Hardware Implementation of the 3GPP KASUMI crypto algorithm

HoWon Kim[1] , YongJe Choi[1], MooSeop Kim[1] and HuiSu Ryu[1]
[1] Department of Information Security Basic,
Electronics and Telecommunications Research Institute(ETRI)
161 Gajeong-Dong YuSeong-Gu, DaeJeon, 305-350, KOREA
Tel : +82-42-860-6228, / FAX : +82-42-860-5611
e-mail :khw@etri.re.kr

Abstract: In this paper, we will present the design and implementation of the KASUMI crypto algorithm and confidentiality algorithm (f8) to an hardware chip for 3GPP system. The f8 algorithm is based on the KASUMI which is a block cipher that produces a 64-bit output from a 64-bit input under the control of a 128-bit key. Various architectures (low hardware complexity version and high performance version) of the KASUMI are made with a Xilinx FPGA and the characteristics such as hardware complexity and ther performance are analyzed.

1. Introduction

In 3GPP system[1,2,3], to make a higher level of security, the following mechanisms for authentication and key agreement are defined for cryptographic functions. The random challenge generating function(f0), the network authentication function(f1), the re-synchronization message authentication function(f1*), the user authentication function(f2), the cipher key derivation function(f3), the integrity key derivation function(f4), the anonymity key derivation function for normal operation(f5), and the anonymity key derivation for re-synchronization(f5*).

And at the ME(Mobile Equipment) and RNC(Radio Network Controller), the f8 confidentiality algorithm and the f9 integrity algorithm are also standardized. These two algorithms are based on the KASUMI which is a modified version of MISTY crypto algorithm[4]. To provide sufficient performance on RNC switch, the KASUMI, f8 and f9 hardware crypto modules should be made with an high performance.

In this paper, we will present the design and implementation of the KASUMI and confidentiality algorithm (f8) to an hardware chip for 3GPP system. The f8 algorithm is based on the KASUMI algorithm which is a block cipher that produces a 64-bit output from a 64-bit input under the control of a 128-bit key. Various architectures (low hardware complexity version and high performance version) of the KASUMI algorithm are made with a Xilinx FPGA and the hardware characteristics are analyzed.

2. Design and Implementation of KASUMI Crypto Algorithm

2.1 KASUMI Crypto Algorithm

The KASUMI is a block cipher algorithm which has a Feistel structure[5]. The block diagram of the KASUMI is depicted in Figure 1. It operates on a 64-bit data block and uses a 128-bit key with eight round operations. The 64-bit input I is divided into two 32-bit strings L_0 and R_0, where $I = L_0 \| R_0$. Then for each integer i with $1 \le i \le 8$, we define: $R_i = L_{i-1}$, $L_i = R_{i-1} \oplus fi(L_{i-1}, RK_i)$. This constitutes the i^{th} round function of KASUMI, where f_i denotes the round function with L_{i-1} and round key RK_i as inputs. The result of the KASUMI is equal to the 64-bit string $(L_8 \| R_8)$ offered at the end of the eighth round.

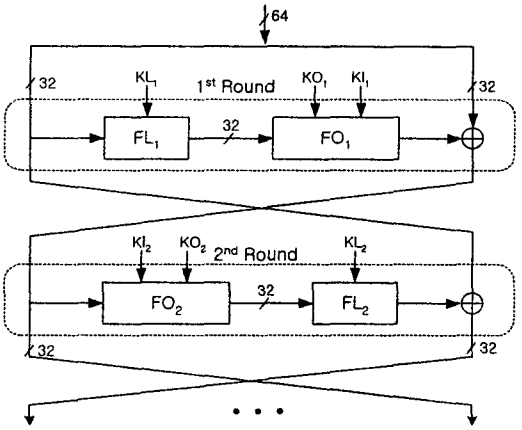

Figure 1. Two round operations(even and odd round) of KASUMI crypto algorithm

The function $f_i( )$ takes a 32-bit input and returns a 32-bit output under the control of a round key RK_i, where the round key comprises the subkey triplet of (KL_i, KO_i, KI_i). The function itself is constructed from two subfunctions; FL and FO with associated subkeys KL_i(used with FL) and subkeys KO_i and KI_i(used with FO). The $f_i()$ function has two different forms depending on whether it is an even round or an odd round.

For rounds 1,3,5 and 7 we define: $f_i(I, RK_i) = FO(FL(I, KL_i), KO_i, KI_i)$ and for rounds 2, 4, 6 and 8 we define: $f_i(I, K_i) = FL(FO(I, KO_i, KI_i), KL_i)$, where $FL( )$, $FO( )$ functions are defined at [2]. The FL function takes a 32-bit data input, a 32-bit subkey KL_i and returns 32-bit as shown in Figure 2. Since the main operations of the FL function are 16-bit AND operation, 16-bit OR operation and 1-bit left rotation operation, we can efficiently make the FL function with an hardware.

