Quantum Modular Multiplication 논문 리뷰

https://youtu.be/hmtkywnc-Xc

IT융합공학부 송경주

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB

Quantum Modular Multiplication

- [1] 의 논문 리뷰
- 곱셈에 Reduction을 사용하지 않고 비트 shift 및 비트 순환 방식의 사용 제안
- 위의 방법으로 복잡성을 줄인 Quantum-Classical, Quantum-Quantum 곱셈기 제안
  - Quantum-Classical 연산 : 비트 Shift 사용
  - Quantum-Quantum 연산 : 비트 순환 사용

- 제안하는 모듈러 곱셈기는 [2]의 Quantum Carry-Lookahead Adder (QCLA) 모듈러 덧셈기를 반복해서 수행하는 방식
  - QCLA : 모든 비트를 병렬로 계산하기 때문에 가산기 중 가장 빠름

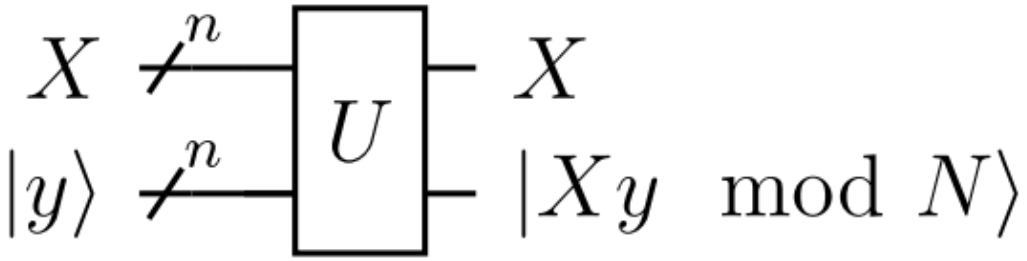
[1] Seong-Min Cho, Aeyoung Kim, Dooho Choi, Byung-Soo Choi, and Seung-Hyun Seo, Quantum Modular Multiplication, IEEE Access, vol.8, pp.213244-213252, 2020
[2] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, "A logarithmicdepth quantum carry-lookahead adder," Quantum Inf. Comput., vol. 6, nos. 4–5, pp. 351–369, 2006.

Quantum Modular Multiplication

- Quantum-Classical operation
  - 연산하는 두 대상 중 1개는 Classic Value 1개는 Quantum Value

