情報保護學會論文誌
第14券 第5號, 2004. 10

스마트카드용 고성능 SEED 프로세서의 구현

최 홍 묵[a)†], 최 명 렬[b)‡]

삼성전자 SOC연구소[a)], 한양대학교[b)]

Implementation of a High Performance SEED Processor for Smart Card Applications

Hong-mook Choi[a)†], Myung-ryul Choi[b)‡]

SOC R&D Center, Samsung Electronics Co., LTD[a)], Hanyang University[b)]

요 약

스마트카드의 응용 분야가 점차 확대됨에 따라 개인 정보에 대한 보안을 어떻게 유지할 것인가의 문제가 최근 가장 큰 이슈가 되고 있다. 스마트카드의 보안 기술은 암호 알고리즘을 이용한다. 빠른 속도의 암호화와 보다 안전한 암호화 처리를 위해 암호 알고리즘의 하드웨어화가 절실히 요구되고 있다.

본 논문에서는 스마트카드 칩 설계 시 가장 중요하게 고려되어야 할 칩 면적을 최소화하기 위하여 라운드 키 레지스터를 사용하지 않는 라운드 키 생성 블록과 한 개의 라운드 함수 블록을 반복 사용하는 구조를 이용하였다. SEED의 F함수와 라운드 키 생성에 사용되는 총 5개의 G 함수를 1개의 G함수로 구현하여 순차적으로 이용하도록 하였다. 따라서 본 논문에서 제안한 SEED 프로세서는 1라운드의 동작을 7개의 부분 라운드로 나누고, 클럭마다 하나의 부분 라운드를 수행하는 구조를 갖는다. 제안한 SEED 프로세서는 기능적 시뮬레이션을 통해 한국정보보호진흥원에서 제공한 테스트 벡터와 동일한 결과를 출력됨을 확인하였으며, 합성 및 FPGA 테스트 보드를 이용하여 기존 SEED 프로세서와의 성능을 비교한 결과 면적이 최대 40% 감소하였음을 알 수 있었다.

ABSTRACT

The security of personal informations has been an important issue since the field of smart card applications has been expanded explosively. The security of smart card is based on cryptographic algorithms, which are highly required to be implemented into hardware for higher speed and stronger security. In this paper, a SEED cryptographic processor is designed by employing one round key generation block which generates 16 round keys without key registers and one round function block which is used iteratively. Both the round key generation block and the F function are using only one G function block with one 5×1 MUX sequentially instead of 5 G function blocks. The proposed SEED processor has been implemented such that each round operation is divided into seven sub-rounds and each sub-round is executed per clock. Functional simulation of the proposed cryptographic processor has been executed using the test vectors which are offered by Korea Information Security Agency.

접수일 : 2004년 3월 12일 ; 채택일 : 2004년 8월 27일
† 주저자 : hongmook.choi@samsung.com
‡ 교신저자 : choimy@asic.hanyang.ac.kr

In addition, we have evaluated the proposed SEED processor by executing VHDL synthesis and FPGA board test. The die area of the proposed SEED processor decreases up to approximately 40% compared with the conventional processor.

Keywords : *SEED, Smart Card, Cryptographic Algorithm, Encryption*

Ⅰ. 서 론

스마트카드는 마이크로 프로세서와 메모리를 내장하고 있어서 카드 내에서 정보의 저장과 처리가 가능한 플라스틱 카드를 말한다[1].

스마트카드는 정보 통신 기술이 발전하면서 그 활용 분야가 전자화폐, ID카드, 로열티카드, 교통카드, 의료 카드 등에 이르기까지 다양하다. 또한 스마트카드는 사용자 인증, 접근 제어, 정보의 저장·관리 등의 기능을 수행함에 있어 안전성과 신뢰성 및 보안성을 확보할 수 있는 기반으로 인정되고 있다.

이와 같이 스마트카드의 활용 분야가 다양해지며 급성장하고 있는 것은 사용자와 공급자 모두에게 보안성, 편리성, 다기능성 등의 사용 이점이 있기 때문이다. 그러나 스마트카드가 현재로서는 보안성이 뛰어난 것으로는 인정되고 있으나 해킹될 경우 카드에 저장된 개인 정보와 비밀키까지도 추출이 가능해 IC 칩에 저장된 정보 등에 대한 위·변조는 물론 개인 정보의 악용까지도 초래할 수 있다[2]. 이러한 문제점으로 인해 개인 정보 보안에 대한 문제가 최근 가장 큰 이슈가 되고 있다.

스마트카드의 보안 기술은 카드에 암호 알고리즘을 탑재하는 방식으로 이루어진다. 암호 알고리즘을 소프트웨어로 구현하는 것은 쉽지만, 처리속도가 현격하게 떨어질 뿐만 아니라 해킹 등에 의한 암호 알고리즘의 불법 접근 및 분석도 가능한 문제점이 있다. 즉, 개인의 정보 유출 가능성이 암호화 알고리즘을 하드웨어로 구현했을 경우에 비해 더 높다는 것을 의미한다. 따라서 빠른 속도의 암호화와 보다 안전한 암호화 처리를 위해 암호 알고리즘의 하드웨어화가 절실히 요구되고 있다.

현재의 스마트카드 칩은 대부분 8비트 마이크로 프로세서가 사용되고 있으나 응용 분야의 확대 및 다양한 기능 제공 등으로 인해 점차 16, 32비트 마이크로 프로세서를 필요로 하고 있는 실정이다.

따라서 본 논문에서는 암호화 과정의 처리 속도를 향상시키기 위해 2004년부터 국내의 현금카드에 탑재될 예정이며 점차 사용 분야가 확대될 국내 표준 암호 알고리즘인 SEED를 이용하여 32비트 마이크로 프로세서에 적합한 SEED 프로세서를 설계하였다. 또한 하드웨어의 반복 구조를 이용하여 면적 효율성을 극대화하는 방식으로 스마트카드 칩 설계에 있어 중요하게 고려되어야 할 칩 면적의 최소화 문제를 해결하였다.

Ⅱ. 스마트카드의 구성 및 주요 기술

스마트카드는 그림 1과 같이 CPU, 보안 모듈, EEPROM, ROM, RAM 등으로 구성되어 있다. CPU는 스마트카드 사용시에 해당 연산을 처리하는 장치이며 RNG(Random Number Generator)는 난수발생기로 CPU에 통합되어 구현되는 경우도 있다. 보안 모듈은 하드웨어적으로 구현되지 않는 경우에는 이 기능을 소프트웨어적으로 처리한다. ROM은 스마트카드를 위한 카드 운영체제인 COS (Card Operating System)가 탑재되며, EEPROM은 COS의 일부분과 응용프로그램이 탑재된다. RAM은 응용프로그램을 구동할 경우에 임시 메모리 기능을 한다[3].

