“Concrete quantum cryptanalysis of binary elliptic curves”

리뷰 및 Future Work

장경배

https://youtu.be/JfToXYV7fxE

ECC에 대한 최신 연구 동향

- **이산 대수 문제 (Discrete Logarithm Problems)**에 그 안전성을 기반하는 **ECC 또한 Shor 알고리즘을 활용하는 양자 컴퓨터의 공격에 안전성이 무너짐**

- Shor의 논문에서 주요 예제로, **정수를 소인수 분해하여 RSA를 깨는 방법을 자세히 설명**했지만
  - **타원 곡선 상에서의 이산 대수 문제로도 Shor 알고리즘이 확장 될 수 있음을 보임**

- **Shor on ECC?**
  - ECC에대한 Shor 알고리즘 적용 연구들은 **타원 곡선에서의 스칼라 곱셈을 양자 회로로 최적화 구현하는 것이 중요**

  - Shor 알고리즘에서 이산 대수 문제를 해결할 때 **가장 많은 비용이 드는 산술**

Thomas Häner, M. Rötteler, K. Svore", Factoring using 2n+2 qubits with Toffoli based modular multiplication, 2016
Craig Gidney, "Factoring with n + 2 clean qubits and n − 1 dirty qubits", 2018

ECC에 대한 최신 연구 동향

- **타원 곡선 상에서의 이산 대수 문제를 해결(Shor 알고리즘)하기 위한 양자 자원들을 추정**
  - **NIST curves 대상**

- **ASIACRYPT(2017)** : “Quantum resource estimates for computing elliptic curve discrete logarithms”
  - 타원곡선에서의 이산대수문제를 해결하는데 필요한 양자 자원들을 추정함으로써 **RSA보다 ECC가 더 양자컴퓨터의 공격에 취약함**을 보임
- **PQCrypto(2020)** : “Improved quantum circuits for elliptic curve discrete logarithms”
  - ASIACRYPT(2017)의 결과보다 **큐비트 수, Depth 모두 줄임**
- **CHES (2020)** : “Concrete quantum cryptanalysis of binary elliptic curves”
  - **Binary ECC에 대한 Shor 알고리즘 적용 시, 더 적은 자원으로 공격 가능함을 보임**
  - **Karatsuba 곱셈을 활용한 곱셈 및 역치 연산 최적화 (양자 컴퓨터 상에서)**

ECC에 대한 최신 연구 동향

- **Banegas, Bernstein, Van Hoof, Lange의 CHES 2020 연구 결과가 가장 적은 자원으로 공격이 가능**
  - **물론 Prime curve(ASIACRYPT, PQCrypto)가 아닌 Binary curve**
  - **하드웨어 친화적인 Binary 산술을 사용하기 때문에 양자 컴퓨터상에서도 더 최적화됨**
    - **Binary 덧셈 → XOR 연산, Binary 곱셈 → AND 연산, 캐리가 X**

	Curve (Prime)	Asiacrypt	PQCrypto
Qubits	P256	**2,338**	**2,124**
	P384	**3,492**	**3,151**
	P521	**4,727**	**4,258**
Depth	P256	-	$\mathbf{1.38 \cdot 2^{32}}$
	P384	-	$\mathbf{1.77 \cdot 2^{34}}$
	P521	-	$\mathbf{1.09 \cdot 2^{36}}$

	Curve (Binary)	CHES (2020)
Qubits	B233	**1,647**
	B283	**1,998**
	B571	**4,015**
Depth	B233	$\mathbf{1.14 \cdot 2^{21}}$
	B283	$\mathbf{1.67 \cdot 2^{21}}$
	B571	$\mathbf{1.57 \cdot 2^{23}}$

<타원곡선 이산대수문제(ECDLP)에 대한 Shor 알고리즘 적용 자원>

CHES Paper

Concrete quantum cryptanalysis of binary elliptic curves

Gustavo Banegas[1], Daniel J. Bernstein[2,3], Iggy van Hoof[4] and Tanja Lange[4]

[1] Chalmers University of Technology, Gothenburg, Sweden gustavo@cryptme.in
[2] University of Illinois at Chicago, Chicago, USA djb@cr.yp.to
[3] Ruhr University Bochum, Bochum, Germany
[4] Eindhoven University of Technology, Eindhoven, The Netherlands
i.v.hoof@tue.nl,tanja@hyperelliptic.org

Abstract. This paper analyzes and optimizes quantum circuits for computing discrete logarithms on binary elliptic curves, including reversible circuits for fixed-base-point scalar multiplication and the full stack of relevant subroutines. The main optimization target is the size of the quantum computer, i.e., the number of logical qubits required, as this appears to be the main obstacle to implementing Shor's polynomial-time discrete-logarithm algorithm. The secondary optimization target is the number of logical Toffoli gates.

