MUTOWA · QUANTUM COMPUTE · WHITEPAPER

Quantum Compute

Quantum readiness, built like infrastructure. The 16-qubit processor explained from the junction up: coherence, cryogenics, control, gates and error correction, with the post-quantum security stack and a method for the transition.

November 2026 · Version 2.1
Winnipeg, Canada · Tokyo, Japan
mutowa.com

Contents

1 Summary 5

1.1 What this edition adds 5

1.2 How to read the figures 5

2 Why quantum, and why now 6

2.1 What a quantum computer is good for 6

2.2 The cryptographic clock 6

2.3 Readiness as infrastructure 7

3 The processor at a glance 7

3.1 Why heavy-hex 8

3.2 What sixteen qubits are for 8

4 Transmon qubits 9

4.1 The circuit 9

4.2 The substrate 9

4.3 Why it has to be cold 9

5 Coherence 10

5.1 Two times 10

5.2 Dynamical decoupling 10

5.4 How coherence is measured 10

6 Cryogenics 11

6.1 How a dilution refrigerator cools 11

6.2 Cooling power 11

6.3 Shielding 11

7 Microwave control 12

7.1 Waveform generation 12

7.2 The frequency plan 12

7.3 The signal path 12

8 The native gate set 13

8.1 Virtual-Z: a gate that takes no time 13

8.2 The √X pulse and DRAG 13

8.3 The cross-resonance gate 13

8.4 How a fidelity is measured . . . 14
8.5 Reading a fidelity . . . 14

9 Readout and reset . . . 14
9.1 Dispersive readout . . . 14
9.2 The Purcell filter . . . 14
9.3 How readout fidelity is measured . . . 15
9.4 Mid-circuit reset . . . 15

10 Error correction . . . 15
10.1 The surface code . . . 15
10.2 One cycle, step by step . . . 16
10.3 Distance . . . 16
10.4 The threshold . . . 16

11 Software and access . . . 17
11.1 Circuits . . . 17
11.2 Compilation . . . 17
11.3 Submission and scheduling . . . 17

12 The post-quantum security stack . . . 18
12.1 The two standards . . . 18
12.2 Hybrid transport . . . 18
12.3 Crypto agility . . . 18

13 Quantum key distribution . . . 19
13.1 BB84 . . . 19
13.2 The link . . . 19
13.3 Where QKD fits . . . 20

14 Hybrid architectures . . . 20
14.1 Three kinds of resource . . . 20
14.2 A shared security mesh . . . 20
14.3 The Quantum node . . . 21

15 Algorithms and use cases . . . 21
15.1 What a small processor runs . . . 21
15.2 Sorting workloads . . . 21
15.3 Kill criteria . . . 21

16 Transition planning 22

16.1 Assess 22

16.2 Prototype 22

16.3 Integrate 22

17 The quantum coin flip 23

18 Limits and assumptions 23

19 Glossary 24

20 References and about MUTOWA 25

20.1 Standards and further reading 25

20.2 About MUTOWA 25

20.3 Next steps 26

01

Summary

MUTOWA Quantum Compute is two things built together: a superconducting quantum processor run as a research system, and a post-quantum security practice that protects ordinary systems against the machines that processors like it will one day become.

The processor is 16 transmon qubits on a heavy-hex lattice, held near 15 millikelvin in a dilution refrigerator and driven by microwave pulses shaped at two billion samples per second. Its published figures are 120 µs of energy relaxation time, 99.94% single-qubit fidelity and 99.4% two-qubit fidelity in 200 ns. Its error-correction specification is a distance-5 surface code with a one-microsecond cycle. Around it sits a security stack built on the NIST post-quantum standards, with quantum key distribution for links that warrant it.

Qubits	16 transmons on a heavy-hex lattice
Coherence	T_1 120 µs, T_2 95 µs
Single-qubit gates	Virtual-Z and DRAG-corrected √X, 25 ns, 99.94% fidelity
Two-qubit gates	Cross-resonance (ECR), 200 ns, 99.4% fidelity
Readout	6.0 to 7.5 GHz resonators with Purcell filter, 98.7% measurement fidelity
Cryogenics	Bluefors XLD400 dilution refrigerator, about 15 mK base, 400 µW at 100 mK
Control	2 GSa/s waveform generation, 0.5 ns pulse resolution
Error correction	Surface code, d=5, 1 µs cycle, about 1×10^{-3} logical error per cycle
Security	Kyber-1024 (FIPS 203), Dilithium-5 (FIPS 204), BB84 QKD; NIST category 5
Software	OpenQASM 3.0; Python, TypeScript and Rust SDKs; gRPC over TLS 1.3; Slurm scheduling

Table 1. Quantum Compute at a glance. These are the figures published at mutowa.com/ventures/quantum.

1.1 What this edition adds

Version 1 of this paper listed the processor's specifications and MUTOWA's readiness method. This edition explains the physics and engineering behind each figure: what a transmon is and why it is cold, what the coherence times allow, how the gates and the readout work, how a surface code turns many imperfect qubits into one better one, and how the post-quantum standards fit together. It closes with a chapter on limits and assumptions.

1.2 How to read the figures

Specifications are the ones MUTOWA publishes for the processor. Where the paper derives something from them (how many gates fit in a coherence time, how cold 15 mK is relative to a qubit's energy), it says so and shows the arithmetic. General explanations of the physics follow the published literature listed in the last chapter.

02

Why quantum, and why now

Two clocks are running. One measures how long until quantum computers do useful work that classical machines cannot. The other measures how long until they break the cryptography that protects today's data. Readiness means working on both without being fooled by either.

2.1 What a quantum computer is good for

A quantum computer is not a faster classical computer. It is a different kind of machine that is dramatically better at a short list of problems and no better at most others. The list includes simulating quantum systems (molecules, materials), certain optimization and sampling problems, and, most consequentially for security, factoring large numbers and computing discrete logarithms, the two problems on which nearly all deployed public-key cryptography rests.

2.2 The cryptographic clock

Shor's algorithm solves factoring and discrete logarithms efficiently on a large, error-corrected quantum computer. Such a machine would break RSA, Diffie-Hellman and elliptic-curve cryptography. It does not exist today, and building one requires error correction at a scale far beyond current processors. But the threat does not wait for the machine: encrypted traffic recorded today can be decrypted whenever the machine arrives. Data with a long shelf life is therefore exposed now.

