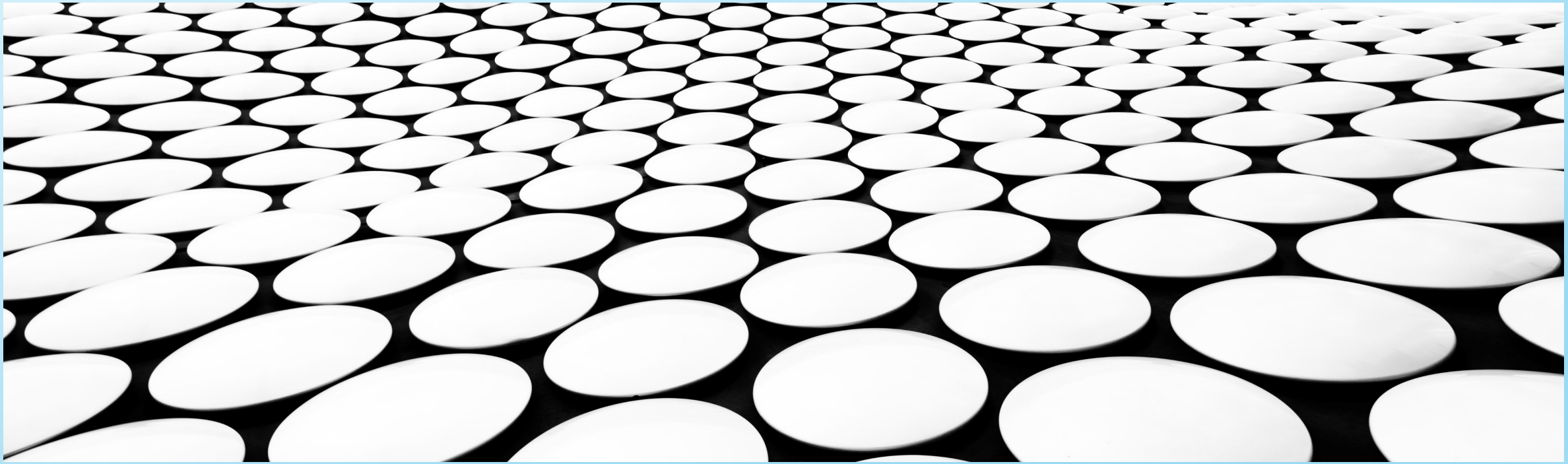
ARHITECTURA SISTEMELOR DE CALCUL

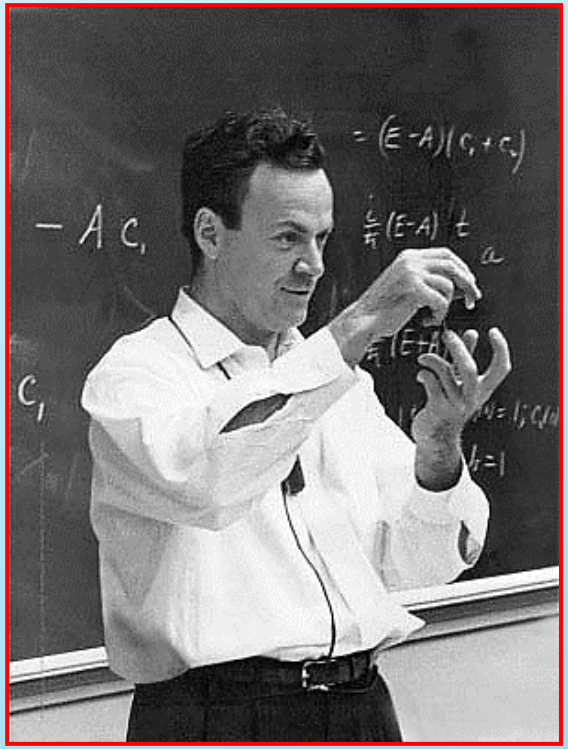
UB, FMI, CTI, ANUL III, 2022-2023



CALCULATOARE CUANTICE

Partea II

ISTORIA CALCULATOARELOR CUANTICE



- 1982 - Feynman a propus ideea de a crea mașini bazate pe legile mecanicii cuantice în loc de legile fizicii clasice.

