情 報 保 護 學 會 誌
第 16 卷 第 3 號, 2006. 6

암호 프로세서의 고속 구현 핵심 기술

장 태 주*

요 약

고속 암호프로세서는 매우 큰 대역폭을 필요로 하는 네트워크 보안 장비, 서버 시스템의 보안의 필수 요소이다. 암호 프로세서는 고속 대용량 처리를 위한 고성능 쪽과 유비쿼터스 등 이동 환경에 적합한 초소형·저전력 쪽으로 크게 두 가지로 나누어 질 수 있다. 이 논문에서는 암호 프로세서의 고속 구현의 몇 가지 요소 기술 들을 살펴 본다. 일반적으로 디지털 논리 설계에 많이 쓰이고 있는 파이프라인 기법과 이를 적용한 결과들을 살펴보고, 여러 개의 암호 코어를 쓰는 방법, 하나의 암호 코어로 여러 개의 세션을 처리할 때 속도 저하를 막기 위한 세션 변경 방법을 설명한다. 끝으로 처리 성능에 영향을 주는 인터페이스 부분을 USB2.0의 보기를 들어 살펴본다.

Ⅰ. 서 론

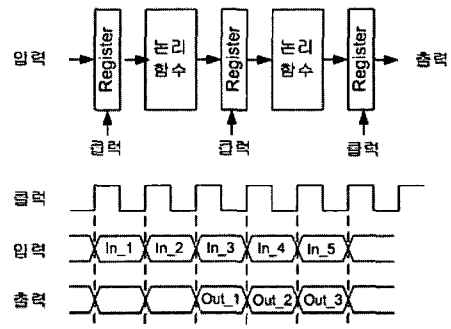
암호 프로세서는 고속 대용량 처리를 위한 고성능 쪽과 유비쿼터스 등 이동 환경에 요구 만족을 위한 초소형·저전력 쪽으로 크게 두 가지로 나누어 질 수 있다. 고성능을 위해서는 전력을 많이 소모하는 것이 일반적이며 최근에는 저전력·고성능으로 발전을 하고 있다. 여기서는 암호 프로세서의 고속 구현의 몇 가지 요소 기술 들을 살펴보고자 한다. 고속의 암호프로세서는 매우 큰 대역폭을 필요로 하는 네트워크 보안 장비, 서버 시스템 보안의 필수 요소이다[1,2].

일반적으로 디지털 논리 설계에 많이 쓰이고 있는 파이프라인 기법과 이를 적용한 결과들을 살펴보고, 여러개의 암호 코어 쓰는 방법, 하나의 암호 코어로 여러 개의 세션을 처리할 때 속도 저하를 막기 위한 세션 변경 방법을 설명한다. 끝으로 처리 성능에 영향을 주는 인터페이스 부분을 USB2.0의 보기를 들어 살펴본다.

Ⅱ. 고속 구현 기술

1. 파이프라인 기법

파이프라인 기법은 여러 명령 또는 단위 연산을 겹쳐서 실행하도록 구현하는 것을 말한다. 파이프라인은 단(stages)으로 나누어지고 각 단은 전체 수행 논리의 일부를 수행한다. 각 단은 서로 연결되어 파이프를 이룬다. 파이프라인 설계자의 목표는 각 파이프 단의 길이의 균형을 잘 맞추는 것이다.



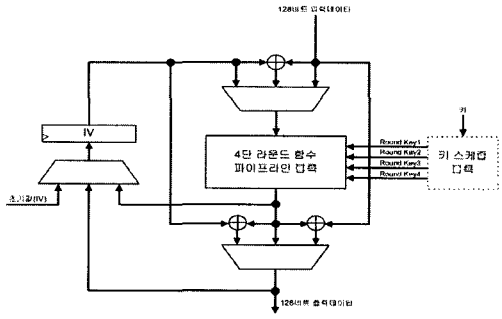
〔그림 1〕 파이프라인 기법

각 구현 결과에 대하여 자원의 사용, 적용 소자 등이 다르기 때문에 직접적인 비교는 어려우며, 특히 같은 구현도 사용 소자의 공정 향상에 따라 처리 속도가 향상된다. 처리 속도 관점에서 AES 블록 암호에 대한 FPGA 구현 결과를 살펴 보면, 2001년 7Gpbs 처리 속도가 발표되었으며[3], 부분 파이프라인 기법을 적용하여 15Gbps 구현 결과도 발표 되었다[4]. 최근 완전