스마트카드의 주요 기술로는 카드 칩 설계, COS, 보안, 응용 서비스 기술 4가지를 들 수 있다. 카드 칩 설계 기술이라 함은 데이터 처리를 위한 논리 회로 설계와 집적회로 제조 공정과 관련하여 요구되는 기술을 의미하고, COS 기술은 다양한 응용 서비스를 수용하기 위해 규모가 작으면서도 복잡한 제어 기능과 파일 관리, 보안 기능을 갖는 OS의 개발 기술을 의미한다. 보안 기술은 COS의 보안 특성을 포함

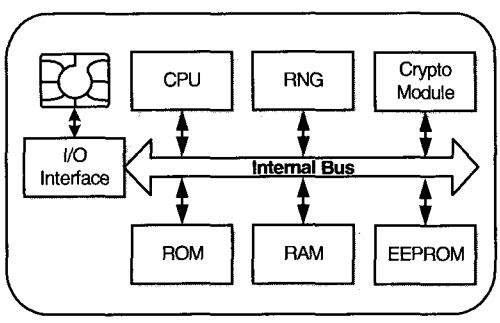


그림 1. 스마트카드의 구성

하며, 소프트웨어와 하드웨어가 조합된 암호학적 측면에서의 안전성과 칩 외부로부터의 허용되지 않는 전기적, 물리적 접근을 막고 내부의 데이터를 보호하는 물리적 측면에서의 안전성을 보장하는 기술을 의미한다. 응용서비스 기술은 카드 단말기와 발행기 등의 인프라 구축과 각 응용 분야에 따른 애플릿 개발 기술 등을 의미한다[4].

카드 칩 설계 기술에 있어서 각 부분들에 대해 공통적으로 소모 전력의 최소화와 칩 면적의 최소화 문제가 고려되어야 한다. 따라서 본 논문에서는 스마트 카드의 중요 구성 요소 중 하나인 보안 모듈의 효율적인 하드웨어 구현에 대하여 기술하였다.

Ⅲ. SEED 알고리즘

1. SEED 알고리즘의 전체 구조

SEED 알고리즘은 128비트의 평문 블록과 128비트 키로부터 생성된 16개의 64비트 라운드 키를 입력으로 사용하여 총 16라운드를 거쳐 128비트 암호문 블록을 출력한다. SEED 알고리즘의 전체 구조는 그림 2와 같이 128비트 입력 평문 블록을 2개의 64비트 블록 (L0(64), R0(64))으로 나누어, 16개의 64비트 라운드 키를 이용하여 16라운드를 수행한 후, 최종 128비트 암호문 블록 (L16(64), R16(64))을 출력한다.

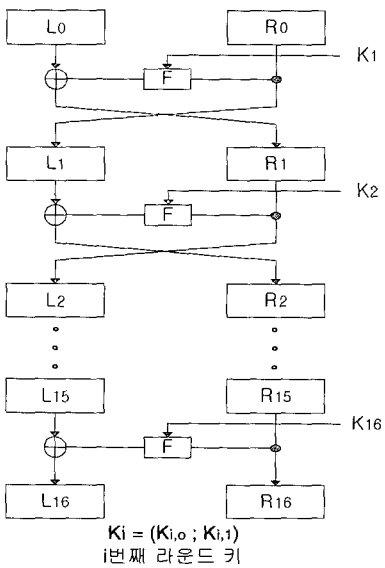


그림 2. SEED 전체 구조

2. F 함수

SEED의 F 함수는 그림 3과 같이 수정된 64비트 feistel 형태로 구성된다. F 함수는 32비트 블록 2개 (C, D)를 입력으로 받아, 32비트 블록 2개(C′, D′)를 출력한다. 즉, 암호화 과정에서 64비트 블록(C, D)과 64비트 라운드 키 $K_i=(K_{i,0};K_{i,1})$를 F 함수의 입력으로 처리하여 64비트 블록(C′, D′)을 출력한다. (i: 라운드 수, ⊕: XOR, $a \boxplus b$: $(a+b) \bmod 2^{32}$)

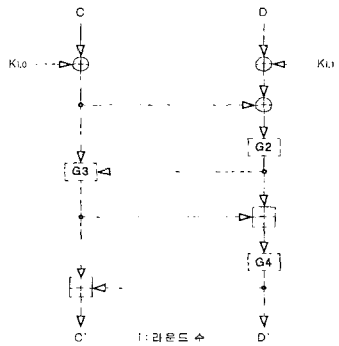


그림 3. F 함수 구조

3. G 함수

G 함수는 그림 4와 같으며 다음과 같이 기술된다.

$$Y_3=S_2(X_3),\ Y_2=S_1(X_2),\ Y_1=S_2(X_1),\ Y_0=S_1(X_0),$$

$$\begin{aligned}
Z_3&=(Y_0\&m_3)\oplus(Y_1\&m_0)\oplus(Y_2\&m_1)\oplus(Y_3\&m_2)\\
Z_2&=(Y_0\&m_2)\oplus(Y_1\&m_3)\oplus(Y_2\&m_0)\oplus(Y_3\&m_1)\\
Z_1&=(Y_0\&m_1)\oplus(Y_1\&m_2)\oplus(Y_2\&m_3)\oplus(Y_3\&m_0)\\
Z_0&=(Y_0\&m_0)\oplus(Y_1\&m_1)\oplus(Y_2\&m_2)\oplus(Y_3\&m_3)
\end{aligned}$$

$$(m_0=0\text{xfc},\ m_1=0\text{xf3},\ m_2=0\text{xcf},\ m_3=0\text{x3f})$$

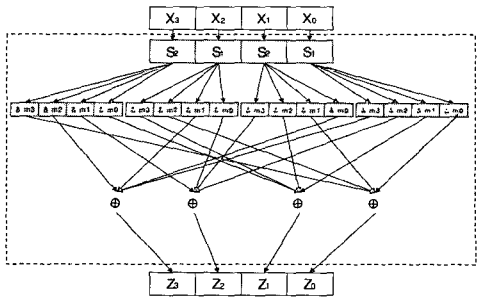


그림 4. G 함수

4. 라운드 키 생성 과정

SEED의 라운드 키는 128비트 입력키를 64비트씩 좌우로 나누어 이것을 교대로 8비트씩 좌/우로 회전 이동한 후, 간단한 산술 연산과 G 함수를 적용하여 라운드 키를 생성한다.