Classic value→

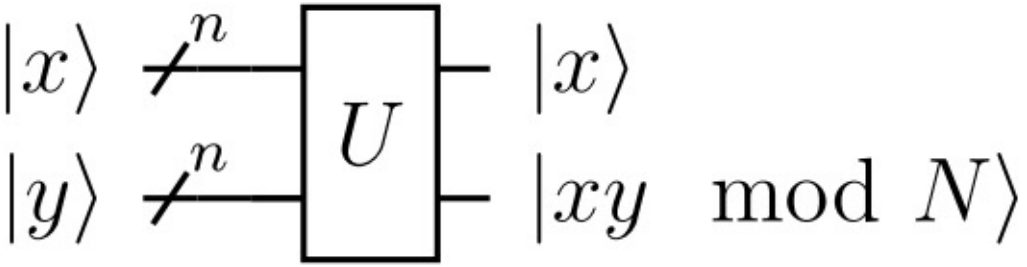
Quantum value→



- Quantum-Quantum operation
  - 연산하는 두 대상 모두 Quantum Value

Quantum value→

Quantum value→

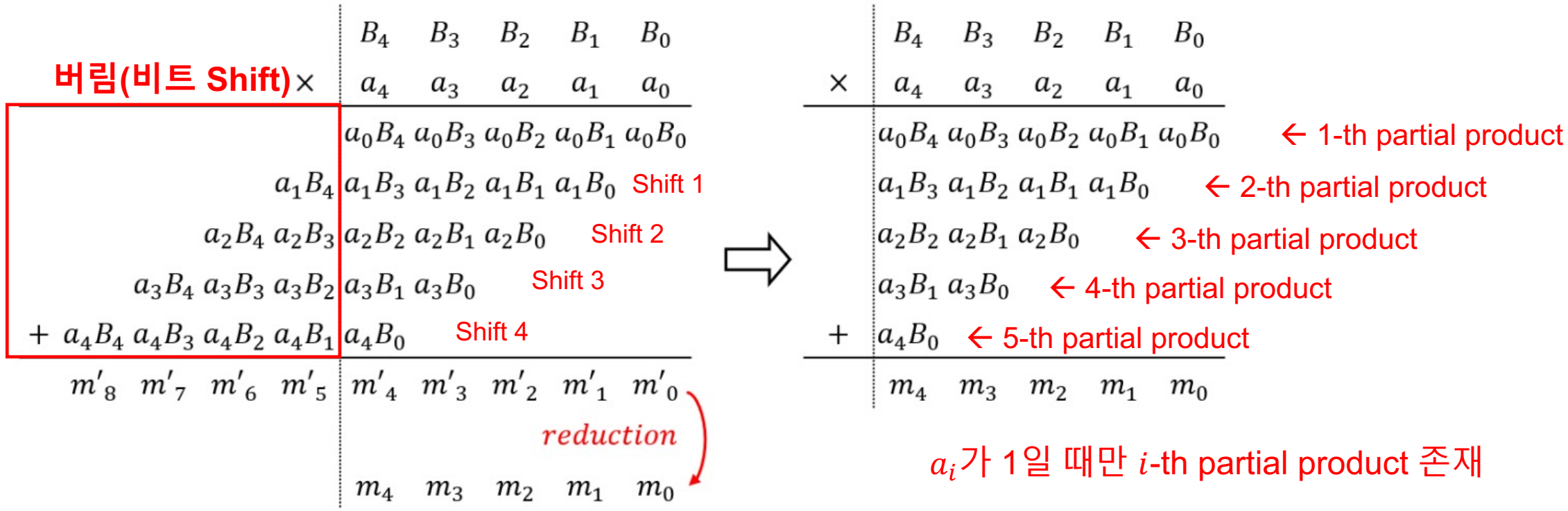


Quantum Modular Multiplication in **GF(2^n)**

Quantum-Classical Multiplication

Quantum Modular Multiplication

- Classic-Quantum Multiplication - GF(2^n)
  - n-bit 의 quantum value a 와 classic value B 의 모듈러 곱셈 수행
  - Partial product setting 단계와 Modular addition 단계로 이루어짐
- Partial product setting

