Full-pipelined CTR-AES를 이용한 Giga-bit 보안모듈 설계

*T.Q. Vinh, 박주현, 김영철
*전남대학교 전자컴퓨터공학부
e-mail : *jhpark@soc.chonnam.ac.kr, yckim@chonnam.ac.kr,

A Design of Giga-bit security module Using Fully pipelined CTR-AES

*T.Q. Vinh, Ju Hyun Park, Young chul Kim
Dept. of Electronics and Computer Engineering Chonnam National University

Abstract

In this paper, we presented our implementation of a counter mode AES based on Virtex4 FPGA. Our design exploits three advanced features: composite field arithmetic SubByte, efficient MixColumn transformation, and On-the-Fly Key-Scheduling for fully pipelined architecture. By pipelining the composite field implementation of the S-box, the area cost is reduced to average 17 percent. By designing the On-the-Fly key scheduling, we implemented an efficient key-expander module which is specialized for a pipelined architecture.

I. 서론

현재 가정과 소규모 사업장에서 재정적인 변화와 개인 커뮤니케이션 그리고 원격의료에 이르기 까지 점점 GPON의 사용이 일반화 되어가고 있다. 이러한 GPON의 사용에 힘입어 개인정보 보호와 커뮤니케이션 보호를 위한 보안의 필요성이 더더욱 커져가고 있다[1]. GPON의 보안을 위한 알고리즘은 5개의 AES(Advanced Encryption Standard)인 전자코드북(ECB), 암호 블럭 chaining(CBC), 암호 피드백 (CFB), 출력 피드백(OFB), counter 모드 중 counter mode를 채택하여 사용하고 있다[2].

현재까지 많은 논문에서 AES 알고리즘을 위한 하드웨어 구조를 제안하고 있다. 특히 대부분의 high throughput 을 위하여 pipeline 구조를 사용하고 있는데 그들의 구조는 outer, inner, 그리고 inner-outer 구조로 다양하게 구현되어 졌다[3][4][5]. AES 알고리즘의 기본 구조는 SubByte, ShiftRow, MixColumn 그리고 AddRoundKey로 구성되어 있고 11라운로 동작하며 마지막 라운드에서는 MixColumn은 제외된다.

이 논문에서 우리는 면적과 성능 면에서 최적화 되어진 full pipelined 구조를 이용하여 CTR-AES를 구현하였다.

본 논문의 구성은 다음과 같다. 본론에서는 우리가 제안한 CTR-AES 구조 특징을 SubByte, MixColumn, 그리고 key expander 측면에서 설명하고 제안된 구조의 구현 및 다른 구조와 비교 설명할 것이다. 그리고 결론을 맺고자 한다.

II. 본론

CTR AES는 Subbytes, ShiftRow, Mixcolumn, AddRoundkey 그리고 Key expander 블록으로 구성되어 있다. AES 총 11 round를 거쳐 ciphertext를 출력한다. 이 논문에서는 속도 향상을 위하여 loop를 사용하지 않고 inner and outer pipeline 구조, Subbyte에서는 composite field operation, 그리고 on-the-fly key expander 구조를 사용하여 속도를 향상시켰다.

2.1 Subbyte transformation

Subbyte transformation 에서는 일반적으로 LUT를 사용하는 경우도 있으나 면적을 줄이고 메모리 요구를 없애기 위하여 우리는 Galios Field 연산을 사용하였다. 기본 연산은 $GF(2^8)$에서 multiplicative inverse를 계산하고 affine transformation을 적용한다. 그러나 $GF(2^8)$에서의 multiplicative inverse 계산은 많은 비용 때문에 $GF(2^4)$을 이용하여 $GF(2^8)$을 계산하였다.

그림 1은 GF 연산을 이용한 non-pipelined S-box 구조를 보여주고 있는데 입력 값은 $GF(2^4)$의 두 연산으로 나누어지고 난 후 multiplicative inverse 값이 계산되어진다. 이 계산되어진 두개의 $GF(2^4)$ 값들은 다시 $GF(2^8)$으로 mapping 되어 지고 마지막 과정으로

* 본 논문은 ETRI 연구비 와 IDEC의 CAD 툴 지원에 의한 것임

affine transformation 이 수행되어진다.