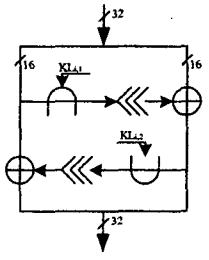

Figure 2. FL function of the KASUMI crypto algorithm

The *FO* function of the KASUMI, as shown in Figure 3, takes a 32-bit data input and two sets of sub-keys, a 48-bit sub-key KO_i and 48-bit sub-key KI_i and returns a 32-bit data output. The *FO* function is comprised of three *FI* function and six XOR operation. The S-box which provides a nonlineality to KASUMI is embedded at *FI* function. The details of the *FI* function and S-box are defined at [2].

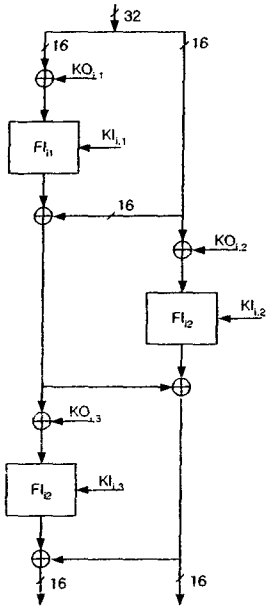

Figure 3. FO function of the KASUMI crypto algorithm

2.2 Hardware Implementation of the KASUMI Crypto Algorithm

For implementing the KASUMI to an hardware block, we can consider the low power version(Type 1) and high performance version(Type 2), which are applicable to ME and RNC switch in 3GPP system, respectively. Figure 4 shows the block diagram of the KASUMI algorithm for low power consumption. It has a simple but efficient architecture for executing KASUMI. The one round of KASUMI is repeated 8 times for completing KASUMI operation. With sacrificing the performance, it achieves simple hardware complexity and low power consumption.

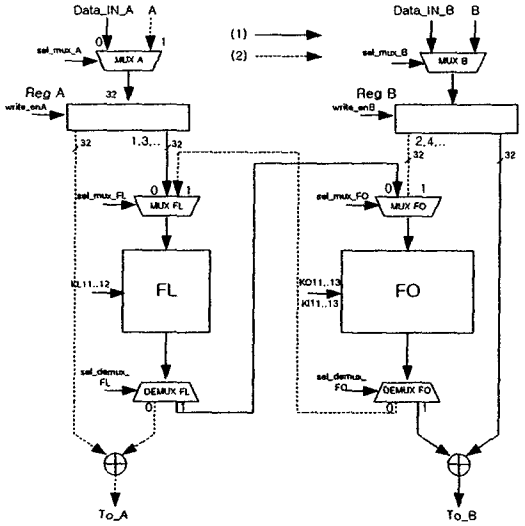

Figure 4. Block diagram of the low power KASUMI hardware (Type 1)

For odd round, the data at the Reg A is processed by *FL* function and then *FO* function with proper key values. The results are feedbacked to Reg B after exclusive operation(XOR) with the data at the Reg B. The datapath for this operation is depicted in Figure 4 with solid line. For even round, the data at the Reg B is processed by *FO* function and then *FL* function. The exclusive operation results of the even round are saved to Reg A.

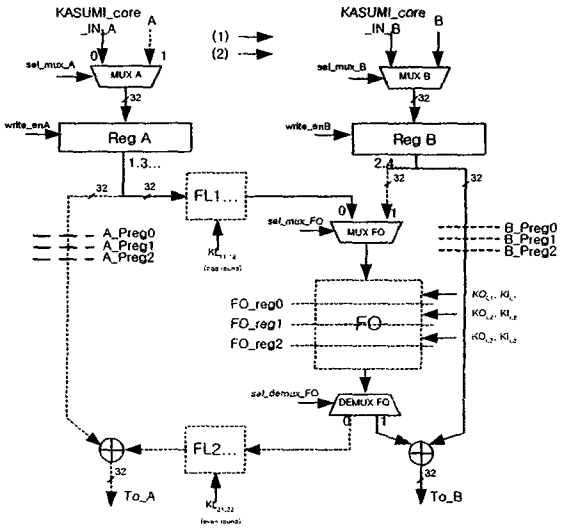

Figure 5. Block diagram of the high performance KASUMI hardware(Type 2)

To make a KASUMI algorithm be applicable to RNC switch which is necessary for high performance, we have made another architecture(Type 2) as shown in Figure 5. It has a four stage pipeline for one round hardware logic. This architecture achieves high performance with a low hardware complexity. The four stage pipeline means that we can execute the four encryption(or decryption) operations for another messages simultaneously. It achieves a higher operation frequency and throughput. This architecture is suitable for RNC switch which needs to handle much number of message traffic.