* ETRI 부설 국가보안기술연구소 책임연구원

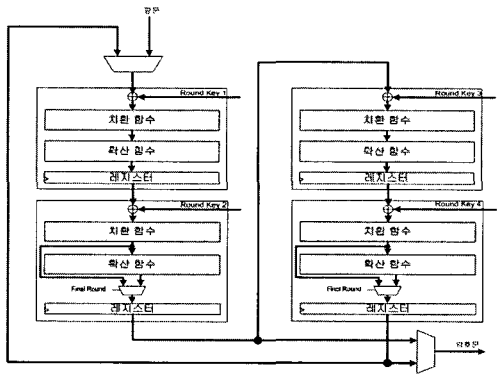
한 파이프라인 기법과 메모리를 쓰지 않고 17.8Gbps 구현 결과가 있으며[5], 각 바이트 치환 함수를 FPGA 내부 블록 RAM을 테이블 구현과 갈로아 체 연산을 혼합하여 VIrtexII-Pro FPGA에서 21.54Gbps 구현 결과가 있다[6].

ARIA의 구현으로는 4단 파이프라인으로 128비트 처리에 14 클럭이 소요되어 삼성 std-130 0.18um 공정 기준으로 최대 125MHz에서 ECB 3.7Gbps, OFB 1.1Gbps(파이프라인 없음)를 얻었다[7]. 고속 ARIA 블록암호 연산기의 구성은 크게 라운드 함수 및 운영모드 처리 블록과 키 스케줄 블록으로 나누어진다. 먼저 라운드 함수 및 운영모드 처리 블록의 구조에 대해 살펴보면 [그림 2]와 같다. 그림에서 보는 것처럼 라운드 함수 및 운영모드 처리 블록은 4단 라운드 함수 파이프라인 블록을 중심으로 입/출력 데이터가 Mux에 의해 선택적으로 결정되는 구조를 가진다. 이러한 입/출력 데이터 Mux 구조를 통해 ECB, CBC, CTR, CFB, OFB 등의 다양한 운영모드를 지원하게 된다. [그림 33]은 라운드 함수 및 운영모드 처리 블록을 구성하는 4단 라운드 함수 파이프라인 블록의 구조를 보여주고 있다. 그림에서 보는 것처럼 4단 라운드 함수 파이프라인 블록은 라운드 함수 4개를 차례로 연결하고 라운드 함수 중간 중간에 레지스터를 삽입하는 전형적인 파이프라이닝 구조를 가진다. 이러한 파이프라인 구조는 병렬처리가 가능한 운영모드인 ECB와 CTR 모드에서 라운드 반복형 구조에 비해 4배의 성능을 가지도록 한다.

