

The 14th Workshop on Current Trends in Cryptology (CTCrypt 2025)

Quantum circuits GOST 34.10-2018, GOST 34.11-2018 and
GOST 34.12-2018 with minimum qubits

Denis Denisenko, Marina Nikitenkova

3-6 June 2025

1. GOST 34.10-2018

Publications

- 1) *Proos J., Zalka C.* Shor's discrete logarithm quantum algorithm for elliptic curves, 2003;
- 2) *Roetteller M., Naehrig M., Svore K.M., Lauter K.* Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms, 2017; ([RoeNSL2017])
- 3) *Häner T., Jaques S., Naehrig M., Roetteler M., Soeken M.* Improved Quantum Circuits for Elliptic Curve Discrete Logarithms, 2020; ([HanJNRS2020])
 - (Low Width) $\approx 8n + 10.2 \lfloor \lg n \rfloor - 1$ logical qubits, $2800n^3 - 1.08 \cdot 2^{31}$ gates¹;
 - (Low T-gates) $\approx 10n + 7.4 \lfloor \lg n \rfloor + 1.3$ qubits, $6262n^3 / \lg n - 1.72 \cdot 2^{24}$ gates;
 - (Low Depth) $\approx 11n + 3.9 \lfloor \lg n \rfloor + 16.5$ qubits, $12478n^3 / \lg n - 1.25 \cdot 2^{29}$ gates.
- 4) *Litinski D., Alto P.* How to compute a 256-bit elliptic curve private key with only 50 million Toffoli gates, 2023;
- 5) *Gouzien E., Ruiz D., Regent F.M., Guillaud J., Sangouard, N.* Computing 256-bit Elliptic Curve Logarithm in 9 Hours with 126 133 Cat Qubits, 2023. (Requires $\approx 9n + w_e + 4$ qubits, $448n^3/w_e$ CNOT and $348n^3/w_e$ Toffoli gates, $w_e = 2 \log_2 n$. If a semi-classical adder from [HanRS2017]² is used, we could get quantum circuit with $8n + w_e + 6$ logical qubits).

We have obtained, that $8n + 4$ qubits are enough for ECDLP.

¹If $n < \sqrt[3]{(1.08 \cdot 2^{31})/2800} = 93$ we get a negative number of gates!!!

²Häner T., Roetteler M., Svore K.M. Factoring using $2n+2$ qubits with Toffoli based modular multiplication, Quantum Information & Computation 17, 673 (2017), 1611.07995

GOST 34.10-2018, based on ECDLP

For $|P\rangle + Q \mapsto |P + Q\rangle$ we need basic operations with $x, y \in GF(p)$:

- 1) $|x\rangle \mapsto |(x + \text{const}) \bmod p\rangle$,
- 2) $|x\rangle |y\rangle \mapsto |x\rangle |(y + x) \bmod 2^n\rangle$
- 3) $|x\rangle |y\rangle \mapsto |x\rangle |(y + x) \bmod p\rangle$,
- 4) $|x\rangle \mapsto |(2x) \bmod p\rangle$,
- 5) $|x\rangle |y\rangle |0\rangle \mapsto |x\rangle |y\rangle |(x \cdot y) \bmod p\rangle$,
- 6) $|x\rangle \mapsto |x^{-1} \bmod p\rangle$.
- 7) $|x\rangle \mapsto |-x \bmod p\rangle$.

We have to clean ancilla qubits for using them again.

Elliptic curves in GOST 34.10-2018, Weierstrass form.

Elliptic curve E over $GF(p)$ ($p > 3$ – prime number), is a set of points (x, y) , $x, y \in GF(p)$ that satisfy a specific cubic equation:

$$y^2 = x^3 + ax + b(\bmod p),$$

where $a, b \in GF(p)$ and $4a^3 + 27b^2 \not\equiv 0 \pmod p$.

For $Q_1(x_1, y_1)$ and $Q_2(x_2, y_2)$ from E operation «+» is defined:

1. If $x_1 \neq x_2$, then $Q_1(x_1, y_1) + Q_2(x_2, y_2) = Q_3(x_3, y_3)$,

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2(\bmod p), \\ y_3 = \lambda(x_1 - x_3) - y_1(\bmod p), \end{cases}$$

where $\lambda = \frac{y_2 - y_1}{x_2 - x_1}(\bmod p)$, i.e. $\lambda = (y_2 - y_1) \cdot (x_2 - x_1)^{-1}(\bmod p)$.

2. Case $x_1 = x_2$ and $y_1 = y_2 \neq 0$ is not considered.
3. Case $x_1 = x_2$ and $y_1 = -y_2(\bmod p)$ is not considered.

Elliptic curve discrete logarithm problem (ECDLP)

For given points P and Q we have to find $k \in \mathbb{N}$, that $Q = P + \dots + P = kP$ (or $[k]P$).

In Digital Signature schemes P is known common parameter, number k is a secret key, point $Q = [k]P$ is a public key. See the recommended parameters sets in P 1323565.1.024-2019.

P.S. There are another representations of Elliptic Curves, but min number qubits is obtained when Weierstrass form is used.