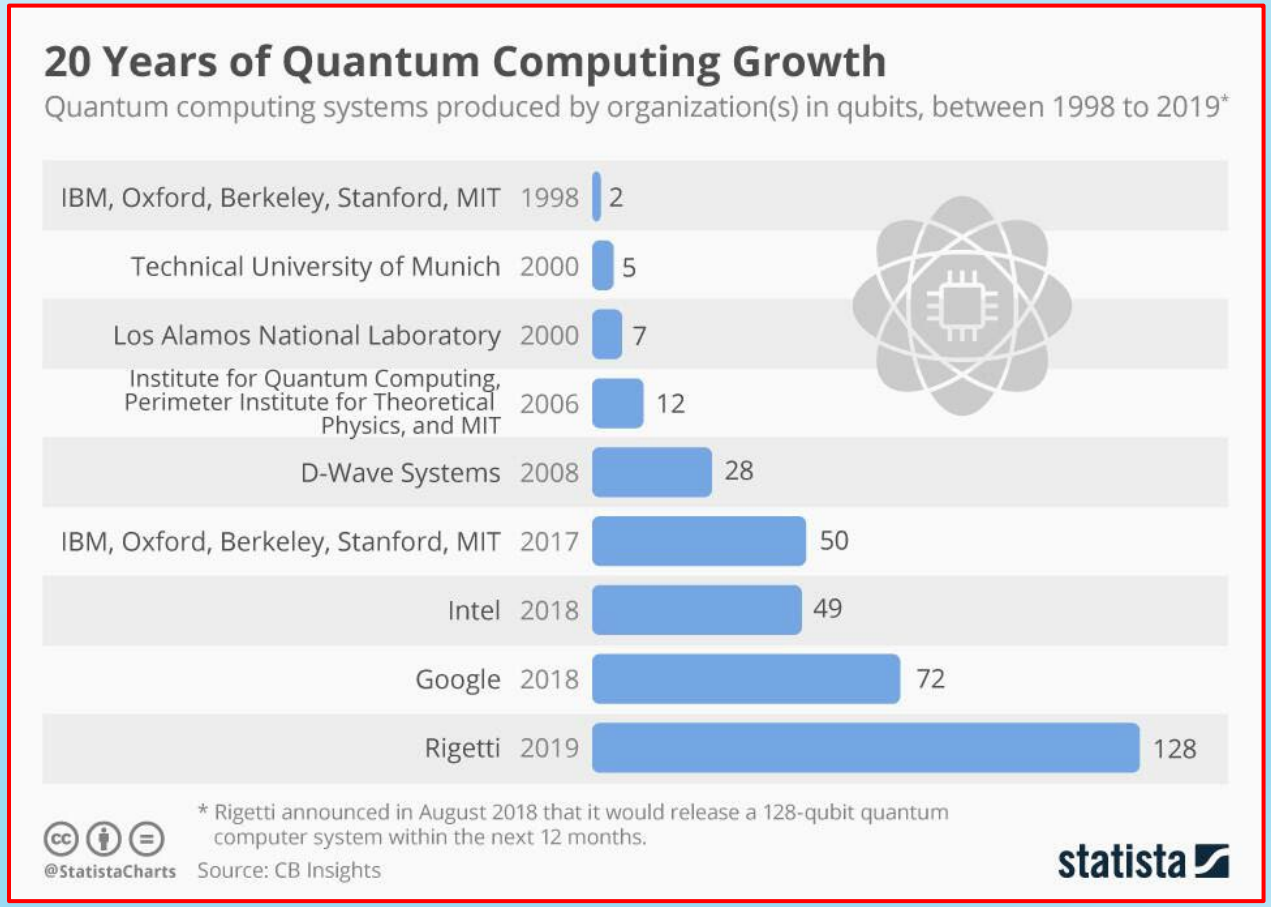
- 1985 - David Deutsch a dezvoltat masina Turing cuantica, arătând că circuitele cuantice sunt universale.
- 1994 - Peter Shor a venit cu un algoritm cuantic pentru a factoriza numere foarte mari.
- 1997 - Lov Grover dezvolta un algoritm cuantic de cautare cu o complexitate de $O(\sqrt{N})$