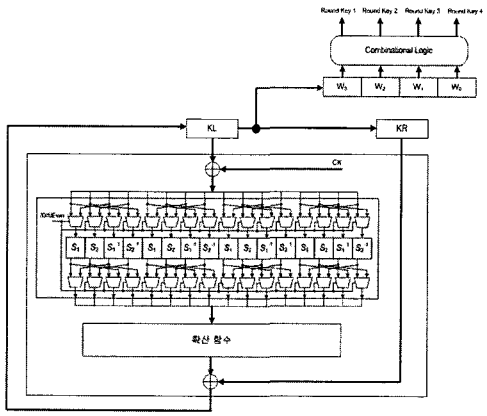


[그림 2] 라운드 함수 및 운영모드 처리 블록의 data path 연결도

키 스케줄 블록은 마스터키를 입력받아 4단 라운드 함수에 필요한 라운드 키를 동작 타이밍에 맞게 출력시켜주는 역할을 담당한다. 키 스케줄 블록의 구조를 살펴보면 [그림 4]와 같다. 그림에서 보는 것처럼 키 스케줄은 라운드 함수와 비슷하게 치환함수, 확산함수를 중심으로 구성된다. 여기서 치환함수(S-Box)는 홀수 라운드인지 짝수 라운드인지에 따라 달라지므로 /OddEven 신호를 기준으로 선택적으로 동작하도록 구현한다. 확산함수는 라운드 함수에서의 확산함수와 동일하다.



[그림 3] 4단 라운드 함수 파이프라인 블록의 구조



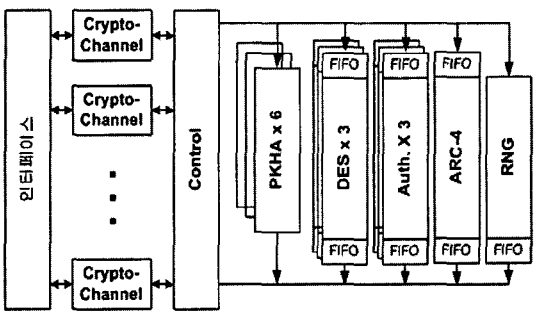
[그림 4] 키 스케줄 블록의 구조

결론적으로 파이프라인의 설계는 각 단의 길이의 균형을 맞추는 것이며, 암호 알고리즘 구조, 파이프라인 개수 등 여러 가지 요인에 따라 달라 질 수 있다.

2. 다중 코어 적용

최근의 고성능 암호 프로세서는 고성능 요구조건을 충족시키기 위하여 여러 개의 코어를 탑재하는 방향으로 구현하고 있다. 여기서는 모토롤러사의 MCP190 프로세서[8]를 살펴보기로 한다.

MPC190 프로세서는 암호 알고리즘 블록의 수를 달리하며 제어기에서 일(Job)을 스케줄 하도록 설계되어 있다. MPC190을 구성하고 있는 암호 알고리즘 블록의 수는 긴 연산 시간이 소요되는 공개키 알고리즘은 6개의 블록으로, 대칭키 알고리듬과 해쉬 알고리즘은 각각 3개의 블록으로 이루어져있다. MPC190의 컨트롤러는 입력 패킷을 분석하여 암호 블록과 패킷의 채널을 스케줄 한다. MPC190의 블록도는 [그림 5]와 같다. PCI 버스 입출력은 PCI v2.2를 지원하고, 32비트와 64 비트 워드 단위로 데이터를 전송한다. MPC190은 PCI를 통해서 디스크립터를 전달받아, Crypto-channel에 전달하도록 프로그램 된다. Crypto-channel의 디스크립터의 명령은 컨트롤러를 통해 해석되고 암호 명령이 수행된다. Crypto-channel은 디스크립터의 포인터를 처리하고, 추가 데이터나 명령을 전달 받기 위해서 PCI 버스를 초기화한다. 처리된 데이터는 각각의 출력 버퍼에 저장되고, PCI 버스를 통해서 시스템 메모리에 저장된다.