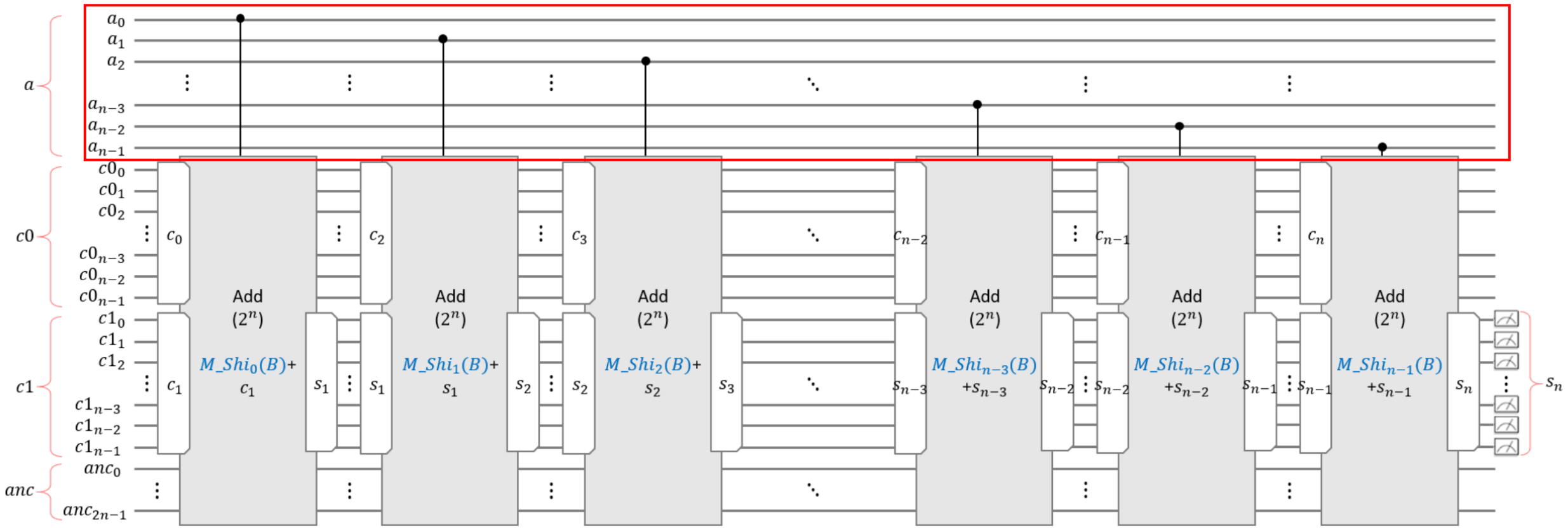


Quantum Modular Multiplication

- Classic-Quantum Multiplication - GF(2^n)

- Partial product setting + Modular addition:(QCLA)사용

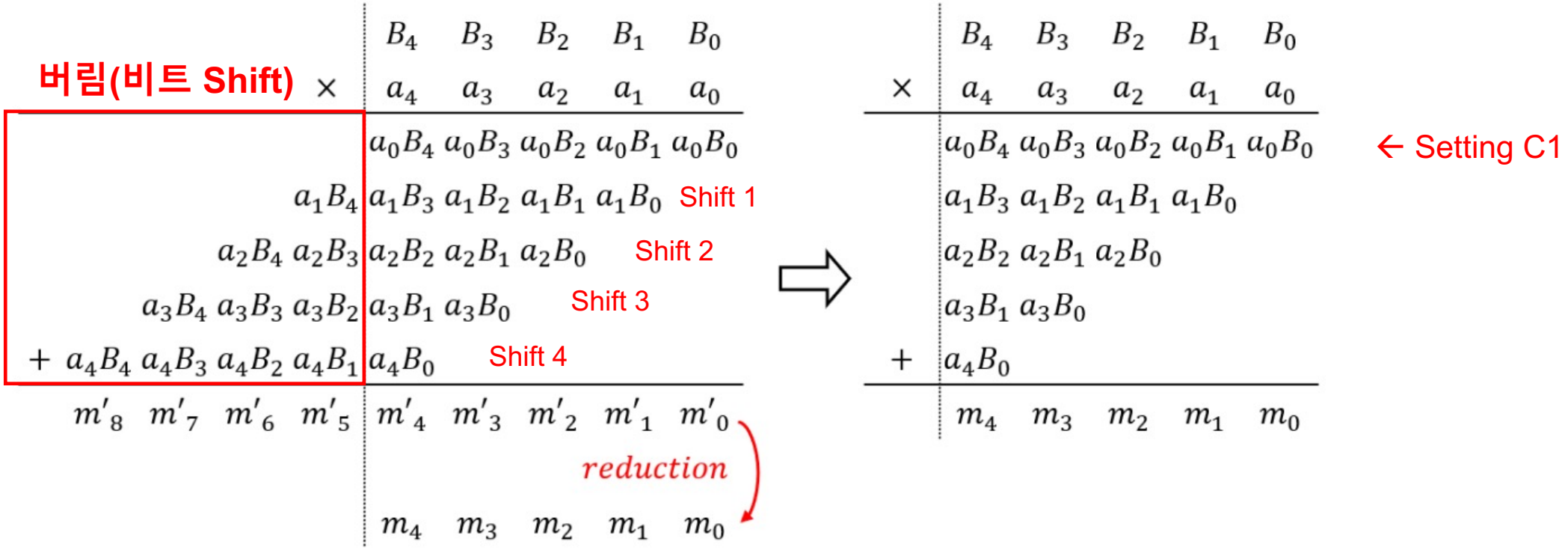
a_i가 1일 때만 i-th partial product 존재하므로 1일 때만 Add 실행



Quantum-Quantum Multiplication

Quantum Modular Multiplication

- Quantum-Quantum Multiplication - GF(2^n)
  - n-bit 의 quantum value a 와 Quantum value b 의 모듈러 곱셈 수행
  - Qubit setting 단계, Modular addition 단계, Inverse setting 단계로 이루어짐
- Partial product setting