Quantum circuit for Shor's ECDLP with classical $QFT^\dagger$

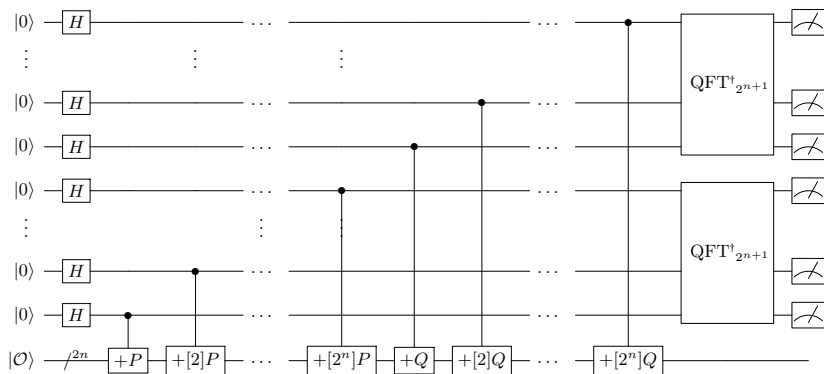


Figure 1: Quantum circuit for Shor's ECDLP from [RoeNSL2017].

In register-accumulator we have to apply controlled additions with points $P, [2]P, [4]P, \dots, [2^n]P$ and $Q, [2]Q, [4]Q, \dots, [2^n]Q$ (i.e. approx $2n + 2$ times).

Quantum circuit for Shor's ECDLP with semi-classical $QFT^\dagger$

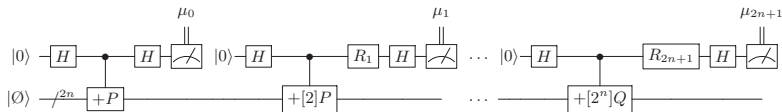


Figure 2: Quantum circuit for Shor's ECDLP with semi-classical $QFT^\dagger$ from [RoeNSL2017]. Here $R_k = \text{diag}(1, e^{i\theta_k})$, $\theta_k = -\pi \sum_{j=0}^{k-1} 2^{k-j} \mu_j$, just start quantum point add about $2n + 2$ times.

We have developed implementation of Shor's quantum algorithm for ECDLP in *Quipper*.

```
f1_myShorECDLP
f2_myRegInit
f3_myECadd2020forShor
f4_my_qftTrue
f5_fromBinary
f6_toBinary
f7_myConstAddModIntP
f8_myConstAddMicrosoft
f9_myCompareWithConst2024
f10_myIncrementTakahashi
f11_mySubtract
f12_mySum
f13_computeCarry
f14_computeCarryCascade
f15_myAdd
f16_mySubstraction
f17_myCompare
f18_myAddModPInt
f19_myIntMultXYModP
f20_myDbLModIntP
f21_myIntSquareModP
f22_myInverseModP2024
f23_myMontBitGCDRound2024
f24_mySpecFunctionForFig15
f25_mySpecFunction
f26_mySWAP
f27_myRightShift
f28_myIntMultConstXModP
f29_myMinusXModPInt
f30_myExtGCD
f31_myInverseXmodN
f32_myPowModN
```

«f23_myMontBitGCDRound2024»

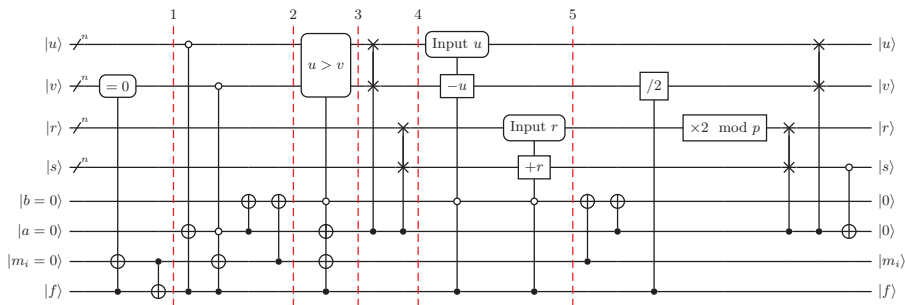


Figure 3: Quantum circuit for Kaliski's algorithm from [GouRRGS2023], f23_myMontBitGCDRound2024.

p.s. Note that there are some typos at (Fig.6) from [HanJNRS2020].

As a results

№1.

For point addition $P(x_1, y_1)$ in quantum register (i.e. $|x_1\rangle |y_1\rangle$) with another point $Q(x_2, y_2)$ ($Q \neq -P, Q \neq O, P \neq O$) it is enough $8n + 3$ qubits.

№2.

To apply Shor's algorithm to ECDLP another $2 \times (n + 1)$ qubits may be required. Thus when canonical representation of EC (Weierstrass form) is used it is enough $10n + 5$ logical qubits.

№3.

To apply Shor's algorithm to ECDLP with semi-classical $QFT^\dagger$ ^a it is enough $8n + 4$ qubits.

^aChiaverini J., Britton J., Leibfried D. et al. Implementation of the Semiclassical Quantum Fourier Transform in a Scalable System, Science 308(5724):997-1000, 2005.

