보안공학연구논문지 *Journal of Security Engineering*
Vol.11, No.1 (2014), pp.123-136
http://dx.doi.org/10.14257/jse.2014.02.08

Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현

강민섭[1], 장태민[2], 김현수[3]

Design and Implementation of High-speed Cryptoprocessor Using Pipeline Technique

Min-Sup Kang[1], Tae-Min Chang[2], Hyean-Soo Kim[3]

요 약

본 논문에서는 pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현에 관하여 기술한다. 암호화를 위한 알고리듬은 DES/3DES, SEED, 그리고 AES를 사용하고 인증을 위한 알고리듬은 HMAC-SHA-1을 이용한다. 또한, 기존 방식 (Iterative 방식)과 제안된 방식 (Pipeline 방식) 에 대한 성능 비교 결과가 제시된다. 제안된 암호 프로세서는 VHDL을 사용하여 구조적 모델링을 행하였으며, Xilinx사의 ISE 6.2i 툴을 이용하여 논리 합성을 수행하였다. 설계 검증을 위해 Modelsim을 이용하여 타이밍 시뮬레이션을 수행하여, 설계된 시스템이 정확히 동작함을 확인하였다.

핵심어 : DSES, 3DES, SEED, AES 암호 프로세서, HMAC-SHA-1, VHDL

Abstract

This paper presents the design and implementation of high-speed cryptoproceesor based on pipeline technique. In this approach, four cipher algorithms of DES/3DES, SEED, and AES are used for data encryption, and a HMAC-SHA-1 algorithm is also used for user authentication. Also, the results of performance comparison are presented for two method of Iterative and pipelined method. The designed cryptosystem was described using Verilog HDL, and it was synthesized for a Xilinx VirtexE FPGA device, using the ISE 6.2i software tools. Also, timing verification of the system is performed by using Mentor Graphics' ModelSim.

Keywords : DSES, 3DES, SEED, AES cipher processor, HMAC-SHA-1, VHDL

접수일(2014년01월16일), 심사의뢰일(2014년01월17일), 심사완료일(1차:2014년01월28일, 2차:2014년02월07일)
게재일(2014년02월28일)

[1](교신저자) 430-714 경기도 안양시 만안구 안양5동, 안양대학교 컴퓨터공학과.
email: mskang@anyang.ac.kr
[2]430-714 경기도 안양시 만안구 안양5동, 안양대학교 컴퓨터공학과.
email: tmchang@anyang.ac.kr
[3]430-714 경기도 안양시 만안구 안양5동, 안양대학교 컴퓨터공학과.
email: hskim@anyang.ac.kr

ISSN: 1738-7531 JSE

1. 서론

네트워크의 이용확대와 함께 기밀성의 높은 데이터 통신에 대한 수요가 증가함에 따라 정보의 흐름을 통제하기가 대단히 어렵기 때문에 내부의 중요한 자원을 인터넷으로부터 보호해 줄 수 있는 인터넷 보안이 가장 심각한 문제로 대두되고 있다[1-3]. 이러한 컴퓨터 네트워크 환경에서는 수없이 많은 데이터 교환이 필요하며, 정보의 안전성과 신뢰성을 보장하기 위한 수단으로서 암호가 적용되고 있다.

암호(cryptography)란 일반적인 평문을 해독 불가능한 암호문으로 변형하거나 또는 암호화된 통신문을 복원 가능한 형태로 변환하기 위한 원리, 수단, 방법 등을 취급하는 기술 이라할 수 있다. 암호 시스템은 대칭형(symmetric) 암호시스템과 비대칭형(asymmetric) 암호시스템으로 분류할 수 있다[1]. 대칭형 암호 알고리듬은 암호화키와 복호화 키가 동일한 암호 알고리듬을 말하며, DES, 3DES(Triple Data Encryption Standard), SEED, AES(Advanced Encryption Standard)등이 있다[2-4]. 비대칭형 암호 은 암호화키와 복호화 키가 동일하지 않은 암호 알고리듬을 말하고, RSA (Rivest-Shamir-Adleman), ECC등이 이에 속한다[5-7].

이러한 암호 시스템은 소프트웨어를 이용하여 쉽게 구현이 가능 하지만 실시간 응용을 위해서는 적합하지 않다. 따라서 보다 향상된 성능 및 안전성을 제공하기 위해 암호 시스템의 하드웨어 구현은 바람직하다. 하드웨어 구현 시 성능의 설계 기법에 따라 크게 차이난다. 특히 암호 시스템과 같이 알고리듬 내에 동일한 동작을 하는 블록이 존재하는 경우에는 이 블록을 설계하고 운영하는 기법에 따라 성능에 큰 영향을 미친다[1].