Quantum Modular Multiplication

- Qubit setting

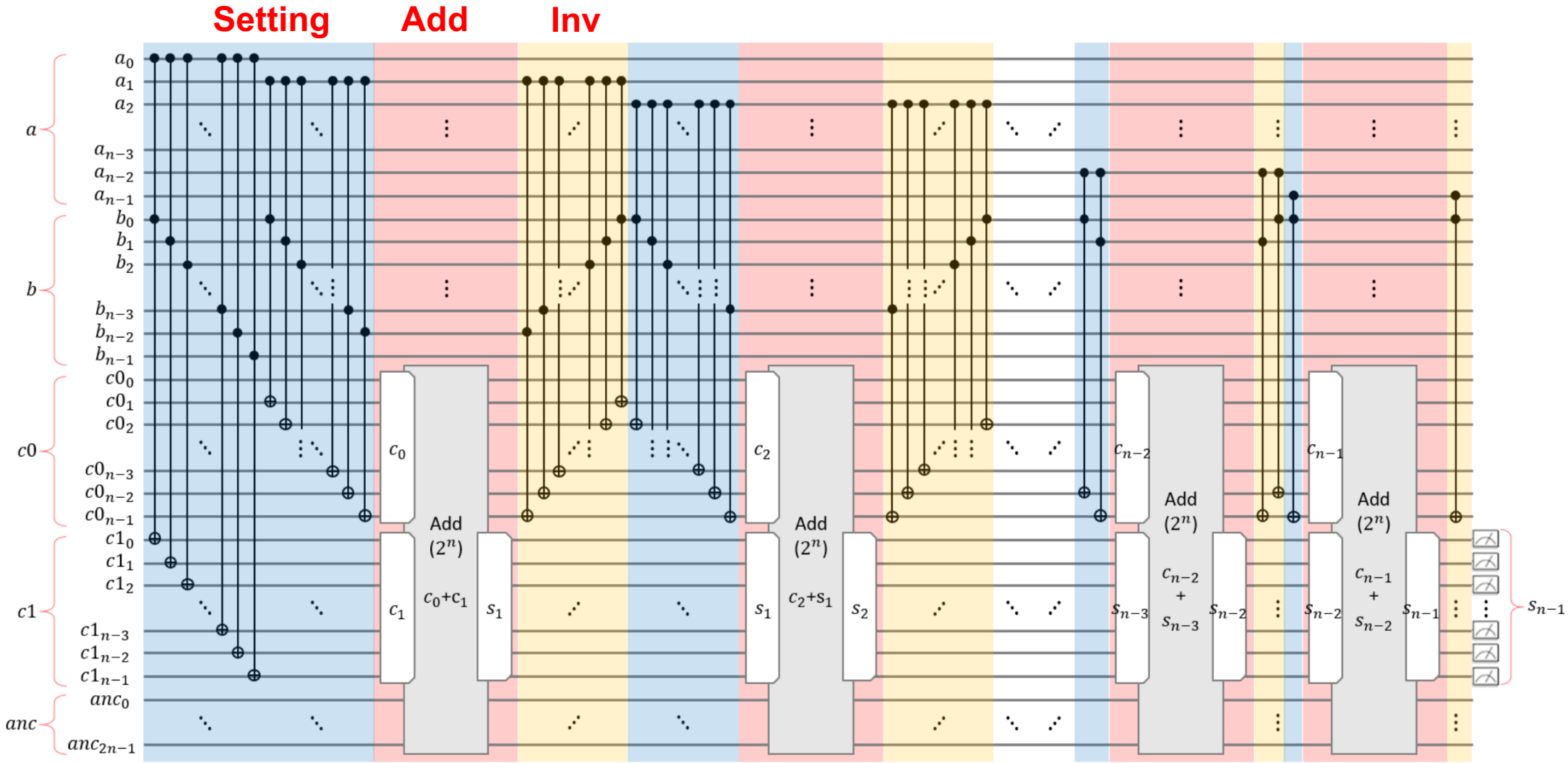
Algorithm 1 Qubit Setting

input : quantum registers a, b, $c0$, and $c1$
output: quantum registers $c0$ and $c1$

1 **for** $i = 0$ **to** $n - 1$ **do**
2 Toffoli($a_0, b_i, c1_i$); → C1[0] ~ C1[n-1] 에 Setting 1 저장
3 **for** $i = 0$ **to** $n - 1$ **do**
4 **for** $j = 0$ **to** $n - 1 - i$ **do**
5 Toffoli($a_i, b_j, c0_{i+j}$); → C1[0] ~ C1[n-1] 에 Setting 2-5 저장
6 Return $c0$, $c1$

