2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

CryptoModule 2000을 이용한 KASUMI 알고리즘의 구현

이옥연*, 정교일*, 조현숙*, 유형준**, 김영국**
*한국전자통신연구원
**프롬투정보통신(주)
e-mail:youhj@from2.co.kr

Implementation of KASUMI Algorithm using CryptoModule 2000

Ok-Yeon Yi*, Kyo-Il Chung*, Hyun-Sook Cho*,
Hyung-Jun Yoo**, Yung-Kook Kim**
*Electronics and Telecommunications Research Institute
*From2 Information & Communications Co., Ltd.

요 약

암호 알고리즘을 디지털 통신에 적용하기 위해서는 고속 처리가 필수적이며, 고속화를 위하여 암호 칩으로 구현하고 있다. 프롬투정보통신에서는 암호칩을 효과적으로 제작하기 위하여 MPC 850 프로세서와 FPGA를 내장하고 통신 포트를 사용할 수 있게 하는 암호툴인 CryptoModule 2000을 발표하였다. 이 논문에서는 CryptoModule 2000을 이용하여 3GPP에서 적용하고 있는 KASUMI 암호 알고리즘을 구현하고 시험 결과를 분석하여 CryptoModule 2000이 암호 시스템 개발용으로 적합함을 확인한다.

1. 서론

현대 사회는 정보통신의 사회이다. 통신망 및 전산망을 통하여 다양한 정보를 대량으로 전송할 수 있도록 하여 편리성을 증대 시켰으나 도청 및 암호 해독 기술이 동시에 발달하여 정보의 노출이라는 역작용을 발생시켰다. 이는 정보보호의 중요성을 일깨웠고, 정보보호의 가장 적극적 기술인 암호 기술이 각광 받도록 하였다.

최근의 정보통신 시스템은 수 Kbps ~ 수십 Gbps에 이르는 다양한 데이터 처리 속도를 지니고 있으며, 사용자들 또한 고속처리에 익숙해져 있다. 따라서 이러한 고속 정보를 암호화하기 위해서는 통신 속도 이상으로 암호 및 복호가 가능해야 하며 암호화 기능이 부가된 이후에 통신 속도의 저하가 최소화되어야 한다.

통신 시스템의 발달과 함께 컴퓨터와 같은 정보처리 시스템의 발달의 암호 해독 기술을 점차 발달시켜서 암호 알고리즘의 구현 복잡도를 높이고 있다. 이는 고속화에 따른 암호 알고리즘의 공격 가능성이 높아지고 공격 방법도 다양해져서 암호 알고리즘이 소프트웨어보다는 하드웨어로 처리되어야 하는 것을 의미하며, 몇 가지 제품은 잘 알려져 있다.

암호 알고리즘은 하드웨어로 구현하는 최상의 방법은 칩으로 구현하는 것이다.[1~5] 그러나 암호 칩 개발에 대한 연구와 보안 통신에 대한 연구가 별도로 이루어지고 있어 암호 칩을 구현하여 통신에 직접 적용하기는 어려운 상태이다.

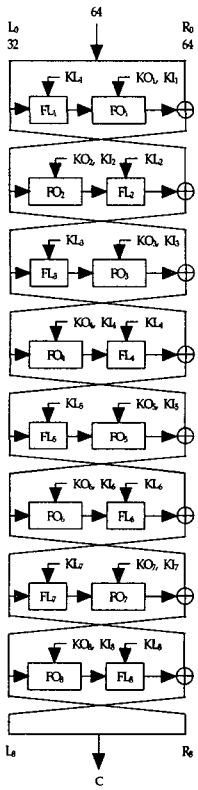
최근에 두 가지 기능을 모두 수용하는 적합한 저작도구로써 CryptoModule 2000이 소개되었다. 이 모듈은 FPGA(Field Programmable Gate Array)를 내장하여 암호 알고리즘을 고속으로 구현할 수 있으며 모토롤라의 MPC 850 프로세서를 내장하고 있어 모듈간의 직접 통신은 물론 PC와의 통신도 가능하게 하였다. 따라서 암호 통신 기능 구현 및 기능 검증을 손쉽게 할 수 있다.

