스마트카드용 HAS-160 프로세서 설계

김해주* · 신경욱*

* 금오공과대학교 전자공학과

A Design of HAS-160 Processor for Smartcard Application

Hae-ju Kim* · Kyung-Wook Shin*

* School of Electronic Eng., Kumoh National Institute of Technology

E-mail : tanatos13@kumoh.ac.kr

요 약

본 논문에서는 한국형 표준 해쉬 알고리듬인 HAS-160을 구현하는 프로세서를 설계하였다. 각 단계연산에 사용되는 4개의 가산기는 연산성능을 높이기 위해 5:3 및 3:2 캐리보존 가산기(carry-save adder)와 캐리선택 가산기(carry-select adder)의 혼합구조를 사용하였다. 설계된 HAS-160 프로세서는 512 비트 메시지로부터 160 비트의 해쉬코드를 생성하는데 82 클록주기가 소요되며, 50 MHz@3.3-V로 동작하는 경우 312 Mbps의 성능을 나타낸다. 0.35-㎛ CMOS 셀 라이브러리로 합성한 결과 약 17,600개의 게이트로 구현되었다.

ABSTRACT

This paper describes a hardware design of hash processor which implements HAS-160 algorithm adopted as a Korean standard. To achieve a high-speed operation with small-area, the arithmetic operation is implemented using a hybrid structure of 5:3 and 3:2 carry-save adders and a carry-select adder. The HAS-160 processor synthesized with 0.35-㎛ CMOS cell library has 17,600 gates. It computes a 160-bit hash code from a message block of 512 bits in 82 clock cycles, and has 312 Mbps throughput at 50 MHz@3.3-V clock frequency.

키워드

Hash algorithm, HAS-160, authentication, information security, smartcard

Ⅰ. 서 론

오늘날 정보통신기술과 인터넷의 발전으로 인해 인터넷 뱅킹, 전자상거래, 전자화폐 등의 서비스가 보편화됨에 따라 마이크로프로세서와 메모리를 내장하고 있어 카드 내에서 정보의 저장과 처리가 가능한 스마트카드의 사용이 급속히 확대되고 있다. 스마트카드와 유·무선 네트워크를 통해 유통되는 개인정보와 금융정보의 보안을 위해 다양한 정보보안 기술이 적용되고 있다. 그러나 컴퓨터 성능의 향상과 보안공격기술의 발달로 인해 정보 유출과 위조를 막기 위한 더 높은 수준의 암호화 기술이 지속적으로 요구되고 있다. 메시지 내용 공개나 트래픽 분석과 같은 수동적 공격은 DES나 AES 같은 대칭키 알고리듬, 그리고 RSA와 같은 비대칭 알고리듬으로 대처할 수 있다[1]. 그러나 단순한 메시지 암호화만으로는 위장, 재전송, 메시지 불법 수정과 같은 능동적 공격에 대해 대처할 수 없으며, 능동적 공격에 대처하기 위한 방법으로 해쉬 함수를 이용한 메시지 인증과 무결성 검증이 사용된다.

해쉬 알고리듬은 임의의 길이의 비트 열을 고정된 길이의 출력 값인 해쉬코드로 압축시키는 암호 알고리듬의 일종이며, 강한 충돌 저항성 즉, 동일한 출력을 갖는 서로 다른 두 개의 입력 메시지를 찾는 것이 불가능한 특성을 갖는다. 해쉬 알고리듬은 크게 DES와 같은 블록암호 알고리듬에 기초한 해쉬 알고리듬과 전용 해쉬 알고리듬으로 나눌 수 있다. 블록암호를 이용한 해쉬 알고리듬은 이미 구현되어 사용되고 있는 블록암호 기술을 사용할 수 있다는 이점이 있으나, 일반적으로 블록암호 알고리듬들은 속도가 느리다는 단점이 있다. 대부분의 응용에서 전용 해쉬 알고리듬이 주로 이용되며, 대표적인 전용 해쉬 알고리듬으로는 SHA-1, MD5, HAS-160 등이 있다.[4-9]

본 논문에서는 한국형 해쉬함수 표준인 HAS-160 알고리듬[2]의 스마트카드 보안에 적합한 하드웨어 설계에

관해 기술한다. 2장에서 HAS-160 알고리듬을 소개하고, 3장에서 HAS-160 프로세서 설계를 기술하며, 4장에서는 설계된 프로세서의 성능을 분석하고, 5장에서 결론을 제시한다.