본 논문에서는 Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현에 관하여 기술한다. 암호화를 위한 알고리듬은 DES/3DES, SEED, 그리고 AES를 사용하고 인증을 위한 알고리듬은 HMAC-SHA-1을 이용한다.

2. 암호 알고리듬

2.1 DES/3DES 알고리듬

DES는 암호화, 복호화 알고리즘이 대칭적이며 치환(permutation)과 대치(substitution) 그리고 S_box로 구성된 블록 암호화 시스템이다[2]. DES 암호화 과정에는 암호화하고자 하는 평문(Plaintext : P)과 키(k1...k16)의 두 가지 입력이 들어간다. 64 비트의 평문 블록은 초기 치환(Initial Permutation : IP)후에 32 비트씩 좌(L0), 우(R0)부분으로 나뉘게 되며 그 다음 16라운드의 계산을 거치게 된다. 16라운드 후에는 오른쪽(R16)과 왼쪽(L16) 부분이 합쳐져서 역 초기 치환(Inverse Initial Permutation: IP-1)을 거침으로써 암호문(ciphertext)이 생성된다. 본질적으로 대치된 64 비트 입력은 16라운드를 거치며 매 라운드의 결과로 64 비트의 중간 값을 생성한다. 각 64 비트

중간 값의 좌우 절반은 분리된 32 비트 값으로 취급되며 L(왼쪽)과 R(오른쪽)로 분류된다. 각 라운드의 전체적인 처리는 다음 공식으로 요약된다[1, 2].

Li = Ri-1
Ri = Li-1 ⊕ f(Ri-1, ki)

여기서 ⊕은 비트 단위 XOR함수를 의미한다. 따라서 각 라운드의 좌측 출력(Li)은 단순히 그 라운드의 우측 입력(Ri-1)과 같다. 우측 출력(Ri)은 Li-1을 Ri-1과 ki의 복합함수 f에 XOR한 결과이다.

입력되는 킷값 64 비트에서 패리트 비트를 제거한 56 비트 킷값은 C0와 D0로 표시된 두 개의 28 비트 값으로 취급된다[2]. 각 라운드 과정에서 Ci와 Di는 1 또는 2 비트씩 순환 좌측 이동하게 된다. 이동된 값들은 다음 라운드 반복 과정의 입력으로 사용한다. 이들은 또한 순열선택2의 입력으로 사용되어, 함수 f(Ri-1, ki)의 입력이 되는 48 비트 출력을 생성한다. 키 ki는 48 비트이며 Ri-1 입력은 32 비트이다. Ri-1입력은 다음 라운드의 Li에 입력되고, 동시에 확장순열을 이용하여 48 비트로 확장된다. 이 과정의 결과인 48 비트는 ki와 XOR된다. 이 48 비트 결과는 32 비트 출력을 생성하는 치환함수(S_box)를 통과하고, 다시 재배치 치환(P)을 거친 32 비트와 Li-1에 입력된 32 비트와 XOR하여 다음 라운드의 Ri에 입력된다.

DES에서 복호화 과정은 암호화 알고리듬과 비슷하기 때문에 구현이 간단하다. 차이가 있다면 단지 반대의 순서로 계산을 행한다는 것뿐이다. 이것은 암호화를 위한 각 라운드의 키가 K1, K2, K3, . . . K16이라면 복호화를 위한 키는 K16, K15, K14, . . . K1이 되는 것이다.

3DES는 두 개의 암호 키를 사용하여 첫 번째 키로 암호화하고 다시 두 번째 키로 복호화한 다음 또 다시 첫 번째 키로 암호화하면 강한 암호를 얻을 수 있다. 3DES는 평문 64 비트와 킷값 56 × 2(k1,i k2,i) = 112 비트가 이용된다. 킷값은 k1,i, k2,i에 각각 64 비트씩 128 비트를 받아들여 패리트 비트를 제외한 56 비트씩 112 비트가 적용된다.

암호화시에 어떤 평문 64 비트와 킷값 k1,i(56 비트), k2,i(56 비트) 112 비트가 입력되면 첫 번째 암호화(Ek1)을 수행하여 중간문 A를 생성하고, 중간문 A를 복호화(Dk2)하여 두 번째 중간문 B를 생성하고, 다시 암호화(Ek1)를 수행하여 최종 암호문을 생성한다.

복호화시에는 암호화 과정과 비슷하나 킷값(k1,i k2,i)을 역순으로 수행하면 평문을 생성한다.

2.2 SEED 알고리듬

SEED는 대칭키 암호알고리듬으로, 블록 단위로 메시지를 처리하는 블록 암호알고리듬이며, 16개의 라운드를 가진 Feistel 구조를 가진다[3, 11]. SEED의 F 함수는 수정된 64 비트의 Feistel 구조를 갖추고 있으며, 32 비트 단위의 2개의 블록(C, D)을 입력으로 받아, 32 비트 단위의 2개의 블록(C′, D′)을 각각 출력한다. G 함수는 F함수 및 라운드키 생성시에 사용되는 주요 함수이다.