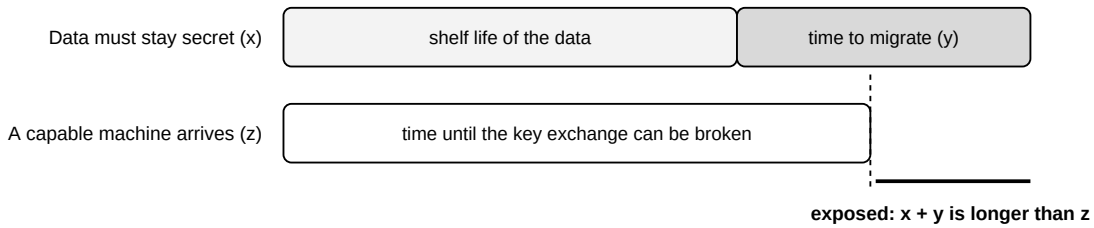

Figure 1. Mosca's inequality. If the time data must stay secret (x) plus the time needed to migrate (y) exceeds the time until the key exchange can be broken (z), the data is already exposed.

Symmetric cryptography is far less affected. Grover's algorithm gives a quadratic speed-up for searching a key space, which roughly halves the effective strength of a symmetric key: AES-256 keeps a comfortable margin. The urgent work is replacing key exchange and signatures, which is what the NIST post-quantum standards do (chapter 12).

Cryptography	Used for	Effect of a large quantum computer	What to do
RSA	Key exchange, signatures	Broken (Shor)	Replace
Diffie-Hellman, ECDH	Key exchange	Broken (Shor)	Replace, starting now: recorded traffic is at risk
ECDSA, EdDSA	Signatures	Broken (Shor)	Replace before such a machine exists
AES-128	Encryption	Weakened (Grover)	Move to 256-bit keys
AES-256	Encryption	Holds, with a reduced margin	Keep
SHA-2 at 256 bits or more	Hashing	Holds	Keep

Table 2. What a large, error-corrected quantum computer would and would not break.

2.3 Readiness as infrastructure

MUTOWA treats quantum capability as a long-horizon systems problem and not a single switch. Its practice has four parts, which the rest of this paper follows:

- **Research systems** for evaluating algorithms, hardware and use cases in controlled, measurable settings (chapters 3 to 11).
- **Post-quantum transition**: inventory, risk assessment and migration planning for systems that must stay secure for decades (chapters 12, 13 and 16).
- **Hybrid architectures** that connect classical, edge and quantum resources without an operational break (chapter 14).
- **Strategic evaluation**: deciding which workloads deserve sustained effort, with technical and economic discipline (chapter 15).

03

The processor at a glance

The MUTOWA QPU is a 16-qubit superconducting processor. Its qubits are transmons, arranged on a heavy-hex lattice in which every qubit is coupled to two or three neighbors.

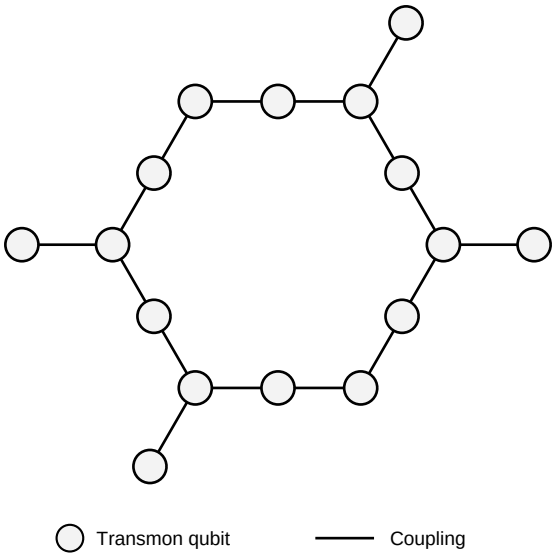

Figure 2. The 16-qubit core: a heavy hexagon of 12 qubits with 4 spurs. Each qubit couples to two or three neighbors.

3.1 Why heavy-hex

On a square lattice each qubit has four neighbors. On a heavy-hex lattice, qubits sit on the corners and along the edges of hexagons, so none has more than three. Fewer neighbors means fewer pairs of qubits whose frequencies can collide and fewer paths for unwanted crosstalk, the two effects that most limit the fidelity of fixed-frequency transmons driven by cross-resonance. The lattice gives up some connectivity to gain cleaner gates.

Active qubits	16 transmons
Topology	Heavy-hex lattice
Coherence T_1 / T_2	120 µs / 95 µs
Single-qubit fidelity	99.94%
Two-qubit fidelity	99.4% (ECR, 200 ns)
Measurement fidelity	98.7%
Base temperature	About 15 mK

Table 3. The processor's published core figures.

3.2 What sixteen qubits are for

Sixteen qubits span a state space of 2^{16} = 65,536 amplitudes. That is small enough for a classical computer to simulate exactly, and that is the point of a processor this size: every result it produces can be checked against an exact calculation. A 16-qubit system is where calibration, control, readout, error-correction cycles and software are developed and measured under real conditions, before any of them is trusted at a scale where checking is no longer possible. It is a research system, and MUTOWA describes it as one.

04

Transmon qubits

A transmon is an electrical circuit that behaves like an artificial atom. Its two lowest energy levels are the qubit.

4.1 The circuit

At the heart of each qubit is a Josephson junction: two superconductors separated by a thin insulating barrier. The junction behaves as an inductor whose inductance depends on the current through it, and it dissipates nothing. Connected to a capacitor, it forms an oscillator whose energy levels are not evenly spaced. That unevenness (anharmonicity) is what makes a qubit possible: a microwave pulse tuned to the gap between the lowest two levels moves the circuit between them without exciting the third.

The transmon design adds a large capacitor across the junction. This makes the circuit almost insensitive to stray electric charge, the noise source that limited earlier superconducting qubits, at the cost of a smaller anharmonicity. The MUTOWA transmons have a transition frequency of 4.5 to 5.5 GHz and an anharmonicity of 250 MHz.