Number of gates? Depends on the $|x\rangle |y\rangle \mapsto |x\rangle |(y + x) \bmod 2^n\rangle$

[HanJNRS2020] present estimates of quantum resources for three cases (minimum number of qubits, T-gates ($T = \sqrt{S}$, $S = \sqrt{Z}$, $Z = \text{HXH}$), quantum depth), each variant used its own adder:

- (Low Width) – TTK³ adder was used;
- (Low T-gates) – CDKMG⁴ adder was used;
- (Low Depth) – DKRS⁵ adder was used;

adder	# Ancilla	# gates	# Toffoli	depth
CDKM	1	$6n + 1$	$2n$	$6n + 1$
CDKM	1	$9n - 8$	$2n - 1$	$2n + 4$
CDKMG	n	$4n$	$2n - 1$	$2n - 2$
Draper ⁶	0	$1.5n^2 + 4.5n + 2$	0	$5n + 3$
DKRS	1	$7n - 6$	$2n - 1$	$2n + 2$
TK ⁷	0	$10n - 9$	$4n - 5$	$8n - 7$
TTK	0	$7n - 6$	$2n - 1$	$5n - 3$

³ Takahashi Y., Tani S., Kunihiro N., Quantum addition circuits and unbounded fan-out, Quantum Information & Computation, v. 10, n. 9, p.872-890, (2010), <http://arxiv.org/abs/0910.2530v1>

⁴ Modification of CDKM from Cuccaro S. A., Draper T. G., Kutin S. A., Moulton D. P. A new quantum ripple-carry addition circuit, (2005), arxiv:quant-ph/0410184, from Gidney C. Halving the cost of quantum addition. Quantum, 2:74, 2018.

⁵ Draper T. G., Kutin S. A., Rains E. M., Svore K. M. A logarithmic-depth quantum carry-lookahead adder, Quantum Information and Computation, 6(4&5):351–369. (2006), arxiv:quant-ph/0406142.

⁶ Draper T. G. Addition on a quantum computer, (2000), arxiv:quant-ph/0008033.

⁷ Takahashi Y., Kunihiro N. A linear-size quantum circuit for addition with no ancillary qubits, Quantum Information and Computation, 5(6):440–448. (2005).

Gate Count in WM (some functions)

```

(*f2_myRegInit::([Int],[Qubit])→Circ ([Int],[Qubit])*)
f2[x, n] := Total[IntegerDigits[x, 2, n]] * gateX; (*n – длина регистра, x-константа, которую записываем в регистр... найдём её вес*)
(*сумма цифр целого числа*)

(*f4_my_qftTrue::([Qubit])→Circ ([Qubit])*)
f4[n_] := (n * (n - 1) / 2) * gateR + n * gateH + 3 * n * CNOT;

(*f11_mySubtract::([Qubit],[Qubit])→Circ ([Qubit],[Qubit])*)
f11[n_] := (n - 1) * CNOT + (n - 2) * CNOT + (n - 1) * Toff + (n - 1) * CNOT + (n - 1) * Toff + (n - 2) * CNOT + (n) * CNOT;

(*f12_mySum::([Qubit],[Qubit])→Circ ([Qubit],[Qubit])*)
f12[n_] := (n - 1) * CNOT + (n - 2) * CNOT + (n - 1) * Toff + (n - 1) * CNOT + (n - 1) * Toff + (n - 2) * CNOT + (n) * CNOT;

(*f15_myAdd::([Qubit],[Qubit],[Qubit])→Circ ([Qubit],[Qubit],[Qubit])*)
f15[n_] := (n - 1) * CNOT + (n - 2) * CNOT + (n - 1) * Toff + (n - 1) * CNOT + (n - 1) * Toff + (n - 2) * CNOT + (n) * CNOT + 1 * CNOT + 1 * Toff;

(*f16_mySubtraction::([Qubit],[Qubit],[Qubit])→Circ ([Qubit],[Qubit],[Qubit])*)
f16[n_] := (n - 1) * CNOT + (n - 2) * CNOT + (n - 1) * Toff + (n - 1) * CNOT + (n - 1) * Toff + (n - 2) * CNOT + (n) * CNOT + 1 * CNOT + 1 * Toff;

(*проверил 05.12.2024*) (*f14_computeCarryCascade::([Int],[Qubit],[Qubit])→Circ ([Int],[Qubit],[Qubit])*)
f14[intX, n] := Module[{res},
  (*программный модуль*)
  res = Total[IntegerDigits[intX, 2, n][[3 ;; n]]] * CNOT + Total[IntegerDigits[intX, 2, n][[3 ;; n]]] * gateX + (n - 3) * Toff; (*нумерация с 0, 0,1,2,3... младшие – внизу схемы*)
  (*сумма цифр целого числа*)
  res = res + IntegerDigits[intX, 2, n][[2]] * CNOT;
  (*цифры целого числа*)
  res = res + IntegerDigits[intX, 2, n][[1]] * IntegerDigits[intX, 2, n][[2]] * gateX + IntegerDigits[intX, 2, n][[1]] * Toff;
  (*цифры целого числа*)
  res = res + (n - 1) * Toff;
  Expand[res]
  (*раскрыть скобки*)
]

