, Key Scheduling Block and Round Block are synchronized. Then, Key Scheduling Block provides partial round key-value to the Round Block, and if partial round is performed up to the number of valid data's word(32-bit) contained in the v-register, one round is completed. Partial round update signal is subrnd_update. As after every round, sum-value

requires an update, FSM activates rnd_update signal and delivers it to the Key Scheduling Block. When every defined round is completed, FSM activates EDResultUpdate signal and delivers the final result value to Result Buffer. The final saved value is synchronized with DataOutValid signal that alerts the validity of the printed value after receiving OutRegShiftEn signal, then transmits 32-bit DataOut register bus externally as many as the number of valid result data.

Next, key scheduling block that is a fundamental block of the XXTEA block cipher will be explained. **Fig. 3** shows the structure of key scheduling block designed in this paper. In case of encryption, the initial sum-value becomes the constant Delta(=0x933779b9) value. However, in case of decryption, the initial sum-value is programmed as [sum=number of rounds x Delta], hence a multiplication machine is essential. However, in this paper, the result value can be found when input message bit value is determined, and only 7 values are tested, therefore the multiplication machine was terminated using mux as **Fig. 3**. After the initial sum-value is determined, with the termination of each round, the result value of sum-value added together with the Delta value is used as the new value for encryption, and for decryption, the result value of sum value subtracted by delta value is used as the new sum-value, therefore addition and reduction machine was used. The key value being employed in each round is used as a selective address for selecting one out of the 4 key values, that were acquired by doing XOR of partial 2-bit of p-value, that informs the number of partial round performance, p[1:0] and sum[3:2]. Next, the design of round block will be explained. **Fig. 4** represents the round block designed in this paper.