- **1973** - Alexander Holevo publishes paper showing that n qubits cannot carry more than n classical bits of information.
- 1976 - Polish mathematical physicist Roman Ingarden shows that Shannon information theory cannot directly be generalized to the quantum case.
- 1981 - *Richard Feynman determines that it is impossible to efficiently simulate a evolution of a quantum system on a classical computer*.
- **1985 - David Deutsch of the University of Oxford, describes the first universal quantum computer.**
- 1993 - Dan Simon, at Universite de Montreal, invents an oracle problem for which quantum computer would be exponentially faster than conventional computer. This algorithm introduced the main ideas which were then developed in Peter Shor's factoring algorithm.
- 1994 - *Peter Shor*, at AT&T's Bell Labs discovers *algorithm* to allow quantum computers to *factor large integers quickly*. Shor's algorithm could theoretically break many of the cryptosystems in use today.
- **1995** - Shor proposs the first scheme for quantum error correction.
- 1996 - Lov Grover, at Bell Labs, invents quantum database search algorithm.
- **1997 - David Cory, A.F. Fahmy, Timothy Havel, Neil Gershenfeld and Isaac Chuang publish the first papers on quantum computers based on bulk spin resonance, or thermal ensembles. Computers are actually a single, small molecule, storing qubits in the spin of protons and neutrons. Trillions of trillions of these can float in a cup of water.**
- **1998** - First working 2-qubit NMR computer demonstrated at University of California, Berkeley.
- 1999 - First working 3-qubit NMR computer demonstrated at IBM's Almaden Research Center. First execution of Grover's algorithm.
- 2000 - First working 5-qubit NMR computer demonstrated at IBM's Almaden Research Center.

➢ 2001 - First working 7-qubit NMR computer demonstrated at IBM's Almaden Research Center.
First execution of Shor's algorithm. The number 15 was factored using 1018 identical molecules, each containing 7 atoms.

2015 Cambridge Quantum Computing releases tket, which could be the first operating system for quantum computing. This enables classical computers interface to quantum computers.

- Oct. 2019. Google claims to have achieved quantum supremacy with a 53-qubit programmable superconducting processor. Named Sycamore, it's based on quantum logic gates. The machine completes a benchmark test in 200 seconds, what a classical supercomputer would take 10,000 years.
- However, IBM claims a supercomputer with more disk storage could solve the problem in 2.5 days.
- In August 2020, 12 qubits of Sycamore are used to simulate a chemical reaction.
- Mar. 2020 Honeywell reports on a demonstration of a quantum computer architecture based on trapped-ion QCCD (Quantum Charge-Coupled Device).



Simulatoare:
https://algassert.com/quirk

https://quantum-circuit.com/

https://qiskit.org/

Documentatie:
https://quantum-computing.ibm.com/docs/

Joc:
https://quantumgame.io/

Limbaje de programare:
QCL (http://tph.tuwien.ac.at/~oemer/qcl.html)

Arhitectura calculatoarelor cuantice

O arhitectură a unui sistem este un model teoretic care arată structura, comportamentul și componentele sistemului. Prin ea sunt descompuse comportamentele complexe ale sistemelor într-un set gestionabil de operațiuni.

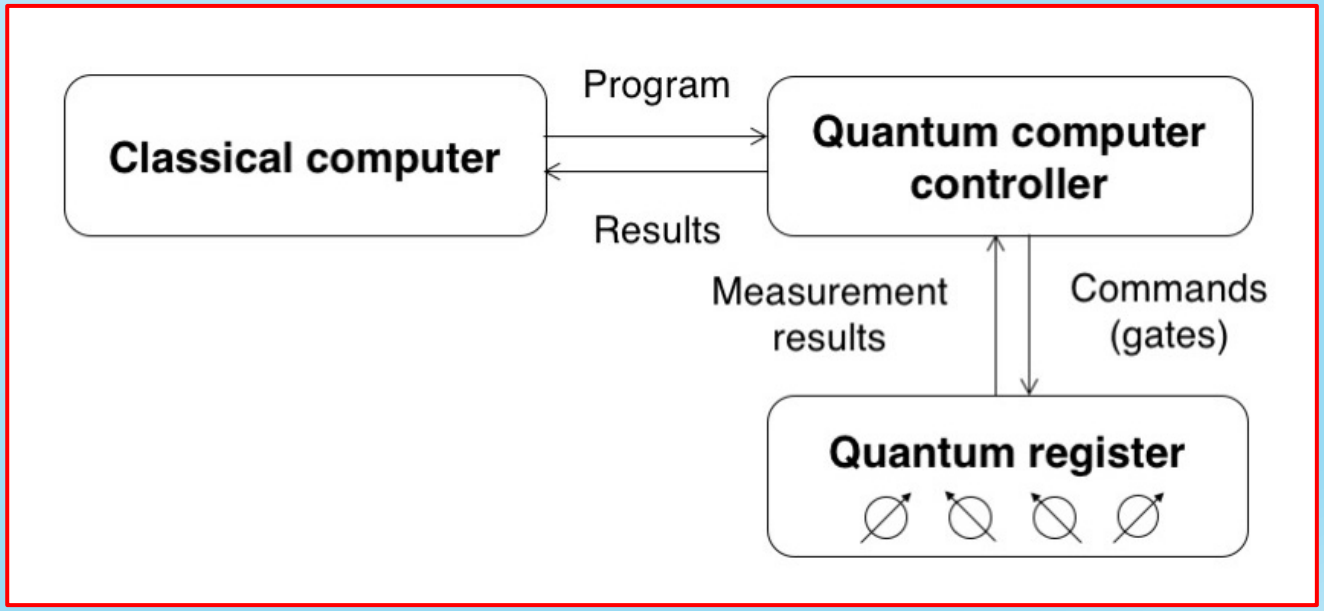
Un computer cuantic are atât părți clasice, cât și cuantice.