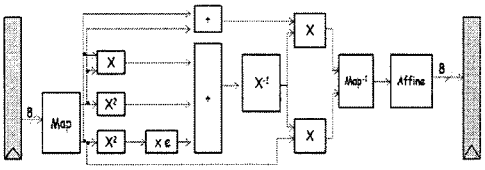

그림 1. GF 연산을 이용한 Non-pipelined S-box

비록 S-box를 위한 composite field 구현이 면적 면에서 효과적이긴 하지만 긴 critical path의 문제점이 있다. 이러한 단점을 해결하기 위하여 파이프라인 구조를 사용하는데 적절한 위치에 사용하지 못하는 파이프라인 구조는 레지스터의 사용을 증가시키므로 면적의 증가를 가져오는 문제점이 있다.

그림 2는 2-stage pipeline 구조를 보여주고 있는데 critical path는 반으로 줄어들었고 단지 3개의 4bit 레지스터만 사용되었다. 또한 그림 3은 3 stage pipeline 구조를 보여주고 있다[4].

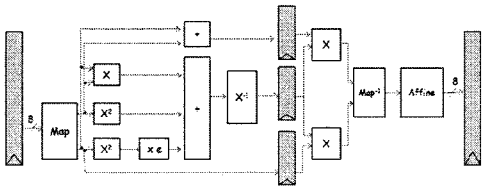

그림 2. 2-stage 파이프라인 S-box

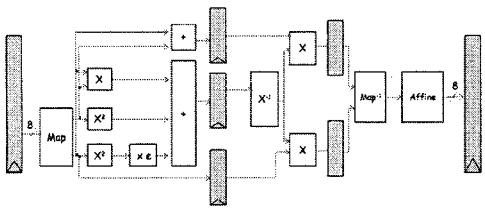

그림 3. 3-stage 파이프라인 S-box

2.2 MixColumn

Mixcolumn transformation에서는 column의 영역은 GF(2^8) 영역의 다항식으로 간주하고 고정된 다항식 c(x) = '03' x^3 + '01' x^2 + '01' x + '02 을 가진 modulo x^4+ 1를 곱하여 계산한다. 기본 MixColumn transformation은 식(1)과 같이 표현할 수 있다.

$$\begin{cases} s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c} \\ s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c} \\ s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}) \\ s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}) \end{cases} \quad (1)$$

MixColumn transformation 구현을 위한 몇몇의 구조들이 제안 되어졌으며 그 중 substructure-shared 구조들이 구현되어졌다 [4][6][7][9]. 우리의 구조에서도 substructure sharing 기법을 적용하였으며 이 기법을 적용하기 위하여 (1)의 식은 다음의 형태로 바꾸어 져야 한다.

$$\begin{cases} s'_{0,c} = \{02\} \bullet (s_{0,c} \oplus s_{1,c}) \oplus s_{1,c} \oplus (s_{2,c} \oplus s_{3,c}) \\ s'_{1,c} = \{02\} \bullet (s_{1,c} \oplus s_{2,c}) \oplus s_{0,c} \oplus (s_{2,c} \oplus s_{3,c}) \\ s'_{2,c} = \{02\} \bullet (s_{2,c} \oplus s_{3,c}) \oplus s_{3,c} \oplus (s_{0,c} \oplus s_{1,c}) \\ s'_{3,c} = \{02\} \bullet (s_{3,c} \oplus s_{0,c}) \oplus s_{2,c} \oplus (s_{0,c} \oplus s_{1,c}) \end{cases} \quad (2)$$

식(2)의 MixColumn transformation 공식은 대칭적 구조를 가지고 있으며 하드웨어 구현의 면적 최적화를 위한 substructure sharing를 적용할 수 있게 되었다. {02} 상수 곱셈은 a= xtime(b) 함수에 의해 계산되어 지는데 함수 xtime()는 left shift 와 subsequent 조건적 bitwise XOR 로서 바이트 level에서 구현 가능하다. 효과적인 xtime() 구조와 XOR-sharing을 적용하므로 최적의 MixColumn transformation을 구현할 수 있으며 구조는 그림 4와 같다.