본 논문에서는 CryptoModule 2000에 KASUMI 알고리즘[6]을 구현하여 시험하였으며, 또한 CryptoModule 2000의 구성도와 개발환경을 기술하였다. 시험 결과는 타이밍도를 통하여 처리속도를 확인하고 암복호가 정확히 수행되었음을 검증하였다.

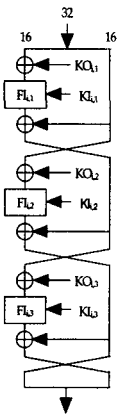
2. KASUMI 알고리즘

3GPP(3rd Generation Partnership Proejct)에서 제시한 KASUMI 알고리즘[6]은 이동통신플랫폼에 최적화된 암호 알고리즘으로 IMT-2000 환경에서 사용자에게 안전하고 경제적으로 쇼핑과 은행 업무 등의 전자상거래 서비스를 제공하기 위한 표준으로 채택되고 있다.

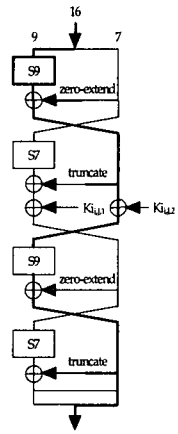
KASUMI 알고리즘은 *fi* 함수 그리고 *FL*, *FO*, *FI* 함수로 구성되며 KAUSMI 암호 알고리즘의 구조는 그림 1~6과 같다.



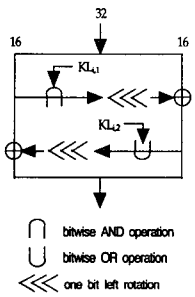
[그림 1] KASUMI



[그림 2] *FO* 함수



[그림 3] *FI* 함수



[그림 4] *FL* 함수

2.1 f_i 함수

fi 함수는 32 비트 입력 데이터 *I*를 입력으로 받아 라운드 키 $RK_i(KL_i, KO_i, KI_i)$ 제어하에 32 비트 출력 데이터 *O*를 출력한다.

fi 함수는 두 개의 서브함수(*FL*, *FO*)로 구성된다.

1과 3, 5, 7 라운드를 정의하면 식 1과 같으며, 2와 4, 6, 8 라운드를 정의하면 식 2와 같다.

$$f_i(I,RK_i)=FO(FL(I,KL_i),KO_i,KI_i) \quad (1)$$

$$f_i(I,K_i)=FL(FO(I,KO_i,KI_i),KL_i) \quad (2)$$

2.2 *FL*, *FO*, *FI* 함수

FO 함수 32 비트 입력 데이터 *I*와 두 개의 서브키(48 비트 서브키 KO_i와 48 비트 서브키 KI_i)를 입력으로 받는다 32 비트 입력 데이터 *I*는 L_0와 R_0 두 개의 데이터로 나누고 48 비트 서브키는 세 개의 16 비트 서브키로 나누어 입력한다. *FO* 함수의 출력 값은 32 비트 값을 반환한다.

FI 함수는 16 비트 입력 데이터 *I*와 16 비트 서브키 $KI_{i,j}$를 입력받는다. *I*는 9 비트 데이터 L_0와 7 비트 데이터 R_0로 나누고 키 $KI_{i,j}$ 역시 7 비트의 $KI_{i,j,1}$과 9 비트 $KI_{i,j,2}$로 나누어 입력한다. *FI* 함수의 출력 값은 16 비트 값을 반환한다.

FI 함수는 두 개의 S-box(S7와 S9)와 *ZE()*와 *TR()* 함수를 사용한다.