II. HAS-160 해쉬 알고리듬

HAS-160[2]은 1998년 10월에 국내 표준화(TTAS.KO-12.0011/R1)를 거쳐 2005년 12월 21일에 개정되었다. HAS-160은 임의의 길이의 비트 열을 512 비트 블록 단위로 처리하여 160 비트의 해쉬코드를 출력한다. HAS-160 알고리듬은 메시지 M을 해쉬함수에 적용하기 위한 메시지 덧붙이기, 메시지 블록 생성, 라운드연산 블록 및 최종 해쉬코드 갱신의 구조로 이루어지며 전체 연산구조는 그림 1과 같다.

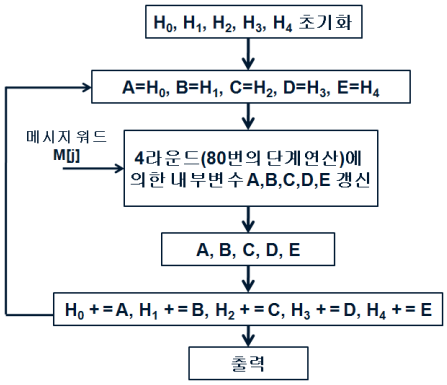

그림 1. HAS-160 해쉬 알고리듬의 연산구조

HAS-160 해쉬 알고리듬은 5개의 32 비트 워드들 사이의 정수연산으로 구성된다. 입력 메시지는 512 비트 단위의 블록으로 분할된 후, 리틀 앤디안(little endian) 방식의 4 바이트(32 비트) 워드 16개로 변환되어 연산에 사용된다. 입력 메시지 M의 비트 수가 512의 정수배가 되지 않는 경우에는 메시지 덧붙이기를 통해 512 비트의 정수배로 만들어 진다.

HAS-160은 4번의 라운드 연산을 통해 해쉬코드를 생성하며, 각 라운드 연산은 20번의 단계연산으로 구성된다. 단계연산은 그림 2와 같이 구성되며, 부울함수 $f[j]$의 계산, 2개의 순환이동(cyclic shift), 4개의 2^{32}-modulo 가산, 내부변수(A, B, C, D, E)의 갱신 등으로 이루어진다. 512 비트의 메시지 M으로부터 생성되는 16개의 32 비트 워드와 부울함수 $f[j]$를 통해 생성되는 4개의 워드를 포함해 20개의 워드 $M[j]$가 80번의 단계연산에 사용된다. 각 라운드 연산에 사용되는 부울함수 $f[j]$와 상수 $K[j]$는 표 1과 같이 정의된다.

순환이동의 크기 (S_1, S_2)는 표 2와 같이 정의된다. S_1은 라운드 마다 동일한 값이 사용되며, 라운드를 구성하는 단계연산에 따라 크기가 달라진다. S_2는 연산단계에 무관하게 동일한 값이 사용되나 라운드 마다 다른 값이 사용된다. 한편, 각 라운드의 연산단계에 적용되는 메시지 워드 $M[j]$의 순서는 표 3과 같이 정의된다. 하나의 512 비트의 메시지 블록에 대한 4번 라운드의 80번 단계연산이 끝나면, 그 결과는 내부 레지스터에 저장되어 다음 메시지 블록의 연산에 사용된다.

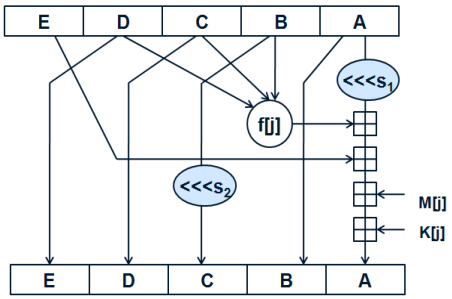

그림 2. HAS-160 알고리듬의 단계연산

표 1. 부울함수 $f[j]$ 및 상수 $K[j]$

라운드	단계연산 j	$f[j]$	$K[j]$
1	$0 \le j \le 19$	$(B \wedge C) \vee (\overline{B} \wedge D)$	00000000
2	$20 \le j \le 39$	$D \oplus C \oplus B$	5A827999
3	$40 \le j \le 59$	$C \oplus (B \vee \overline{D})$	6ED9EBA1
4	$60 \le j \le 79$	$D \oplus C \oplus B$	8F1BBCDC