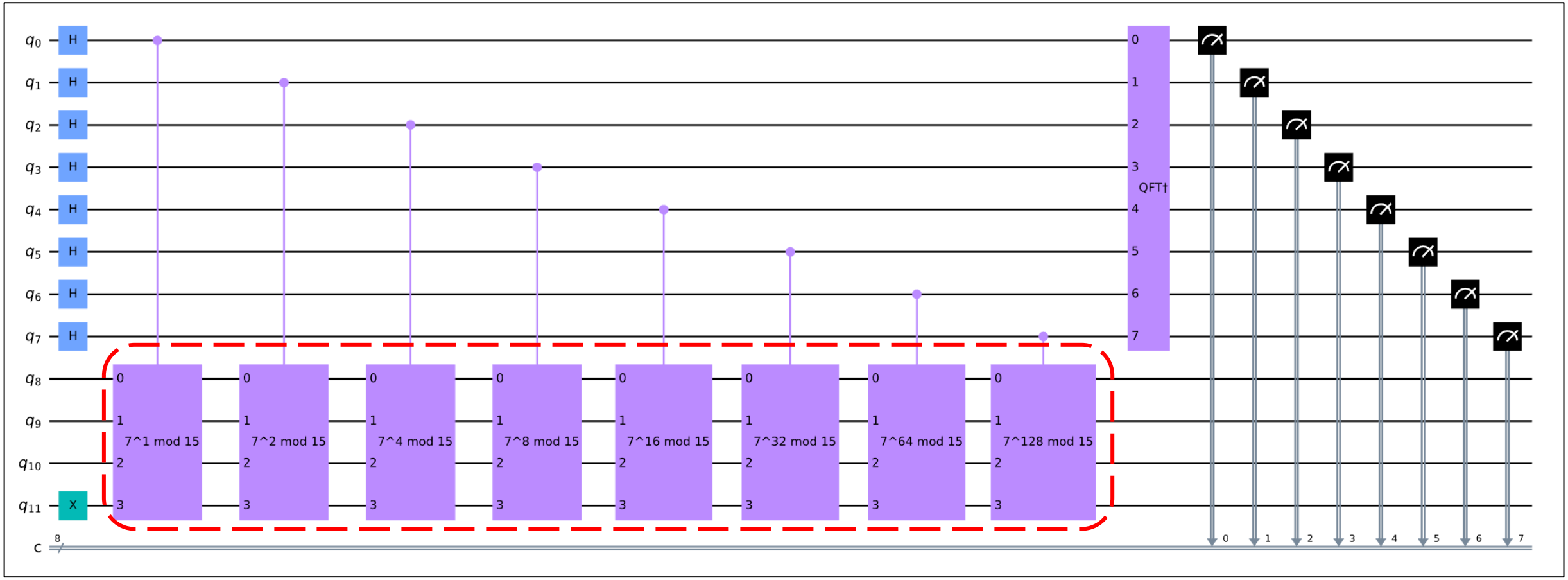
For an elliptic curve over a field of 2^n elements, this paper reduces the number of qubits to $7n + \lfloor \log_2(n) \rfloor + 9$. At the same time this paper reduces the number of Toffoli gates to $48n^3 + 8n^{\log_2(3)+1} + 352n^2 \log_2(n) + 512n^2 + O(n^{\log_2(3)})$ with double-and-add scalar multiplication, and a logarithmic factor smaller with fixed-window scalar multiplication. The number of CNOT gates is also $O(n^3)$. Exact gate counts are given for various sizes of elliptic curves currently used for cryptography.