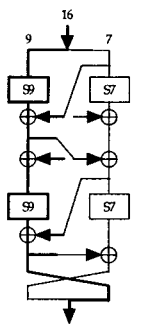
- *ZE(x)* : 7 비트 값 *x*를 받아들이고 최하위에 두 개의 '0' 비트를 추가함으로써 9 비트 값으로 바꾼다.
- *TR(x)* : 9 비트 값 *x*를 받아들이고 최하위 비트를 버림으로써 7 비트 값으로 바꾼다.

2.3 S-box

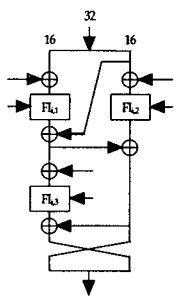
두 개의 S-box는 look-up 테이블뿐만 아니라 논리조합으로도 쉽게 구현될 수 있도록 설계되었으며, 입력 *x*는 출력 *y*의 비트 수에 따라 7 비트나 9 비트로 구성된다.

다음은 KASUMI가 하드웨어로 구현되어야 하는 특징은 다음과 같다.

- 키 스케줄 또한 하드웨어로 구현하는 것이 효과적이다.
- S-Boxe는 다수의 look-up 테이블보다는 소수의 조합에 의해 구현될 수 있도록 설계되었다.
- *FI* 함수에서 S7-Box와 S9-Box 연산은 그림 5와 같이 병렬로 수행되어야 한다.
- $FI_{i,1}$과 $FI_{i,2}$ 연산은 그림 6과 같이 병렬로 수행되어야 한다.



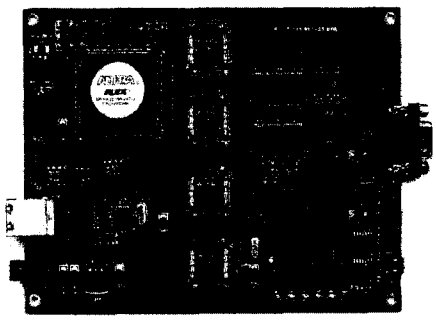
[그림 5] *FI* Function



[그림 6] *FO* Function

3. CryptoModule 2000 설계 및 개발

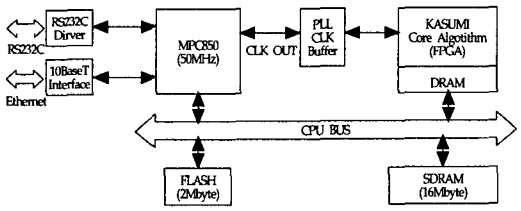
그림 7은 CryptoModule 2000의 실물 사진이다.



[그림 7] CryptoModule 2000

3.1 하드웨어

그림 8은 CryptoModule 2000의 하드웨어 구조를 나타낸다.



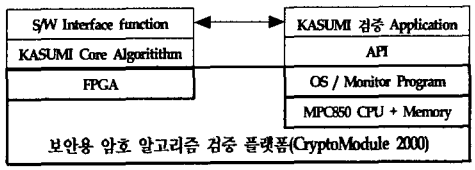
[그림 8] CryptoModule 2000 하드웨어 구조

CryptoModule 2000의 하드웨어 특징은 다음과 같다.