라운드 키 생성 과정은 그림 5와 같다. 주어진 128비트 암호키 K=A||B||C||D를 32비트 레지스터 A, B, C, D로 나눈 후 각 라운드 i에 사용되는 라운드 키 Ki는 다음과 같은 방식으로 생성한다.

```
for( i=1; i<=16; i++) {
  Ki,0 ← G(A+C-KCi);
  Ki,1 ← G(B-D+KCi);
  if( i%2==1 ) A||B ← (A||B)>>8;
  else C||D ← (C||D)<<8;
}
```

위에서 (B||A)〉〉8, (D||C)〈〈8는 각각 64비트 수를 오른쪽/왼쪽으로 8비트 회전 이동시킨 것이다[5].

라운드 키 생성 과정에 사용된 상수는 [5]를 참조한다.

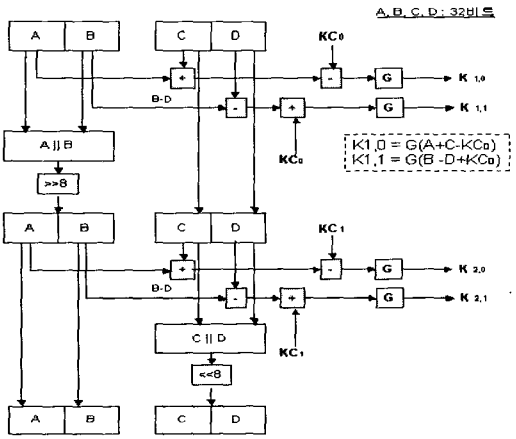


그림 5. 라운드 키 생성 과정

Ⅳ. 제안한 SEED 프로세서

본 논문에서 제안한 SEED 프로세서는 32비트 마이크로 프로세서에 적합하도록 설계하였으며, 그림 6과 같이 6개의 블록으로 구성되어 있다.

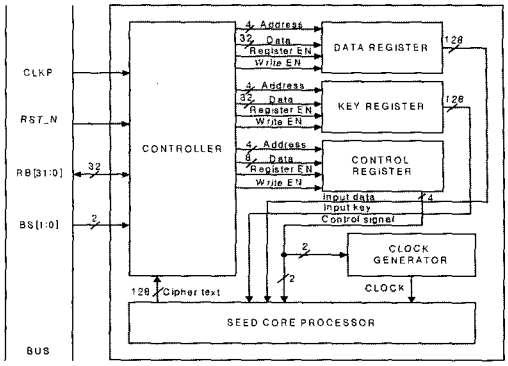


그림 6. 제안한 SEED 프로세서의 전체 구조

1. 컨트롤러

컨트롤러는 SEED 코어 프로세서 블록의 암·복호화 과정 수행에 필요한 데이터를 생성하기 위해 RB와 BS를 입력으로 받아 그 입력에 따라 어드레스, 데이터, 레지스터 이네이블(enable) 신호와 쓰기 허용(write enable) 신호를 각 레지스터에 출력하는 기능을 하는 블록이다.

여기서 RB는 BS에 따라 어드레스 또는 데이터로 사용되며 입출력으로 사용된다. SEED 프로세서에서 사용하는 입출력 신호는 표 1과 같다. 표 2에서 볼 수 있듯이 RB는 BS가 "00"일 경우는 데이터, "01"일 경우는 읽기/쓰기 및 어드레스 신호로 인식되며, "11"일 경우는 버스가 유휴 상태가 된다.

표 1. 제안한 SEED 프로세서의 입출력 신호

항 목	I/O	내 용
CLKP	I	클럭
RST_N	I	RST_N = '0' → 리셋
RB[31:0]	I/O	읽기/쓰기(R/W) 신호 어드레스/데이터 버스
BS[1:0]	I	"00" : 데이터 입력 "01" : 읽기/쓰기(R/W) 및 어드레스 입력 "10" : 사용안함 "11" : Bus idle

표 2. SEED의 테스트 벡터

입력키	00000000000000000000000000000000
평 문	000102030405060708090A0B0C0D0E0F
암호문	5EBAC6E0054E166819AFF1CC6D346CDB

컨트롤러는 어드레스(BS가 01일 경우의 RB 신호)가 입력으로 들어올 경우 그 어드레스에 해당하는 레지스터를 활성화시키기 위해 레지스터 이네이블 신호를 출력한다. 또한 RB의 읽기/쓰기 신호(RB[14:13])에 따라 레지스터의 데이터를 읽고 쓰는 것이 가능하다. SEED 프로세서에서 사용하지 않는 어드레스가 액세스할 경우 각 레지스터를 활성화시키지 않아 데이터를 읽고 쓸 수 없다.

2. 데이터, 키, 컨트롤 레지스터

데이터 및 키 레지스터는 컨트롤러로부터 입력된 32비트의 데이터 및 키를 이용하여 128비트의 데이터 및 키를 SEED 코어 프로세서 블록에 제공하며, 컨트롤 레지스터는 SEED의 시작 신호, 암·복호화 모드 선택 신호, 클럭 분주 제어 신호를 생성한다.

컨트롤 레지스터의 암·복호화 시작 신호는 컨트롤 레지스터의 입력 Data[0]이고, 모드 선택 신호는 Data[1], 클럭 분주 제어 신호 Data[4:3]이다. 암·복호화 모드 선택 신호는 '0'일 경우는 암호화, '1'일 경우는 복호화를 수행한다. 암·복호화 시작 신호는 '1'일 경우에 암·복호화를 시작한다.

3. 클럭 생성기

클럭 생성기는 컨트롤 레지스터에서 생성된 클럭 분주 제어 신호를 이용하여 클럭을 2분주, 4분주, 정지하는 기능을 수행하는 블록으로 제어 신호가 "00"일 경우에는 클럭 정지 기능을 수행하며, "01"일 때는 기준 클럭, "10"일 때는 2분주, "11"일 때는 4분주 클럭을 SEED 코어 프로세서 블록에 제공한다.