Keywords: Quantum cryptanalysis · elliptic curves · quantum resource estimation · quantum gates · Shor's algorithm

Elliptic Curve Discrete Logarithm Problem(ECDLP)

- **Elliptic Curve Diffie-Hellman**
  - 타원 곡선상에서의 **Secret Point를 공유하여 키 교환을 수행**하는 프로토콜
  - **Known**
    - Point P on Curve
  - **Secret**
    - Bob : Integer α
    - Alice : Integer β
  - **ECDLP**
    - $P_\alpha = [\alpha]P$일 때, P_α로부터 α를 알아내는 문제
    - Bob → $P_\alpha = [\alpha]P$를 Alice에게 전달
    - Alice → $P_\beta = [\beta]P$를 Bob에게 전달
    - Bob과 Alice만의 Shared Secret Point : $\boldsymbol{P_{\alpha\beta}} = [\alpha \cdot \beta]P = [\alpha]P_\beta = [\beta]P_\alpha$
      - **(전달받은 Point) × (자신의 Secret)**

Shor on ECDLP



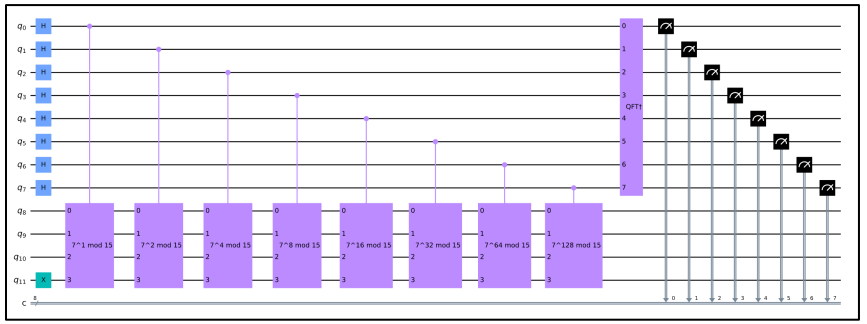
< Factoring 회로 >

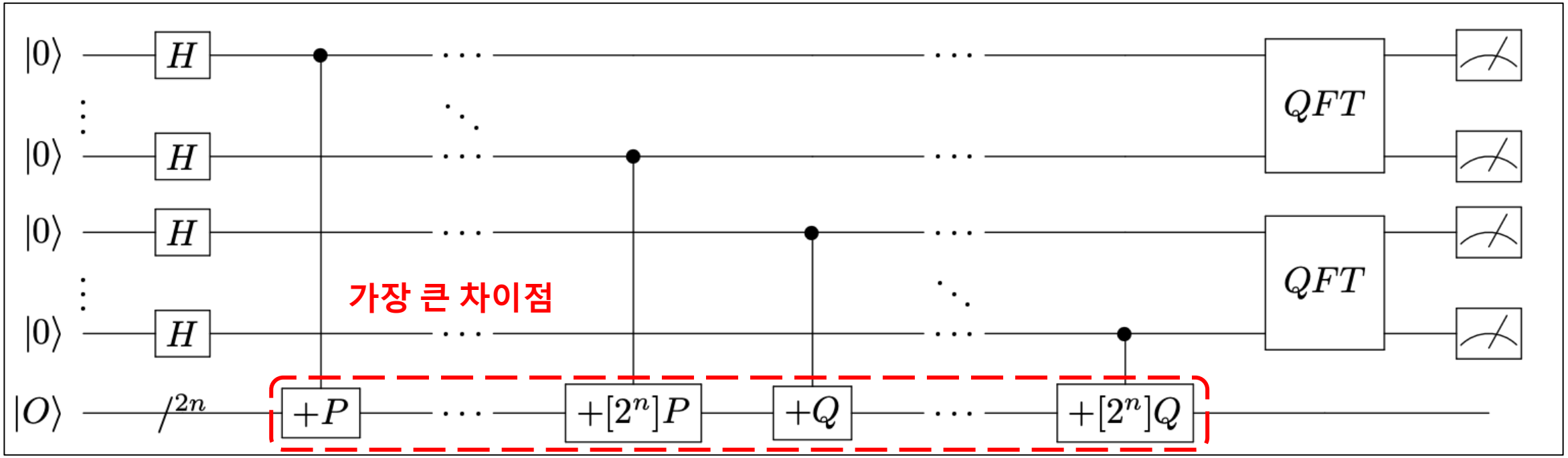
Shor on ECDLP

- $E(\mathbb{F}_{2^n})$상에서의 $Q = [\alpha]P,\ \text{Secret} = \ \alpha,\ \alpha \in \{1, \dots, r\}$

- Factoring하는 Shor 알고리즘 양자 회로와 유사
  - 2개의 레지스터 k, l 을 사용
  - $2n$ qubits의 세번째 레지스터