[그림 5] MPC190 블록도

MPC190 암호 프로세서는 모토로라의 S1 암호 프로세서 계열 제품이다. MPC190은 PCI를 지원하는 네트워킹 장비나 컴퓨터에서 사용하기 쉬운 고성능의 프로세서이다. MPC190 프로세서는 IPsec, IKE, WTLS/WAP, SSL/TLS와 같은 프로토콜에 최적화되어 있다. 지원하는 암호 알고리즘은 ECC/RSA/Diffe-Hellman, DES/3DES, SHA-1, MD-4/5 및 ARC-4이다. 인터넷 키 교환(IKE) 프로토콜에서 1024bit Diffe-Hellman 알고리즘을 사용하는 경우 520 connection/sec, 155bit ECC 알고리즘인 경우 1000 connection/sec의 성능을 낼 수 있다. 블록 암호와 인증에서는 3DES_HMAC-SHA-1을 사용하는 경우 0.5 Gbps, DES를 사용하는 경우 1.13 Gbps, 3DES를 사용하면 0.68 Gbps의 성능을 낼 수 있다. MD5를 이용한 해쉬 연산은 0.97 Gbps의 성능을 보이고 있다.

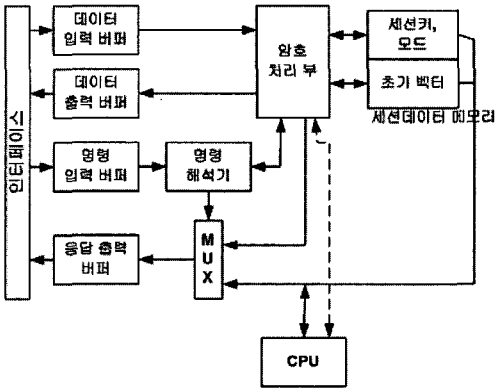
[7]의 구현에서도 4개까지의 코어를 구성하여 시뮬레이션 해 보았으며, 하드웨어 자원이 허용되는 범위까지 확장이 가능하다.

여러 개의 암호 코어를 써서 최적의 성능을 내기 위해서는 적용환경을 잘 분석하여 효율적인 작업 분배가 이루어지도록 하는 것이 필요하다[9]. 지금까지 대부분의 분배기 연구는 소프트웨어적인 처리하고 있으며, 하드웨어 관점에서 연구는 미흡하며 구현되어 쓰이고 있는 것도 적다. 분배기는 응용분야 및 사용환경에 따라 장단점을 비교하여 가장 우수한 성능을 낼 수 있는 기법을 연구하는 것이 필요하며, 하드웨어로 구현한다면 구조 변경이 어려우므로 환경에 대한 분석을 잘 하여야 한다.

3. 고속 세션 변경

각종 암호 알고리즘 처리동작에서의 세션이란 동일한 키를 이용하여 암호화되는 시간 구간을 의미한다. 한 세션에서 사용되는 블록 암호 알고리즘의 키를 세션 키(Session Key)라 부르며 세션 키는 암호 통신을 하고 있는 쌍방이 정해진 키 공유 알고리즘에 의해 서로 공유하게 된다. 하나의 세션이 정의되면 세션 키뿐만 아니라 암호 운용 모드, 초기화 벡터(initial vector) 등도 같이 결정되어 진다. 세션 변경은 암호화 장치의 암호 통신 대상이 바뀜으로 인해 암호 함수의 세션 키, 운용모드, 초기화 벡터 등의 데이터가 변경되는 동작으로 정의하기로 한다.

암호화 장치에서 세션 변경 동작은 일반적으로 장치 내의 CPU에 의해 수행되어 진다. 이러한 CPU의 세션 변경 동작은 세션 변경 지연시간을 유발하게 되며 이러한 처리 지연 현상은 세션 변경이 고속으로 일어나는 네트워크 환경에서는 암호 처리 성능을 저하 시키는 주요 요인이 된다. 세션 변경 동작을 고속으로 수행하기 위하여 하나의 장비 내에 다수의 암호칩 또는 암호 모듈을 내장하여 여러 세션에 대한 암호화 동작들을 몇 개의 암호칩 또는 암호 모듈로 분산시키는 형태로 구현할 수도 있으나 이 경우엔 장비의 구현 복잡도를 증가시킬 뿐만 아니라 가격을 상승시키는 요인이 되어 제품의 경쟁력을 떨어뜨리게 된다. 이 절에서는 이러한 문제를 해결하기 위한 구현 기술로 고속 세션 변경 방법[10]에 대하여 살펴보기로 한다.