4.2 The substrate

The qubits are transmon arrays on annealed sapphire (99.999% Al_2O_3), with lithographically defined Josephson junctions. Sapphire is chosen because it absorbs very little microwave energy: a qubit's stored energy leaks away through defects in the materials around it, and a cleaner, more ordered substrate means fewer defects for it to leak into.

4.3 Why it has to be cold

Two requirements set the operating temperature. The metal must be superconducting, which for the metals commonly used in such circuits happens at around a kelvin or above. And the qubit must sit in its lowest state when left alone, which requires the surrounding thermal energy to be far smaller than the energy of one quantum of the qubit. The second requirement is the stricter one.

A photon at 5 GHz carries an energy equivalent to a temperature of 0.24 K (Planck's constant times the frequency, divided by Boltzmann's constant). At the processor's 15 mK, that is 16 times the thermal energy, and the chance of finding the qubit thermally excited is of order e^{-16}, about one in ten million. At 4 K, the temperature of liquid helium, the qubit would be excited almost half the time and would hold no information at all.

Physical qubits	Transmon arrays on annealed sapphire substrate (99.999% Al_2O_3), lithographically defined Josephson junctions
Transition frequency	4.5 to 5.5 GHz
Anharmonicity	250 MHz
Operating temperature	15 mK

Table 4. Published qubit figures.

05

Coherence

A qubit holds its state for a limited time. Everything a quantum processor does must fit inside that time, which makes coherence the budget every other figure is spent from.

5.1 Two times

T_1, energy relaxation, is how long an excited qubit takes to decay to its ground state. It is 120 µs at the processor's 15 mK base temperature. **T_2, dephasing**, is how long a superposition keeps a definite phase between its two parts. It is 95 µs with dynamical decoupling. T_2 can never exceed twice T_1, and in practice it is shorter, because slow drifts in a qubit's frequency scramble its phase before its energy is lost.

5.2 Dynamical decoupling

The published T_2 is measured with a CPMG-XY4 sequence. Dynamical decoupling inserts a train of fast, precisely timed pulses that flip the qubit back and forth. Phase picked up from a slowly varying disturbance during one interval is undone during the next, so only noise that changes faster than the pulse spacing gets through. The XY4 variant alternates the axis of the pulses so that small errors in the pulses themselves cancel instead of accumulating.

5.4 How coherence is measured

Coherence figures of this kind are measured in a standard way. For T_1, the qubit is excited, left alone for a chosen delay and then measured; repeating this over many delays traces an exponential decay whose time constant is T_1. For T_2, the qubit is put in a superposition, left for a delay (with or without decoupling pulses) and rotated back; the decay of the returning signal gives T_2. Each point is an average over thousands of repetitions, because a single measurement yields only a 0 or a 1.

Operation	Duration	How many fit in T_1 (120 µs)
Single-qubit gate (√X)	25 ns	4,800
Two-qubit gate (ECR)	200 ns	600
Ancilla reset	200 ns or less	600 or more
Error-correction cycle	1 µs	120

Table 5. Operation times against the relaxation time. Arithmetic on the published figures.

The ratio of coherence time to gate time bounds how good a gate can be: a two-qubit gate that takes 1/600 of T_1 loses a fraction of a percent to decay before any control error is counted. Longer coherence and shorter gates are the two levers, and they pull against each other, since driving a qubit harder to make a gate faster also drives transitions it should not (chapter 8).

06

Cryogenics

The processor sits at the coldest point of a dilution refrigerator, at about 15 millikelvin: roughly 180 times colder than the 2.7 K of deep space.

6.1 How a dilution refrigerator cools

Mechanical coolers (pulse tubes) take the system from room temperature to about 4 K. Below that, cooling comes from a property of helium. A mixture of the two isotopes, helium-3 and helium-4, separates into two phases at low temperature, one rich in helium-3 and one dilute. Moving helium-3 atoms from the rich phase into the dilute one absorbs heat, much as evaporation does. The refrigerator circulates helium-3 continuously so that this happens without pause in the mixing chamber, where the processor is mounted.

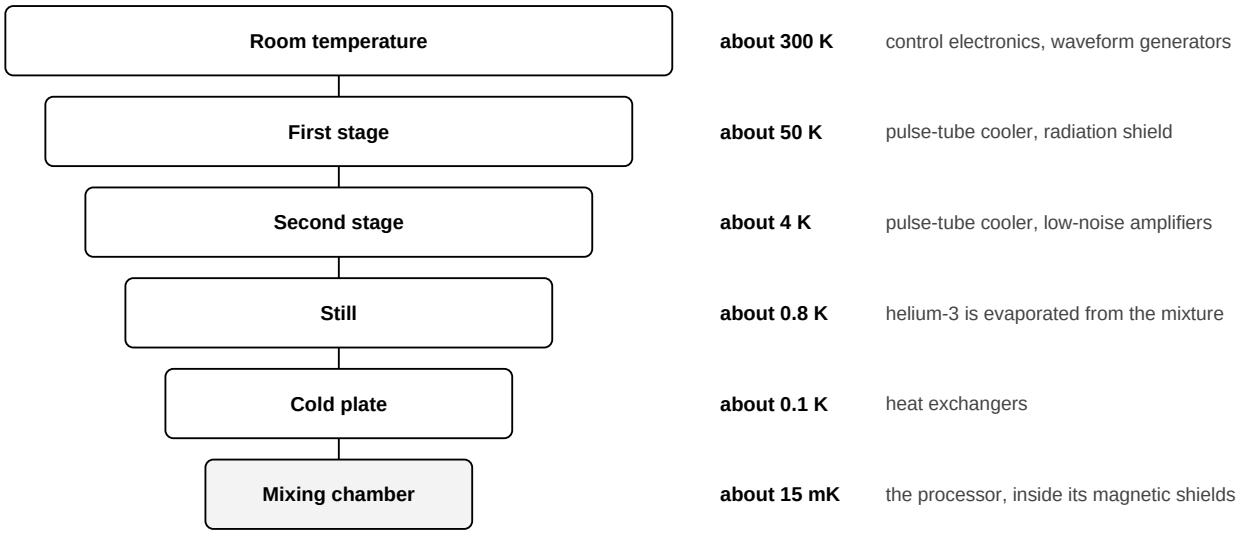

Figure 3. The stages of a dilution refrigerator. Stage temperatures other than the base are typical values for this class of refrigerator; the base temperature is the published figure.