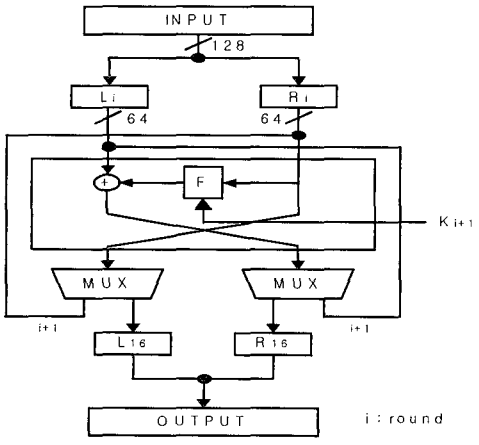


그림 7. 제안한 SEED 코어 프로세서의 전체 구조

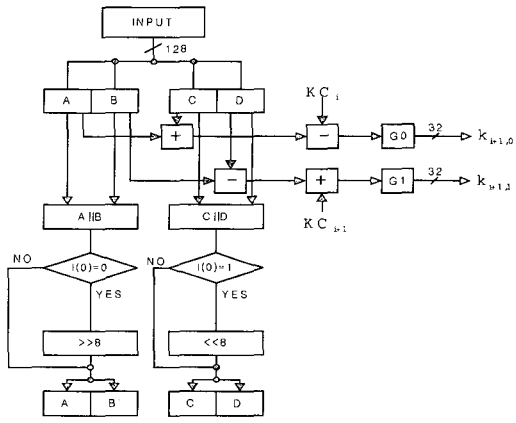


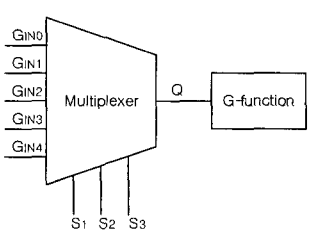
그림 8. 제안한 SEED 코어 프로세서의 라운드 키 생성 블록

4. SEED 코어 프로세서

제안한 SEED 코어 프로세서는 그림 7과 같이 한 라운드의 하드웨어 구조를 반복하여 이용하도록 설계하였다. 또한 라운드 키 생성 과정도 그림 8과 같이 반복 구조로 설계하여 하드웨어의 면적 효율성을 높일 수 있도록 하였다. 또한 제안한 SEED 프로세서는 라운드 키가 라운드의 동작이 수행될 때 동시에 생성되므로 라운드 키의 저장을 위한 별도의 레지스터가 필요하지 않다.

이 외에도 면적 효율성을 높이기 위해 5개의 G 함수를 하나의 G 함수로 공유해서 사용할 수 있도록 설계하였다. SEED는 그림 3과 8에서와 같이 총 5개의 G 함수를 사용한다. 이 5개의 G 함수를 그림 9에서 보는 바와 같이 멀티플렉서를 이용하여 하나의 G 함수로 공유하는 방식을 채택하였다.

그림 10은 SEED 코어 프로세서의 상태 천이도를 나타낸다. SEED 코어 프로세서가 동작하지 않



선택 신호 S3 S2 S1	Q
0 0 0	GIN0
0 0 1	GIN1
0 1 0	GIN2
0 1 1	GIN3
기 타	GIN4

그림 9. G함수의 공유

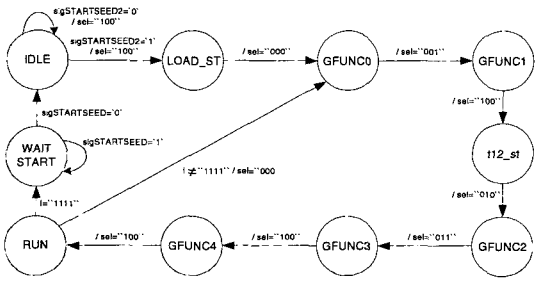


그림 10. 제안한 SEED 코어 프로세서의 상태 천이도

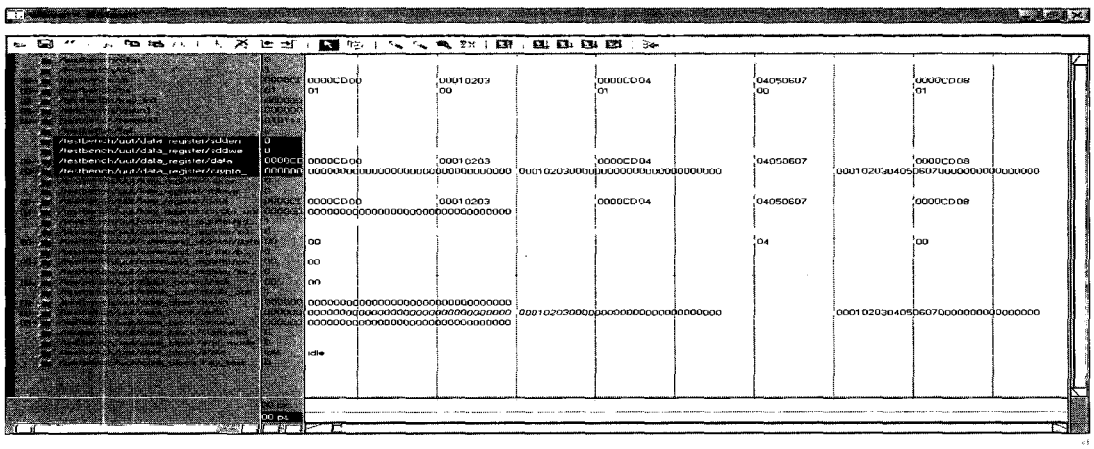
을 때는 IDLE 상태를 유지하게된다. START-SEED 신호 '1'이 입력된 후 두 클럭 후에 sig-STARTSEED2 신호는 '1'이 되고 IDLE 상태에서 LOAD_ST 상태로 천이된다. LOAD_ST 상태에서는 KEY_TEXT_LOAD 신호가 '1'이 되어 평문/암호문과 키를 읽어들이게 된다. 이 상태에서 GFUNC0 상태로 천이되며, sel 신호는 "000"이 된다. GFUNC0 상태에서는 sel 신호가 "000"이므로 GOUT0_enable='1'이 되고, GIN0이 선택되어 GOUT0이 계산되고 GFUNC1 상태로 천이된다. GFUNC1 상태 또한 GFUNC0 상태에서와 같은 동작이 이루어져 GOUT1이 계산되고 t12_st 상태로 천이된다. t12_st 상태에서는 t12_enable ='1'이 되어 F함수의 라운드 키와 C, D가 XOR 연산하는 부분의 동작이 수행된다. 여기서 GFU-NC2 상태로 천이되며 GFUNC4 상태까지의 동작은 GFUNC1의 상태에서와 동일하게 수행된다. GFUNC4 상태에서는 RUN 상태로 천이된다. RUN 상태에서는 I가 "1111"이 아닐 경우에는 GFUNC0 상태로 천이되어 위의 동작이 동일하게 반복된다. 즉, 16라운드 과정을 모두 수행할 때까지 위의 과정을 반복하게 된다. 16라운드 과정을 모두 수행하여 I="1111"이 되면 WAIT_START 상태로 천이된다. WAIT_START는 SEED 프로세서의 반복 수행을 방지하기 위한 상태로 6클럭이후에 IDLE 상태로 천이된다.