SEED 알고리즘의 전체 구조는 Feistel 구조로 이루어져 있으며, 128비트의 평문 블록과 128비트

키를 입력으로 사용하여 총 16라운드를 거쳐 128비트 암호문을 블록으로 출력한다.

SEED에서 사용하는 F함수는 수정된 64비트 Feistel 형태로 구성된다. F함수는 각 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 F함수의 입력으로 처리 하여 64비트 블록(C′,D′)을 출력한다[3].

G함수는 F함수 및 라운드 키 생성 시에 사용되는 주요 함수이다. G함수는 매우 좋은 특성을 갖는 두 개의 8비트 S-Box를 이용하여 입력의 각 바이트를 비선형 변환 후, 그 결과 32비트를 4비트 왼쪽 회전 이동한 후 출력한다. 즉, G함수의 입력 32비트를 4개의 8비트 블록 (X_3, X_2, X_1, X_0)으로 분할하여 2개의 S-Box를 (S_3, S_2, S_1, S_0)순서로 적용시켜 (Y_3, Y_2, Y_1, Y_0)를 생성하고, 4비트 왼쪽 회전이동 후, 4개의 8비트 블록 (Z_3, Z_2, Z_1, Z_0)을 생성한다.

G함수 내부에서 쓰이는 두 S-Box는 전단사 함수 $x^n (0 \leqq n \leqq 255)$ 중 차분 특성 및 선형 근사식의 확률이 균일(uniform)하게 분포하는 두 개의 $n = \{247, 251\}$을 이용하여 생성되었다. $GF(2^8)$에서의 0이 아닌 모든 원소를 원시다항식(primitive polynomial) $x^8 + x^6 + x^5 + x + 1$의 근 a의 멱승으로 나타낸다. 고정점(fixed point)이 발생하는 것을 막기 위해 아핀변환(affine transformation)이 사용되었다[1, 3].

SEED의 키 생성 알고리듬은 128비트의 암호키를 64비트씩 좌우로 나누어 이들을 교대로 8비트씩 좌/우로 회전이동 후, 결과의 4워드들에 대한 간단한 산술연산과 G함수를 적용하여 라운드 키를 생성한다.

2.3 AES 알고리듬

AES(Advanced Encryption Standard)는 2001년 2월에 미국 NIST (National Institute of Standard and Technology)에 의해 연방 정보처리 표준 (FIPS 197, Federal Information Processing Standard)으로 지정된 차세대 표준 대칭키 암호화 방식이다. AES는 효율, 보안, 성능, 구현, 유연성 면을 고려할 때 기존 암호화 표준인 3-DES에 비해 탁월한 성능을 보이며, 특히 Rijndael 알고리듬은 AES 조건을 만족하는 타 알고리듬에 비해 월등한 효과을 나타낸다[1].

블록 암호화 알고리듬인 Rijndael 은 블록 크기는 16, 24 또는 32 바이트가 사용가능하며, 각각의 블록 크기에 역시 16, 24 또는 32 바이트의 키를 지원하지만, AES에 선정되면서 블록 길이는 128bits로 고정이 되었다[1, 4].

AES 알고리듬은 암호화 시128bits plantext와 128bits키 값을 입력으로 받는다. 최초 AddRounKey 연산을 실행 한 후 Bytesub, ShiftRow, MixColumn, AddRoundKey 순으로 연산을 실행 한다. 1라운드부터 9라운드 까지는 4개의 블록이 순차 연산이 하며 최종 10라운드에서는 MixColumn 연산을 제외한 나머지 3개의 연산이 이루어진 후 암호화된 data가 출력 된다. 이 때

각 라운드 키 값은 Keyschedule을 통하여 계산되어 진다. 복호화 시는 암호화와는 약간 다른 방식으로 복호화가 된다. 최초 AddRoundKey 연산을 거친 후 Inverse ShiftRow, Inverse Bytesub, AddRoundKey 그리고 Inverse MixColumn 순으로 1라운드부터 9라운드까지 연산을 실행 한다. 마지막 10라운드는 암호화와 비슷한 방식으로 Inverse MixColumn을 제외한 data 연산을 거친 후 암호화된 data를 복호화 하여 평문 data를 출력 하게 된다.

Bytesub는 상태 행렬내의 각 바이트 단위 요소변환을 S-box를 사용하여 변환된 data를 적용하는 비선형 치환 연산이다. 즉, 1바이트 단위로 S-box 연산을 통하여 치환되어 진다. Inverse Bytesub는 복호화에 사용되는 블록으로 Inverse S-box를 거쳐 복호화 과정을 연산한다.