6.2 Cooling power

The refrigerator is a Bluefors XLD400, which delivers 400 µW of cooling power at 100 mK. The number looks small and is the binding constraint of the whole system. Every control line that reaches the processor carries heat down with it, and every microwave pulse deposits a little more. The number of qubits a refrigerator can host is set less by space than by how much heat its wiring brings to the coldest stage.

6.3 Shielding

The processor is enclosed in Cryoperm and µ-metal magnetic shields. Stray magnetic fields shift qubit frequencies and can trap magnetic flux in the superconducting films, both of which shorten coherence. The two materials are high-permeability alloys that draw field lines into themselves and around the volume they enclose; Cryoperm is formulated to keep that property at cryogenic temperatures.

Base temperature	About 15 mK
Dilution refrigerator	Bluefors XLD400
Cooling power at 100 mK	400 µW
Magnetic shielding	Cryoperm + µ-metal

Table 6. Published cryogenic figures.

07

Microwave control

Qubits are controlled with shaped microwave pulses a few tens of nanoseconds long. The control electronics decide how precisely those shapes can be drawn.

7.1 Waveform generation

Arbitrary waveform generators sample at 2 GSa/s, which is one sample every 0.5 ns. A 25 ns single-qubit pulse is therefore drawn with 50 samples, and a 200 ns two-qubit pulse with 400. The generator shapes a pulse's envelope (its rise, its amplitude, its phase); that envelope modulates a carrier at the qubit's own frequency. Resolution matters because gate errors at the level of a few parts in ten thousand come from details of the pulse shape.

7.2 The frequency plan

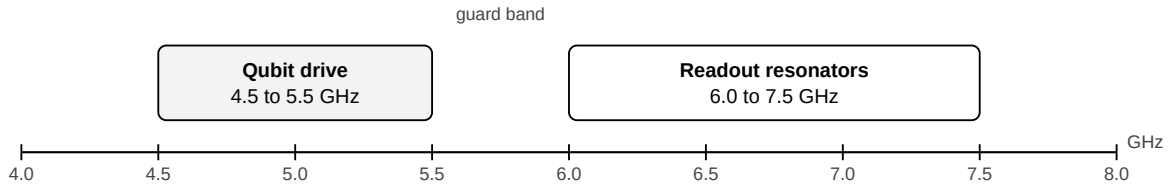

Figure 4. Qubit drive and readout bands.

Qubit transitions lie between 4.5 and 5.5 GHz. The readout resonators lie above them, between 6.0 and 7.5 GHz. Keeping the two bands apart lets a readout tone probe a resonator without driving the qubit attached to it, and lets drive pulses and readout signals share wiring while being separated cleanly by frequency.

7.3 The signal path

Pulses travel from room-temperature electronics down coaxial lines to the processor. Along the way the lines are attenuated at successive cold stages, so that the thermal noise of the warm end of a cable does not reach the qubits. Returning readout signals are tiny and are amplified on their way back up, first at the cold stages where an amplifier adds the least noise.

Waveform generator sample rate	2 GSa/s
Pulse resolution	0.5 ns
Qubit drive frequency	4.5 to 5.5 GHz, with 250 MHz anharmonicity
Readout resonators	6.0 to 7.5 GHz

Table 7. Published control-plane figures.

08

The native gate set

Every algorithm is compiled down to three physical operations: a phase shift done in software, one calibrated single-qubit pulse and one two-qubit interaction.

8.1 Virtual-Z: a gate that takes no time

A rotation of a qubit about its Z axis is the same as a shift in the phase reference of every pulse that follows. The control software therefore applies Z rotations by updating a number: no pulse is sent, no time passes and no error is added. This is why the physical single-qubit gate set can be so small.

8.2 The √X pulse and DRAG

The one physical single-qubit gate is √X, a quarter turn about the X axis, 25 ns long. Any single-qubit operation can be built from at most two √X pulses and three virtual-Z rotations.

A 25 ns pulse is short enough that its spectrum is tens of megahertz wide. The transmon's third level lies only 250 MHz from the transition being driven, so a plain pulse would push a little of the qubit's state out of the two levels that hold information. DRAG (derivative removal by adiabatic gate) corrects for this by adding a second pulse component, proportional to the derivative of the first and a quarter cycle out of phase with it, that cancels the leakage. It is what allows short pulses on a weakly anharmonic qubit, and it is behind the 99.94% single-qubit fidelity.

8.3 The cross-resonance gate

Two qubits are entangled with the cross-resonance effect. One qubit (the control) is driven with a microwave tone at the frequency of its neighbor (the target). Because the two are coupled, the target then rotates in a direction that depends on the state of the control. No tunable element is needed: the qubits stay at fixed frequencies, which is good for coherence.

The raw interaction comes with unwanted side effects. The echoed form of the gate (ECR) splits the drive into two halves of opposite sign with a flip of the control qubit between them, so that the wanted interaction adds and the unwanted terms cancel. One ECR gate takes 200 ns and reaches 99.4% fidelity. Together with single-qubit gates it is equivalent to the CNOT gate that circuits are usually written in.

Gate	Implementation	Duration	Fidelity
Z rotation	Virtual: a phase update in software	0	Exact
√X	DRAG-corrected microwave pulse	25 ns	99.94% (single-qubit)
ECR	Echoed cross-resonance	200 ns	99.4%
Measurement	Dispersive readout, Purcell filter		98.7%
Reset	Mid-circuit ancilla reset	200 ns or less	

Table 8. The native operations and their published figures.

8.4 How a fidelity is measured

Gate fidelities are usually measured by randomized benchmarking. Random sequences of gates of increasing length are applied, each ending with the one gate that should return the qubits to their starting state. The fraction of runs that do return falls off with sequence length, and the rate of that fall gives the average error per gate. Because the sequences are random, the result is insensitive to errors in preparing and measuring the qubits, which would otherwise be mistaken for gate error. Interleaving one particular gate among the random ones isolates the fidelity of that gate.