V. 시뮬레이션 및 합성 결과

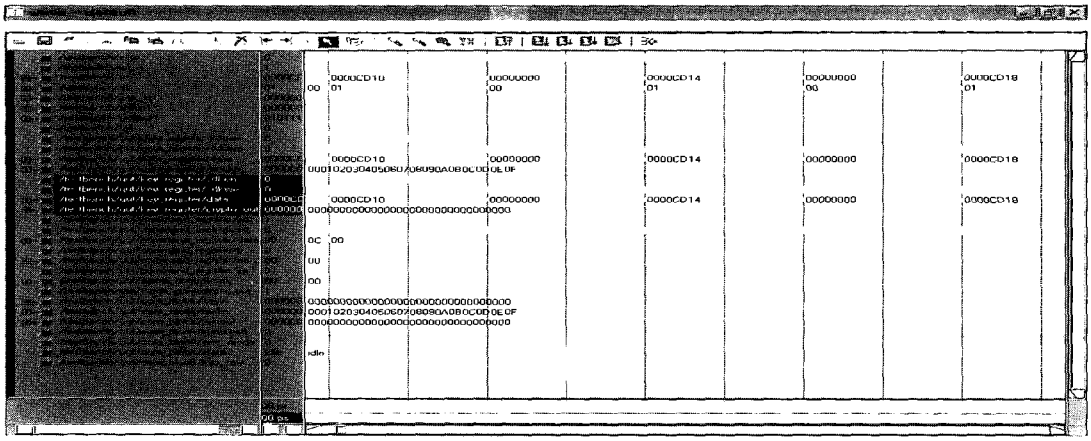
1. 시뮬레이션 결과

그림 11과 12는 32비트 데이터 버스를 이용한 SEED 프로세서의 레지스터, SEED 코어 프로세서의 기능적 시뮬레이션을 수행한 결과 파형이다. 표 2는 한국정보보호진흥원에서 제공한 테스트 벡터[6]이다.

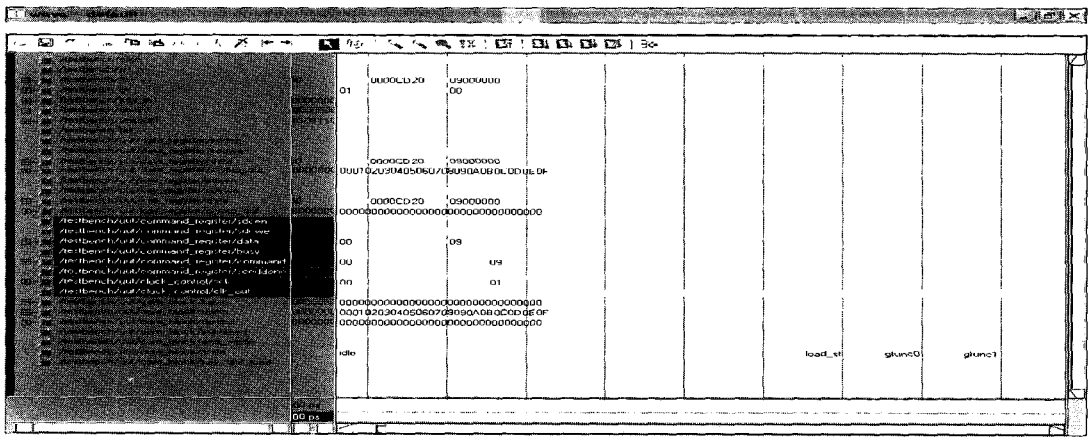
그림 11의 (a)는 0D00번지에 "00010203", 0D04번지에 "04050607", 0D08번지에 "08090A0B", 0D0C번지에 "0C0D0E0F"를 각각 써서 데이터 레지스터에 "000102030405060708090A0B0C0D0E0F"의 평문을 쓰는 과정을 나타내고 있다. (b)는 0D10번지, 0D14번지, 0D18번지와 0D1C번지에 각각 "00000000"을 써서 키 레지스터에 입력키 "000000-00000000000000000000000000000000"를 쓰는 과정을 나타낸다. (c)는 0D20번지의 컨트롤 레지스터에 SEED 제어 신호("00001001": 기준 클럭 사용, 암호화 모드, SEED 시작)를 쓰는 과정으로 클럭 생성기에 기준 클럭 사용 신호 "01"이 입력되어 기준 클럭이 생성되는 과정을 나타낸다. (d)는 라운드 키 생