Modelul de bază al calculatoarelor cuantice

Atât computerul clasic, cât și cel cuantic constau în esență din trei părți :

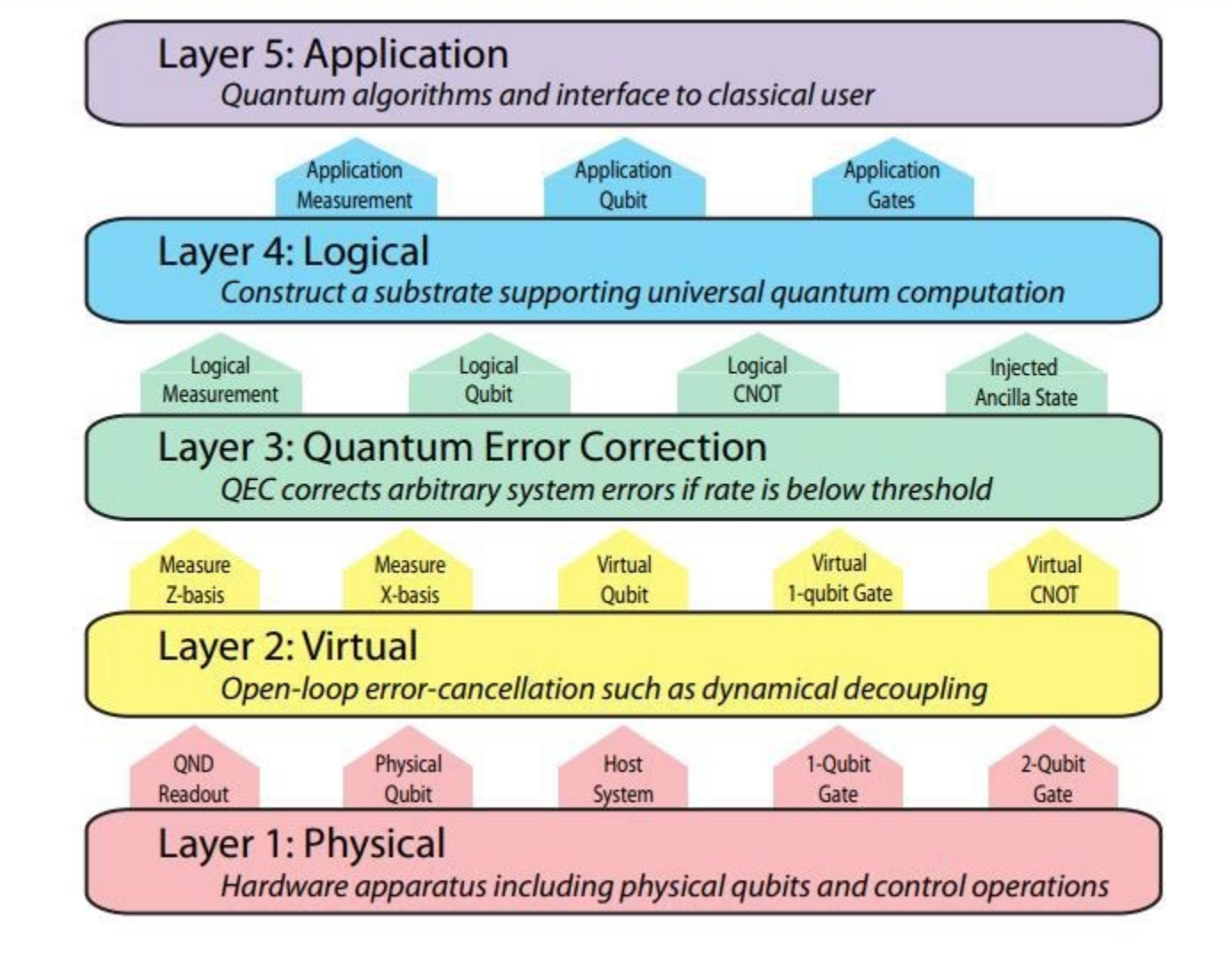
- **Memoria** - Conține/stocheaza starea curentă a mașinii.
- **Un procesor sau un controler** - Efectuează operații elementare asupra stării mașinii.
- **Dispozitiv de intrare/ieșire** - Face posibilă definirea stării inițiale și obținerea stării finale de calcul.

