# 경량 블록 암호의 최적화를 위한 GPU 구현 연구 동향

송민호* ,김상원* ,서화정**

*한성대학교 (대학원생)

*한성대학교 (교수)

# Trends in GPU-Accelerated Implementation of Lightweight Block Ciphers

Min-Ho Song*, Sang-Won Kim*, Hwa-Jeong Seo**

*Hansung University(Graduate student)

**Hansung University(Professor)

**요 약**

본 논문에서는 GPU 환경에서 경량 블록 암호의 최적화를 위한 최적 구현 연구에 대한 동향을 조사하였다. HIGHT, LEA GIFT 등을 포함한 여러 경량 블록 암호의 GPU 기반 구현 사례를 분석하고 각 알고리즘에 적용된 최적화 기법의 특징과 성능 향상 효과를 정리하였다. 분석 결과 다양한 GPU 최적화 기법이 활용되었으며 이를 통해 기존의 CPU 기반 구현에 비해 높은 처리 성능을 달성할 수 있음을 확인하였다.

## I. 서론

사물인터넷의 급속한 발전과 다양한 응용 분야로의 확산에 따라 수많은 저전력, 저사양 장치들이 무선 네트워크를 통해 상호 연결되고 있다. 이러한 환경에서는 기기 간 주고받는 데이터를 보호하는 암호 기술이 필수적이며 이에 따라 제한된 연산 자원에서도 동작 가능한 경량 블록 암호에 대한 관심이 높아지고 있다. 경량 블록 암호는 메모리와 연산 능력이 제한된 환경에서도 효율적으로 동작할 수 있도록 설계된 알고리즘이다. 최근에는 이러한 경량 암호를 IoT 단말 뿐만 아니라 대량의 데이터를 처리해야 하는 환경에서도 효율적으로 처리하기 위한 연구가 진행되고 있다[1]. 특히 많은 IoT 장치로부터 동시에 들어오는 데이터를 처리하는 경우 CPU 기반 처리만으로는 성능 한계가 존재한다. 이러한 문제를 해결하기 위한 방안으로 병렬 연산에 특화된 GPU를 활용한 경량 블록 암호의 고속 구현이 주목받고 있다.

본 논문에서는 GPU 상에서의 경량 블록 암호 최적 구현 동향에 대해 알아보고자 한다. 다양한 경량 블록 암호들의 GPU 기반 구현 사례를 분석하고 각 알고리즘에 적용된 최적화 기법의 특징과 성능 향상 효과를 정리한다. 이러한 분석은 GPU 기반 경량 블록 암호의 효율적인 구현을 위한 참고 자료로 활용될 수 있다.

## II. 관련 연구

### 2.1 경량 블록 암호

경량 블록 암호는 제한된 연산 자원, 메모리, 전력 소비 환경에서도 효율적인 암호화를 제공할 수 있도록 설계된 암호 알고리즘이다[2]. 기존의 표준 블록 암호인 AES는 높은 보안성과 일반 하드웨어에서의 효율성은 우수하지만 연산 성능과 메모리가 극히 제한된 환경에서는 그 적용이 어렵다. 이에 따라 연산 구조를 단순화하고 하드웨어 구현에 필요한 자원을 줄인 경량 블록 암호

가 제안되었다. 경량 블록 암호는 일반적으로 SPN 또는 ARX 구조를 기반으로 하며 비트 수준의 연산으로 구성된 함수로 인해 저전력 환경에서 높은 성능을 발휘할 수 있다.

### 2.2 GPU Processor

GPU는 본래 그래픽 처리를 위해 설계된 병렬 처리 프로세서였지만 최근에는 범용 연산(GPGPU, General-Purpose computing on GPU)을 위해 널리 사용되고 있다. GPU는 다수의 스레드를 동시에 실행할 수 있어 반복적이고 병렬성이 높은 연산을 빠르게 처리하는 데 매우 적합하다. GPU는 여러 개의 Streaming Multiprocessor(SM)으로 구성되어 있으며 각 SM은 다수의 연산 코어를 내장하고 있다. 하나의 SM은 동시에 여러 개의 스레드 블록을 실행할 수 있으며 각 블록은 다시 다수의 스레드로 구성된다. 이를 통해 대량의 데이터를 병렬로 빠르게 처리할 수 있도록 설계되어 있다.

GPU 내부에는 다양한 속도와 역할을 가진 계층적 메모리 구조가 존재하며 그 구조는 [그림 1]과 같다.. 모든 스레드가 접근 가능하지만 접근 속도가 느린 전역 메모리, 각 스레드에 할당되는 가장 빠른 레지스터, 동일한 블록 내의 스레드들이 공유하는 공유 메모리 등이 있으며 이러한 메모리 구조는 연산 성능을 극대화하기 위한 핵심 요소로 작용한다.