- 고집적 FPGA를 기반으로 융통성 있는 개발 환경 제공한다.
  - 20만 Gate급의 FPGA 사용 : 별도의 물리적인 하드웨어 수정 없이 다양한 암호 알고리즘의 하드웨어 시제품 제작 및 검증이 가능하다.
  - FPGA 내부에서 다양한 별도 메모리 제공 : 98,304 비트 (DPRAM, FIFO 등)의 메모리를 제공한다.
  - VHDL 설계 : 상호 소스공유의 편리성 제공 및 ASIC화 할 경우 별도의 작업이 필요 없다.
  - 편리한 FPGA 설계 및 개발 환경 제공 : PC 기반에서 설계 및 타이밍 시뮬레이션 기능을 제공한다.
- 고성능 CPU와 메모리를 활용한 다양한 소프트웨어 개발 및 기능 검증 환경을 제공한다.
  - Motorola사의 MPC850 50MHz CPU 사용 : 충분한 기능 검증 플랫폼으로서의 성능을 제공한다.
  - 16Mbyte/32Mbyte/64Mbyte의 SDRAM 및 2/4Mbyte의 Flash 메모리 : FPGA에 구현된 Core 알고리즘의 검증을 위한 충분한 소프트웨어개발 환경 및 기능을 제공한다.
  - 다양한 Real time OS Porting 가능 : VxWorks, pSOS 등
- 편리한 사용자 인터페이스를 제공한다.
  - RS232C Port 제공 : 9600/19200/38400bps의 다양한 속도를 제공(Default 19200bps)한다.
  - Ethernet 정합 기능 제공 : 10BaseT 인터페이스 제공으로 개발환경의 네트워킹 기능을 제공한다.

3.2 검증 플랫폼

그림 9는 CryptoModuel 2000에 적용한 KASUMI 알고리즘을 검증하기 위한 플랫폼이다.



[그림 9] CryptoModule 2000에 적용한 KASUMI 알고리즘

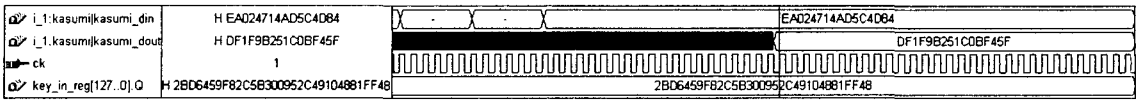
KASUMI 알고리즘의 FPGA 설계 및 시뮬레이션은 ALTERA사의 MAXplus-II를 사용하였다.

CryptoModule 2000의 FPGA 개발 및 검증 플랫폼 특징은 다음과 같다.

- KASUMI Core 알고리즘 : VHDL(Very high speed circuits Hardware Description Language)로 설계되어 FPGA에 Porting된다.
- 소프트웨어 인터페이스 기능 : KASUMI Core 기능과 소프트웨어와의 인터페이스 기능을 수행한다.
- OS/Monitor 프로그램 : KASUMI 알고리즘의 검증 프로그램을 개발하기 위한 환경을 제공한다.
- API(Application Programming Interface) : 응용 프로그램을 개발하기 위한 인터페이스를 제공한다.

4. 시험 결과 및 분석

CryptoModule 2000에 구현된 KASUMI 알고리즘에 대한 시뮬레이션은 ALTERA사의 MAXplus-II를 사용하였다. 시뮬레이션은 64 비트(EA024714AD5C4D84: HEX) 입력 데이터를 128 비트 키(2BD6459F82C5B300952C49104881FF48: HEX)로 암호화하였으며 그 결과 64 비트의 암호화된 데이터 (DF1F9B251C0BF45F: HEX)를 출력하였다. 시뮬레이션 결과는 그림 10에서 나타내었다.