표 2. 순환이동의 크기 S_1, S_2

라운드	단계연산 j	S_1	S_2
1	$0 \le j \le 19$	5, 11, 7, 15, 6, 13, 8, 14, 7, 12, 9, 11, 8, 15, 6, 12, 9, 14, 5, 13	10
2	$20 \le j \le 39$		17
3	$40 \le j \le 59$		25
4	$60 \le j \le 79$		30

표 3. 메시지 워드 $M[j]$의 연산 순서

라운드	단계연산 j	$M[j]$
1	$0 \le j \le 19$	18, 0, 1, 2, 3, 19, 4, 5, 6, 7, 16, 8, 9, 10, 11, 17, 12, 13, 14, 15
2	$20 \le j \le 39$	18, 3, 6, 9, 12, 19, 15, 2, 5, 8, 16, 11, 14, 1, 4, 17, 7, 10, 13, 0
3	$40 \le j \le 59$	18, 12, 5, 14, 7, 19, 0, 9, 2, 11, 16, 4, 13, 6, 15, 17, 8, 1, 10, 3
4	$60 \le j \le 79$	18, 7, 2, 13, 8, 19, 3, 14, 9, 4, 16, 15, 10, 5, 0, 17, 11, 6, 1, 12

III. HAS-160 프로세서 설계

3.1 HAS-160 프로세서의 구조

본 논문의 HAS-160 해쉬 프로세서는 스마트카드에 내장되는 마이크로프로세서의 암호 보조 프로세서로 동작하도록 설계하였다. 32 비트 입출력 인터페이스를 가지며, 입력 메시지의 비트 열과 워드 열 사이의 변환과 메시지 덧붙이기는 마이크로프로세서에서 전처리(preprocessing)되어 입력되며, HAS-160 프로세서는 전처리된 512 비트의 입력에 대한 160 비트의 해쉬코드를 계산한다.

설계된 HAS-160 해쉬 프로세서의 전체 구조는 그림 3과 같으며, 512 비트의 입력 메시지를 받아 4개의 워드를 추가하여 20개의 워드 열을 생성하는 M_Gen 블록, 80번의 단계연산을 수행하는 HAS160_round 블록, 해쉬코드 초기값을 저장하는 LUT 블록, 그리고 유한상태머신(FSM)의 제어블록 등으로 구성된다.

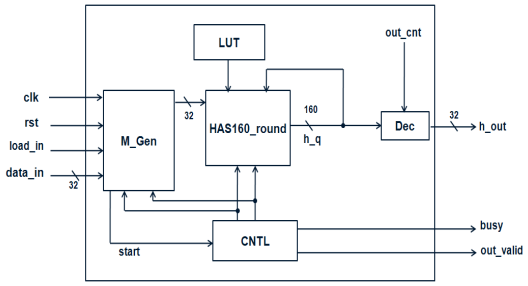

그림 3. HAS-160 프로세서의 구조

M_Gen 블록은 그림 4와 같이 20개의 32 비트 메시지 워드를 저장하는 레지스터, MUX, XOR 및 내부 제어회로 등으로 구성된다. 512 비트의 입력 메시지로부터 16개의 32 비트 메시지 워드 $M[15:0]$를 생성하며, HAS-160 표준문서[2]에 규정된 방식에 의해 16개의 메시지 워드 중 일부를 선택하여 XOR 연산을 통해 4개의 메시지 워드 ($M[16:19]$)를 생성하여 레지스터에 저장한다.

단계연산 블록은 표 1에 정의된 부울함수 $f[j]$를 계산하는 f_func 회로, 라운드 연산에 사용되는 상수 $K[j]$을 결정하는 k_init 회로, 표 2에 정의된 순환이동 크기를 결정하는 s1_rotl 및 s2_rotl 회로, 그리고 2^{32} -modulo 가산기 등으로 구성되며, 그림 5와 같이 설계하였다. 제어회로에서 입력되는 라운드 계수기의 값에 따라 $f[j]$와 $K[j]$, S_2 순환이동 크기가 결정되며, 단계계수기의 값에 따라 S_1 순환이동 크기가 결정된다. 그림 5의 단계연산 블록에서 연산속도에 가장 큰 영향을 미치는 부분은 다음 단계연산에 사용될 내부변수 A를 갱신하는 가산기 회로이다. 본 논문에서는 5:3 및 3:2 carry-save 가산기와 carry-select 가산기를 사용하여 설계하였으며, 이를 통해 최소의 게이트로 고속 동작하도록 하였다.