〔그림 6〕 고속 세션 변경 개념도

〔그림 6〕은 고속 세션 변경의 개념도이다. 구성은 인터페이스, CPU 및 메모리, 버퍼 메모리 등의 구성 요소들을 갖는 범용성 있는 암호화 장치의 일반적인 구조를 기본으로 하고, 장치내에 세션 변경에 필요한 데이터를 저장하는 세션 데이터 메모리 및 하드웨어 명령 해석기가 있으며, 암호처리부는 고속 세션 변경이 쉬운 구조를 갖는다.

인터페이스는 외부 호스트 장비와 버퍼 사이의 정합 기능을 담당하는 블록으로 외부로는 PCI, PCMCIA 등과 같은 표준 정합 기능을 수행하며 내부는 버퍼 스위치 기능을 수행한다. 버스 스위치는 호스트의 입력 데이터를 명령 데이터와 처리용 데이터로 구분하여 각각 명령 입력 버퍼와 데이터 입력 버퍼로 전달하고, 응답 출력 버퍼로부터 전달되는 응답 데이터와 데이터 출력 버퍼를 통해 전달되는 암복호 처리 결과 데이터를 외부 인터페이스로 전달하는 기능을 수행한다.

명령 해석기는 명령 입력 버퍼에 저장된 호스트의 명령 데이터를 해석하여 그 명령이 암복호 처리 명령인지 여부를 판단한다. 암복호 처리용 명령이면 CPU 대신 응답 데이터를 생성하여 응답 출력 버퍼로 전달하고, 세션 ID, 암복호 처리용 입력 데이터 길이 등과 같은 암복호 동작을 수행하는데 필요한 변수들을 암호칩에 전달하여 암복호 처리 동작의 시작을 명령한다. 이는 암복호 처리 동작에 대한 명령 해석, 명령에 대한 응답 과정들을 CPU의 중개 없이 하드웨어에 의해 처리하도록 함으로서 세션 변경 동작을 고속으로 수행하기 위함이다. 호스트의 명령이 암복호 처리용 명령이 아니면 CPU가 처리해야 할 명령이므로 명령 데이터를 CPU로 전달한다.

CPU는 호스트 장비로부터 전달되는 명령을 해석하여 이를 수행할 수 있도록 암호화 장치 전반을 제어하고 명령 수행 결과에 해당되는 응답 데이터를 생성하여 호스트 장비로 전달한다. 단, 암복호 처리 명령의 경우 CPU는 응답 데이터를 생성하지 않고 명령 해석기가 응답 데이터를 생성한다.

세션 데이터 메모리는 세션 관련 데이터가 저장되는 메모리로 각 세션에 따른 세션 키, 초기화 벡터, 운용 모드 및 세션의 정상 여부를 나타내는 데이터 등이 저장된다. 각 세션에 따른 세션 관련 데이터의 저장 어드레스 값은 해당 세션의 ID 값에 의해 결정되며 초기화 벡터와 세션 키 데이터를 하나의 메모리에 저장할 수도 있으나 세션 변경 속도를 고속화하기 위해서는 각각의 메모리를 따로 둠이 바람직하다. 암호화 장치가 세션 변경을 위한 동작을 수행할 경우에는 암호처리부가 세션 데이터 메모리를 직접 액세스한다. CPU가 세션 초기화 동작을 수행할 경우에는 직접 액세스 할 수 있다. 암호처리부는 블록 암호 알고리즘이 구현되는 칩으로 블록 암호화 혹은 복호화 동작을 수행한다.

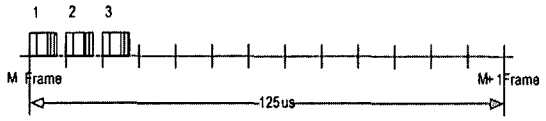
결론적으로 암복호 동작 수행에 있어 세션 변경이 일어나면 명령해석기가 처리하고 암호처리부에서 세션 메모리에 직접 접근하여 처리함으로써 처리 속도를 매우 향상 시킬 수 있다.