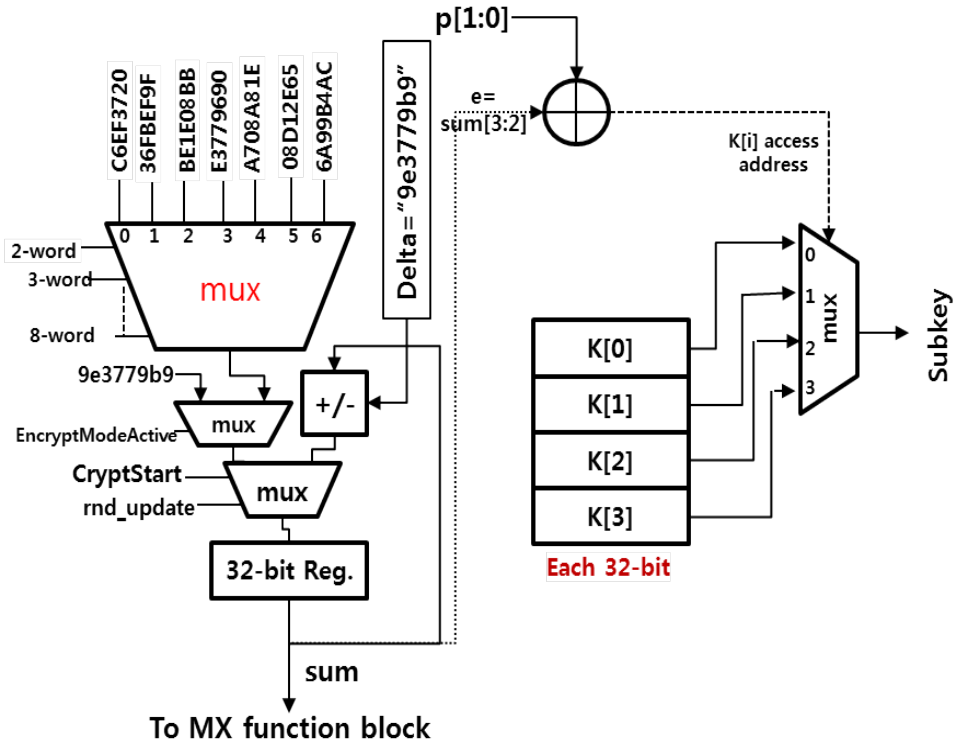

Fig. 3. The key scheduling block for XXTEA cipher

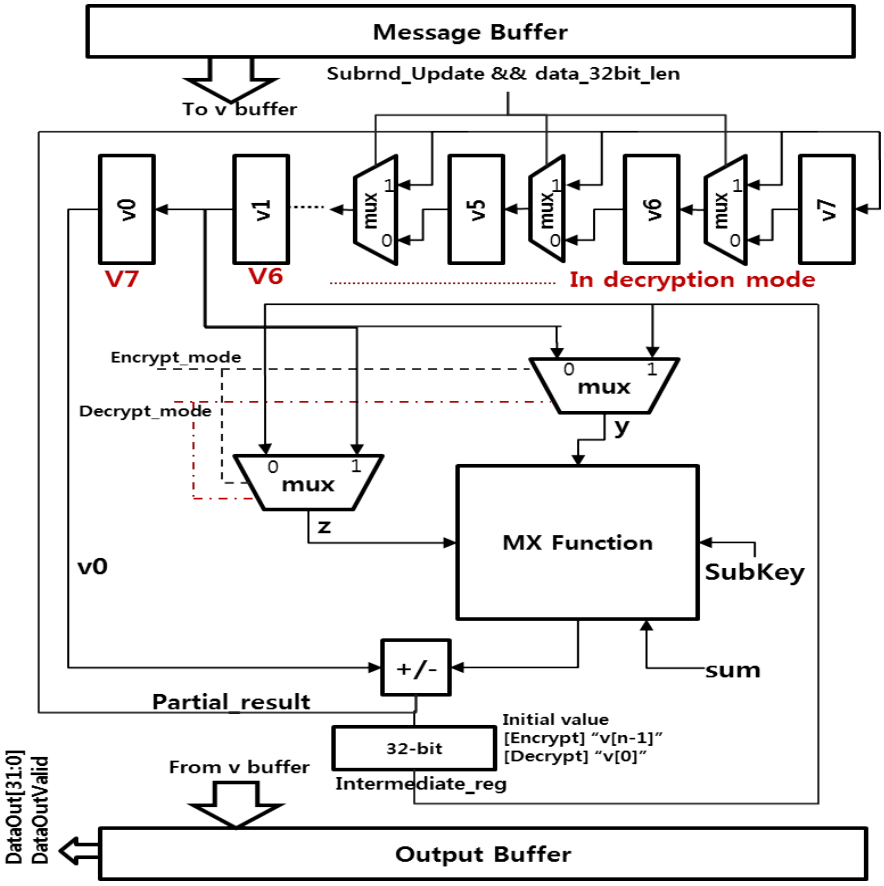

Fig. 4. The round block for XXTEA cipher

It was concluded from the pseudo C code analysis that in a round block, the input x and y of MX function block are exchanged and used. Thus, the round block was designed using mux, based on the encryption mode for y and z input to be exchanged. The 32-bit register Intermediate_reg below **Fig. 4** is a saved value of MX function result value added with v-value(in case of encryption), or subtracted by v-value(in case of decryption). This register value's hardware structure is simplified at the next clock cycle, if used as y or z depending on the mode. Also, the value of MX Function value and v-value added/subtracted together corresponds to an updated v-value, and therefore is packed into different positions regarding valid number of words; for 2 words(64-bit), it is stored in v1, for 3 words – v2, 8 words(256-bit) – v7. Moreover, vi registers connected in a Serial Shift Register are shifted to the left direction for the next partial round operations.

Fig. 5 represents a hardware implementation of MX function block. MX function is called a round function. MX function consists of shift, xor, and modulo addition. The y, z, sum, and round keys are inputs to MX function. Finally, MX result is produced. The sum value of each round is equal to previous sum+delta. p is the sub-round iteration counter. The value e is LSB two bits of two-bit right shifted sum value for each round.