제어회로는 라운드 및 단계계수기와 FSM으로 구성된다. FSM은 M_Gen 블록에서 생성되는 start 신호를 감지하기 전의 IDLE 상태, 단계연산을 수행하는 CALC 상태, 160 비트의 해쉬코드를 갱신하고 h_d 레지스터에 저장하는 VALID_OUT 상태로 구성된다. 입력 메시지가 512 비트 이상인 경우에는 use_prev_h 신호를 이용하여 h_d 레지스터를 갱신된 값으로 유지시켜 다음 메시지 블록의 연산에 사용되도록 하였다. 라운드계수기와 단계계수기는 FSM의 CALC 상태에서 동작하며, 단계연산 블록의 부울함수, 순환이동 크기, 상수 값 등을 결정한다.

한 클록 당 하나의 단계연산이 수행되도록 설계되었으며, 라운드 당 20번의 단계연산으로 구성되고 4라운에 총 80번의 단계연산을 수행하는데 80 클록이 사용된다. 512 비트의 입력 메시지로부터 4개의 워드 열을 추가로 생성하는데 한 클록이 사용되며, 단계연산 종료 후 h_d 레지스터의 초기값을 내부변수와 더하여 최종 해쉬코드를 생성하는데 한 클록이 사용된다. 따라서 512 비트의 메시지로부터 160 비트의 해쉬코드를 생성하는데 소요되는 총 82 클록 사이클이 소요되도록 설계하였다.

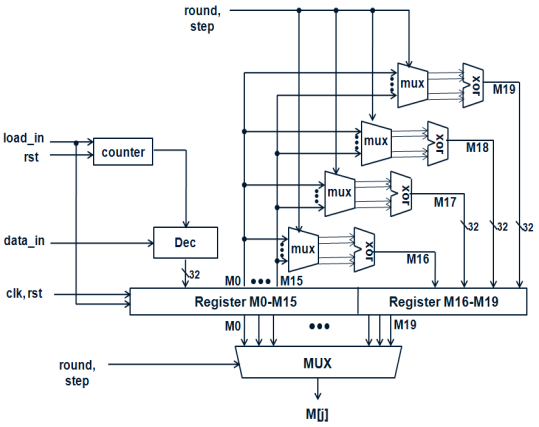

그림 4. M_Gen 블록

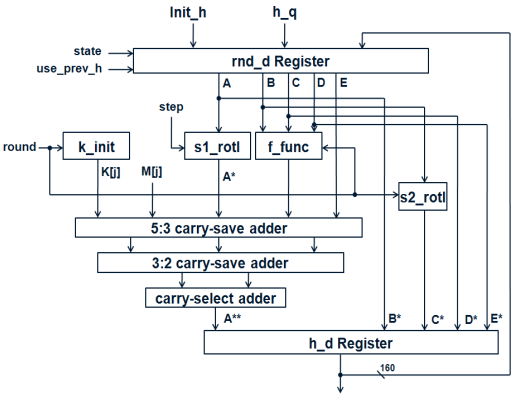

그림 5. 단계연산 블록

IV. 설계 검증 및 성능 평가

Verilog HDL로 설계된 HAS-160 해쉬 프로세서를 표준문서에 제시된 1,024 비트의 테스트 벡터를 사용

하여 기능을 검증하였다. 1,024 비트의 테스트 벡터는 512 비트의 블록 2개로 분할되어 순차적으로 연산에 사용되며, 첫 번째 512 비트의 메시지 블록으로부터 얻어진 160 비트의 해쉬코드와 두 번째 메시지 블록을 연산하여 최종 160 비트의 해쉬코드가 얻어진다. 시뮬레이션 결과는 그림 6과 같으며, 표준문서에 제시된 해쉬코드와 일치하는 결과가 얻어져 설계된 해쉬 프로세서의 논리기능이 정상 동작함을 확인하였다.

설계된 해쉬 프로세서를 0.35-㎛ CMOS 셀 라이브러리를 사용하여 합성한 결과, 17,600 게이트로 구현되었다. 타이밍 분석 결과 50 MHz의 주파수로 안전하게 동작 가능한 것으로 평가되었다. 160 비트의 해쉬코드 계산에 82 클록 사이클이 사용되므로, 설계된 해쉬 프로세서는 312 Mbps의 성능을 갖는다.