```

Gate Count in WM ($f3 : (|xP\rangle |yP\rangle, (xQ, yQ)) \mapsto |x(P + Q)\rangle |y(P + Q)\rangle$)

```
f3[paramP_, n_, xQ_, yQ_] := Module[{res, tmpX, f},
  [программный модуль]
```

```
res = 0;
```

```
res =
```

```
f7[paramP, xQ, n] +
```

```
f7[paramP, yQ, n] +
```

```
f22[paramP, n] +
```

```
f19[paramP, n] +
```

```
f22[paramP, n] +
```

```
f19[paramP, n] +
```

```
n * CNOT +
```

```
f19[paramP, n] +
```

```
f7[paramP, Mod[(3 * xQ), paramP], n] +
```

```
[остаток от деления]
```

```
gateX +
```

```
f21[paramP, n] +
```

```
f18[paramP, n] +
```

```
f21[paramP, n] +
```

```
gateX +
```

```
f19[paramP, n] +
```

```
f22[paramP, n] +
```

```
f19[paramP, n] +
```

```
f22[paramP, n] +
```

```
f7[paramP, yQ, n] +
```

```
f7[paramP, xQ, n] +
```

```
f29[(2^n - 1 - paramP), n] (* в f29 в качестве параметра указываем поправку, которую необходимо вычесть из 2^n-1, чтобы в результате инверсии бит получить число -x mod p;
```

```
в рассматриваемом примере при p=29 эта поправка равна 31-29 = 2. *)
```

```
Expand[res]
```

```
[раскрыть скобки]
```

```
]
```

```
tmp = f3[29, 5, 15, 2];
```

```
(* tmp = f3[513, 512, 15, 2]; *)
```

```
tmp = f3[13, 4, 12, 12];
```

```
In[ ]:= tmp
```

```
Out[ ]:= 3473 CNOT + 14 CNOTc0 + 960 CNOTc0c1 + 631 gateX + 102 gateXc0 + 2 gateXc0c0c0 + 32 gateXc0c0c0c0c1 +
32 gateXc0c0c1 + 32 gateXc0c1 + 260 SWAP + 608 SWAPc1 + 4660 Toff + 2 Toffc0 + 448 Toffc0c1 + 1818 Toffc1 + 168 Toffc1c1
```

Shorted notations:

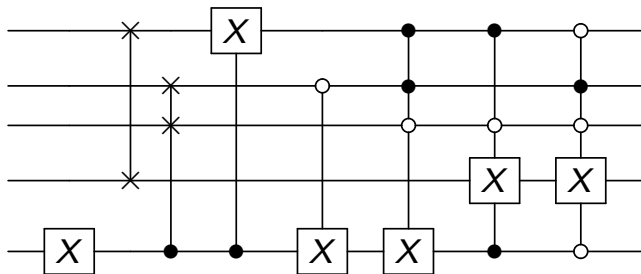


Figure 4: Examples: gateX, SWAP, SWAP_c1, gateX_c1, gateX_o1, gateX_c2o1, gateX_c2o1, gateX_c1o3.

P 1323565.1.024-2019

Let's calculate the exact value of the number of quantum gates required to add some point $Q(x_2, y_2)$ to the point $P(x_1, y_1)$ in quantum register $(|x_1\rangle |y_2\rangle)$ for the parameters from P 1323565.1.024-2019.

	$Q = G + G$	$Q = G + G + G + G$	$Q = [8]G$	$Q = [16]G$
gateX	17 464 706	17 451 107	17 450 492	17 449 919
SWAP	1 242 368	1 242 368	1 242 368	1 242 368
gateX_c1	74 084 542	74 072 065	74 071 736	74 071 362
gateX_c1o1	7193	7193	7193	7193
gateX_c1o2	2048	2048	2048	2048
gateX_c1o512	2048	2048	2048	2048
gateX_c2	169 789 615	169 788 275	169 787 301	169 787 403
gateX_c2o1	5 239 784	5 239 784	5 239 784	5 239 784
gateX_c3	171 471 068	171 471 068	171 471 068	171 471 068
gateX_c3o1	2 097 992	2 097 992	2 097 992	2 097 992
gateX_c4	67 603 736	67 603 736	67 603 736	67 603 736
gateX_c4o1	64	64	64	64
gateX_c5	9 301 696	9 301 696	9 301 696	9 301 696
gateX_c6	114 688	114 688	114 688	114 688
gateX_o1	7 241	7 241	7 241	7 241
gateX_o512	2	2	2	2
SWAP_c1	2 619 392	2 619 392	2 619 392	2 619 392
Total gates (TTK)	521 048 183	521 020 767	521 018 849	521 018 004

Table 1: Quantum gates for $|P\rangle + Q \mapsto |P + Q\rangle$, for id-tc26-gost-3410-2012-256-paramSetB.