SubKey is a value provided by key scheduling block and employs same value while performing one round. When the final round accomplishes, the final result encryption/decryption value that is positioned at v register is transferred to the Output Buffer and this value transfers 32-bit print data DataOut externally along with the DataOutValid

signal. The change to the hardware can be minimized by positioning v-register storing sequence oppositely. That is, using 8-word message mode as an example, in case of encryption the v-value is stored in v0, v1, …, v7 order from the left, and in case of decryption the v-value is stored in v7, v6, …, v0 order. This is why the message buffer is designed to reorder data sequence.

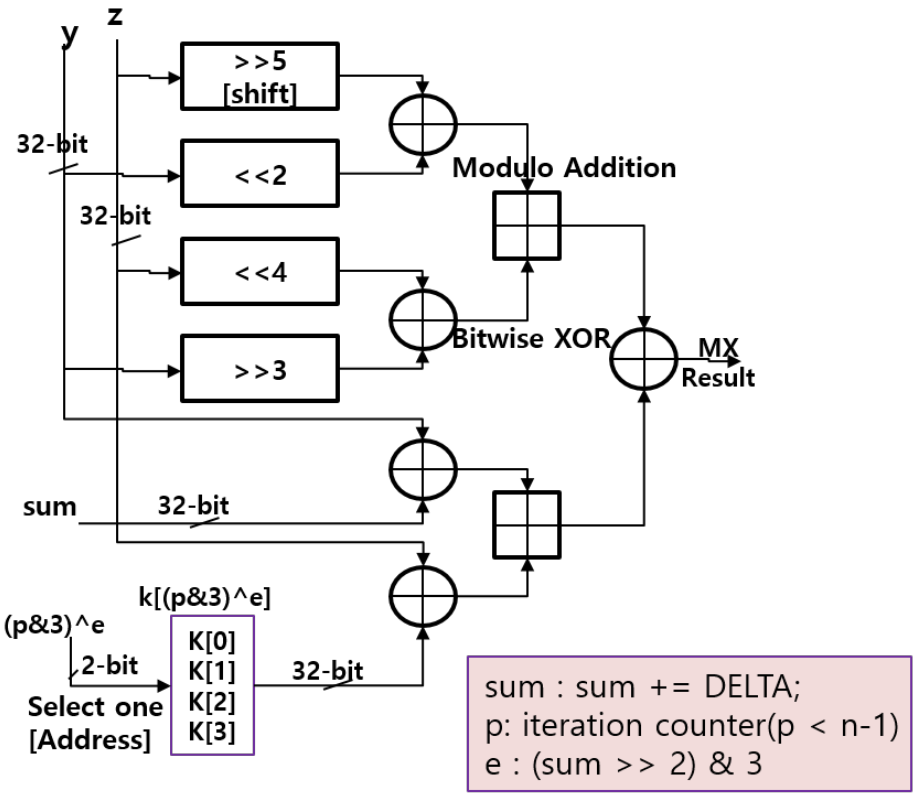

Fig. 5. Hardware implementation of MX function

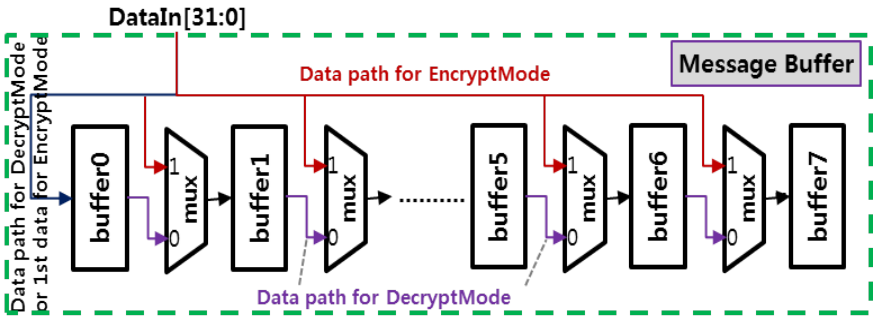

Fig. 6. Message buffer with data reordering

In encryption mode, when data is stored in the Message Buffer of **fig. 6**, it is stored sequentially from buffer0. However, for decryption mode, data is always entered from buffer0. When new data is inserted to be saved, existing data in buffer0 is transferred to buffer1 and the

new data is saved in buffer0. By doing so, when data is transferred from bufferi to vi, reordered data can be transferred to a register that carries an identical i-value. In this paper, the design was completed in a way that minimum hardware could be used.