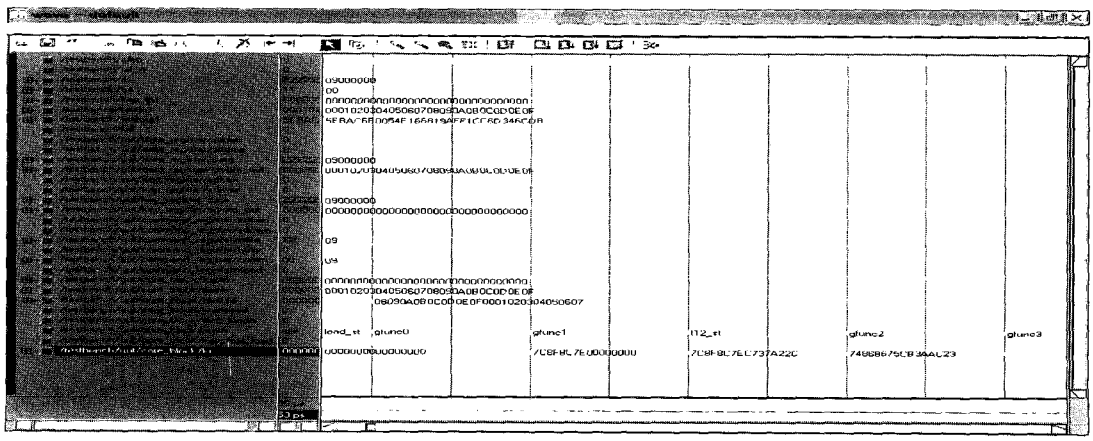
(a) 데이터 레지스터에 평문 쓰기



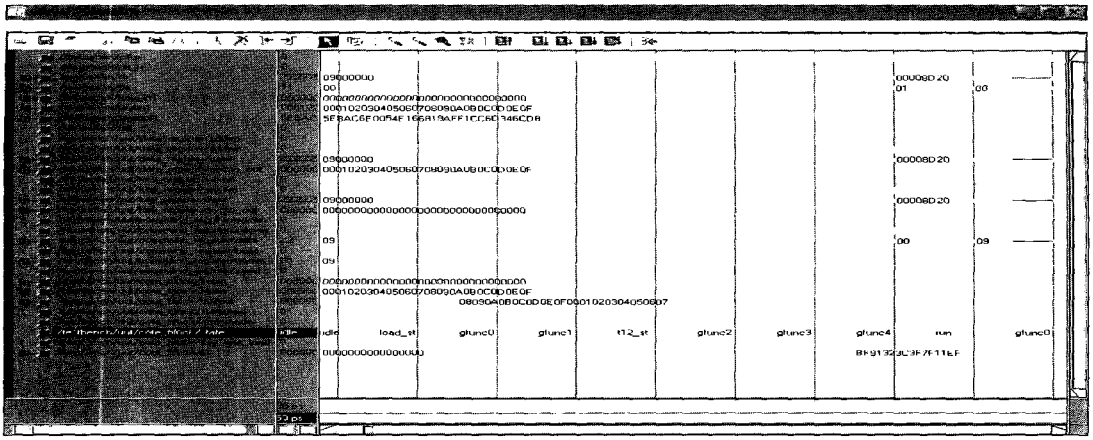
(b) 키 레지스터에 입력키 쓰기



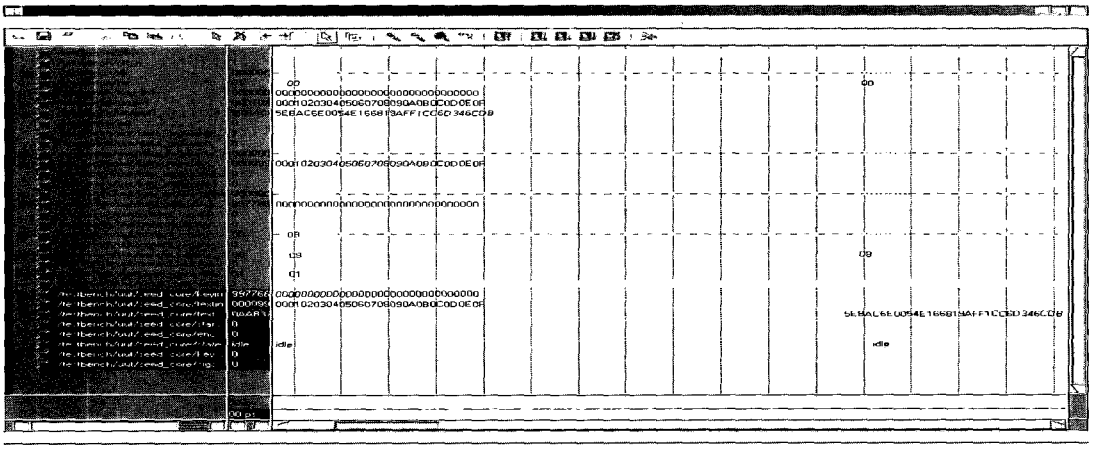
(c) 컨트롤 레지스터에 SEED 제어 신호 쓰기 (암호화 모드)