	$Q = G + G$	$Q = G + G + G + G$	$Q = [8]G$	$Q = [16]G$
gateX	618 125	614 939	614 560	614 671
SWAP	1 242 368	1 242 368	1 242 368	1 242 368
gateX_c1	35 458 997	35 456 298	35 455 644	35 455 808
gateX_c1o1	10 197	10 197	10 197	10 197
gateX_c1o2	2048	2048	2048	2048
gateX_c1o512	2048	2048	2048	2048
gateX_c2	139 933 039	139 932 389	139 932 311	139 932 297
gateX_c2o1	5 240 260	5 240 260	5 240 260	5 240 260
gateX_c3	162 439 656	162 439 656	162 439 656	162 439 656
gateX_c3o1	2 097 992	2 097 992	2 097 992	2 097 992
gateX_c4	66 690 744	66 690 744	66 690 744	66 690 744
gateX_c4o1	64	64	64	64
gateX_c5	9 300 928	9 300 928	9 300 928	9 300 928
gateX_c6	114 688	114 688	114 688	114 688
gateX_o1	10 346	10 346	10 346	10 346
gateX_o512	2	2	2	2
SWAP_c1	2 619 392	2 619 392	2 619 392	2 619 392
Total gates (TTK)	425 780 894	425 774 359	425 773 248	425 773 509

Table 2: Quantum gates for $|P\rangle + Q \mapsto |P + Q\rangle$, for id-tc26-gost-3410-2012-256-paramSetC.

	$Q = G + G$	$Q = G + G + G + G$	$Q = [8]G$	$Q = [16]G$
gateX	8 303 257	8 303 113	8 301 859	8 303 188
SWAP	1 242 368	1 242 368	1 242 368	1 242 368
gateX_c1	53 701 401	53 701 428	53 699 719	53 701 439
gateX_c1o1	8 753	8 753	8 753	8 753
gateX_c1o2	2 048	2 048	2 048	2 048
gateX_c1o512	2 048	2 048	2 048	2 048
gateX_c2	155 336 645	155 336 279	155 336 317	155 336 323
gateX_c2o1	5240008	5240008	5240008	5240008
gateX_c3	167 982 364	167 982 364	167 982 364	167 982 364
gateX_c3o1	2 097 992	2 097 992	2 097 992	2 097 992
gateX_c4	67 402 328	67 402 328	67 402 328	67 402 328
gateX_c4o1	64	64	64	64
gateX_c5	9 304 256	9 304 256	9 304 256	9 304 256
gateX_c6	114 688	114 688	114 688	114 688
gateX_o1	8 746	8 746	8 746	8 746
gateX_o512	2	2	2	2
SWAP_c1	2 619 392	2 619 392	2 619 392	2 619 392
Total gates (TTK)	473 366 360	473 365 877	473 362 952	473 366 007

Table 3: Quantum gates for $|P\rangle + Q \mapsto |P + Q\rangle$, for id-tc26-gost-3410-2012-256-paramSetD.

	$Q = G + G$	$Q = G + G + G + G$	$Q = [8]G$	$Q = [16]G$
gateX	81 339 196	81 339 646	81 337 186	81 337 850
SWAP	4 975 104	4 975 104	4 975 104	4 975 104
gateX_c1	340 097 276	340 097 452	340 094 796	340 096 422
gateX_c1o1	15 849	15 849	15 849	15 849
gateX_c1o2	4096	4096	4096	4096
gateX_c1o512	4096	4096	4096	4096
gateX_c2	797 078 021	797 078 177	797 078 203	797 077 513
gateX_c2o1	20 972 460	20 972 460	20 972 460	20 972 460
gateX_c3	819 484 944	819 484 944	819 484 944	819 484 944
gateX_c3o1	8392328	8392328	8392328	8392328
gateX_c4	325 258 936	325 258 936	325 258 936	325 258 936
gateX_c4o1	128	128	128	128
gateX_c5	44 520 320	44 520 320	44 520 320	44 520 320
gateX_c6	458 752	458 752	458 752	458 752
gateX_o1	14 429	14 429	14 429	14 429
gateX_o512	2	2	2	2
SWAP_c1	10 481 664	10 481 664	10 481 664	10 481 664
Total gates (TTK)	2 453 097 601	2 371 758 737	2 453 093 293	2 453 094 893

Table 4: Quantum gates for $|P\rangle + Q \mapsto |P + Q\rangle$, for id-tc26-gost-3410-2012-512-paramSetA.