Quantum Modular Multiplication

- Quantum-Quantum Multiplication - GF(2^n)

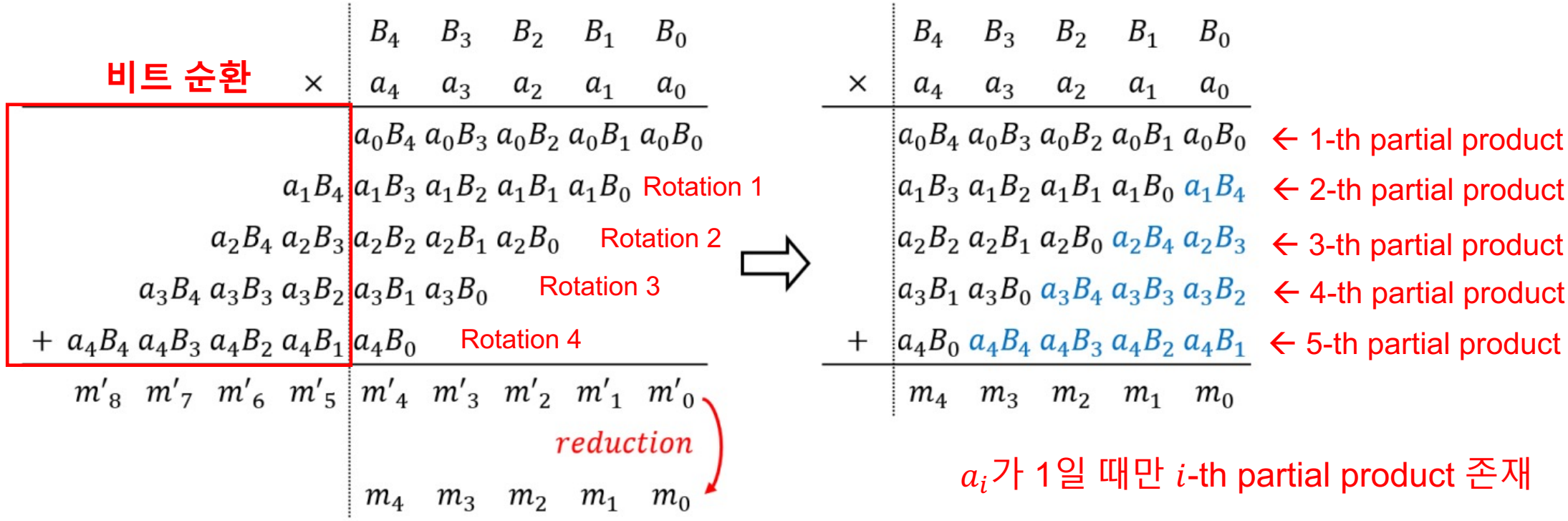


Quantum Modular Multiplication in **GF($2^n - 1$)**

Quantum-Classical Multiplication

Quantum Modular Multiplication

- Classic-Quantum Multiplication - GF(2^n-1)
  - n-bit 의 quantum value a 와 classic value B 의 모듈러 곱셈 수행
  - Partial product setting 단계와 Modular addition 단계로 이루어짐
- Partial product setting