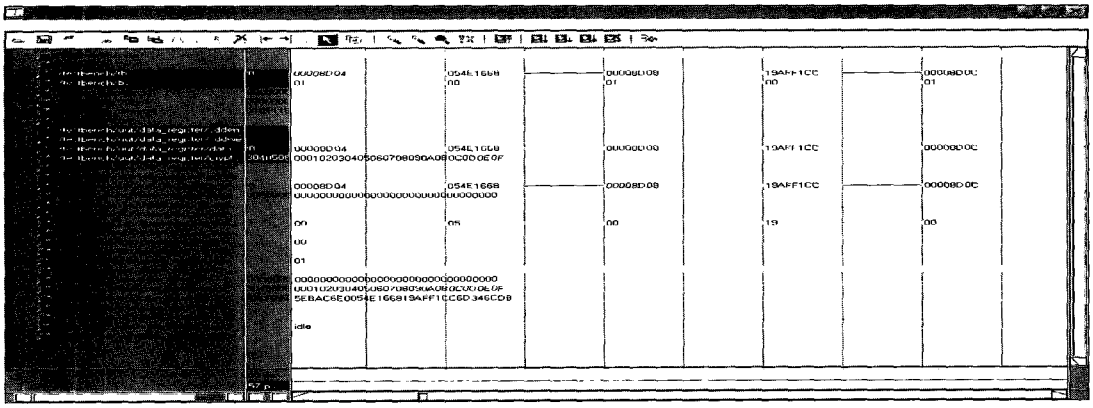
(d) 라운드 키 생성



(e) 1라운드 처리 과정



(f) 제안한 SEED 코어 프로세서의 암호화 과정



(g) 데이터 레지스터에서 암호문 읽기

그림 11. 제안한 SEED 프로세서의 암호화 과정

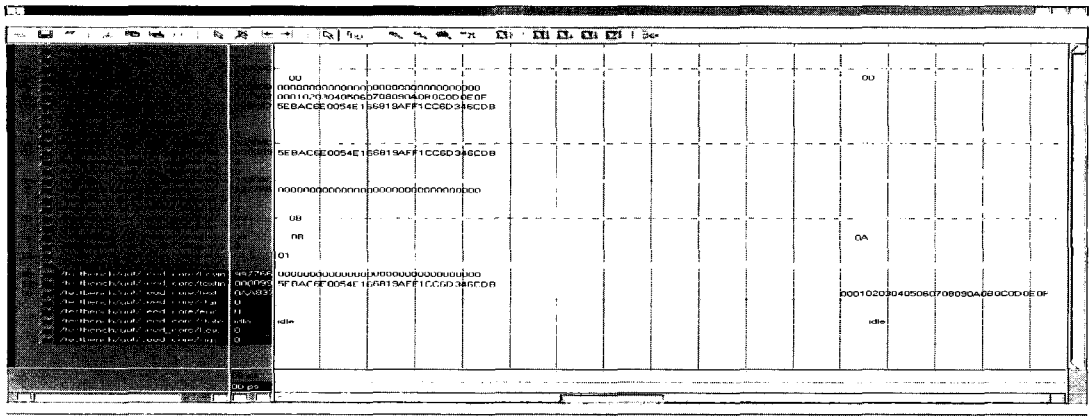


그림 12. 제안한 SEED 코어 프로세서의 복호화 과정

성 과정을 나타낸 것으로 G함수를 공유하여 GFUNC0와 GFUNC1 상태를 거쳐 t12_st 상태에서 라운드 키가 생성됨을 알 수 있다. 이 과정을 통해 1라운드 키가 테스트 벡터의 "7C8F8C7EC737A22C"와 동일하게 생성됨을 알 수 있다. (e)는 1라운드가 총 7개의 부분 라운드로 이루어져 있음을 나타낸다. 즉, 제안한 SEED 프로세서는 112개(16×7)의 부분 라운드를 거쳐 암/복호화 과정을 수행함을 알 수 있다. (f)는 SEED 코어 프로세서가 16라운드 과정을 수행하여 "5EBAC6E0054E166819AFF1CC6D346CDB"의 암호문을 출력하는 과정, (g)는 암호화 과정이 모두 수행된 후 데이터 레지스터에서 암호문을 읽는 과정을 나타낸 것으로 0D00번지에서 "5EBAC6E0", 0D04번지에서 "054E1668", 0D07번지에서 "19AFF1CC", 0D0A번지에서 "6D346CDB"을 읽어 암호화 과정이 정상적으로 수행됨을 알 수 있다. 그림 12는 위의 과정과는 반대로 복호화 과정을 수행한 결과를 나타낸다. 모든 과정이 동일하므로 SEED 코어 프로세서에서의 복호화 과정만 나타냈다. 그림 12는 SEED 코어 프로세서의 복호화 과정을 수행한 결과로 평문 "000102030405060708090A0B0C0D0E0F"가 출력됨을 확인할 수 있다. 이 시뮬레이션 결과는 표 2와 동일한 결과로 SEED 프로세서가 정상적으로 동작함을 확인할 수 있다.

또한 위의 결과 이외에도 다양한 테스트 벡터를 이용하여 본 논문에서 제안한 SEED 프로세서를 하기 위해 한국정보보호진흥원에서 제공한 C언어 코드와 [7]에서 제공한 테스트 벡터를 이용하였다.

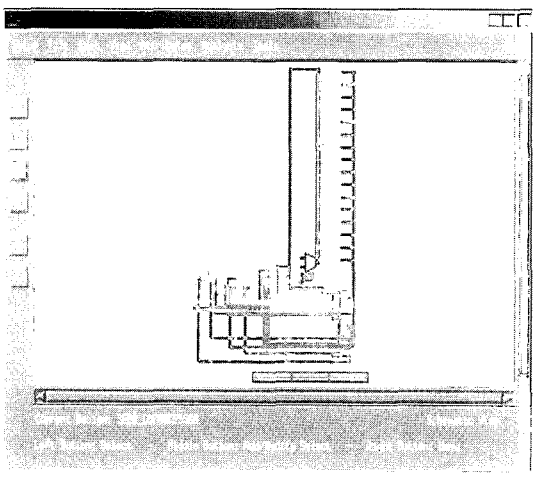


그림 13. 제안한 SEED 프로세서의 합성 결과

2. 합성 결과

그림 13은 Fujitsu uc_core_66 라이브러리를 이용하여 32비트 데이터 버스용 SEED 프로세서를 합성한 결과를 RTL schematic으로 표현한 것이다. 표 3은 각 블록별로 소요된 게이트 수를 나타낸 것이다.

표 3. 제안한 SEED 프로세서의 블록별 게이트 수

블 록	게이트 수
Controller	439
Data Register	928
Key Register	928
Control Register	67
Clock Generator	17
SEED Core Processor	8,094
Top (SEED Coprocessor)	10,610

Ⅵ. FPGA 검증 및 성능 비교

1. FPGA를 이용한 검증

본 논문에서 제안한 SEED 프로세서는 VHDL을 이용하여 설계하였다. ModelSim을 이용하여 시뮬레이션 검증을 하였고 그림 14와 같이 FGPA 테스트 보드를 이용하여 테스트를 완료하였다. 테스트 벡터를 FPGA에 입력으로 주기 위해 RS-232 통신과 89C51을 이용하였으며 암·복호화 결과를 LCD에 표시되게 함으로써 검증하였다.

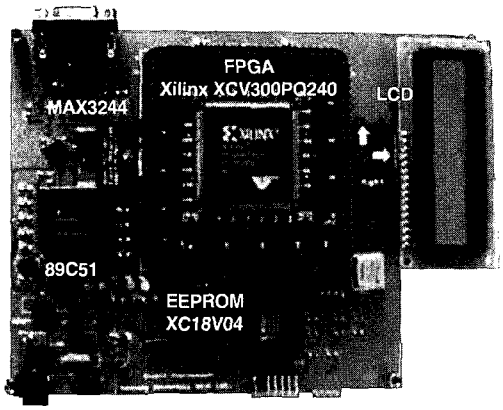


그림 14. FPGA를 이용한 테스트 보드

2. 성능 비교

일반적으로 스마트카드는 13.56MHz로 동작하므로 본 논문에서 제안한 SEED 프로세서는 40MHz 정도를 목표로 설정하여 합성하였다. 또한 라이브러리는 Fujitsu cs66_uc_core를 사용하였다. 합성 결과 최장 지연 패스는 24.98ns로서 최대 동작 주파수는 40.03MHz이며, 총 게이트 수는 약 10,610이다. 시뮬레이션 결과에서 알 수 있듯이 128비트의 데이터를 입력하는데 8 클럭, 128비트의 키를 입력하는데 8 클럭, 컨트롤 데이터를 입력하는데 2 클럭, 암·복호화 과정을 처리하는데 113 클럭(데이터와 키 로드, 각 라운드 당 7 클럭), WAIT_START 상태 6 클럭, 암·복호화 처리 결과를 읽는데 8 클럭 총 145 클럭이 소요되어 총 소요 시간은 145÷(40.03×106)으로 3.622㎲이다. 따라서 처리속도는 35.34Mbps이다.