Since the major time delay is due to *FO* function block, we have divided the KASUMI block into four partitions by inserting three pipeline registers to *FO* block as shown in Figure 5. The dotted line at the *FO* block means that three pipeline registers are inserted. As described at section 2.1, the *FO* block is comprised of *FI* block and the *FI* block is comprised of S-Blox.

The four stage pipeline structure of the KASUMI achieves higher operating frequency and higher throughput. Ideally, it achieves 4 times faster than the Type 1 as shown in Figure 4. For avoiding *FL* block conflicts between another data, we have inserted another *FL* block, *FL2*.

Figure 6 shows the key scheduling hardware block for four stage pipeline architecture. Due to its regularity of the key scheduling mechanism of the KASUMI, the key scheduling block only needs registers to store constant values and key values, hardwired right rotation block, exclusive OR gates and the registers for pipeline synchronization.

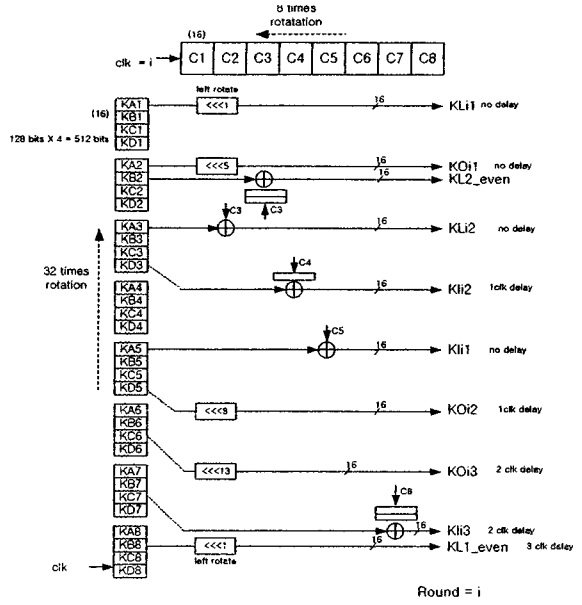

Figure 6. Key scheduling block for high performance KASUMI

3. Hardware Implementation of the F8 Encryption Algorithm

Within the security architecture of the 3GPP system, the f8 algorithm is standardized to provide a confidentiality in message traffic. The confidentiality algorithm f8 is a stream cipher that is used to encrypt and decrypt blocks of data under a confidentiality key CK. The block of data may be between 1 and 5114 bits long. The algorithm uses KASUMI in a form of output-feedback mode as a keystream generator. The structure of the confidentiality function f8 is shown in [2] and the block diagram for hardware implementation is shown in Figure 7.

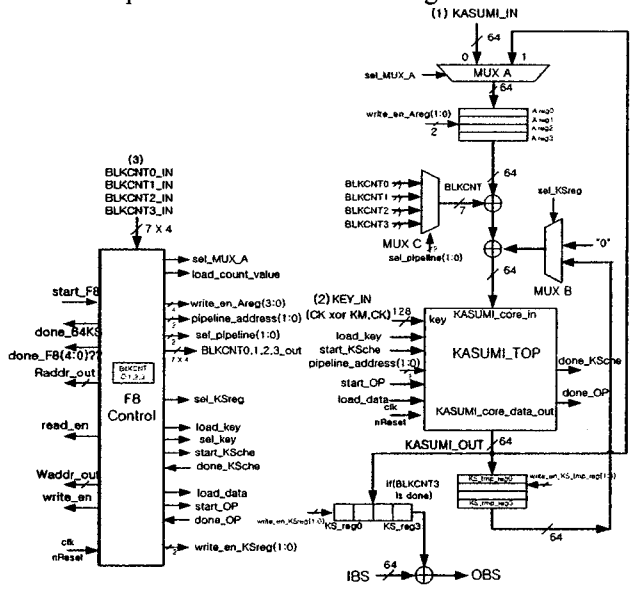

Figure 7: Block diagram of the F8 encryption algorithm

The f8 crypto block has a KASUMI crypto core, input/output buffers, a control logic, multiplexers, and data path. It has four stage pipeline registers for manipulating four different messages. To make the initial *A* value, the KASUMI crypto block reads COUNT, BEARER, DIRECTION, and 64-bit input data with properly padded zeros. With a proper key values(CK ⊕ KM), KASUMI make the initial *A* value with 8 round operations. This initial *A* value and BLKCNT, KASUMI key streams are fed with KASUMI crypto module with proper exclusive OR operations. The key stream values are exclusive ORed with an input bit stream to make an output bit stream.