< Factoring 회로 >





Shor on ECDLP

- $E(\mathbb{F}_{2^n})$상에서의 $Q = [\alpha]P,\ \text{Secret} = \ \alpha,\ \alpha \in \{1, \ldots, r\}$
  - **내부 수식이 ECDLP 타겟으로 바뀜**

$$\frac{1}{2^{n+1}} \sum_{k,\ell=0}^{2^{n+1}-1} |k, \ell, [k]P + [\ell]Q\rangle.$$

< ECDLP에 대한 Shor 알고리즘 >

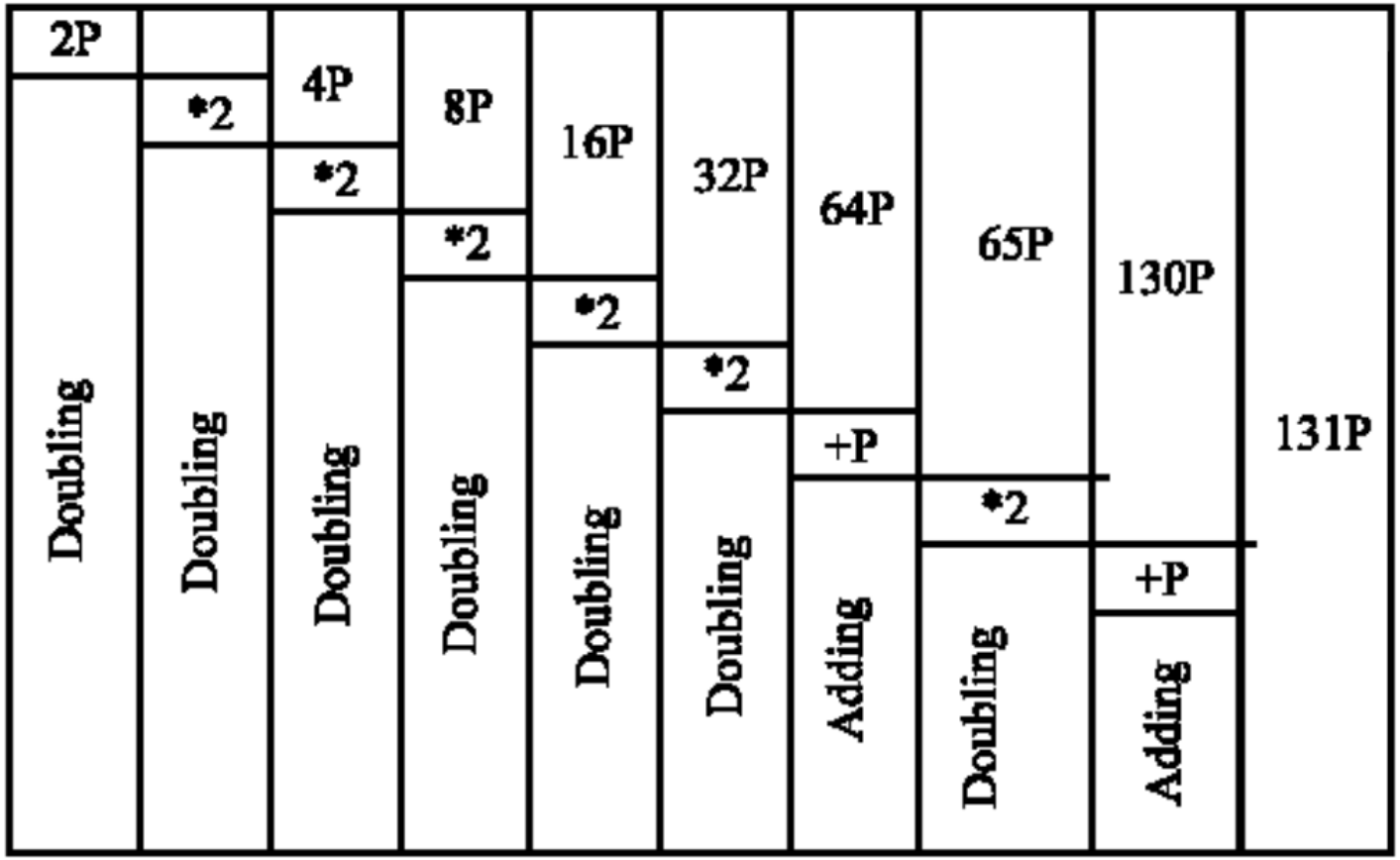
$$|u_0\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |a^k \bmod N\rangle$$