- **Registrii cuantici** sunt memoria calculatoarelor cuantice. Dețin date cuantice pentru algoritm.
- **Porțile cuantice** sunt echivalentul instructiunilor.
- **Controler de calculator** deține programul și le spune dispozitivelor care controlează fiecare qubit să efectueze acțiuni conform instrucțiunilor.



Arhitectură stratificată

- Arhitectura computerului cuantic constă din cinci straturi, în care fiecare strat are propriul set de sarcini sau funcții.
- Pentru a executa o operație, un strat trebuie să primească comanda sau instrucțiunile de la stratul de mai jos și să le proceseze în consecință.

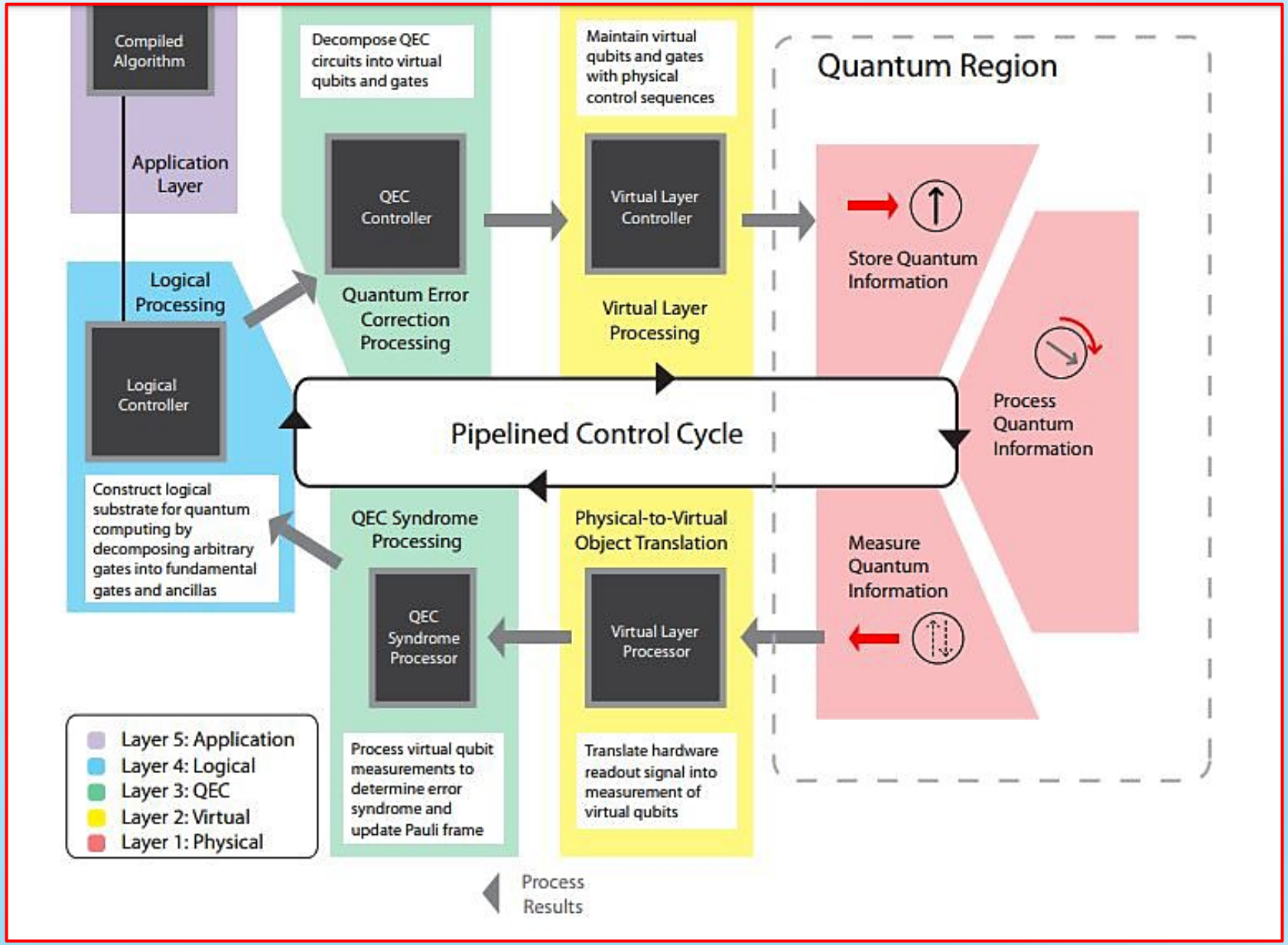


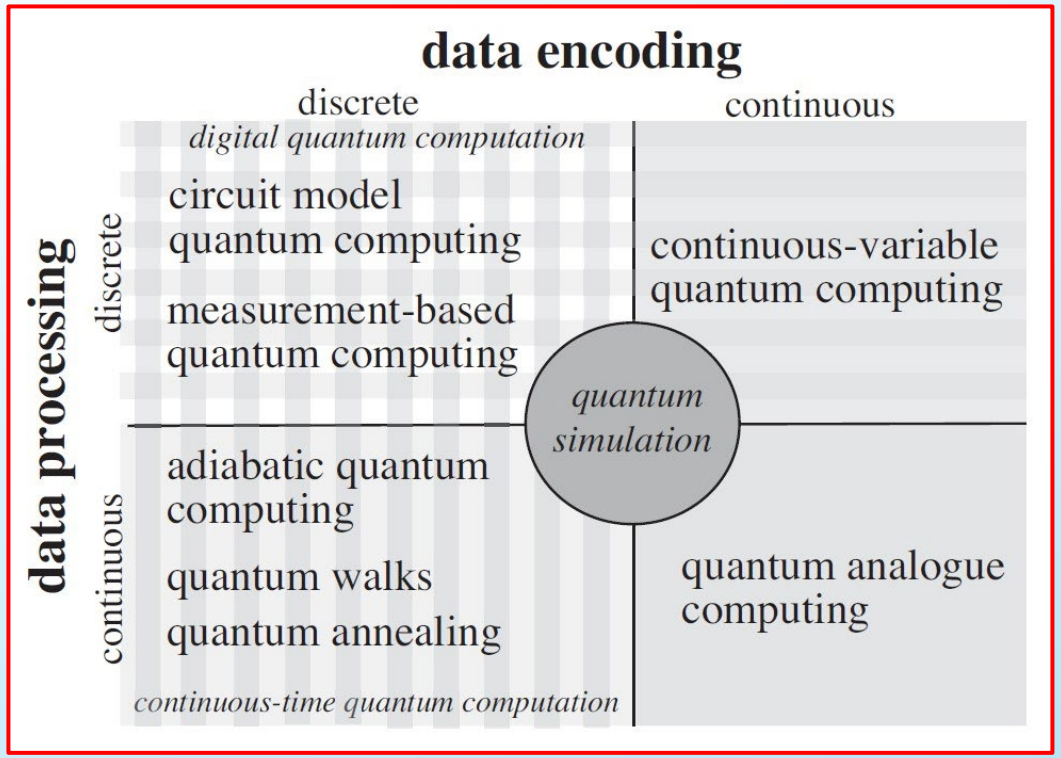
Ciclul de control primar definește comportamentul dinamic al computerului cuantic în această arhitectură, deoarece toate operațiunile trebuie să interacționeze cu această buclă.