4. 인터페이스

최근 암호모듈의 인터페이스로서 고성능의 서버나 네트워크 장비의 경우 PCI가 많이 사용되고 있으며 휴대용으로는 유연성, 확장성 그리고 사용 편이성 등의 이유로 USB가 많이 사용되고 있다. 이 절에서는 USB 인터페이스에 대한 고려사항을 살펴본다[11].

USB 인터페이스의 규격은 USB1.1과 USB2.0 규격이 있으며 USB2.0의 규격은 최대 480Mbps의 이론적 성능을 낼 수 있다. PCI 규격 역시 요즘 많이 사용되는 PCI2.1, PCI-X, PCI-Express등 수 Gbps에서 수십 Gbps 까지 다양한 규격이 제시되어 사용되고 있다. 그러나 이들 인터페이스의 최대 성능은 대량의 데이터가 한 번에 전송되었을 경우에 측정되는 성능으로 작은 크기의 데이터가 전송될 경우 그 성능은 상당한 차이가 난다. 따라서 암호 프로세서의 모든 데이터 입출력은 인터페이스를 통하여 이루어지기 때문에 암호 코어 처리 속도와 상관없이 인터페이스 제약조건에 따라 성능이 저하가 일어난다. 이러한 현상은 USB2.0규격을 분석해보면 알 수 있다.

USB2.0 규격에서 마이크로 프레임이란 High Speed 버스에서 USB 통신이 이루어지는 기본 시간 단위를 말하며 1 마이크로 프레임을 125㎲로 정의한다. 즉 USB2.0 규격에서는 마이크로 프레임 단위로 데이터 통신이 발생하며 아무리 적은 데이터라도 데이터 입/출력 동작에 125㎲의 시간을 소모하게 된다. 따라서 데이터의 전송 크기가 작을수록 전송 성능이 떨어진다. 예를 들어 1,500 바이트 데이터 전송을 위해서는 한 번의 데이터 입출력 동작에서 전송할 수 있는 데이터의 최대 크기는 512 바이트이므로 1마이크로 프레임 동안 3번의 전송(transfer)만 발생하며 나머지 10회는 데이터 전송 없이 시간만 소비하게 된다. 따라서 1,500 바이트 데이터의 전송 속도는 이론적으로 1500x8비트/125㎲=96Mbps가 되며 최대 성능 480Mbps 의 1/5 정도로 제한된다. [그림 7]은 이러한 현상을 그림으로 나타낸 것이다.



[그림 7] 1마이크로 프레임에서 1,500바이트 데이터 전송

[표 1]은 USB2.0 규격에서 입출력 데이터 크기별 이론적 최대 전송 속도를 나타낸 것이다. 페이로드(payload)의 크기가 512 바이트일 경우엔 1 마이크로 프레임 동안 최대 13회의 데이터 전송이 가능하다. 따라서 데이터 전송 속도는 (512 × 8(bit)) × 13 / 125㎲ = 426.0Mbps로 거의 최대 전송 속도(480Mbps)에 가까운 수치를 보이고 있으나, 데이터 페이로드의 크기가 4바이트 일 경우 최대 127회의 전송이 가능하므로 (4 × 8(bit)) × 127 / 125㎲ = 32Mbps로 전송 성능이 떨어지는 것을 알 수 있다.