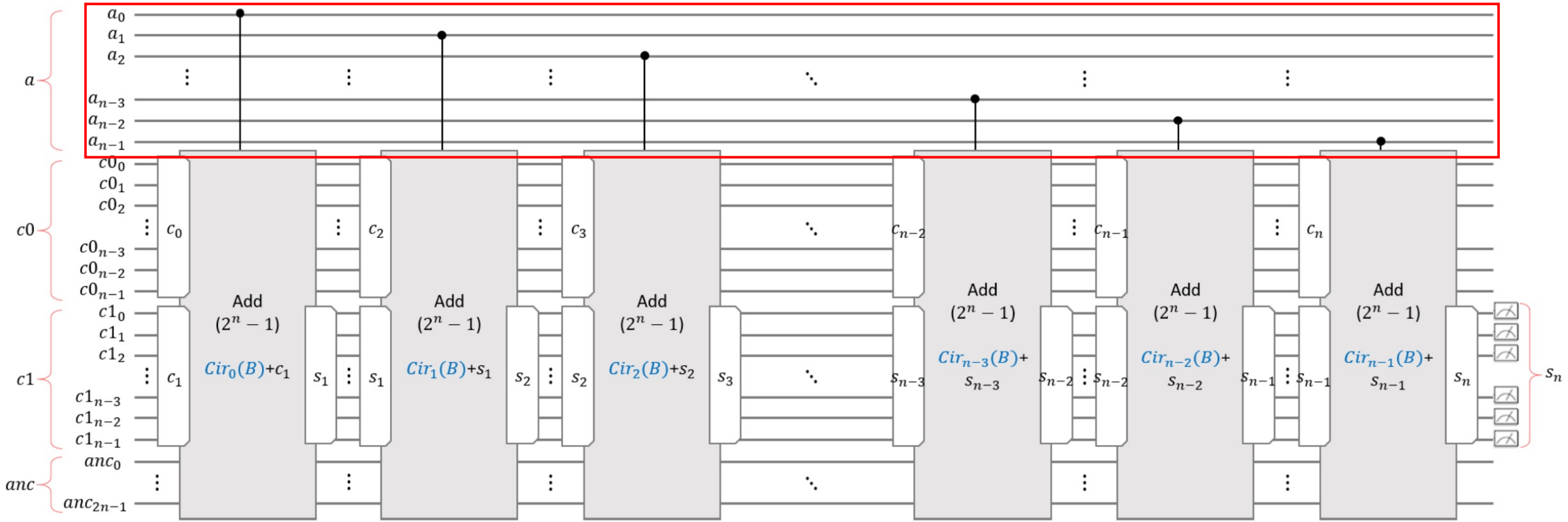


Quantum Modular Multiplication

- Classic-Quantum Multiplication - GF(2^n-1)

- Partial product setting + Modular addition

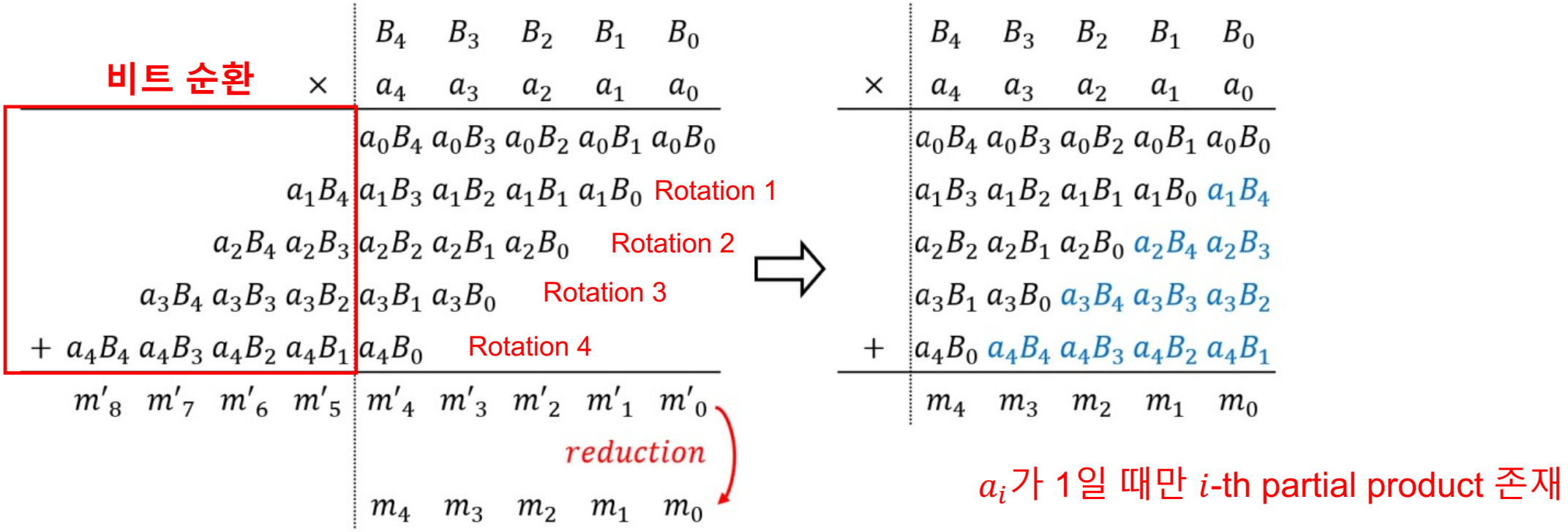
a_i가 1일 때만 i-th partial product 존재하므로 1일 때만 Add 실행