ShiftRow 변환은 data 값을 변경하지 않고 위치만을 변경하는 행 단위 연산을 수행한다. 0번째 행은 변환을 하지 않으며 1번째, 2번째, 3번째 행들은 각각 1, 2, 3 바이트씩 좌측으로 순환이동을 한다. Inverse ShiftRow는 복호화에 사용되는 변환으로 0번째 행은 변환을 하지 않고 1번째, 2번째, 3번째 행들은 각각 1, 2, 3 바이트씩 우측으로 순환이동 연산을 수행한다.

MixColumn 변환은 32bits data를 $GF(2^8)$위에 3차 다항식 b(x)로 대응시키고 주어진 다항식 c(x) = $03x^3 + 01x^2 + 01x + 02$ 를 이용하여 b(x) c(x)를 계산하고 $x^4 + 1$로 나눈 나머지를 32bits로 대응 시키는 변환을 말한다[4].

AddRoundKey 변환은 키 확장에 의해 생성된 128bits 라운드 키 값과 128bits 입력 data 간의 단순한 bits 단위의 XOR 연산을 적용한 것이다.

KeySchedule 변환은 AES 암·복호화에 쓰이는 라운드 키를 생성하는 블록이다. AES 암호 알고리듬에서 10번의 라운드 연산에 사용되는 라운드 키는 생성 방식에 따라 온라인 방식과 오프라인 방식으로 나누어진다. 암·복호화마다 매번 스케줄러를 통해 라운드 키를 생성하여 사용하는 온라인 방식은 라운드 키를 따로 저장할 필요가 없으므로 추가적인 메모리나 레지스터를 사용하지 않는다. 그러나 초기키가 동일하더라도 매번 키 스케줄러를 통해 라운드 키를 생성해야 하는 단점이 있다. 미리 초기키 값에 대해 별도의 키 스케줄러 과정을 통해 라운드 키를 생성한 후 메모리에 저장하여 사용하는 오프라인방식은 추가적인 메모리 사용과 초기키의 변경 시 라운드 키를 갱신해야 하지만, 동일한 초기키에 대해서는 라운드 키를 재생성할 필요가 없다[1, 4].

KeySchedule을 통해 라운드 키를 생성하기 위해서는 4번의 S-box 연산, data 로테이션 그리고 rcon이라는 32bits 라운드 상수 값을 연산하여야한다.

2.4 HMAC-SHA-1 알고리듬

HMAC(The Keyed-Hash Message Authentication Code)은 메시지의 무결성과 함께 메시지의 출처의 인증을 위해 사용되며 암호학적 해쉬함수와 대칭키로 구성된다[8]. HMAC은 암호학적 해쉬함수 H와 비밀키 K를 요한다. H는 데이터 블록에 기본압축함수를 반복하여 데이터를 해쉬하는 암호학적 해쉬함수라고 가정한다. B는 그러한 블록의 바이트 단위의 길이이며, 앞서 언급된 해쉬

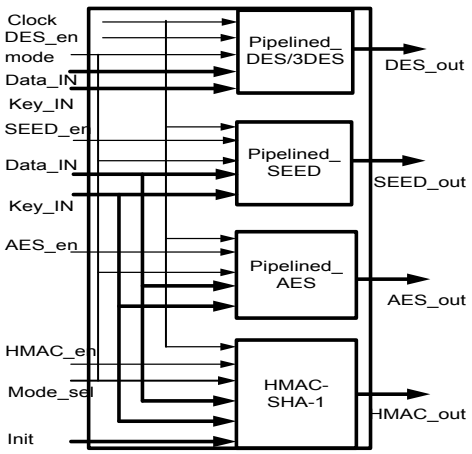
함수들 에서는 B=64이다. L은 해쉬 출력의 바이트 단위의 길이이며, MD5에서는 L=16이고 SHA-1에서는 L=20이다. 인증키 K의 길이는 해쉬함수의 블록 길이인 B이하의 임의의 값이다.

NIST에서 개발된 해쉬 알고리듬인 SHA-1은 264 비트 이하의 메시지에서 160 비트의 메시지 요약을 생성한다. 표준 해쉬 알고리듬 SHA-1은 임의의 길이를 가지는 입력 메시지를 512비트 블록 단위로 처리하여 160비트의 출력을 낸다. 512비트 단위 블록을 처리하는 압축 함수는 모두 4 라운드, 80단계로 구성되며, 해쉬코드를 계산하는 연쇄변수는 5개이다. 또한 각 라운드에 적용될 메시지 변수의 개수는 512비트 입력블록으로부터 생성된 16워드와 이로부터 추가로 생성되는 4개의 워드를 포함하여 20개가 된다[8].