8.5 Reading a fidelity

A fidelity of 99.4% means an error of 0.6% per two-qubit gate. Errors compound: a circuit with 100 two-qubit gates succeeds without any error a little over half the time ($0.994^{100} \approx 0.55$). That is why uncorrected processors run shallow circuits, and why going further requires error correction (chapter 10).

09

Readout and reset

Measuring a qubit without touching it directly, and returning it to zero quickly, are what make repeated error-correction cycles possible.

9.1 Dispersive readout

Each qubit is coupled to its own microwave resonator, tuned well away from the qubit's frequency. At that detuning the two do not exchange energy; instead, the resonator's frequency shifts slightly depending on whether the qubit is in 0 or 1. A weak probe tone sent to the resonator comes back with a phase that reveals which. The qubit is measured by looking at something else.

9.2 The Purcell filter

A resonator that is strongly connected to the outside world gives a fast, clear measurement, but it also gives the qubit a path to lose its energy through that connection. A Purcell filter is a band-pass element between resonator and line that passes signals at the readout frequency and blocks them at the qubit frequency. It allows fast readout without paying for it in T_1. The published measurement fidelity is 98.7%.

9.3 How readout fidelity is measured

Readout fidelity is the simplest figure to define: prepare the qubit in 0 many times and count how often it reads 1, prepare it in 1 and count how often it reads 0, and average the two error rates. A measurement fidelity of 98.7% is an assignment error of 1.3%. Readout errors are larger than gate errors on most superconducting processors, partly because a measurement lasts long enough for the qubit to decay while it is being read.

9.4 Mid-circuit reset

Error correction measures the same helper qubits (ancillas) over and over. Waiting several T_1 for an ancilla to relax between cycles would take hundreds of microseconds. The processor resets an ancilla actively, in 200 ns or less, so that a full cycle fits in one microsecond.

10

Error correction

No physical qubit is good enough to run a long computation. Error correction spreads one logical qubit across many physical ones, and keeps checking them, so that errors are caught faster than they accumulate.

10.1 The surface code

In a surface code, data qubits sit on a grid. Between them are checks, each of which measures a joint property (the parity) of the two or four data qubits around it, through an ancilla qubit. There are two kinds: Z-type checks detect bit flips and X-type checks detect phase flips. A check reveals whether its neighbors agree, never what their values are, so the encoded information is not disturbed by the measuring.

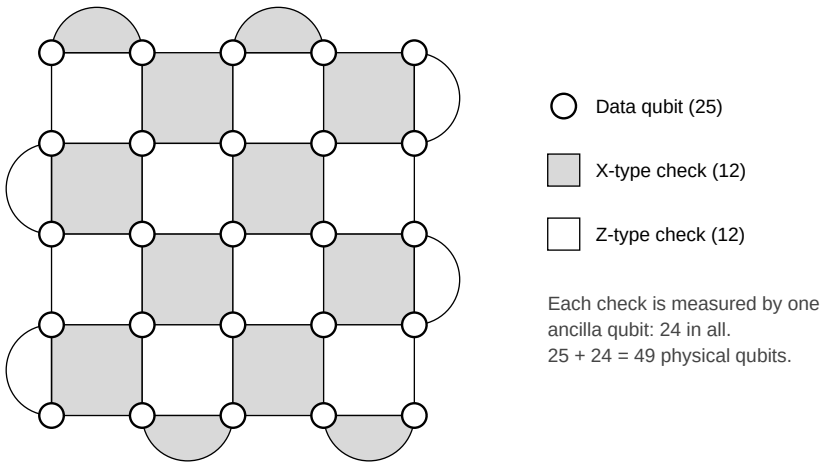

Figure 5. A distance-5 rotated surface code: 25 data qubits and 24 checks.

When a data qubit suffers an error, the checks beside it change their answer on the next cycle. The pattern of changed checks is the syndrome. A classical program, the decoder, reads the syndromes cycle after cycle and works out the most likely set of errors that explains them, so that they can be undone or simply tracked.

10.2 One cycle, step by step

1. **Reset.** Every ancilla is returned to 0.
2. **Entangle.** Each ancilla interacts in turn with the data qubits around it, through two-qubit gates applied in a fixed order across the whole grid.
3. **Measure.** The ancillas are read out. Each result is the parity of that check's data qubits.
4. **Compare.** The decoder compares each result with the same check's result in the previous cycle. A change marks the end of an error chain.
5. **Repeat.** The data qubits are never measured directly until the computation ends.

Comparing cycles, and not single results, is what makes the scheme tolerant of faulty measurements: a wrong readout appears as a change that reverses one cycle later, and the decoder treats it as one more kind of error to be explained.

10.3 Distance

The distance, d, is the smallest number of physical errors that can change the encoded information without being noticed. A distance-5 code can correct any two errors in a cycle. The rotated layout uses d × d data qubits and one fewer ancilla: 25 + 24 = 49 physical qubits for each logical qubit.

10.4 The threshold

Error correction only helps if the physical error rate is below a threshold. Above it, the extra qubits and gates add more errors than the code removes. Below it, each increase in distance suppresses the logical error rate by a further factor, roughly as the ratio of physical error to threshold raised to the power (d + 1) / 2. The specification puts the threshold at 0.95% two-qubit gate error and the processor's two-qubit error at 0.6%, an operating margin of 0.3% below threshold.

Code family	Surface code, d=5 rotated planar
Physical qubits per logical qubit	49
Syndrome cycle	1 µs measurement, with mid-circuit ancilla reset in 200 ns or less
Threshold	0.95% two-qubit gate error; operating margin 0.3% below threshold
Logical error rate	About 1×10^{-3} per cycle on the d=5 code

Table 9. The published error-correction specification.

> **Scope.** A distance-5 logical qubit needs 49 physical qubits, more than a single 16-qubit processor provides. The figures in this chapter specify the code, its cycle and what it asks of the gates, readout and reset; they are not a count of logical qubits on the 16-qubit core.

11

Software and access

A quantum processor is used through a classical software stack that turns a circuit into pulses and pulses back into numbers.