< Factoring에 대한 Shor 알고리즘 >

$$[2^0]P, [2^1]P, \ldots, [2^n]P \text{ and } [2^0]Q, [2^1]Q, \ldots, [2^n]Q$$

Shor on ECDLP

- Curve 상에서의 Scalar Multiplication? → **Double and ADD** Scalar Multiplication
  - Point addition의 연속
    - **Ex) 5P**
      1. $P + P = 2P$
      2. $2P + 2P = 4P$
      3. $4P + 1P = 5P$



Shor on ECDLP

- **Point Addition?**
  - **다양한 연산들의 조합**

Two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2) \neq \pm P_1$ are added to produce $P_1 + P_2 = P_3 = (x_3, y_3)$ as

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a, \quad y_3 = (x_2 + x_3)\lambda + x_3 + y_2 \text{ with } \lambda = (y_1 + y_2)/(x_1 + x_2).$$

and $P_1 \neq -P_1$ is doubled to produce $[2]P_1 = (x_3, y_3)$ as

$$x_3 = \lambda^2 + \lambda + a, \quad y_3 = x_1^2 + (\lambda + 1)x_3 \text{ with } \lambda = x_1 + y_1/x_1.$$

Shor on ECDLP

- **Point Addition?**
  - **해당 논문에서는 Point Addition을 위한 다양한 Binary Filed Arithmetic의 양자 구현에 대해 설명하고 있음**

5 Basic arithmetic

In this section we discuss reversible in-place algorithms for the basic arithmetic of binary polynomials modulo a field polynomial $m(z)$, i.e. elements of $\mathbb{F}_{2^n}$.

5.1 Addition and binary shift

5.2 Multiplication

> For multiplication we use a space-efficient Karatsuba algorithm by Van Hoof [Hoo20] which uses $O(n^2)$ CNOT gates, $O(n^{\log_2 3})$ Toffoli gates and $3n$ total qubits: $2n$ qubits for the input, f, g, and n separate qubits for the output, h. The algorithm is detailed in Appendix A.

공동 저자 Van Hoof의 양자 카라추바 곱셈을 사용

5.3 Squaring

6 Inversion and division in binary finite fields

6.1 Inversion using extended GCD

6.2 Inversion using FLT

Shor on ECDLP

- **Inversion 비교**
  - **해당 논문에서는 GCD 방식의 inversion을 구현하였지만, FLT 방식의 구현이 더 좋을 것 같다는 생각도 듬**
    - **GCD → 적은 큐빗, 많은 Toffoli gates, FLT → 더 많은 큐빗, 적은 Toffoli gates**

Table 2: Comparison of various instances of division Algorithms 1 and 2. Field polynomials from Table 1. Depths and gate count are upper bounds since a generic algorithm is used rather than optimizing for specific fields.

n	GCD				FLT			
	TOF	CNOT	qubits	depth	TOF	CNOT	qubits	depth
8	3,641	1,516	67	4113	243	2,212	56	1314
16	10,403	5,072	124	12,145	1,053	10,814	144	5968
127	277,195	227,902	903	378,843	50,255	502,870	1,778	203,500
163	442,161	375,738	1,156	612,331	83,353	906,170	1,956	451,408
233	827,977	743,136	1,646	1,172,733	132,783	1,486,464	3,029	640,266
283	1,202,987	1,088,400	1,997	1,708,863	236,279	2,708,404	3,962	1,434,686
571	4,461,673	4,266,438	4,014	6,494,306	814,617	10,941,536	9,136	6,151,999

Shor on ECDLP

- **최종 Point addition 양자 회로 → Addition, Multiplication, Inversion, Squaring 모두 사용**

Algorithm 3: Point addition for binary elliptic curves.

Fixed input : A constant field polynomial m of degree $n > 0$. A fixed constant a from the elliptic curve formula. A fixed point $P_2 = (x_2, y_2)$.

Quantum input : A control qubit q. An elliptic curve point P_1 represented as two binary polynomials x_1, y_1 stored in x, y of size n. A size-n array λ initialized to an all-$|0\rangle$ state. Ancillary qubits for division.

Result: (x, y) as $P_1 + P_2 = P_3 = (x_3, y_3)$ if $q = 1$, P_1 if $q = 0$, λ and ancillary qubits same as input $\lambda = 0$.