3. Verification and Performance Analysis of Designed XXTEA Cipher

For the verification of the designed XXTEA encryption algorithm, the test case from reference[10] was used. Using Verilog HDL, XXTEA module was designed and the verification was processed by comparing suggested input and result value of the test case corresponds to the input and result values of the designed module. The XXTEA cipher algorithm designed in this paper employs 128-bit length of fixed master key and plain text or cipher text, the input message, can have varying lengths of the multiples of 32-bit from 64-bit, that is 64-bit, 96-bit, 128-bit, 160-bit, 192-bit, 224-bit, and 256-bit. The design tool used in this paper is Xilinx ISE[14,15].

This paper will only explain the simulation results for 96-bit input message. First, **Fig. 7** employed “ffffffff”, “ffffffff”, “ffffffff”, “ffffffff” for 128-bit Master Key. Plain text, the input for designed XXTEA cipher, employed “a8137c15”, “575eba50”, “91776d30”. It was confirmed that encryption using the following inputs yielded final cipher “ef1c60b2”, “72b778b0”, “6abaccab”, which matched the proposed values from the test case. The total time taken for yielding the result value was 70-clock cycle.

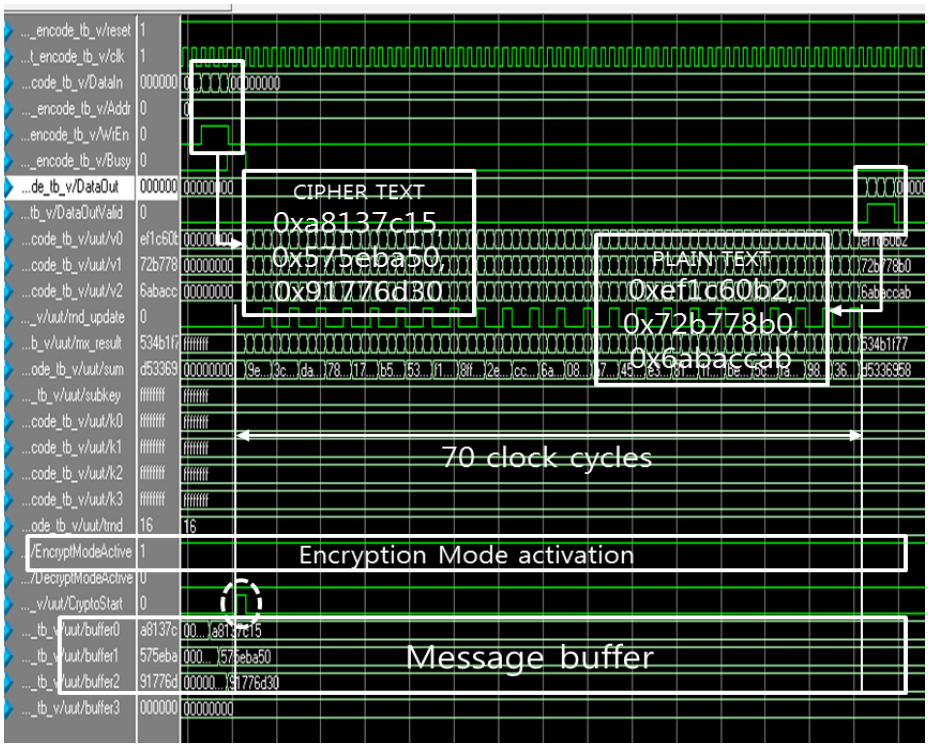

Fig. 7. Encryption simulation of 96-bit plain text for XXTEA cipher

Fig. 8 shows the decryption simulation of 96-bit cipher text for XXTEA cipher algorithm. Similar to **Fig. 7**, under the identical master key setting excluding the decryption mode, when

decryption is processed using cipher text as an input, like **Fig. 8**, it can be retrieved back to a plain text.