표 4는 설계된 HAS-160 프로세서의 성능을 비교한 것이다. 본 논문의 해쉬 프로세서는 문헌 [3] 보다 최대 동작 주파수가 느린 것으로 나타났으나 이는 논리합성에 사용된 공정의 차이로 인한 것이며, 동일한 공정을 사용하여 합성하면 비슷한 동작 속도가 얻어질 것으로 예상된다. 본 논문의 해쉬 프로세서는 문헌 [3]의 결과에 비해 약 16%의 적은 게이트 수로 구현되었으며, 따라서 저전력 소모가 요구되는 스마트카드의 보조프로세서에 적합하다. 동일한 동작주파수로 환산하면, 본 논문의 해쉬 프로세서는 문헌 [3]과 동일한 312 Mbps의 성능을 갖는다.

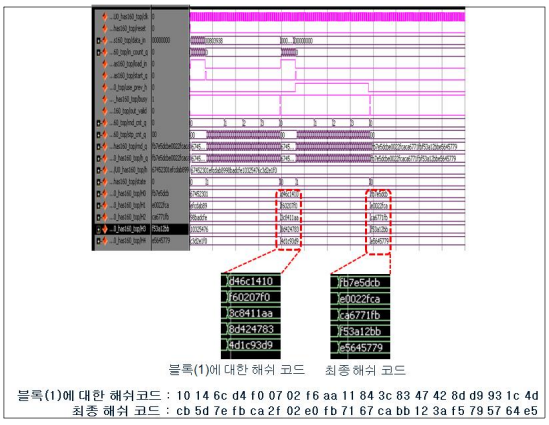

그림 6. 기능검증 결과

V. 결 론

한국형 표준 해쉬 알고리듬인 HAS-160을 구현하는 프로세서를 설계하였으며, 스마트카드 보안 보조프로세서로 사용될 수 있도록 동작속도 보다는 게이트 수와 저전력에 초점을 두어 구현하였다. 설계된 해쉬 프로세서는 0.35-μm CMOS 셀 라이브러리를 사용하여 50 MHz의 동작주파수로 합성한 결과 약 17,600개의 게이트로 구현되었으며, 312 Mbps의 성능을 가져 스마트카드에 IP 형태의 모듈로 활용될 수 있을 것이다.

표 4. 성능 비교

구분	[3]	본 논문
알고리듬	HAS-160	
단계 당 클록 수	1	
총 소요 클록 수	82	
데이터 I/O	32비트	
동작 주파수	175 MHz	50 MHz (Max 75 MHz)
연산 시간 (50 MHz 환산)	467 ns (1.6 us)	1.6 us
게이트 수	21,000 (1.0)	17,600 (0.84)
성능 (50 MHz 환산)	1,093 Mbps (312 Mbps)	312 Mbps
공정	0.25-㎛	0.35-㎛

참고문헌

[1] William Staillings, *Cryptography and Network Security, Principle and Practice*, 1999

[2] TTA, "Hash Function Algorithm Standard (HAS-160)", 12. 2000.

[3] Ju-Dai Hyun, Byeong-Yoon Choi, "Hardware Design of HAS-160 Algorithm", Dong-eui Univ., vol. 37, pp. 415-421, 08. 2002.

[4] 전신우, 김남영, 정용진, "SHA-1과 HAS-160과 의사 난수 발생기를 구현한 해쉬 프로세서 설계", *한국통신학회 논문지*, vol.27, pp. 112-121, 2002.

[5] Yongje Choi, Mooseop Kim, Taesung Kim, Howon Kim, "Low power implementation of SHA-1 algorithm for RFID system", *IEEE Int. Symp. on Consumer Electronics*, pp. 1-5, June, 2006.

[6] 성수학, "해쉬함수의 최근 동향", *http://mathnet.kaist.ac.kr/real/2006/6/text/sungsuhak.pdf*, June, 2006.

[7] Charanjit S. Jutla and Anindya C. Patthak, "Provably Good Codes for Hash Function Design", *IEEE Trans. on Information Theory*, vol. 55, no. 1, pp. 33-45, Jan. 2009.

[8] SHA-1 Standard, National Institute of Standards and Technology (NIST), Secure Hash Standard, FIPS PUB 180-1, *www.itl.nist.gov/fipspubs/fip180-1.htm*, 2003.

[9] Yadollah Eslaim, Ali Sheikholeami, P. Glenn Gulak, Shoichi Masui, Kenji Mukaida, "An Area Efficient Universal Cryptography Processor for Smart Cards", *IEEE Trans. on VLSI Systems*, vol. 4, pp. 43-56, Jan. 2006.

※ 2009년도 IT SoC 핵심설계인력양성 사업의 SoC 전공실습프로젝트 지원에 의한 연구 결과의 일부임.
※ 반도체설계교육센터(IDEC)의 CAD Tool 지원에 감사드립니다.