보안서비스 모듈 또는 SoC 암호 프로세서는 명령

[표 1] USB2.0 High Speed Bulk Transaction Limits

Data Payload	Max Bandwidth (Mbps)	Max Transfer
1	8.512	133
4	32.512	127
16	107.52	86
128	327.68	40
512	426.00	13

[표 2] 암호처리 명령 수행 절차

수행 단계	설명
① 명령 읽기	호스트 장비로부터 전달된 암호처리 명령을 읽어옴
② 응답 출력	명령 해석 결과를 호스트 장비로 전달함
③ 암호처리용 데이터 읽기	호스트 장비로부터 암호처리용 데이터를 읽어옴. 동시에 입력 데이터에 대한 암호처리동작을 수행함
④ 암호처리 결과 데이터 출력	암호 처리 결과 데이터를 호스트 장비로 전달함

의 종류에 따라 차이가 있기는 하지만 보통 암호처리 동작을 수행하기 위해 기본적으로 다음과 같은 절차에 따라 호스트 장비와 통신을 하게 된다.

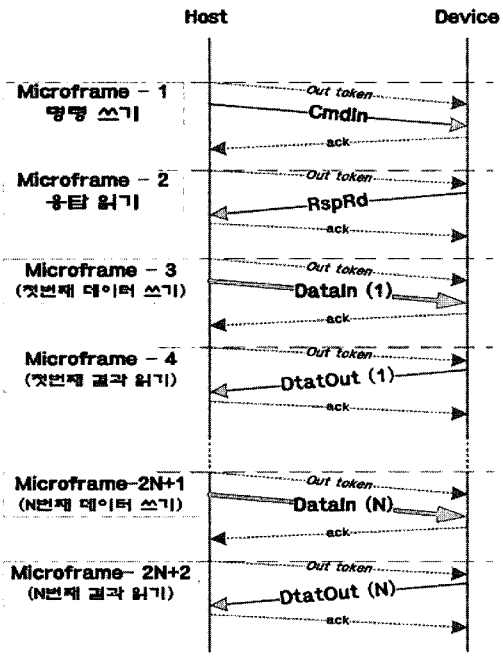
명령 수행 절차상의 명령 및 데이터의 모든 입출력 동작은 USB 통신을 통해 이루어지며 명령 읽기와 같이 작은 크기의 명령 데이터를 읽어오는 동작에서도 125㎲의 한 마이크로 프레임을 소모하게 된다.

명령 수행 단계에서 암호화 동작이 일어나는 단계는 암호처리용 데이터 읽기 단계와 암호처리 결과 데이터 출력 단계이므로 명령 읽기 단계나 응답 출력 단계는 일종의 부가적인 단계로서 암호 처리 성능을 저하시키는 주요 요인으로 작용한다. 이러한 현상은 패킷 암호화와 같이 작은 크기의 데이터를 암호화해야 하는 경우엔 급격한 성능저하 현상을 유발할 수 있다.

[그림 8]은 바이트에 대한 암호 명령 처리 절차를 USB2.0 통신 개념과 접목시켜 나타낸 것이다. 명령 읽기 및 응답 쓰기 단계도 1 마이크로 프레임을 사용한다. 따라서 N바이트를 암호화하는데 명령 읽기와 응답 출력에 2마이크로프레임, N바이트를 전송하는데 한 마이크로 프레임당 6,656바이트를 전송할 수 있으므로 데이터 읽기와 출력에 각각 (⌈N/6656⌉) 마이크로프레임이 전송된다. 따라서 총 전송되는 프레임은 2+(2×(⌈N/6656⌉))이 소요되므로 암호성능은 (식 1)과 같이 표현될 수 있다.

$$\text{성능} = \frac{N \times 8}{\{2+(2\times \lceil N/6656 \rceil)\}\times 125}[Mbps] \quad (1)$$

N의 값이 충분히 클 경우엔 명령 읽기와 응답 쓰기 단계에서 소모한 2마이크로 프레임의 시간이 전체 시간에 비해 그리 크지 않으나 N의 값이 작은 경우엔 이



〔그림 8〕 N바이트 암호처리

시간이 실제 암호화 동작을 수행하는 시간보다 더 커지게 된다. 여기서 명령입력과 데이터 입력을 합치고, 응답출력을 데이터 출력과 합치면 2배로 속도 향상이 일어남을 쉽게 알 수 있다. 따라서, 사용되는 시스템 환경을 고려하여 인터페이스를 설계하여야 한다.