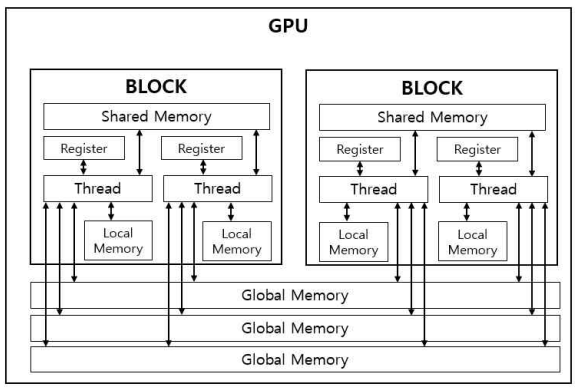


Fig. 1. Structure of the GPU Memory

## III. GPU 상에서의 블록암호 최적 구현

An et al.[3]은 HIGHT, LEA, CHAM 세 가지 ARX 기반 경량 블록 암호를 GPU 환경에서 최적화하여 구현하는 방안을 제시하였다. 일반적인 ECB 모드 대신 병렬 처리에 유리한 CTR 모드를 활용하여 성능을 개선하였다. CTR 모드는 블록 암호를 스트림 암호처럼 사용할 수 있게 하는 운영 모드로 고정 값인 Nonce와 증가하는 Counter로 구성된다. 해당 연구에서는 GPU의 각 스레드 ID를 Counter 값으로 활용하여 별도의 메모리 복사 없이 병렬 암호화를 수행할 수 있도록 하였다. 이를 통해 CPU와 GPU 간의 데이터 복사 오버헤드를 줄여 전체 처리 성능을 향상시켰다. 또한 Nonce 값은 고정되므로 반복적인 연산이 불필요하다는 점을 활용하여 Nonce 관련 연산을 사전에 계산해 Lookup Table 형태로 대체함으로써 HIGHT의 경우 2라운드에 해당하는 연산을 절감할 수 있었다.

라운드 키는 접근 속도가 느린 전역 메모리 대신 빠른 공유 메모리에 저장하여 처리 속도를 높였다. 이 때 뱅크 충돌을 방지하기 위해 각 스레드가 고유한 뱅크를 할당받도록 메모리 구조를 조정하였으며 라운드 키는 해당 뱅크에 저장되어 스레드 간 충돌 없이 병렬 연산이 가능하도록 하였다. 추가적으로 CUDA의 저수준 어셈블리 언어인 PTX를 활용하여 로테이션 연산 등의 명령어를 최적화함으로써 연산 효율을 개선시켰다. RTX 2070 환경에서 구현된 HIGHT, LEA, CHAM의 GPU 구현은 OpenMP 기반 CPU 구현에 비해 각각 445배, 530배, 1856배에 달하는 처리량 향상을 달성하였으며 제안된 최적화 기법의 효과성과 실용성을 입증하였다.

Kim et al.[3]은 PRESENT와 GIFT 블록 암호에 대해 GPU 상에서 비트 슬라이싱 기법을 중심으로 최적화한 구현 방식을 제안하였다. 비트 슬라이싱은 동일한 인덱스의 비트를 묶어 여러 데이터를 동시에 처리할 수 있도록 하는 방식으로 병렬 처리에 매우 효과적이다. 이를 통해 Sbox 연산을 LUT 방식 대신 논리 회로 기반 구조로 변환하여 구현하였다. PRESENT의 경우 기

존 연구를 바탕으로 14개의 논리 연산만으로 Sbox 연산을 수행하였고 GIFT 역시 유사한 논리 구조를 통해 LUT 없이 Sbox 처리가 가능하도록 설계되었다. 또한 비트 슬라이싱 방식은 P-layer 연산 역시 단순한 레지스터 간의 비트 이동으로 대체할 수 있어 연산 복잡도를 크게 줄일 수 있다.

일반적인 GPU 기반 암호 구현에서는 라운드 키를 사전에 계산하여 메모리에 저장하는 방식을 사용한다. 하지만 비트 슬라이싱 환경에서는 데이터 블록 수의 증가에 따라 필요한 라운드 키도 선형적으로 증가하게 되어 메모리 자원에 과부하가 발생할 수 있다. 이를 해결하기 위해 해당 연구에서는 라운드 키를 매 라운드마다 즉시 계산하는 on-the-fly 방식을 도입하였다. PRESENT 및 GIFT는 키 스케줄 구조가 비교적 단순하여 해당 방식이 적합하다.