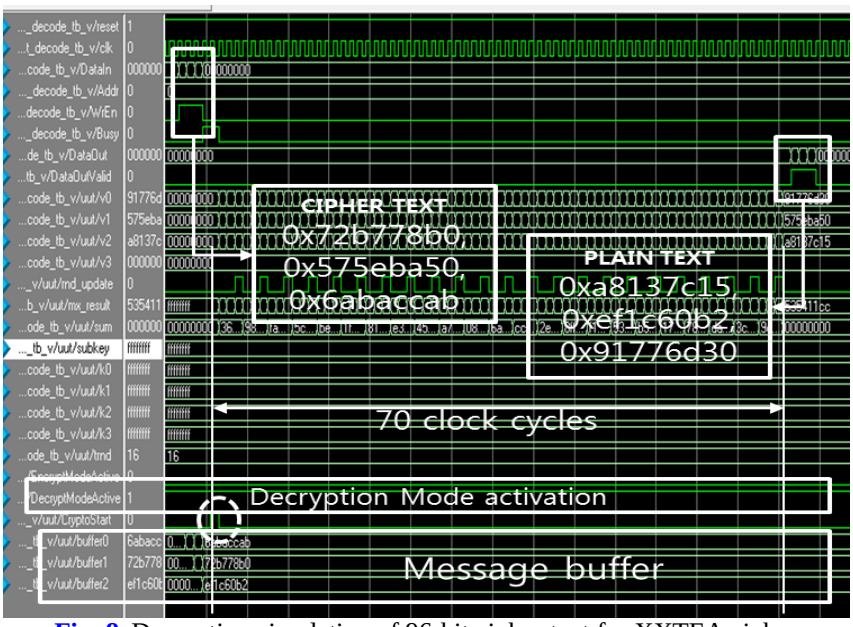

Fig. 8. Decryption simulation of 96-bit cipher text for XXTEA cipher

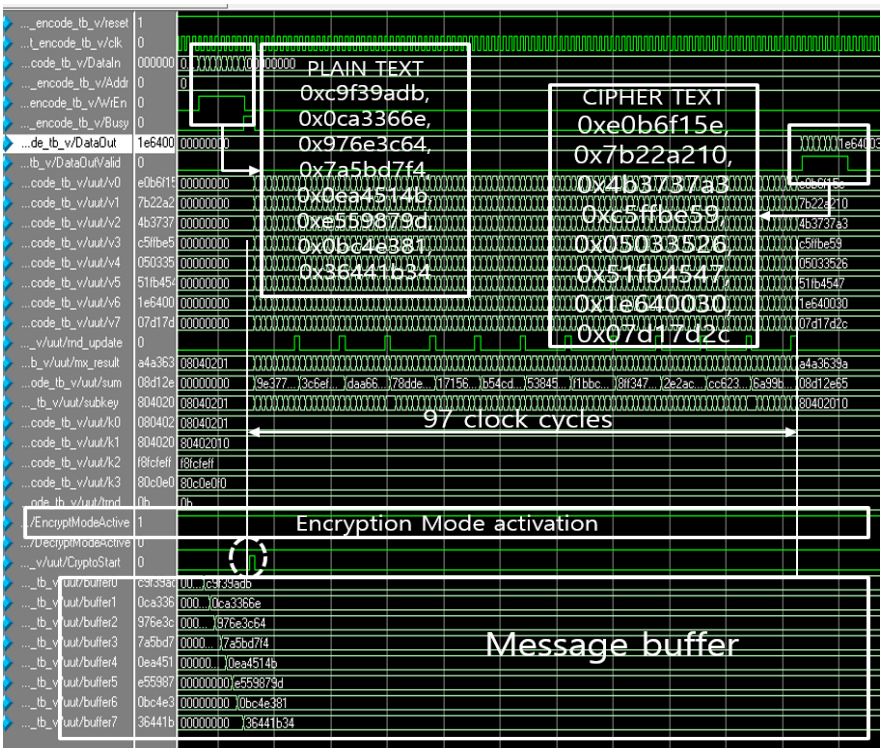

Fig. 9. Encryption simulation of 256-bit cipher text for XXTEA cipher

In this paper, the designed XXTEA cipher processor supports encryption and decryption for successive messages. However, depending on the message block sizes, the execution cycles are variable. The execution cycle for 96-bit and 256-bit has already been referred. 65-clock cycle is taken for 64-bit message, 69-clock cycle for 128-bit message, 85-clock cycle for 192-bit message.

Next, the encryption/decryption simulation of 256-bit message input will be explained. **Fig. 9** employed “08040201”, “80402010”, “f8fcfeff”, “80c0e0f0” for master key. Employing “c9f39adb”, “0ca3366e”, “91776d30”, “7a5bd7f4”, “0ea4514b”, “e559879d”, “0bc4e381”, “36441b34”, as the plain text message input, and carrying out encryption, the final cipher is “e0b6f15e”, “7b22a210”, “4b3737a3”, “c5ffbe59”, “05033526”, “e51fb4547”, “1e640030”, “07d17d2c”, corresponding to the values suggested at the test case. The total time taken for yielding the result was 97 Clock cycles.

Fig. 10 is a decryption simulation of 256-bit cipher text for XXTEA cipher, which the decryption mode is activated and uses the same master key as **Fig. 9** as an input. It shows that when the cipher text is entered, the original plain text can be obtained through the activation of decryption mode.