	$Q = G + G$	$Q = G + G + G + G$	$Q = [8]G$	$Q = [16]G$
gateX	1 681 086	1 681 719	1 681 565	1 681 025
SWAP	4 975 104	4 975 104	4 975 104	4 975 104
gateX_c1	158 567 150	158 567 997	158 567 768	158 567 109
gateX_c1o1	23 025	23 025	23 025	23 025
gateX_c1o2	4 096	4 096	4 096	4 096
gateX_c1o512	4 096	4 096	4 096	4 096
gateX_c2	662 475 465	662 475 899	662 475 329	662 475 771
gateX_c2o1	20 973 444	20 973 444	20 973 444	20 973 444
gateX_c3	784 496 552	784 496 552	784 496 552	784 496 552
gateX_c3o1	8 392 328	8 392 328	8 392 328	8 392 328
gateX_c4	322 791 800	322 791 800	322 791 800	322 791 800
gateX_c4o1	128	128	128	128
gateX_c5	44 531 584	44 531 584	44 531 584	44 531 584
gateX_c6	458 752	458 752	458 752	458 752
gateX_o1	21 862	21 862	21 862	21 862
gateX_o512	2	2	2	2
SWAP_c1	10 481 664	10 481 664	10 481 664	10 481 664
Total gates (TTK)	2 019 878 138	2 019 880 052	2 019 879 099	2 019 878 342

Table 5: Quantum gates for $|P\rangle + Q \mapsto |P + Q\rangle$, for id-tc26-gost-3410-2012-512-paramSetB.

$C^n(X)$ decomposition by $O(n)$ Toffoli, CNOT and single qubit gates

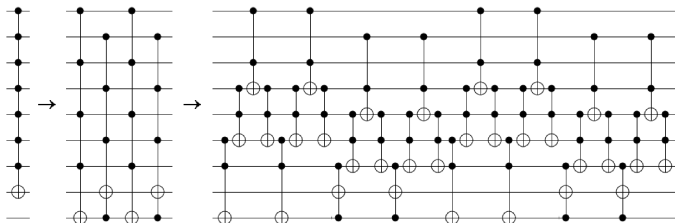


Figure 5: Example 1. *Craig Gidney*. StackExchange: Creating bigger controlled nots from single qubit, Toffoli, and CNOT gates, without workspace. 2015.

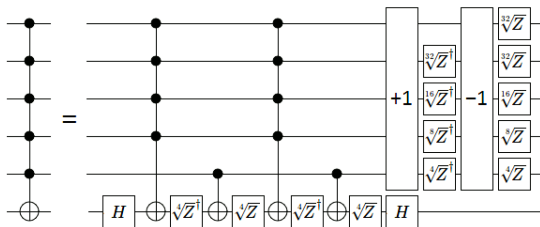


Figure 6: Example 2. *Craig Gidney*. Using Quantum Gates instead of Ancilla Bits.

param set	Qubit count for ECDLP by Shor ($8n + 4$, see fig. 2)	Gate count for $ P\rangle + Q \mapsto P + Q\rangle$
256-paramSetB	2052	$5.21 \cdot 10^8$
256-paramSetC	2052	$4.25 \cdot 10^8$
256-paramSetD	2052	$4.73 \cdot 10^8$
512-paramSetA	4100	$2.453 \cdot 10^9$
512-paramSetB	4100	$2.019 \cdot 10^9$

Table 6: Quantum resources for Shor's ECDLP for P 1323565.1.024-2019.

2. GOST 34.11-2018

Hash function GOST 34.11-2018 (Streebog) – Merkle-Damgård construction:

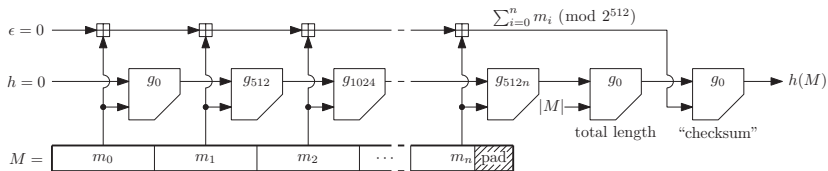


Figure 7: from Markku-Juhani O. Saarinen «STRIBOB: Authenticated Encryption from GOST R 34.11-2012 LPS Permutation», eprint: 2014-271.

The compression function operates in Miyaguchi-Preneel mode:

$$g(h, m) = E(h, m) \oplus h \oplus m.$$

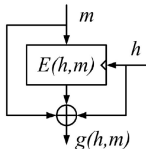


Figure 8: Miyaguchi-Preneel.

GOST 34.11-2018

For message $M \in V^*$:

$$g_N : V_{512} \times V_{512} \rightarrow V_{512}, \quad N \in V_{512},$$

$$g_N(h, m) = E(LPS(h \oplus N), m) \oplus h \oplus m,$$

where

$$E(K, m) = X[K_{13}]LPSX[K_{12}] \dots LPSX[K_2]LPSX[K_1](m),$$

$$K_i \in V_{512}, \quad i = 1, \dots, 13,$$

$$K_1 = K;$$

$$K_i = LPS(K_{i-1} \oplus C_{i-1}), \quad i = 2, \dots, 13.$$

Let

$$g_N : V_{16} \times V_{16} \rightarrow V_{16}, \quad N \in V_{16},$$

$$g_N(h, m) = E(S(h \oplus N), m) \oplus h \oplus m,$$

$$E(K, m) = X[K_{13}]SX[K_{12}] \dots SX[K_2]SX[K_1](m),$$

$$K_i \in V_{16}, \quad i = 1, \dots, 13,$$

$$K_1 = K; K_i = S(K_{i-1} \oplus C_{i-1}), \quad i = 2, \dots, 13.$$

Let's define $S : V_{16} \rightarrow V_{16}$

$$S(x_1, \dots, x_{16}) = L(\pi_0(x_1, \dots, x_4) || \pi_1(x_5, \dots, x_8) || \pi_2(x_9, \dots, x_{12}) || \pi_3(x_{13}, \dots, x_{16}))$$

with $\pi_0, \pi_1, \pi_2, \pi_3$ from GOST 34.12-2018 «Magma» and linear transform $L(\vec{x}) = \vec{x} \cdot A$.