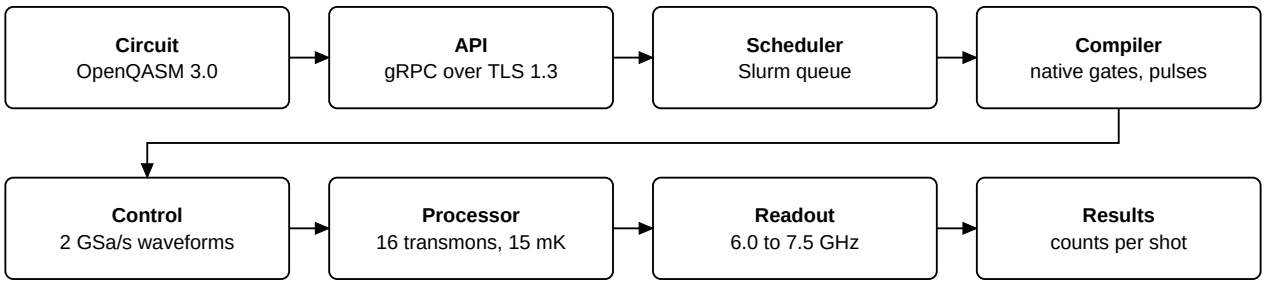

Figure 6. The path of a job, from circuit to results.

11.1 Circuits

Circuits are written in OpenQASM 3.0, an open assembly language for quantum programs. It expresses gates and measurements, and also the classical control flow and timing that error-correction experiments need. A Bell-pair circuit, the smallest demonstration of entanglement, looks like this:

```
OPENQASM 3.0;
include "stdgates.inc";

qubit[2] q;
bit[2] c;

h q[0];            // put the first qubit in superposition
cx q[0], q[1];     // entangle the second with it
c = measure q;     // both bits now always agree
```

11.2 Compilation

The compiler rewrites a circuit in the native gate set. A Hadamard gate becomes a √X pulse between two virtual-Z rotations. A CNOT becomes one ECR gate dressed with single-qubit gates. Operations between qubits that are not neighbors on the heavy-hex lattice are routed through intermediate qubits. The result is a schedule of pulses with exact start times.

11.3 Submission and scheduling

Client SDKs are available for Python, TypeScript and Rust. Jobs are submitted over gRPC, protected with TLS 1.3, and queued by a Slurm scheduler, the same scheduler widely used on classical high-performance clusters, which lets quantum jobs be queued, prioritized and accounted for in a familiar way. A job runs its circuit many times (shots) and returns the count of each measured outcome.

Client SDKs	Python / TypeScript / Rust
Circuit format	OpenQASM 3.0
Transport	gRPC + TLS 1.3
Job scheduler	Slurm on QPU

Table 10. Published software figures.

12

The post-quantum security stack

Post-quantum cryptography runs on ordinary computers and resists attack by quantum ones. It is the part of quantum readiness that can, and should, be deployed today.

12.1 The two standards

In August 2024 NIST published its first post-quantum standards. MUTOWA's stack uses the strongest parameter set of each.

Role	Algorithm	Standard	Public key	Ciphertext or signature
Key encapsulation	Kyber-1024 (ML-KEM-1024)	FIPS 203	1,568 bytes	1,568 bytes
Signature	Dilithium-5 (ML-DSA-87)	FIPS 204	2,592 bytes	4,627 bytes

Table 11. The post-quantum primitives and their sizes, from the public standards.

Key encapsulation lets two parties agree on a secret key over an open channel; it replaces the Diffie-Hellman and RSA key exchanges. **Signatures** prove who sent a message and that it was not altered; they replace RSA and elliptic-curve signatures. Both algorithms rest on lattice problems for which no efficient quantum algorithm is known, and both are specified at NIST security category 5, the highest: at least as hard to break as AES-256.

12.2 Hybrid transport

Transport is hybrid: a classical X25519 exchange and a Kyber-1024 encapsulation run together, and the session key depends on both. The reasoning is defense in depth. Lattice cryptography has been studied intensely but for fewer years than elliptic curves. With a hybrid, traffic stays protected unless both are broken, so it is at least as safe as today's connections against today's attackers and designed to stay safe against a quantum one.

12.3 Crypto agility

Algorithms will change again. A system is crypto-agile when its algorithms are named in configuration and negotiated between peers, so that replacing one is an update and not a redesign. Agility is the property that makes the next migration cheaper than this one.

PQC KEM	CRYSTALS-Kyber-1024 (NIST FIPS 203) for key encapsulation
PQC signature	CRYSTALS-Dilithium-5 (NIST FIPS 204) for authentication
Crypto agility	Hybrid X25519 + Kyber-1024 transport, for defense in depth against harvest-now, decrypt-later attacks
NIST level	Category 5

Table 12. The published security stack.

13

Quantum key distribution

Quantum key distribution uses the physics of single photons, not a mathematical problem, to let two parties detect whether a key was observed in transit.

13.1 BB84

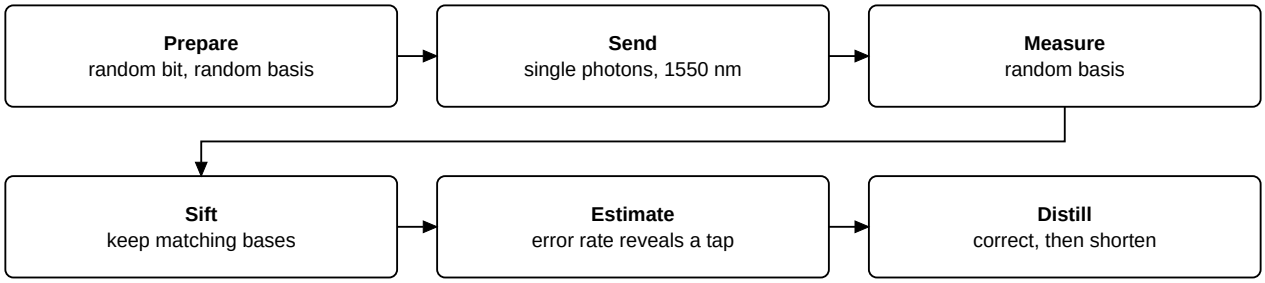

Figure 7. The steps of BB84 prepare-and-measure key distribution.