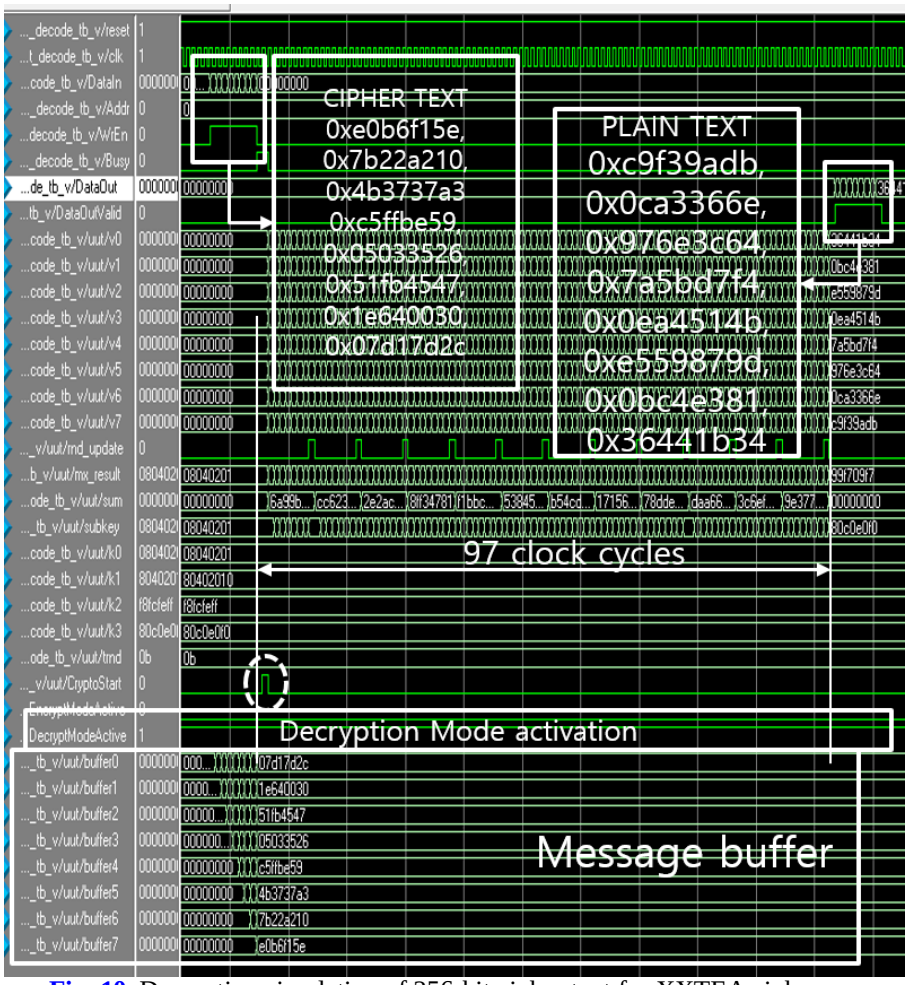

Fig. 10. Decryption simulation of 256-bit cipher text for XXTEA cipher

Table 2 shows the performance analysis of lighweight cryptographic algorithms. XTEA[11] was implemented using Samsung 0.35um technology. It has a maximum throughput of 125Mbps. HIGHT[12] was proposed in South Korea in 2005. HIGHT is a lightweight cipher with 64-bit block length and 128-bit key length. It has a maximum throughput of 271Mbps. The XXTEA processor designed in this paper has features that it can process different lengths of input message blocks, hold an operation frequency of 177MHz at Vertex4, the maximum throughput for 64-bit message input is 174Mbps, and finally for 256-bit message input, it shows a performance of 467Mbps.

Table 2. Performance analysis of lightweight cryptographic alogorithms

Items	Our XXTEA	XTEA[11]	HIGHT[12]
Encryption/ Decryption	Both supported		
I/O data bus	32-bit	64-bit	32-bit
Message size	64, 96, 128, 160, 192, 224, 256-bit	64-bit	64-bit
key	128-bit		
Operation frequency	177MHz	125MHz	144MHz
# of Slices	1618	Samsung 0.35um	1122
Block ROM	No	2300 gates	4 256x8-bit ROMs
Clock cycles of encryption/decryption for a message block	65-clock @64-bit message 97-clock @ 256-bit message [Variable clock cycles according to message sizes]	32-clock	36-clock
Maximum Throughput	174Mbps @64-bit message block 467Mbps/sec @256-bit message block	125Mbps	271Mbps

4. Conclusions

This paper designed an XXTEA cipher processor that uses a 128-bit fixed master key and can encrypt or decrypt varying message lengths for the multiples of 32-bit. It aimed to minimize the usage of hardware resources by suggesting an efficient hardware structure through using a message buffer that can relocate the data or using a multiplexer without the multiplier required for the initial sum value. The maximum data processing rate of the designed XXTEA cipher algorithm for 64-bit input message is 174Mbps, for 256-bit input message is 467Mbps. It is expected that the XXTEA cipher processor designed in this paper can be used as a security module for many areas including smart card, internet banking, e-commerce and IoT(Internet of Things).