Let $C_i \in V_{16}, (i = 1, 2, \dots, 12) = \{30299, 28346, 13331, 44219, 41318, 64991, 62659, 52792, 62737, 5031, 27713, 20813\}$.

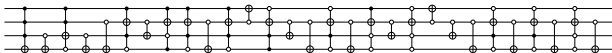


Figure 9: $\pi_0 = (12, 4, 6, 2, 10, 5, 11, 9, 14, 8, 13, 7, 0, 3, 15, 1)$.

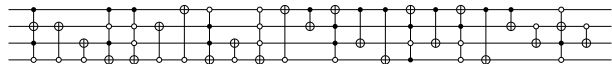


Figure 10: $\pi_1 = (6, 8, 2, 3, 9, 10, 5, 12, 1, 14, 4, 7, 11, 13, 0, 15)$.

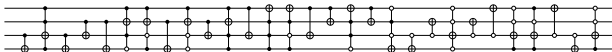


Figure 11: $\pi_2 = (11, 3, 5, 8, 2, 15, 10, 13, 14, 1, 7, 4, 12, 9, 6, 0)$.

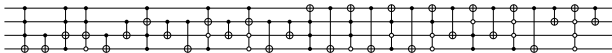


Figure 12: $\pi_3 = (12, 8, 2, 1, 13, 4, 15, 6, 7, 0, 10, 5, 3, 14, 9, 11)$.

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix},$$

To get PLU-decomposition we could use «LUdecomposition» from WM.

For matrix A we obtain $p=(2,1,3,4,6,5,7,8,10,9,11,12,14,13,15,16)$,

[illegible]

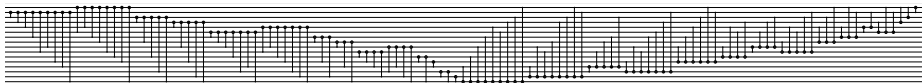


Figure 13: Quantum circuit for $|\vec{x}\rangle \rightarrow |\vec{x} \cdot A\rangle$ with reordering qubits. For $A_{16 \times 16}$ it is enough 123 CNOT.

№4.

We could implement linear transform «L» without any ancilla qubits by approx $n^2/2$ CNOT.

For $\ell : V_{64} \rightarrow V_{64}$ from GOST 34.11-2018 it is enough 2031 CNOT (1959 CNOT and 24 SWAP).

For $L : V_{128} \rightarrow V_{128}$ from GOST 34.12-2018 «Kuznechik» it is enough 8360 CNOT.

Gates for Sbox GOST 34.11-2018

Permutation τ from GOST 34.11-2018 could be implemented by 224 SWAP.

808: "not, arity 1" controls 0+1
2: "not, arity 1" controls 0+7
88: "not, arity 1", controls 1
11: "not, arity 1" controls 1+6
19: "not, arity 1" controls 2+5
56: "not, arity 1" controls 3+4
68: "not, arity 1" controls 4+3
63: "not, arity 1" controls 5+2
29: "not, arity 1" controls 6+1
6: "not, arity 1", controls 7
Total gates: 1150
Inputs: 8
Outputs: 8
Qubits in circuit: 8

Table 7: S-box $\pi : V_8 \rightarrow V_8$ from GOST 34.11-2018 and GOST 34.12-2018 «Kuznechik» (output «GateCount» in Quipper)⁸. For S-box $\pi : V_8 \rightarrow V_8$ we need 896 CNOT and 254 CCNOT(7,1).

«88: "not, arity 1" controls 1» == 88 gateX_c1;
 «808: "not, arity 1" controls 0+1» == 808 gateX_o1;

⁸by algorithm from Denisenko D.V., Nikitenkova M.V. Optimization of S-boxes GOST R 34.12-2015 «Magma» quantum circuits without ancilla qubits. Matematicheskie Voprosy Kriptografii v.11(2), p. 43–52, 2020

Minimum qubits for GOST 34.11-2018

2 cases:

- If $|M| < 16$, then for getting $|M\rangle |h(M)\rangle$ we have to compute compression function g three times and it is enough $|M| + 3 \cdot 16$ qubits (i.e. **64** qubits).
- If $16 \leq |M| < 32$, then for getting $|M\rangle |h(M)\rangle$ we have to compute compression function g four times and it is enough $|M| + 3 \cdot 16$ qubits (i.e. **80** qubits).

Thus, for GOST 34.11-2018:

3 times g , $|M| < 512$

For getting $|M\rangle |h(M)\rangle$ it is enough $|M| + 3 \cdot 512$ qubits (i.e. **$512 + 1536 = 2048$** qubits).