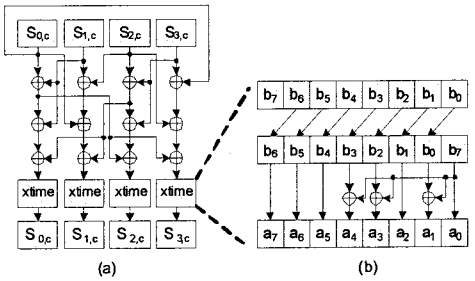

그림 4. (a) MixColumn의 효과적인 구조
(b) xtime() 함수의 구현 구조

2.3 Key Expander 블록

128 bit AES 알고리즘에서 Key expansion 함수는 초기 키 값으로부터 11 라운드 키 값을 생성한다. 파이프라인 AES 구조의 경우, 모든 라운드 키 값은 동시에 가능해야만 한다. 그러므

로 몇몇 연구자들은 첫 번째 라운드 키 값을 계산한 후 10라운드를 위하여 하드웨어를 10번 반복 복사하여 구현하였다[4][5]. 이러한 구조는 동시에 모든 라운드 키 값을 계산할 수 있는 장점이 있지만, 많은 면적을 소모하게 된다. Xinmiao Zhang는 on-the-fly 방법에서 동작할 수 있는 key expander를 제안하였다[9].

이 논문에서 우리는 면적 효과적이면서 on-the-fly 방법에서 라운드 키를 계산할 수 있는 key expander를 구현하였다. Sub-pipelined 라운드 프로세서의 대칭적 동작을 위하여 r개의 sub-stage로 key expander를 나누었다.

우리는 11 라운드 키 값을 저장하기 위하여 11 레지스터를 사용하는데 이 구조는 [9]에서 사용하는 구조와는 다르다. [9]에서 사용된 구조는 모든 라운드 키 값과 내부 pipeline stage를 위한 임시 값 저장을 위하여 r sets의 레지스터를 사용하였는데 이 방법을 이용하여 우리는 면적을 줄일 수 있었다.

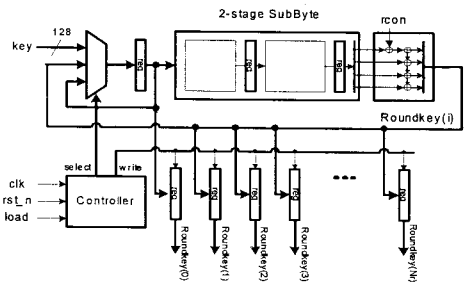

그림 5. On-the-fly key expander 구조

3-stage를 갖는 on-the-fly key expander 용 sub-pipelined 구조는 그림 5와 같다.

Ⅲ. 구현 및 성능비교

우리는 Virtex-4 VLX100-12 에서 128bit CTR-AES의 full pipelined 구조를 구현하였다. Xilinix ISE 8.2i를 이용하여 합성하였고 post placement timing를 확인하였다. 암호화 및 복호화 동작의 시뮬레이션을 위하여 ModelSim 5.8c을 사용하였다.

LUT's, 레지스터, slides, 그리고 최대 주파수에 관점에서 우리의 설계를 평가하였다.

이 논문에서 3 sub-pipelined stage를 가진 full-pipe
lined 구조를 구현하였는데 전체 31 stage가 소모되었다. 그러므로 31 clock 사이클 후에 해당되는 입력 값의 암호화 된 블록이 처음 출력되며 매 clock 마다 다음 암호화된 블록을 출력한다.

구현된 구조를 통하여 우리는 26.7 Gbits/s의 throughput 을 얻을 수 있었고 표 1은 지금까지 구현되어진 AES 와 우리의 결과를 비교하였다.

기존의 AES 는 VirtexE device에서 구현하였으므로 정확한 비교를 위하여 우리의 결과도 VirtexE- family에서 다시 합성 및 검증하였다.