References

[1] Satish K. Vishwakarma and Shivam Khare, "XXTEA An Optimized Encryption Design with High Feedback Substitution Box Architecture," *International Journal of Modern Engineering & Management Research*, Vol.2, Issue 3, pp.12-16, Sep. 2014. Article (CrossRef Link).
[2] Issam Damaj, Samer Hamade, and Hassan Diab, "Efficient Tiny Hardware Cipher under Verilog," in *Proc. of the 2008 High Performance Computing & Simulation Conference*, 2008. Article (CrossRef Link).
[3] Mi-Ji Sung, Kyung-Wook Shin, "An Efficient Hardware Implementation of Lightweight Block Cipher LEA-128/192/256 for IoT Security Applications," *JKIICE*, Vol.19, No. 7, pp.1608-1616, Jul. 2015. Article (CrossRef Link).
[4] Shweta Gaba, Iti Aggarwal, and Sujata Pandey, "Design of Efficient XTEA Using Verilog," *International Journal of Scientific and Research Publications*, Vol. 2, Issue 6, pp.1-5, June 2012. Article (CrossRef Link).
[5] Seungil Sonh, Byeongyoon Choi, Mingoo Kang, "Technology Trend of Cipher Chips," *KSII*, Vol.1, No.2, pp.1491-1500, Oct. 2001..
[6] J. Kelsey et al., "Related-key Cryptanalysis of 3-way, Biham-DES, cast, DES-XNew DES, RC2, and TEA," in *Proc. of 1st International Conference on Information and Communication Security*, pp.233-246, 1997. Article (CrossRef Link).
[7] David J. Wheeler and Roger M. Needham, "Correction to XTEA," *http://www.movable-type.co.uk/scripts/xxtea.pdf*, Oct. 1998. Article (CrossRef Link).
[8] Ion Sima, et al., "XXTEA, an Alternative Replacement of KASUMI Cipher Algorithm in A5/3 GSM, F8, F8 UMTS Data Security Functions," in *Proc. of 9th International Conference on Communications(IEEE)*, pp.323-326, 2012. Article (CrossRef Link).
[9] Elias Yarrokov, "Cryptanalysis of XXTEA," *International Association for Cryptologic Research*, pp.1-6, May 2010. Article (CrossRef Link).
[10] http://read.pudn.com/downloads187/ sourcecode/windows/other/877059/xxtea.cpp__.htm Article (CrossRef Link).
[11] Joohong Kim, "Hardware Design and Performance Evaluation of a Lightweight Cryptographic Algorithm for RFID/USN," *NDSL*,. 2006. Article (CrossRef Link).
[12] Seungil Sonh, "Design of Encryption/Decryption Core for Block Cipher HIGHT," *JKIICE*, vol. 16, no. 4, pp. 778-784, Apr. 2012. Article (CrossRef Link).
[13] Seungil Sonh, Hyeopgoo Yeo, "A Design of XXTEA for Variable-Length Message Block Cipher," in *Proc. of APIC-IST 2019*, pp. 115-116, Jun. 2019. Article (CrossRef Link).
[14] *Tutorial: Xilinx ISE 14.4 and Digilent Nexys 3*. Article (CrossRef Link).
[15] Hoang Nguyen, "Xilinx ISE Simulator Tutorial V14.4," pp. 1-38, Jun. 2015. Article (CrossRef Link).

Hyeopgoo Yeo received his B.S. and M.S. degrees in electronic engineering from Yonsei University in Seoul, South Korea, in 1991 and 1993, respectively. He also received his M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida in Gainesville, FL, USA, in 2003 and 2007, respectively. From 1993 to 1999, he worked as a design engineer at Samsung Electronics Co., Ltd, where he performed ASIC cell library and high-speed digital I/O design using various CMOS technologies for gate-array and standard cell. In 2008, Dr. Yeo joined the hardware R&D group at Samsung, where he was involved with mobile hardware design for wireless communications. In March 2009, he joined Hanshin University and he is currently an Assistant Professor. His research interests include RF/analog circuit.

SeungIl Sonh is a professor in the Division of Information and Telecommunications at Hanshin University, Osan South Korea from 2002. He has received the B.S., M.S., and Ph.D. degrees from Yonsei University, Seoul, Korea all in Electronic Engineering in 1989, 1991 and 1998, respectively. Also He served as a visiting professor at the Georgia Institute of Technology in the U.S. in 2015. His research interests include cryptographic algorithm, IoT, and FPGA design.

MinGoo Kang is a professor in the Dept. of IT Contents at Hanshin University, Osan South Korea from 2000. He has received the B.S., M.S., and Ph.D. degrees from Yonsei University, Seoul, Korea all in Electronic Engineering in 1986, 1989 and 1994, respectively. He was a research engineer at Samsung Electronics from 1985 to 1997. His research interests include wireless communication algorithm, smart mobile IoT devices, and blockchain security. He is a chair of KSII(Korean Society of Internet Information), also served ICONI 2014, and APIC-IST 2017 hosted by KSII as the conference chair.