4 times g , $512 \leq |M| < 1024$

For getting $|M\rangle |h(M)\rangle$ it is enough $|M| + 3 \cdot 512$ qubits (i.e. **$1024 + 1536 = 2560$** qubits).

Example A.1 from GOST 34.11-2018

18 819: "X, arity 1"
7 756 800: "not, arity 1" controls 0+1
19 200: "not, arity 1" controls 0+7
3 241 168: "not, arity 1", controls 1
105 600: "not, arity 1" controls 1+6
182 400: "not, arity 1" controls 2+5
537 600: "not, arity 1" controls 3+4
652 800: "not, arity 1" controls 4+3
604 800: "not, arity 1" controls 5+2
278 400: "not, arity 1" controls 6+1
57 600: "not, arity 1", controls 7
62 400: "swap, arity 2"
Total gates: 13 517 587
Inputs: 2048
Outputs: 2048
Qubits in circuit: 2048

Table 8: Quantum resources for GOST 34.11-2018 example A.1.

3. GOST 34.12-2018

Kuznechik

3992: «not, arity 1»
943 744: «not, arity 1» controls 0+1
2336: «not, arity 1» controls 0+7
12 480: «not, arity 1» controls 1
12 848: «not, arity 1» controls 1+6
22 192: «not, arity 1» controls 2+5
65 408: «not, arity 1» controls 3+4
79 424: «not, arity 1» controls 4+3
73 584: «not, arity 1» controls 5+2
33 872: «not, arity 1» controls 6+1
7 008: «not, arity 1», controls 7
960: «swap, arity 2»
Total gates: 1 966 944
Inputs: 384
Outputs: 384
Qubits in circuit: 384

Table 9: Quantum resources for GOST 34.12-2018 «Kuznechik».

Magma

S-boxes GOST 34.12-2018 «Magma»

Best circuits for 4-bit sboxes obtained by LIGHTER-R. For example:

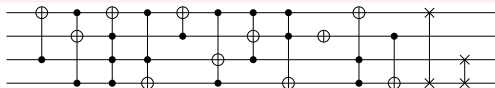


Figure 14: $\pi_0 = (12, 4, 6, 2, 10, 5, 11, 9, 14, 8, 13, 7, 0, 3, 15, 1)$.

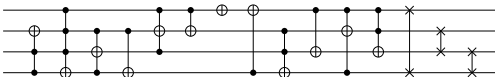


Figure 15: $\pi_1 = (6, 8, 2, 3, 9, 10, 5, 12, 1, 14, 4, 7, 11, 13, 0, 15)$.

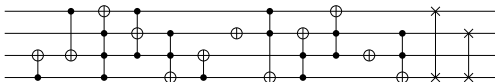


Figure 16: $\pi_2 = (11, 3, 5, 8, 2, 15, 10, 13, 14, 1, 7, 4, 12, 9, 6, 0)$.

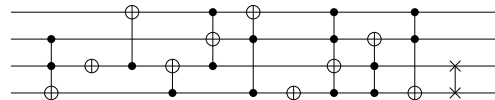


Figure 17: $\pi_3 = (12, 8, 2, 1, 13, 4, 15, 6, 7, 0, 10, 5, 3, 14, 9, 11)$.

Magma

768: «not, arity 1»
12 480: «not, arity 1», controls 1
6 720: «not, arity 1», controls 2
640: «not, arity 1», controls 3
960: «swap, arity 2»
Total gates: 21 568
Inputs: 320
Outputs: 320
Qubits in circuit: 320

Table 10: Quantum resources for GOST 34.12-2018 «Magma».

Conclusion

№1.

For point addition $P(x_1, y_1)$ in quantum register (i.e. $|x_1\rangle |y_1\rangle$) with another point $Q(x_2, y_2)$ ($Q \neq -P, Q \neq O, P \neq O$) it is enough $8n + 3$ qubits.

№2.

To apply Shor's algorithm to ECDLP another $2 \times (n + 1)$ qubits may be required. Thus when canonical representation of EC (Weierstrass form) is used it is enough $10n + 5$ logical qubits.

№3.

To apply Shor's algorithm to ECDLP with semi-classical $QFT^\dagger$ ^a it is enough $8n + 4$ qubits.

^aChiaverini J., Britton J., Leibfried D. et al. Implementation of the Semiclassical Quantum Fourier Transform in a Scalable System, Science 308(5724):997-1000, 2005.

№4.

We could implement linear transform «L» without any ancilla qubits by approx $n^2/2$ CNOT. For $\ell: V_{64} \rightarrow V_{64}$ from GOST 34.11-2018 it is enough 2031 CNOT (1959 CNOT and 24 SWAP). For $L: V_{128} \rightarrow V_{128}$ from GOST 34.12-2018 «Kuznechik» it is enough 8360 CNOT.

Conclusion

№5.

For GOST-34.11-2018 example A.1 (512-bit) it is required 2048 qubits and 13 517 587 gates.

№6.

For GOST-34.12-2018 example A.2 (Kuznechik) it is required 384 qubits and 1 966 944 gates.

№7.

For GOST-34.12-2018 example A.3 (Magma) it is required 320 qubits and 21 568 gates.

Thanks for your attention!