3. 고속 암호 프로세서 설계

3.1 암호 프로세서 설계

[그림 1]은 Pipeline방식을 사용한 암호 프로세서의 블록도를 나타낸다. 암호 프로세서는 DES/3DES, SEED, 그리고 AES를 채용한 암호엔진과 HMAC-SHA-1 인증엔진으로 구성되며, 각 모듈들은 각각의 Enable 신호에 의하여 작동하도록 설계되었다.



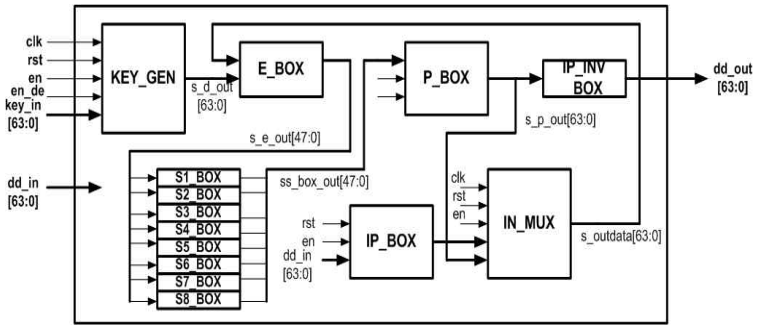
[그림 1] Pipeline 방식을 사용한 암호프로세서의 블록도

[Fig. 1] Block diagram of cryptoprocessor using pipeline technique

(1) Pipelined DES/βDES 설계

[그림 2]는 DES 알고리듬을 하드웨어로 구현하기 위한 Iterative 형태의 내부 블록도를 나타낸다

[2].

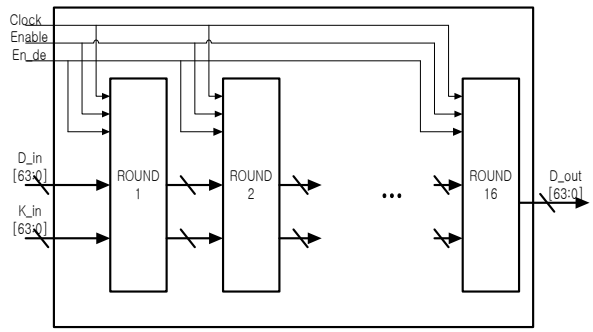


[그림 2] DES의 내부 블록도
[Fig. 2] Block diagram of DES

KEY_GEN블록은 DES의 각각의 라운드키를 생성하는 블록이며, IP_BOX블록은 64비트의 평문이 치환입력을 생성하기 위해 비트열의 순서를 재조정 하는 초기순열 블록이며, IP_INV_BOX블록은 초기순열의 역초기 순열블록이며, IN_MUX블록은 DES의 각각의 라운드에 맞게 데이터를 입력해 주는 블록이다.

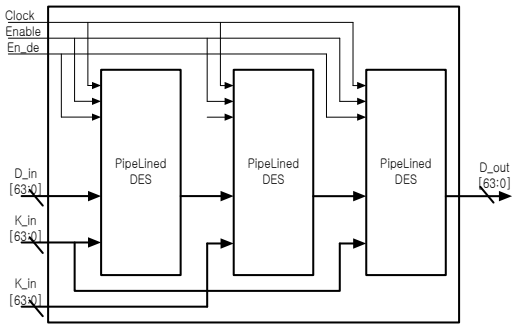
3DES는 3개의 DES블록으로 구성이 되며, 각각의 DES블록을 순차적으로 거쳐서 최종 3DES의 결과 값을 얻는다. 3DES의 킷값은 64비트의 2개의 키를 사용하며, Key1은 첫 번째 DES블록과 세 번째 DES블록에 사용되어지고, Key2는 두 번째 DES블록의 키 값으로 사용되어진다.

[그림 3]은 파이프라인방식을 적용한 DES알고리듬의 내부구조를 나타낸다. 각각의 ROUND에는 키생성을 위한 블록과 암호과정을 위한 블록이 포함되며, ROUND1과 ROUND16에 각각 초기 치환(initial permutation : IP)과 역 초기 치환(inverse initial permutation: IP-1)을 포함하고 있다.



[그림 3] DES의 파이프라인 구조
[Fig. 3] Pipelined structure of DES

[그림 4]는 파이프라인 방식을 적용한 3DES알고리듬의 내부 구조를 나타낸다.