Quantum-Quantum Multiplication

Quantum Modular Multiplication

- Quantum-Quantum Multiplication - GF(2^n-1)
  - n-bit 의 quantum value a 와 classic value B 의 모듈러 곱셈 수행
  - Qubit setting 단계, Modular addition 단계, inverse setting 단계로 이루어짐
- Partial product setting



Quantum Modular Multiplication

- Qubit setting

Algorithm 2 Qubit Setting

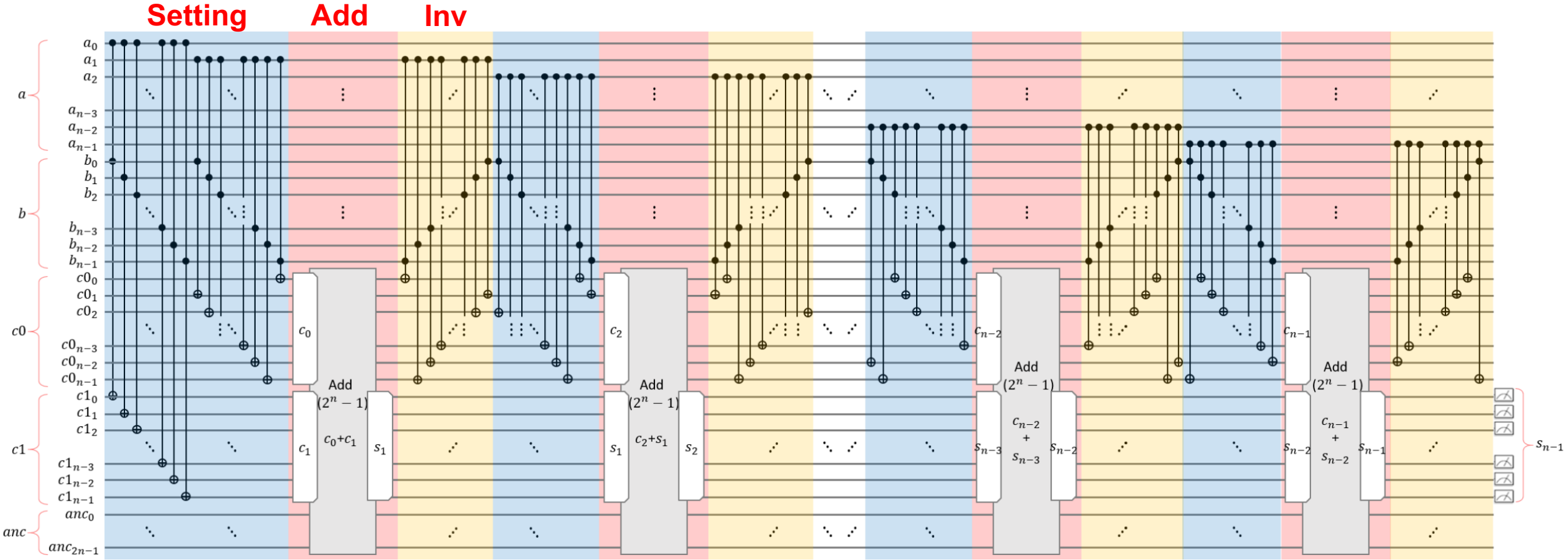
input : quantum registers a, b, $c0$, and $c1$
output: quantum registers $c0$ and $c1$

1 **for** $i = 0$ **to** $n - 1$ **do**
2 Toffoli($a_0, b_i, c1_i$);
3 **for** $i = 1$ **to** $n - 1$ **do**
4 **for** $j = 0$ **to** $n - 1$ **do**
5 Toffoli($a_i, b_j, c0_{(i+j) \bmod n}$);
6 Return $c0, c1$

Rotation 해서 저장

Quantum Modular Multiplication

- Quantum-Quantum Multiplication - GF(2^n-1)



Q & A