1. **Prepare.** The sender encodes each random bit on a single photon, choosing at random between two incompatible ways of encoding (two bases).
2. **Measure.** The receiver measures each photon in a basis of their own random choosing.
3. **Sift.** Over an ordinary channel the two compare which bases they used, not the results, and keep only the bits where the bases matched.
4. **Estimate.** They compare a sample of the kept bits. Measuring a photon in the wrong basis disturbs it, so an eavesdropper necessarily introduces errors. An error rate above a known bound means the key is discarded.
5. **Distill.** Remaining errors are corrected, and the key is shortened by an amount that removes whatever an eavesdropper could have learned.

13.2 The link

MUTOWA's QKD is BB84 prepare-and-measure over 1550 nm optical fiber, within a loss budget of under 0.1 dB/km. 1550 nm is the wavelength at which optical fiber absorbs the least light. Loss is what limits QKD, because a lost photon cannot be amplified without measuring it: every 10 dB of loss means only one photon in ten arrives, and the key rate falls with it.

13.3 Where QKD fits

QKD and post-quantum cryptography answer the same threat in different ways and are used together. QKD needs an authenticated classical channel for its sifting step, or an attacker could simply sit in the middle; that authentication is provided by post-quantum signatures. And QKD needs a dedicated optical link, so it suits a small number of high-value fixed routes, while post-quantum cryptography protects everything else.

Protocol	BB84 prepare-and-measure
Medium	1550 nm optical fiber
Loss budget	Under 0.1 dB/km
Authentication	Dilithium-5 (FIPS 204)

Table 13. Published key-distribution figures.

14

Hybrid architectures

A quantum processor is never used alone. It is one accelerator in a system that is mostly classical, and the engineering is in the joins.

14.1 Three kinds of resource

Resource	What it does well	Where it sits
Classical compute	Orchestration, compilation, decoding, almost all of every workload	Data center and edge
Edge compute	Low-latency inference and routing near the user	MUTOWA Private Edge Compute nodes
Quantum compute	Specific subroutines: simulation, sampling, and research into error correction	The QPU, in its cryostat

Table 14. The parts of a hybrid system.

MUTOWA builds hybrid interfaces that connect these without compromising operational continuity: classical orchestration, edge inference and routing, and quantum acceleration composed behind governed APIs. A caller submits work to one interface; where each part runs is the system's decision.

14.2 A shared security mesh

The quantum stack shares its post-quantum cryptography with the Private Edge Compute network, whose nodes link to one another with Kyber-1024 key encapsulation and keys rotated every 60 seconds. A result leaving the quantum system travels the same protected mesh as any other traffic. The network is described in the companion paper, *Private Edge Compute*.

14.3 The Quantum node

The Quantum tier of the Sunlight node program adds a small QPU allocation for quantum compute workloads to everything in the Compute tier, at $2,500 USD per license. A Sunlight node is hosted by MUTOWA or shipped to its holder, as the holder chooses; its QPU allocation is time on the processor this paper describes either way. Compensation depends on real usage of the streams a tier includes and on node uptime; earnings vary and are not guaranteed.

15

Algorithms and use cases

The honest question about any quantum use case is not whether a quantum computer could help one day, but what would have to be true for it to help, and when.

15.1 What a small processor runs

- **Characterization.** Randomized benchmarking, coherence measurements and crosstalk maps: the experiments behind every figure in this paper.
- **Entanglement demonstrations.** Bell pairs and larger entangled states, checked against exact prediction.
- **Variational algorithms.** Short circuits tuned by a classical optimizer, for small chemistry and optimization problems.
- **Error-correction primitives.** Repeated parity checks, mid-circuit measurement and reset, and real-time decoding.

15.2 Sorting workloads

MUTOWA's evaluation method sorts an organization's workloads into three groups, each answered with measurements and not narratives.

Group	Test	Action
Could benefit from quantum acceleration	Is there a known quantum algorithm with a real advantage, and how many error-corrected qubits would it need?	Prototype at small scale; track hardware against the requirement
Needs post-quantum protection	Does it rely on public-key cryptography, and must its data stay secret for years?	Migrate now (chapter 16)
Stays classical	Everything else, which is most things	No quantum work; revisit when the first two change

Table 15. Sorting workloads by what quantum means for them.

15.3 Kill criteria

A prototype is useful only if it can fail. Each one starts with a hypothesis, a success threshold and a condition under which the approach is dropped. Hardware and algorithms are compared on equal footing, and results are reported as measurements: fidelity, coherence, and cost.

16

Transition planning

Moving to post-quantum cryptography is mostly an inventory problem. The cryptography is standardized; finding everywhere the old cryptography lives is the work.

16.1 Assess

Stage one maps cryptographic exposure, computational bottlenecks and mission-critical data dependencies. Its output is an inventory with, for every system:

Field	Question it answers
Where cryptography is used	Protocols, libraries, certificates, hardware modules, vendor products
Which algorithms	Key exchange, signatures, symmetric ciphers, hashes, and their key sizes
Data lifetime	How long must what this system protects stay secret?
Exposure	Does the data cross networks where it could be recorded?
Owner and upgrade path	Who can change it, and can it be changed in place?

Table 16. Fields of a cryptographic inventory.

Ranking the inventory by data lifetime and exposure gives a migration order: long-lived secrets that travel over public networks come first.

16.2 Prototype

Stage two tests the migration in controlled, measurable environments. Post-quantum keys and signatures are larger than the ones they replace (chapter 12), which affects handshake sizes, certificate chains, constrained devices and anything with a fixed message format. A prototype finds those limits before production does.

16.3 Integrate

Stage three creates the hybrid interfaces and governance for production: post-quantum cryptography by default, hybrid transport as the bridge, crypto agility so the next change is cheaper, and operating procedures that evolve as the technology matures.

1 Inventory every cryptographic dependency and data lifetime.

2 Assess exposure to harvest-now, decrypt-later collection.

3 Migrate key exchange first, to NIST FIPS 203, with hybrid transport.

4 Migrate signatures to NIST FIPS 204 as certificates and firmware-signing chains are renewed.

5 Govern the transition with measurable, staged milestones.

17

The quantum coin flip

The simplest quantum program prepares one qubit in an equal superposition and measures it. The result is 0 or 1, each exactly half the time.