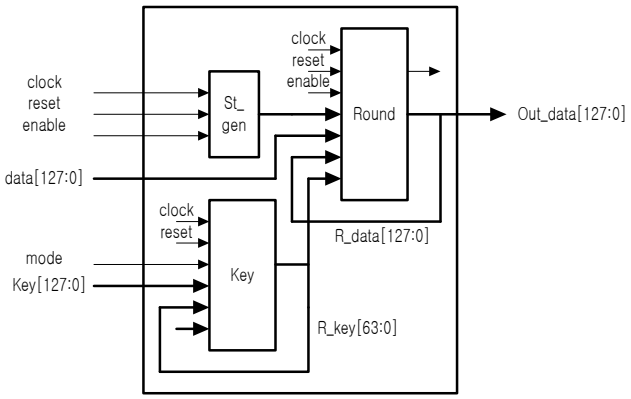
[그림 4] 3DES 시스템의 구조

[Fig. 4] Block diagram of 3DES

그림 4에서 알 수 있듯이 3DES는 DES파이프라인 구조를 3개를 사용하여 구성되어지며, 64비트의 Key값 2개를 받아 처음과 마지막 DES블록에서 첫 번째 64비트 키를 사용하고 두 번째 DES블록에서는 두 번째 64비트 키를 사용한다. 48clocks이후에 매 clock마다 64비트의 암호문을 생성한다.

(2) Pipelined SEED 설계

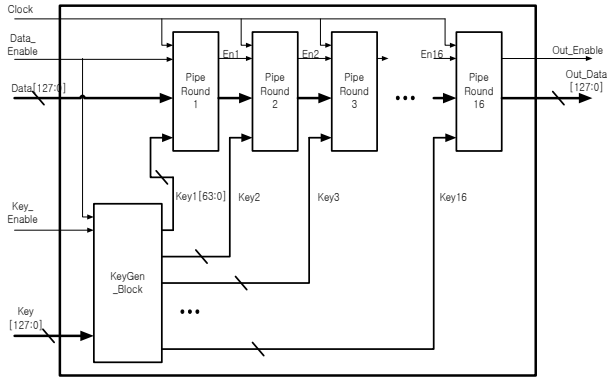
[그림 5]는 설계된 SEED알고리듬[3]의 내부 블록도를 나타낸다. SEED의 내부 블록도는 St_gen, Key, Round 블록으로 구성되어져 있다. St_gen 블록은 Round블록의 데이터 처리에 필요한 제어신호를 발생시키고, Key 블록은 매 라운드에 필요한 R_key를 생성한다.



[그림 5] SEED의 내부 블록도

[Fig. 5] Block diagram of SEED

Round 블록은 St_gen 블록에서 나온 제어신호와 Key 블록에서 생성된 R_key로 SEED의 내부 16 라운드 처리를 수행한다. [그림 6]은 SEED의 파이프라인 구조를 나타낸다.

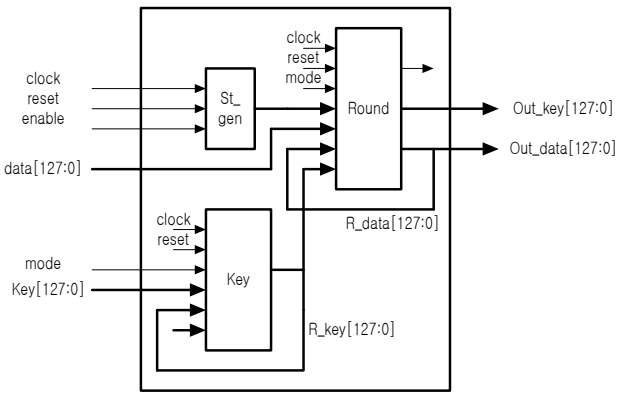


[그림 6] SEED의 파이프라인 구조
[Fig. 6] Pipelined structure of SEED

[그림 6]에서 알 수 있듯이 시스템의 구조는 하나의 KeyGen_Block과 16개의 Pipel Round블록으로 구성된다. KeyGen_Block은 매 라운드에 필요한 Key를 생성하고 Pipe Round 블록은 라운드 내부 연산을 수행하도록 구성한다.

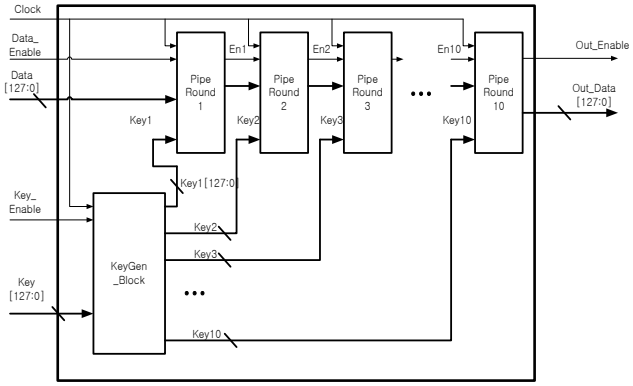
(3) Pipelined AES 설계

[그림 7]은 설계된 AES 알고리듬[4]의 내부 블록도를 나타낸다[4].