해당 연구는 대부분의 연산을 GPU의 레지스터에서 직접 수행하도록 하여 전역 메모리 접근을 최소화하였다. GIFT-128과 같이 상대적으로 키 크기가 큰 경우 일부 라운드 키를 공유 메모리에 저장하였으나 공유 메모리의 사용 여부에 따른 성능 차이는 크지 않은 것으로 나타났다. 이는 레지스터 기반의 연산 구조가 전체 성능에 큰 영향을 미친다는 점을 보여준다. 성능 평가 결과 RTX 3060 환경에서 측정된 PRESENT의 처리량은 기존 최신 GPU 구현에 비해 4배 이상 향상된 것으로 나타났다.

Yang et al.[5]는 하드웨어 친화적으로 설계된 경량 블록 암호 PRINCE를 GPU 상에서 고속으로 구현하기 위해 다양한 최적화 기법을 적용하였다. 해당 연구는 PRINCE의 연산 구조가 병렬 처리에 적합하다는 점을 기반으로 GPU 아키텍처에 최적화된 구현 방식을 통해 암호화 처리 성능을 향상시키고자 하였다. PRINCE는 4비트 Sbox를 사용하며 64비트 상태에 이를 적용하기 위해 총 16번의 Sbox 연산이 필요하다. 이를 최적화하기 위해 두 개의 4비트 Sbox를 사전에 결합하여 8비트 단위의 LUT을 생성하고 이를 GPU의 Constant Memory에 저장하였다. Constant Memory는 캐시 기반으로 접근되기 때문에 LUT 접근 속도가 빠르며 Sbox 호출 횟수도 절감되어 전반적인 메모리 접근 비용이 감소하고 성능이 향상되었다.

또한 PRINCE의 선형 계층인 M-Layer는 64비트 행렬 곱셈으로 구성되며 일반적으로 비트 단위의 AND 및 XOR 연산을 반복적으로 수행해야 하는 계산량이 많은 구조이다. 해당 구현에서는 상태 값을 상위 8비트와 하위 8비트로 분리하고 각각에 대해 사전 계산된 LUT를 이용하여 변환하는 방식으로 행렬 연산을 대체하였다. 이로써 복잡한 행렬 곱셈을 단 2회의 LUT 접근과 1회의 XOR 연산으로 처리할 수 있게 되어 성능이 크게 개선되었다.

CPU에서 GPU로 데이터를 전달하는 과정 역시 암호화 성능에 직접적인 영향을 미친다. 해당 연구는 cudaMemcpy()를 이용한 일반적인 메모리 복사 방식 cudaMallocManaged()를 이용한 Unified memory 방식을 적용하여 CPU와 GPU가 동일한 메모리 공간을 공유할 수 있도록 하였다. 이를 통해 데이터 복사 오버헤드를 줄이고 전체 암호화 처리량을 높일 수 있었다. GTX 1070 환경에서 성능 측정한 결과 기존 CPU 구현 대비 4000배 빠른 속도를 달성하였다.

## IV. 결론

본 논문에서는 GPU 상에서의 경량 블록 암호 최적 구현 연구 동향에 대해 살펴보았다. 다양한 경량 블록 암호의 GPU 기반 구현 사례를 분석하고 각 알고리즘에 적용된 최적화 기법의 특징과 성능 향상 효과를 정리하였다. 분석된 구현들은 비트 슬라이싱, LUT 활용, 공유 메모리 최적화 등 GPU 아키텍처의 병렬 처리 특성을 극대화하는 방식이였으며 이를 통해 기존 CPU 기반 구현에 비해 높은 성능 향상을 달성한 것을 확인할 수 있었다. 이러한 최적화 기법들은 특정 알고리즘에 한정되지 않고 구조적으로 유사한 다른 경량 블록 암호에도 적용이 가능하다. 본 논문의 분석 결과는 GPU 기반 경량 블록 암호의 효율적인 구현과 최적화를 위한 기초 자료로 활용될 수 있을 것이다.

## V.Acknowledgment

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2025-02306395, Development and Demonstration of PQC-Based Joint Certificate PKI Technology, 50%) and this work was partly supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00264, Research on Blockchain Security Technology for IoT Services, 50%).

## [참고문헌]

[1] An, SangWoo, and Seog Chung Seo. "Highly efficient implementation of block ciphers on graphic processing units for massively large data." Applied Sciences 10.11 (2020): 3711.

[2] Hatzivasilis, George, et al. "A review of lightweight block ciphers." Journal of cryptographic Engineering 8 (2018): 141-184.

[3] An, SangWoo, et al. "Parallel implementations of ARX-based block ciphers on graphic processing units." Mathematics 8.11 (2020): 1894.

[4] Kim, Hyunjun, et al. "Secure and robust Internet of Things with high-speed implementation of PRESENT and GIFT block ciphers on GPU." Applied Sciences 12.20 (2022): 10192.

[5] Yang, Jingchun, et al. "A GPU implementation of PRINCE block cipher." 2024 IEEE 7th International Conference on Computer and Communication Engineering Technology (CCET). IEEE, 2024.