Starting from 0, a single Hadamard gate puts the qubit in the state written |+>: an equal superposition of 0 and 1. Measurement forces an outcome. Quantum mechanics gives the probability of each outcome as the square of its amplitude, which here is one half for each. The outcome is not hidden in the qubit beforehand and is not produced by any formula: it is random in the strongest sense physics knows.

```
OPENQASM 3.0;
include "stdgates.inc";

qubit q;
bit c;

h q;               // equal superposition of 0 and 1
c = measure q;     // 0 or 1, each with probability one half
```

On the native gate set the Hadamard is one 25 ns √X pulse between two virtual-Z rotations. On real hardware the split is 50/50 to within the gate and measurement fidelities.

The Quantum page at mutowa.com/ventures/quantum has a coin-flip demonstration with a running tally of results. It is a demonstration: the visitor's browser's cryptographic random generator stands in for measuring a real qubit.

18

Limits and assumptions

What the figures in this paper cover, and what they do not.

- **The processor is a research system.** Sixteen qubits can be simulated exactly on a classical computer. The processor's purpose is developing and measuring the hardware, control and error-correction methods that larger systems depend on.
- **Error-correction figures specify a code.** A distance-5 logical qubit uses 49 physical qubits. The published cycle, threshold and logical error figures describe that code and its requirements (chapter 10).
- **Fidelities and coherence times vary.** On any superconducting processor, individual qubits and pairs differ and figures drift between calibrations. The figures here are the processor's published specifications.
- **Post-quantum means designed to resist.** ML-KEM and ML-DSA are public standards that have withstood years of open analysis. No cryptography is proven unbreakable, which is why transport is hybrid.
- **QKD protects a link, not an endpoint.** It needs authentication and dedicated fiber, and its range is limited by loss.
- **No date is given for cryptographically relevant quantum computers.** Nobody knows it. Transition planning is justified by the lifetime of data, not by a forecast.
- **Stage temperatures in Figure 3** other than the base are typical values for a dilution refrigerator, shown for explanation.

- **Specifications are subject to change** as research systems evolve.

19

Glossary

Term	Meaning
Ancilla	A helper qubit used to measure a check without disturbing the data
Anharmonicity	The difference between a transmon's first and second transition frequencies
BB84	The first quantum key distribution protocol (Bennett and Brassard, 1984)
Coherence time	How long a qubit keeps its state: T_1 for energy, T_2 for phase
Cross-resonance	Driving one qubit at its neighbor's frequency to entangle the two
Decoder	Classical software that infers errors from syndrome measurements
Dilution refrigerator	A cryostat that cools to millikelvin by mixing helium-3 into helium-4
Distance (d)	The smallest number of errors a code cannot detect
DRAG	A pulse-shaping technique that suppresses leakage out of the qubit levels
ECR	Echoed cross-resonance: the native two-qubit gate
Fidelity	How closely an operation matches the ideal one; one minus the error
Heavy-hex	A lattice in which every qubit has two or three neighbors
Logical qubit	One protected qubit encoded in many physical qubits
ML-KEM / ML-DSA	The NIST post-quantum standards for key encapsulation and signatures (FIPS 203 / 204)
OpenQASM	An open assembly language for quantum circuits
Purcell filter	A filter that stops a qubit losing energy through its readout line
QKD	Quantum key distribution
QPU	Quantum processing unit
Syndrome	The pattern of check results that signals an error
Transmon	A superconducting qubit made from a Josephson junction and a large capacitor
Virtual-Z	A Z rotation applied by shifting the phase of later pulses

20

References and about MUTOWA

20.1 Standards and further reading

1 J. Koch et al. *Charge-insensitive qubit design derived from the Cooper pair box.* Physical Review A 76, 042319, 2007.

2 P. Krantz et al. *A quantum engineer's guide to superconducting qubits.* Applied Physics Reviews 6, 021318, 2019.

3 F. Motzoi et al. *Simple pulses for elimination of leakage in weakly nonlinear qubits.* Physical Review Letters 103, 110501, 2009.

4 D. C. McKay et al. *Efficient Z gates for quantum computing.* Physical Review A 96, 022330, 2017.

5 S. Sheldon et al. *Procedure for systematically tuning up cross-talk in the cross-resonance gate.* Physical Review A 93, 060302, 2016.

6 C. Chamberland et al. *Topological and subsystem codes on low-degree graphs with flag qubits.* Physical Review X 10, 011022, 2020.

7 A. G. Fowler et al. *Surface codes: Towards practical large-scale quantum computation.* Physical Review A 86, 032324, 2012.

8 A. Cross et al. *OpenQASM 3: A broader and deeper quantum assembly language.* ACM Transactions on Quantum Computing 3, 2022.

9 P. W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.* SIAM Journal on Computing 26, 1997.

10 M. Mosca. *Cybersecurity in an Era with Quantum Computers: Will We Be Ready?* IEEE Security & Privacy 16, 2018.

11 C. H. Bennett, G. Brassard. *Quantum cryptography: Public key distribution and coin tossing.* Proceedings of IEEE ICCSSP, 1984.

12 NIST. *FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard* and *FIPS 204: Module-Lattice-Based Digital Signature Standard.* August 2024.

13 MUTOWA. *Private Edge Compute* and *Private Quantum-Edge Telephony.* Whitepapers, November 2026. mutowa.com/press.

20.2 About MUTOWA

Rooted in the philosophy of **無永遠** (MUTOWA), the Japanese concept of boundless permanence, MUTOWA builds enduring, fault-tolerant infrastructure for the modern world. MUTOWA develops a connected ecosystem of edge intelligence, applied AI and quantum-ready systems designed for the long horizon.

MUTOWA is led by its co-founders, Nicolai Kazuyuki Diaz Tsuzuki (Co-Founder and President) and Parker Czuba (Co-Founder and Chief Technology Director), from Winnipeg, Canada and Tokyo, Japan.

20.3 Next steps

- See the processor and its figures: mutowa.com/ventures/quantum
- Plan a migration: mutowa.com/contact
- Write to MUTOWA: hello@mutowa.net

© 2026 MUTOWA. Specifications are subject to change as research systems evolve.