Ⅲ. 결 론

고성능의 암호 프로세서 설계를 위한 몇 가지 요소 기술들을 살펴보았다. 암호 코어 자체의 성능 향상을 위해서 일반적으로 파이프라인 기법을 적용하고 있으며, 암호 코어를 여러 개 써서 바라는 처리 속도를 얻을 수도 있다. 여러 개의 코어를 사용 할 때는 작업을 분배하는 방법이 더 중요 할 수 있다. 고속 세션 변경은 주로 서버급 프로세서에 적용되나 클라이언트용 암호 프로세서에도 적용되기도 한다. 암호 프로세서 자체의 성능도 중요하지만 인터페이스의 제약 때문에 성능저하가 일어나므로 프로세서 설계 시 이점도 고려하여야 한다.

참 고 문 헌

〔1〕 주학수, 주홍돈, 김승주, "고속 암호연산 프로세서 개발 현황", 정보보호학회지, 제12권3호, 2002. 6.

〔2〕 김기현, 한종욱, "Giga급 보안 프로세서 및 VPN 장비 기술 동향", 주간기술동향1131, 2004. 3.

〔3〕 McLoone at al, "High performance single-chip FPGA Rijndael algorithm implementation," *CHESS 2001*, Paris, France, pp. 68-79, May 2001.

〔4〕 구본석, 이상한, "Rijndael 블록암호 알고리즘 FPGA 구현", *한국정보보호학회 종합학술발표회 논문집*, Vo. 11, No. 1, pp. 403-406, Nov. 2001.

〔5〕 Jarvinen et al, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," *Int. Symp. on Field Programmable Gate Arrays*, pp. 207-215, Monterey, USA, Feb. 2003.

〔6〕 Alireza Hodjat and Ingrid Verbauwhede, "A 21.54 Gbits/s fully pipelined AES processor on FPGA," *Proc., 12th Annual IEEE Symp. on Field-Programmable Custon Computing Machine*, pp. 308-309, Apr. 2004.

〔7〕 Bon-seok Koo, Gwon Ho Ryu, and Taejoo Chang, "High speed ASIC implementation of ARIA," in preparation.

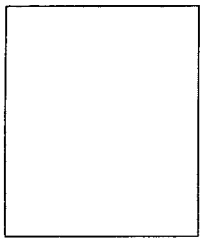
〔8〕 Motorola, MPC190 Security Processor User's Manual, http://www.motorola.com/, 2003.

〔9〕 윤연상, 이선영, 박진섭, 권순열, 김용대, 양상운, 장태주, 유영갑, "IPSec 보안서버의 성능분석 모델", *대한전자공학회논문지TC*, 제41권9호, pp.9-16, 2004.

〔10〕 이상한, 고행석, 장태주, 김영수, 양상운, 박상현, 구본석, "고속 세션 변경이 가능한 블록 암호화 장치 및 그 구동 방법", 특허번호 10-0420555, 2004. 2. 17.

〔11〕 Sang-Hyun Park, Hoon Choi, Sang-Han Lee, Taejoo Chang, "The High-Speed Packet Ciphers System suitable for small sized data," accepted for presentation, IWSEC2006, 2006.

〈著 者 紹 介〉



장 태 주 (Chang, Taejoo)
정회원

1982년 2월 : 울산대학교 전기공학과 졸업
1990년 2월 : 한국과학기술원 전기및전자공학고 석사 졸업
1998년 2월 : 한국과학기술원 전기및전자공학과 박사 졸업
1982년 1월~2000년1월 : 국방과학연구소 선임연구원
2000년 2월~현재 : ETRI 부설 국가보안기술연구소 책임연구원
관심분야 : 암호프로세서 설계, 정보보호, 통계학적 신호처리