Ⅳ. 결론 및 향후 연구 방향

이 논문에서 우리는 AES 알고리즘을 위한 full pipelind 구조를 구현하였다. 우리의 설계에서는 3가지 주요한 특징이 있는데 (1) composite field arithmetic SubByte, (2) 면적 효과적인 MixColumn, (3) on-the-fly sub-pipeline 된 key expander 이다. 전체 31 stage가 소모되었으며 Virtex4 VLX 100 device에서 26.7 Gbits/s throughput을 구현하였다. 그러므로 우리 구조는 GPON system의 암호화에 적당한 구조이다.

표 1. AES 알고리즘의 FPGA 구현의 비교

Design	Device	Frequency (MHz)	Throughput (Mbps)	slices	BRAMs	Mbps/slice
Shuenn-Shyang [3]	XCV1000e-8	125.38	1604	1857	0	0.867
Jae-Gon Lee [4]	XCV3200e-8	40	5120	8009	104	0.639
Saqib, N.A. [5]	XCV812e-8	20.192	2584	2744	0	0.942
Jarvinen [7]	XCV1000e-8	129.2	16500	11719	0	1.408
Xinmiao Zhang (r=3) [9]	XCV812e-8	93.5	11965	9406	0	1.272
Our design (r=3)	XCV1000e-8	121.24	15518	10957	0	1.416
Our design (r=3)	XC4VLX100-12	208.49	26686	9478	0	2.816

참고문헌

[1] "Gigabit-capable Passive Optical Networks (G-PON) : Transmission convergence layer specification", ITU-T G.984.3 Amendment 1, July. 2005

[2] Morris Dworkin, "Recommendation for Block Cipher Modes of Operation", NIST Special Publication, http://csrc.nist.gov/ CryptoToolkit/ modes/, 2001.

[3] Shuenn-Shyang Wang; Wan-Sheng Ni, "An efficient FPGA implementation of advanced encryption standard algorithm", Proceedings of the 2004 International Symposium on Circuits and Systems, Vol 2, 23-26 May 2004 Page(s):II 597-600 Vol.2

[4] Jae-Gon Lee, Woong Hwangbo, Seonpil Kim, Chong-Min Kyung, "Top-down implementation of pipelined AES cipher and its verification with FPGA-based simulation accelerator", 6th International Conference On ASIC Proceedings, page(s): 68- 72, 24-27 Oct. 2005.

[5] Saqib, N.A.; Rodriguez-Henriquez, F.; Diaz-Perez, A, "AES algorithm implementation - an efficient approach for sequential and pipeline architectures", Proceedings of the Fourth Mexican International Conference on Computer Science, page(s):126 - 130, 8-12 Sept. 2003.

[6] Nedjah, N.; de Macedo Mourelle, L.; Cardoso, M.P., "A Compact Pipelined Hardware Implementation of the AES-128 Cipher", ITNG 2006. Third International Conference on Information Technology: New Generations, page(s):216 - 221, 10-12 April 2006.

[7] Yongzhi Fu; Lin Hao; Xuejie Zhang; Rujin Yang "Design of an extremely high performance counter mode AES reconfigurable processor", Second International Conference on Embedded Software and Systems , 16-18 Dec. 2005 Page(s):7 pp.

[8] Hodjat, A.; Verbauwhede, I., "Area-throughput trade-offs for fully pipelined 30 to 70 Gbits/s AES processors", IEEE Transactions on Computers, Volume 55, Issue 4, page(s):366 - 372, April 2006

[9] Xinmiao Zhang; Parhi, K.K., "High-speed VLSI architectures for the AES algorithm", IEEE Transactions on Very Large Scale Integration(VLSI) Systems, Vol 12, Issue 9, page(s):957 - 967, Sept. 2004.

[10] V. Rijmen, "Efficient Implementation of the Rijndael SBox", http://www.esat.kuleuven.ac.be/~rijmen/rijndael/ .

[11] J. Wolkerstorfer, E. Oswald, and M. Lamberger, "An ASIC Implementation of the AES Sboxes", Proc. RSA Conf. 2002, Feb. 2002