1	$x \leftarrow \text{const\_ADD}(x, x_2)$	// $x = x_1 + x_2$
2	$y \leftarrow \text{ctrl\_const\_ADD}_q(y, y_2)$	// $y = y_1 + q \cdot y_2$
3	$\lambda \leftarrow \text{DIV}(x, y, \lambda)$	// $\lambda = y/x$
4	$y \leftarrow \text{MODMULT}(x, \lambda, y)$	// $y = y + x \cdot (y/x) = 0$
5	$y \leftarrow \text{SQUARE}(\lambda, y)$	// $y = \lambda^2$
6	$x \leftarrow \text{ctrl\_const\_ADD}_q(x, a + x_2)$	// $x = x_1 + x_2 + q(a + x_2)$
7	$x \leftarrow \text{ctrl\_ADD}_q(x, \lambda)$	// $x = x_1 + x_2 + q(\lambda + a + x_2)$
8	$x \leftarrow \text{ctrl\_ADD}_q(x, y)$	// $x = x_1 + x_2 + q(\lambda + \lambda^2 + a + x_2)$
9	$y \leftarrow \text{SQUARE}(\lambda, y)$	// $y = \lambda^2 + \lambda^2 = 0$
10	$y \leftarrow \text{MODMULT}(x, \lambda, y)$	// $y = x \cdot \lambda$
11	$\lambda \leftarrow \text{DIV}(x, y, \lambda)$	// $\lambda = \lambda + (x \cdot \lambda)/x = 0$
12	$x \leftarrow \text{const\_ADD}(x, x_2)$	// $x = x_1 + q(\lambda + \lambda^2 + a + x_2)$
13	$y \leftarrow \text{ctrl\_ADD}_q(y, x)$	// $y = y + q \cdot x_3$
14	$y \leftarrow \text{ctrl\_const\_ADD}_q(y, y_2)$	// $y = y + q \cdot y_2$

Shor on ECDLP

- **ECC에 대한 Shor 알고리즘 최적화 포인트 →** $[2^0]P, [2^1]P, ..., [2^n]P$ and $[2^0]Q, [2^1]Q, ..., [2^n]Q$
  - **Addition, Squaring은 별다른 차별화가 없음 → Multiplication이 주요해 보임**

- **Van Hoof Multiplication은 최소 큐비트, Toffoli gate를 사용하지만**
  이번 WISA에 제출한 제안 곱셈기보다 Depth, $TD \cdot M$이 높음

Table 6: Comparison of quantum resources required for multiplication of $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$.

Field size 2^n	Source	#CNOT	#1qCliff	#T	Toffoli depth	#Qubits	Full depth	$TD \cdot M$
$n = 8$	This work (Sec. 3.5)	323	54	189	1	81	32	81
	[7]	405	30	448	28	24	216	672
	[8]	270	54	189	8	43	88	344
	[6]	382	54	189	N/A	24	N/A	N/A

TD = Toffoli depth, M = number of qubits.

Shor on ECDLP

- **Result**

Table 5: Qubit and gate count for Shor's algorithm for binary elliptic curves. Field polynomials from Table 1. CNOT count given is an upper bound.

		Single step			Total
n	qubits	TOF gates	CNOT gates	depth upper bound	TOF gates
8	68	7,360	3,522	8,562	132,480
16	125	21,016	11,686	25,205	714,544
127	904	559,141	497,957	776,234	143,140,096
163	1,157	893,585	827,623	1,262,280	293,095,880
233	1,647	1,669,299	1,615,287	2,406,230	781,231,932
283	1,998	2,427,369	2,359,187	3,503,964	1,378,745,592
571	4,015	8,987,401	9,081,061	13,238,554	10,281,586,744

Future Work

- **Multiplication 및 Point addition 산술 구조에서의 개선점 찾기**
  - **WISA 곱셈은 연속적인 곱셈에서 Qubits 오버헤드가 상쇄되어 더 효과적**

- **Depth와 Qubits를 고려한 최적화 포인트 찾기**
  - **저자들도 언급하듯, Depth 최적화에 대한 고려를 크게 하지 않았음**

> In this work, depth will not be explored in-depth, but will be reported and optimization left to future work.

  - **사실 Error correction까지 고려하면, Toffoli depth도 결국 Qubit수로 이어짐**

- **Inversion 알고리즘 변경?**
  - **FLT Inversion이 더 좋을 것 같기도? 함**

- **박시와 협업..?**