We have implemented the Type 1 and Type 2 KASUMI crypto module and f8 crypto algorithm with the Xilinx Virtex-E FPGA chip. The designs were implemented in VHDL and synthesized with the Synopsys FPGA Express. Xilinx Foundation 4.2 was used for FPGA implementation. After we have verified the full functionalities of the KASUMI and f8 FPGA chip, we have made a PCI interfaced PCB board with this FPGA and then verified its normal functionality and measured its performance under the Windows operating system. Figure 8 shows the PCB board for KASUMI and f8 crypto algorithms. The input/output buffer and PCI interface logics with KASUMI/f8 hardware block are implemented in single Xilinx FPGA chip.

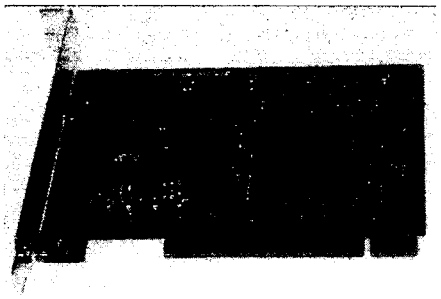

Figure 8: Photograph of the KASUMI and F8 crypto PCB board

Table 1 shows the characteristics of KASUMI and f8 crypto module which are implemented in a Xilinx FPGA chip. Since we have made the FPGA chip compatible with a 33MHz PCI 2.2 bus, the four stage pipelined KASUMI core and corresponding f8 FPGA chip have a 33MHz operating frequency. However, with a Xilinx Foundation tool, we have confirmed the maximum operating frequency of the KASUMI crypto block(Type 2) achieves 60MHz operating frequency. To compare the performance of our results, we have referenced the Marinis's results[6] with our results as shown in Table 1 and Table 2.

Table 1 : Main features of the KASUMI crypto module when implemented for Xilinx Virtex-E

	Our Result		Marinis's Result[6]
	Type 1	Type 2	
Frequency (MHz)	20	33(60)	20.88
Logic Size (slices)	650	1100	1287
Performance (Mbps)	110	234(410)	167.07

The Marinis's KASUMI core shown in Table 1 is a two round architecture with an even round circuit and odd round circuit. As shown in Table 1, our Type 2 KASUMI core(four stage pipeline architecture) achieves higher performance with a low hardware complexity when we compared the Marinis's result.

The characteristics of our f8 crypto algorithm is also shown in Table 2. The maximum operating frequency of the f8 crypto block is 52MHz and the operating frequency with a PCI interface block is 33MHz. Table 2 shows that our result(especially Type 2) shows that our results achieve higher performance than Marinis's results with a low hardware complexity.

Table 2 : Main features of the F8 crypto module when implemented for Xilinx Virtex-E

	Our Result		Marinis's Result[6]
	Type 1	Type 2	
Frequency (MHz)	19.5	33(52)	20.52
Logic Size (slices)	920	1650	2781
Performance (Mbps)	103	211(321)	N/A

4. Concluding Remarks

We have presented the design and implementation of a KASUMI crypto algorithm and f8 confidentiality crypto algorithm to an hardware chip for 3GPP system. To make a high performance f8 crypto chip, we have made the KASUMI crypto core with a four stage pipeline. And for low hardware complexity, only one round circuit of KASUMI crypto algorithm is used. This KASUMI and f8 crypto algorithm are implemented with a Xilinx Virtex-E FPGA chip and fully verified with a test-bed (which is composed of test programs in Windows operating system, PCI interface board, PCB board, etc.).

References

[1] ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms Document 1: *f8 and f9 Specification*, Sept. 2000.

[2] ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms Document 2: *KASUMI Specification*, Dec. 1999.

[3] 3GPP TS 33.102 v3.7.0 , Dec. 2000.

[4]Mitsuru Matsui: New block encryption algorithm MISTY. In *Fast Software Encryption'97*, vol. 1267 of LNCS, pages 54-68.

[5] Alfred J. Menezes, Paul C. van Oorschot, and Scot A. Vanstone. *Handbook of Applied Cryptography*,1997. Press, 1997.

[6]Kostas Marinis, Nikos K. Moshopoulos, Fotis Karoubalis, and Kiamal Z. Pekmestzi, *On the Hardware Implementation of the 3GPP Confidentiality and Integrity Algorithms*, ISC 2001, LNCS 2200.