기존의 SEED 프로세서 [8]은 5개의 G 함수를 1개로 공유하여 사용하였으나 라운드 키 생성을 위해 별도의 레지스터를 사용하였다. 즉, 암·복호화 과정을 수행하기 전에 라운드 키를 생성하여 레지스터에 저장하게 된다.

이 방법은 레지스터가 추가되어 칩의 면적이 커질 뿐만 아니라 암·복호화 이전에 라운드 키를 생성하기 때문에 암·복호화시 소요되는 클럭 수가 증가하게 된다. [8]에는 게이트 수가 언급되어 있지 않으므로 임의로 라운드 키를 저장하기 위한 레지스터를 설계하여 그 결과 값을 예측하였다. 그 결과 [8]은 약 16,770게이트 정도이다. 또한 G함수를 [9]에서는 2개, [10]에서는 3개를 사용하여 칩 면적은 증가하게 된다. 각 논문에서 제안한 SEED 프로세서를 합성하는데 있어서의 제약 조건이 다르므로 본 논문과 동일한 조건으로 합성을 한 결과 G함수 하나의 면적은 약 3,500게이트 정도이다. 따라서 본 논문과 동일한 조건임을 가정한다면 [9]는 약 14,110 게이트, [10]은 17,610게이트 정도의 면적이 될 것으로 예상된다.

표 4. SEED 프로세서의 성능 분석

성능 분석 항목	[8]	[9]	[10]	제 안
(최대) 동작 주파수	5㎒	100㎒	97.1㎒	40.03㎒
소요 클럭 수	244	-	50	145
소요 시간	48.8㎲	-	-	3.622㎲
처리 속도[Mbps]	2.62	237	258.9	35.34
게이트 수	16,770	14,110	17,610	10,610

표 4는 본 논문에서 제안한 SEED 프로세서와 기존의 SEED 프로세서의 성능을 분석한 표이다. 그 결과 제안한 SEED 프로세서는 [8]보다는 속도와 면적 모든 면에서 우수한 성능을 보였으며, [9], [10]에 비해서는 속도는 느린 반면 면적에서 우수한 성능을 보였다. 제안한 SEED 프로세서는 [8]에 비해 37%, [9]에 비해 약 25%, [10]에 비해 약 40% 정도 감소하였다. 단 [8]의 주파수는 최대 동작 주파수가 아닌 [8]에 기재된 동작 주파수이다.

Ⅶ. 결 론

본 논문에서는 스마트카드의 활용 분야가 점차 다양해지며 개인의 보안 문제가 중요한 이슈가 됨에 따라 스마트카드의 보안 기능을 강화하고자 국내 표준 암호 알고리즘인 SEED를 VHDL을 이용하여 하드

웨어로 설계하였다. 32비트 마이크로 프로세서에 적합한 SEED 프로세서를 설계하여 암호화 과정의 처리 속도를 향상시켰으며, 스마트카드 칩 설계시 중요하게 고려되어야 할 칩 면적의 최소화 문제를 해결하기 위해 하드웨어의 반복 구조를 이용하여 칩 면적의 효율성을 극대화하였다.

제안한 SEED 프로세서는 한국정보보호진흥원에서 제공한 테스트 벡터를 이용하여 기능적, 타이밍 시뮬레이션을 통해 정상적인 동작이 이루어짐을 확인하였다. 또한 합성 및 FPGA 검증을 통해 기존 SEED 프로세서와의 성능을 비교한 결과 본 논문에서 제안한 SEED 프로세서는 기존의 SEED 프로세서보다 칩 면적이 최대 약 40% 감소하였다. 스마트카드는 칩 면적을 가장 중요하게 고려해야 하므로 본 논문에서 제안한 SEED 프로세서가 적합할 것으로 판단된다.

참 고 문 헌

[1] 박천교, 이윤철, "국내외 스마트카드 기술 및 시장 동향," 전자통신동향분석, 16(5), pp. 77-84, 2001.

[2] 이성은, 장홍종, 박인재, 한선영, "다중 암호화 기법을 활용한 하이브리드 스마트카드 구현," 정보보호학회논문지, 13(2), April 2003.

[3] 이기한, 스마트카드 칩 기술, TTA 저널, 90, pp. 69-74, 2003.

[4] 김승철, 김원종, 조한진, 정교일, "32 비트 저전력 스마트카드 IC 설계," 대한전자공학회 하계종합학술대회 논문집, 25(1) pp. 349-352, 2002.

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

[6] Korea Information Security Agency, Test vectors of SEED, April 1999.

[7] 김역, 정창호, 장윤석, 이상진, 이성재, "SEED 구현 적합성 검증 시스템에 관한 연구," 정보보호학회논문지, 13(1), pp.69-85, 2003.

[8] Y. H. Seo, J. H. Kim, and D. W. Kim, "Hardware Implementation of 128-bit Symmetric Cipher SEED," APASIC 2000, Proceedings of the Second IEEE Asia Pacific Conference on ASIC, pp. 183-186, 2000.

[9] 최병윤, 서정욱, "SEED 알고리즘용 암호 보조 프로세서의 설계," 한국통신학회논문지, 25(9), pp. 1609-1616, 2000.

[10] 전신우, 정용진, "128비트 SEED 암호 알고리즘의 고속처리를 위한 하드웨어 구현," 정보보호학회논문지, 11(1), pp. 13-23, 2001.

〈著者紹介〉

최 홍 묵 (Hong-mook Choi) 정회원
2002년 2월 : 한양대학교 전자공학과 졸업
2004년 2월 : 한양대학교 전자전기제어계측공학 공학석사
2004년 1월~현재 : 삼성전자 SOC연구소 연구원
〈관심분야〉 암호화 프로세서, 멀티미디어 컨텐츠 보호 시스템, 스마트 카드 응용, ASIC

최 명 렬 (Mryung-ryul Choi) 정회원
1983년 2월 : 한양대학교 전자공학과 졸업
1985년 12월 : 미시간주립대학교 컴퓨터공학 공학석사
1991년 12월 : 미시간주립대학교 컴퓨터공학 공학박사
1991년 3월~1991년 10월 : 생산기술연구원 전자정보실용화센터 조교수
1991년 11월~1992년 8월 : 생산기술연구원산하 전자부품종합기술연구소 선임연구원
1992년 9월~현재 : 한양대학교 전자컴퓨터공학부 교수
〈관심분야〉 ASIC, 3D 디스플레이, RFID, 스마트카드 응용, ITS, EFC