Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현

°박상조* 김우성* 장태민** 강민섭**
*호서대학교 컴퓨터공학과
**안양대학교 컴퓨터공학과
sancho@davan.co.kr, mskang@anyang.ac.kr

Design and Implementation of High-speed Crypto Processor Using Pipeline Technique

°Sang-Cho Park* Woo-Sung* Tae-Min Chang** Min-Sup Kang**
*Department of Computer Engineering, HoSeo University
**Department of Computer Engineering, Anyang University

요 약

본 논문에서는 Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현에 관하여 기술한다. 암호화를 위한 알고리듬은 DES 와 SEED를 사용하고 인증을 위한 알고리듬은 HMAC-SHA-1을 이용한다.

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.

1. 서 론

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. 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 알고리듬은 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 알고리즘내에 동일한 동작을 하는 블록이 존재하는 경우에는 이 블록을 설계하고 운영하는 기법에 따라 성능에 큰 영향을 미친다.

본 논문에서는 Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현에 관하여 기술한다. 암호화를 위한 알고리듬은 DES와 SEED를 사용하고 인증을 위한 알고리듬은 HMAC-SHA-1을 이용한다.

2. 암호 시스템

2.1 DES/3DES 알고리듬

대표적인 블록 암호화 알고리듬인 DES는 64 비트의 데이터와 56 비트 길이의 키를 사용하여 64 비트의 암호화 결과를 생성한다[3]. 암호화를 위해 64 비트의 평문이 초기 치환(initial permutation)후에 32 비트씩 좌), 우 부분으로 나뉘게 되며 그 다음 16라운드의 계산을 거치게 된다. 16라운드 후에는 오른쪽과 왼쪽 부분이 합쳐져서 역 초기 치환(inverse initial permutation)을 거침으로써 암호문(Ciphertext)이 생성된다.

3DES는 두 개의 암호 키를 사용하여 첫 번째 키로 암호화하고 다시 두 번째 키로 복호화 한 다음 또 다시 첫 번째 키로 암호화하여 강력한 암호를 얻는 방식이다.

2.2 SEED 알고리듬

SEED는 대칭키 암호알고리듬으로, 블록 단위로 메시지를 처리하는 블록 암호알고리듬이며, 16개의 라운드를 가진 Feistel 구조를 가진다[4,9]. SEED의 F 함수는 수정된 64 비트의 Feistel 구조를 갖추고 있으며, 32 비트 단위의 2개의 블럭(C, D)을 입력으로 받아, 32 비트 단위의 2개의 블럭(C', D')을 각각 출력한다. G 함수는 F함수 및 라운드키 생성시에 사용되는 주요 함수이다.

2.3. 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 해쉬함수들 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 2^{64} 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.

3. 고속 암호프로세서의 설계

3.1 암호 프로세서 설계

그림 1은 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.

(1) Pipelined DES 설계

그림 1은 DES 알고리듬을 하드웨어로 구현하기 위한 Iterative 형태의 내부 블록도를 나타낸다[2].

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 입력해주는 블록이다.

3DES는 3개의 DES블록으로 구성이 되며, 각각의 DES블록을 순차적으로 거쳐서 최종 3DES의 결과 값을 얻는다.

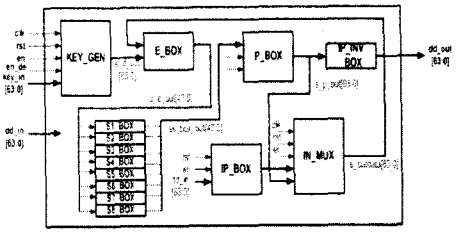

그림 1. DES의 내부 블록도

3DES의 킷값은 64비트의 2개의 키를 사용하며, Key1은 첫 번째 DES블록과 세 번째 DES블록에 사용되어지고, Key2는 두 번째 DES블록의 키 값으로 사용되어진다.

그림 2는 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.

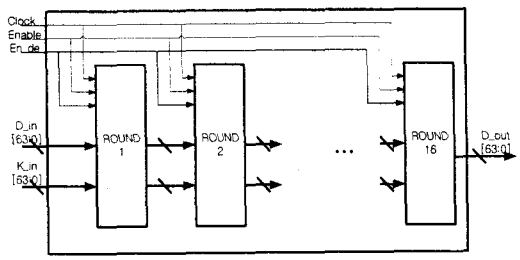

그림 2. DES의 파이프라인 구조

(2) Pipelined SEED 설계

그림 3은 설계된 SEED알고리듬의 내부 블록도를 나타낸다[3].