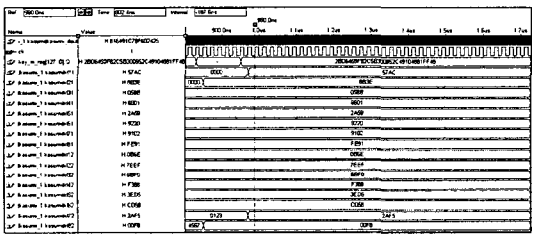


[그림 10] 암호화된 결과 값

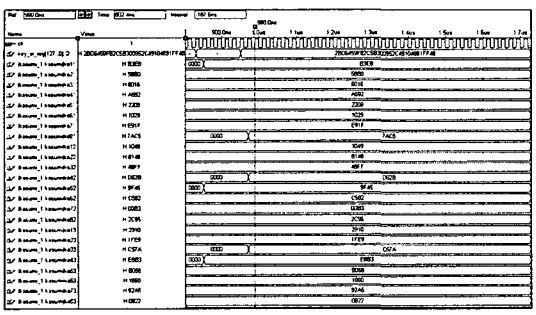
그림 11, 12, 13은 KASUMI 알고리즘의 라운드 키 $Rk_i(KL_{i,j}, KO_{i,j}, KI_{i,j})$를 생성한 결과이다. i는 KASUMI 알고리즘의 라운드 수($1<i<16$)이며, j는 $FL(1<j<2)$, $FO(i<j<3)$ 함수의 라운드 수를 의미한다.

시뮬레이션 결과 KASUMI 알고리즘을 FPGA로 구현하였을 경우 MPC 850 50MHz CPU에서 약 164Mbps(≒20Mbyte)의 속도로 암호화되었다. 시뮬레이션 결과는 소프트웨어로 구현된 KASUMI 알고리즘의 암호화 속도보다 빠르게 수행되었다.

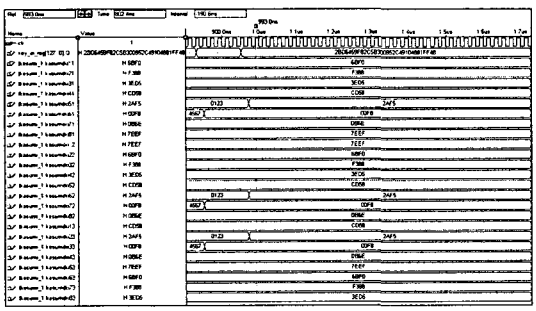
- FPGA 속도 : 64bit/390ns = 164.1Mbps(≒20Mbyte)
- 소프트웨어 속도 : 49Mbps(≒6.1Mbyte)



[그림 11] $KL_{i,j}$ 키 생성 결과



[그림 12] $KO_{i,j}$ 키 생성 결과



[그림 13] $KI_{i,j}$ 키 생성 결과

5. 결론

본 논문에서는 CryptoModule 2000을 이용하여 3GPP에서 적용하고 있는 KASUMI 알고리즘을 구현하였다. 알고리즘의 구현은 FPGA로 이루어졌으며, 타이밍도를 통하여 암복호 처리가 우수한 수행 속도로 정확히 처리되었음을 확인하였다. 본 논문에서 기술하지 않았으나 데이터 통신 시험에서도 만족할만한 결과를 얻었다. 본 논문에서는 일반적인 방법으로 KASUMI를 구현하였으므로 구현 방법을 개선하면 알고리즘 처리 속도를 향상시킬 수 있을 것이며, CryptoModule 2000을 이용하면 효과적일 것이다. 따라서 CryptoModule 2000을 이용하여 암호 알고리즘을 구현하면 암호 칩의 선행 구현을 기반으로 암호 시스템 개발에 활용도가 높을 것으로 기대된다.

참고문헌

[1] "*Encryption Policy and Market Trends*", http://guru.cose.georgetown.edu/~denning/crypto/trends.html

[2] M. Matsui, "Linear Crytanalysis of DES Cipher(Ⅰ)", *Symposium on Cryptography and Information Security '93*, 1993.

[3] 이재철, 강민섭, "3중 DES 알고리듬의 FPGA 설계 및 구현," 정보처리학회 춘계학술발표논문집, 1999, pp. 820-823.

[4] 이명동, 장경선, "SEED 암호 알고리즘의 FPGA 구현," 한국통신정보학회 충청지부, 2000, pp. 431-439.

[5] 최광윤, 오명신, "개선된 DES를 이용한 고속 암호칩의 구현," 정보처리학회 추계학술논문발표논문집, 1998, pp. 789-792.

[6] *Specification of the 3GPP Confidentiality and Integrity Algorithms*, ETSI/SAGE Specification, Version 1.0, December, 1999.