[그림 7] AES의 내부 블록도
[Fig. 7] Block diagram of AES

블록에서 생성된 R_key로 AES의 내부 10 라운드 처리를 수행한다. [그림 8]은 AES파이프라인 구조를 나타낸다.

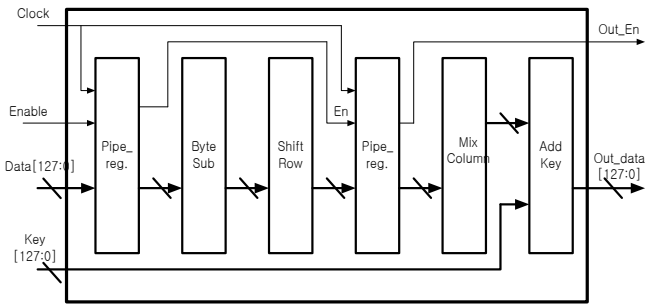


[그림 8] AES 파이프라인 구조

[Fig. 8] Pipelined structure of AES

[그림 8]에 나타난 블록도는 10개의 파이프 라운드 블록과, 하나의 키 생성 블록으로 구성되어 있다. 키 생성 블록은 매 라운드 마다 사용하게 될 키들을 생성 시켜 10개의 파이프 라운드 블록에 키 입력을 담당한다. 각 파이프 라운드 블록은 내부적으로 같은 동작을 하는 블록들로 구성되어져 있다. 단, Pipe Round1과 Pipe Round10은 암ㆍ복호 동작이 다소 차이가 나기 때문에 각각 수정을 하여 작성하였다.

[그림 9]는 각 파이프 라운드 블록의 내부 구조를 보여주고 있다.



[그림 9] 라운드 내부의 파이프라인 구조

[Fig. 9] Pipelined structure for inner Round

ByteSub블록에서는 입력 값에 대한 ByteSub함수 값을 출력한다. 이 출력값은 클럭과 무관하게 ShiftRow함수를 통과하여 다음 PipeReg의 입력값에 연결된다. PipeReg블록은 이전 PipeReg블록에

서 나온 en신호와 전체 clock신호에 따라 ShiftRow를 통과되며, ShiftRow를 통과하고 나온 입력값은 MixColumn에 연결된다. MixColumn을 통과한 출력값은 라운드의 Key값과 함께 AddKey블록에서 XOR연산을 거치고 다음 라운드 블록으로 전송된다.

4. 시스템 구현 및 성능평가

본 논문에서 제안된 pipelined 암호 프로세서의 각 모듈은 VHDL을 이용하여 설계하였다. 또한, Xilinx ISE 6.2i 툴을 이용하여 합성을, 설계 검증을 위한 타이밍 시뮬레이션은 Modelsim을 이용하였고, Xilinx FPGA VertexII(XC8000)를 타겟으로 FPGA를 구현하였다. 본 연구는 IDEC의 EDA Tool 에서 지원하여 수행하였다. [표 1]과 [표 2]는 각각 Iterative 와 Pipelined 암호 프로세서에 대한 성능평가를 나타낸다.

[표 1] Iterative 암호프로세서의 성능 평가
[Table 1] Performance evaluation of Iterative cryptoprocessor

Iterative	# gates (Silces)	Frequency (MHz)	Throughput (Mbps)
DES	17,199 (637)	120	320
3DES	53,649 (1,987)	120	106
AES	111,024 (4,112)	59	546
SEED	150,336 (5,568)	34	546
HMAC-SHA-1	132,570 (4,910)	43	233

[표 2] Pipelined 암호프로세서의 성능 평가
[Table 2] Performance evaluation of Pipelined cryptoprocessor

Pipelined	# gates (Silces)	Frequency (MHz)	Throughput (Gbps)
DES	50634 (1,746)	228	1.4
3DES	181337 (6,253)	225	1.4
AES	562861 (19,409)	116	1.4
SEED	610595 (21,055)	91	1.1

표 1에서 알 수 있듯이 처리속도에 있어서 DES와 3DES는 120Mhz로 동작하며, 처리율은 각각 320Mbps와 106Mbps 정도였다. AES와 SEED 경우 처리속도는 각각 49Mhz 와 34Mhz로 동작하며, 처리율은 각각 546Mbps과 218Mbps 정도를 보였다. 인증모듈의 경우 SHA-1 은 43MHz에서 동작

하고, 지연시간은 15 ns으로 약 537Mbps의 처리율을 보였다. HMAC과 연동하여 HMAC-SHA-1으로 동작할 경우 43Mhz에서 233Mbps의 처리율을 보였다.