SEED의 내부 블록도는 St_gen, Key, Round 블록으로 구성되어져 있다. St_gen 블록은 Round블록의 데이터 처리에 필요한 제어신호를 발생시키고, Key 블록은 매 라운드에 필요한

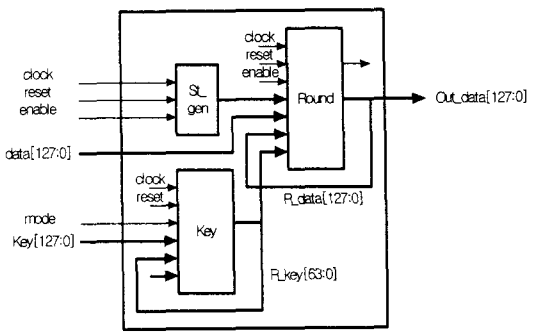

그림 3. SEED의 내부 블록도

R_key를 생성한다.

Round 블록은 St_gen 블록에서 나온 제어신호와 Key 블록에서 생성된 R_key로 SEED의 내부 16 라운드 처리를 수행한다. 그림 4는 SEED의 파이프라인 구조를 나타낸다.

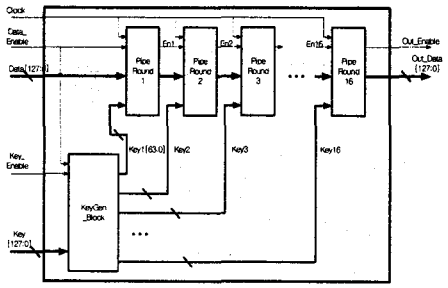

그림 4. SEED의 파이프라인 구조

그림 4에서 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.

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.

표 1과 표2는 각각 Iterative 와 Pipelined 암호 프로세서에 대한 성능평가를 나타낸다.

표 1 Iterative 암호프로세서의 성능 평가

Iterative	# gates (Silces)	Throughput (Frequency)
DES	28107 (1041)	320Mbps (80Mhz)
SEED	166725(6175)	218Mbps (30Mhz)
HMAC-SHA-1	140130(5190)	233Mbps (37Mhz)

표 2 Pipelined 암호프로세서의 성능 평가

Pipelined	# gates (Silces)	Throughput (Frequency)
DES	61776(2288)	5.1Gbps (80Mhz)
SEED	550881(21055)	6.7Gbps (53Mhz)

표 1과 표 2의 성능평가를 통하여 Throughput은 Pipelined 방식이 대폭 개선되었음을 알 수 있다. 그러나 #gates 수의 경우 Iterative 방식보다 약 3배 이상 증가되어 면적에서는 더 많은 hardware overhead를 보여준다.

5. 결론

본 논문에서는 Pipeline 기법을 이용한 고속 암호 프로세서의 설계 및 구현에 관하여 기술하였다. 본 논문에서는 암호 프로세서의 Iterative와 Pipelined 방식을 설계하였고, 두 시스템의 성능 분석에 관하여 기술하였다. Iterative 방식은 적은 면적을 사용하며, Pipelined 방식은 많은 면적을 필요하지만, Iterative 방식에 비하여 높은 성능을 나타냄을 확인하였다.

제안된 암호 프로세서는 VHDL을 사용하여 구조적 모델링을 행하였으며, Xilinx사의 ISE 6.2i 툴과 Modelsim을 이용하여 시뮬레이션 및 합성을 수행하였다.

[참고문헌]

[1] 인터넷보안기술포럼 "Implementation Technology for secure VPN in IP Layers", 2001.

[2] NBS, Data Encryption Standard, FIPS Pub. 46, U.S, National Bureau of Standards, Jan. 1977.

[3] 한국정보보호센터, "128비트 블록 암호알고리듬(SEED) 개발 및 분석보고서", KISA, 2003.

[4] an Daemen, Vincent Rijmen, "AES Proposal Rijndal" (http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf)

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

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

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

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

[9] 강민섭, 남승용, 김주한, "IPSec 암호 프로세서를 위한 SHA-1 해쉬 엔진의 하드웨어 구현", 대한전자공학회, 추계학술발표논문집, 2003.

[10] 이광호, 강민섭, 류대현, "병렬처리 기법을 이용한 AES 암호 알고리듬의 FPGA 구현", 대한전자공학회, 추계학술발표논문집, 2003.

[11] 이광호, 남승용, 강민섭, "개선된 LUT 방식을 이용한 SEED 알고리듬의 FPGA 구현", 대한전자공학회, SOC 설계발표논문집, 2004.