[표 1]과 [표 2]의 성능평가를 통하여 Throughput은 Pipelined 방식이 높음을 알 수 있다. 그러나 #gates가 Iterative 방식보다 많이 나타나 면적에서는 많은 hardware overhead를 보여준다.

5. 결론

본 논문에서는 pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 FPGA 구현에 관하여 기술하였다. 암호화를 위한 알고리듬은 DES/3DES, SEED, 그리고 AES를 사용하였고 인증을 위한 알고리듬은 HMAC-SHA-1을 사용하였다. 제안한 방법에 있어서 고속 암호 프로세서의 실현을 위하여 우선, 3종류의 암호화 모듈이 Iterative 방식으로 설계하였고, 이들을 기반으로 한 Pipelined 방식의 암호 프로세서가 FPGA로 구현되었다.

제안된 암호 프로세서는 VHDL을 사용하여 구조적 모델링을 행하였으며, Xilinx사의 ISE 6.2i 툴과 Modelsim을 이용하여 시뮬레이션 및 합성을 수행하였다. 시스템의 성능평가를 통하여 Iterative 방식은 적은 면적을 사용하는 대신에 Pipelined 방식은 많은 면적을 필요하지만, Iterative 방식에 비하여 높은 성능을 나타냄을 확인하였다.

Reference

[1] Behrouz A. Forouzan, *Cryptography And Network Security*, McGraw-Hill Education (India) Pvt Limited, (2011).

[2] NBS, Data Encryption Standard, FIPS Pub. 46, U.S, National Bureau of Standards, Jan. (2003).

[3] 128-bit Block Cipher Algorithm (SEED), KISA, (2003) December, Korea.

[4] Joan Daemen, Vincent Rijmen, AES Proposal Rijndal, (http://csrc.nist.gov/encryption/aes/rijndael/) (1988).

[5] R. L. Rivest, A. Shamir and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communication of the ACM, (1978), Vol. 21, No. 2, pp. 120-126.

[6] W. Diffie, and M. E. Hellman, New directions in cryptography, IEEE Trans. Computers, June (1976), Vol. IT-22, No. 6, pp. 644-654.

[7] T. ElGmal, A public-key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. on Information Theory, (1985), Vol. IT-31, No. 4, pp. 469-472.

[8] C. Madson, R. Glenn, The use of HMAC-SHA1 within ESP and AH, RFC 2040, November (1998).

[9] A. J. Elbirt, W.Yip, et.al, An FPGA-Based performance evaluation of the AES block cipher candidate algorithm finalists, IEEE Transactions on VLSI System, August (2001), Vol. 9, No. 4.

[10] K. H. Lee, M. S. Kang, D. H. Ryu, FPGA Implementation of AES cipher algorithm using parallel processing technique, IEIE, Fall Conference, (2003).

[11] K. H. Lee, S. Y. Nam, and M. S. Kang, FPGA Implementation of SEED algorithm using modified LUT method, IEIE, Conference of SOC Design, (2004).

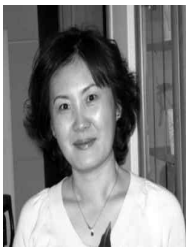
Authors

강민섭(Min-Sup Kang)

1979년 2월 광운대학교 전자통신공학과 (학사)
1984년 8월 한양대학교 전자공학과 (공학석사)
1992년 2월 (일) 오사카대학교 전자공학과 (공학박사)
1984년 ~ 1993년 한국전자통신연구원 선임연구원
2001년 ~ 2002년 Univ. of California, Irvine 전기전자공학과 객원연구원
1993년 ~ 현재 안양대학교 컴퓨터공학과 교수
관심분야: 네트워크 보안, 암호 시스템, 임베디드 시스템, RFID/USN, ASIC 설계

장태민(Tae-Min Chang)

2006년 2월 방송통신대학교 컴퓨터과학과 (학사)
2008년 2월 안양대학교 컴퓨터공학과 (공학석사)
2008년 3월 ~ 현재 안양대학교 컴퓨터공학과 (박사과정)
2008년 ~ 현재 안산1대학 디지털정보통신과 겸임교수
관심분야 : VLSI 설계, 암호프로세서 설계, 임베디드 설계, IBS 통신 보안, RFID/USN



김현수(Hyean-Su Kim)

1989년 2월 계명대학교 전자계산학과 (학사)
1996년 2월 고려대학교 전산교육학과 (교육학석사)
1996년 ~ 현재 안양대, 성결대, 가천대 시간강사
2013 9월 ~ 현재 안양대학교 컴퓨터공학과 (박사과정)
관심분야: 컴퓨터 보안, 전자상거래