Scopul principal al ciclului de control este implementarea cu succes a **corectării erorilor cuantice**.

Calculatorul cuantic trebuie să funcționeze suficient de rapid pentru a corecta erorile; totuși, unele operațiuni de control implică în mod necesar întârzieri, așa că acest ciclu nu emite pur și simplu o singură comandă și așteaptă rezultatul înainte de a continua - conducta este esențială.

Straturile 1 până la 4 interacționează în buclă, în timp ce stratul Aplicație interacționează numai cu stratul logic, deoarece este agnostic cu privire la designul de bază al computerului cuantic.





Analog quantum computers (including quantum annealer, adiabatic quantum computers, and direct quantum simulation).

These systems operate using coherent manipulation of the qubits, changing the analog values of the Hamiltonian.

It does not use quantum gates.

Fully error-corrected gate-based quantum computers.

Like NISQs, these are gate-based systems that operate on qubits.

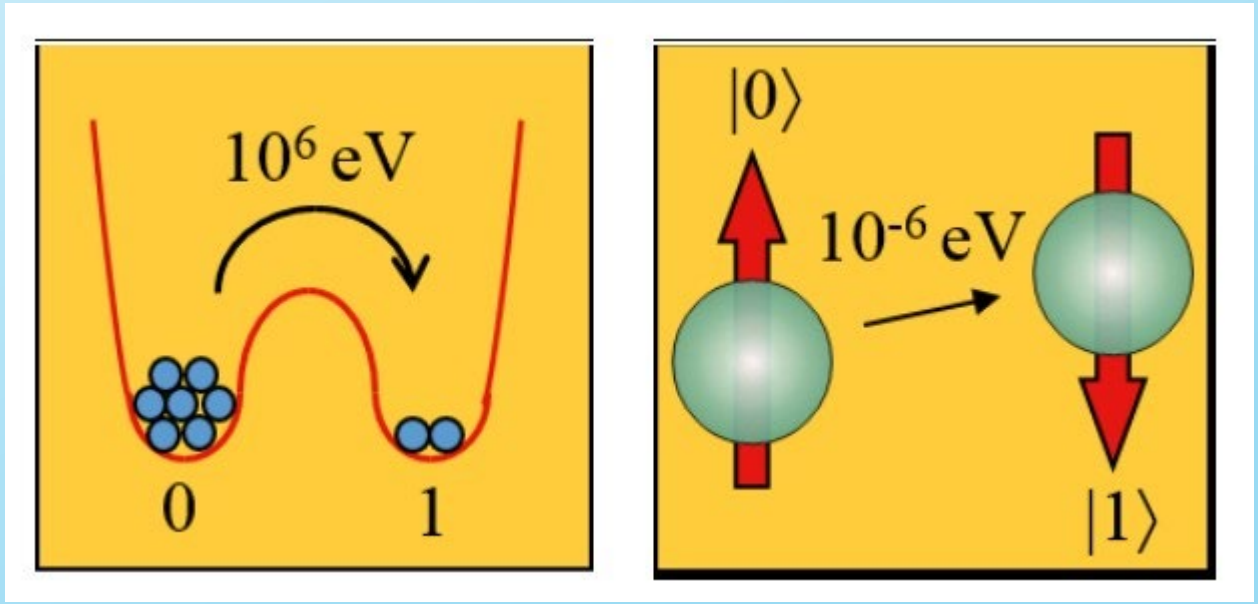
Still, these systems are complex and implement quantum error correction to eliminate the negative effects of system noise (including errors introduced by imperfect control signals or device fabrication or unintended coupling of qubits to each other or the environment).

These systems enable reductions in error probability rates sufficient that the computer is reliable for all computations.

Fully error-corrected gate-based quantum computers are expected to scale to thousands of logical qubits, enabling massive computational capabilities.

Decoerența cuantică este o sursă majoră de erori atunci când se lucreaza cu computere cuantice. Orice interacțiune a qubiților cu mediul exterior în moduri care perturbă comportamentul lor cuantic va duce la decoerență.

Calculatoarele cuantice de astăzi pot funcționa doar pentru perioade scurte de timp (adesea mai puțin de o secundă, și chiar și design-urile bune funcționează doar câteva secunde) înainte ca decoerența să interfereze cu funcționarea lor.



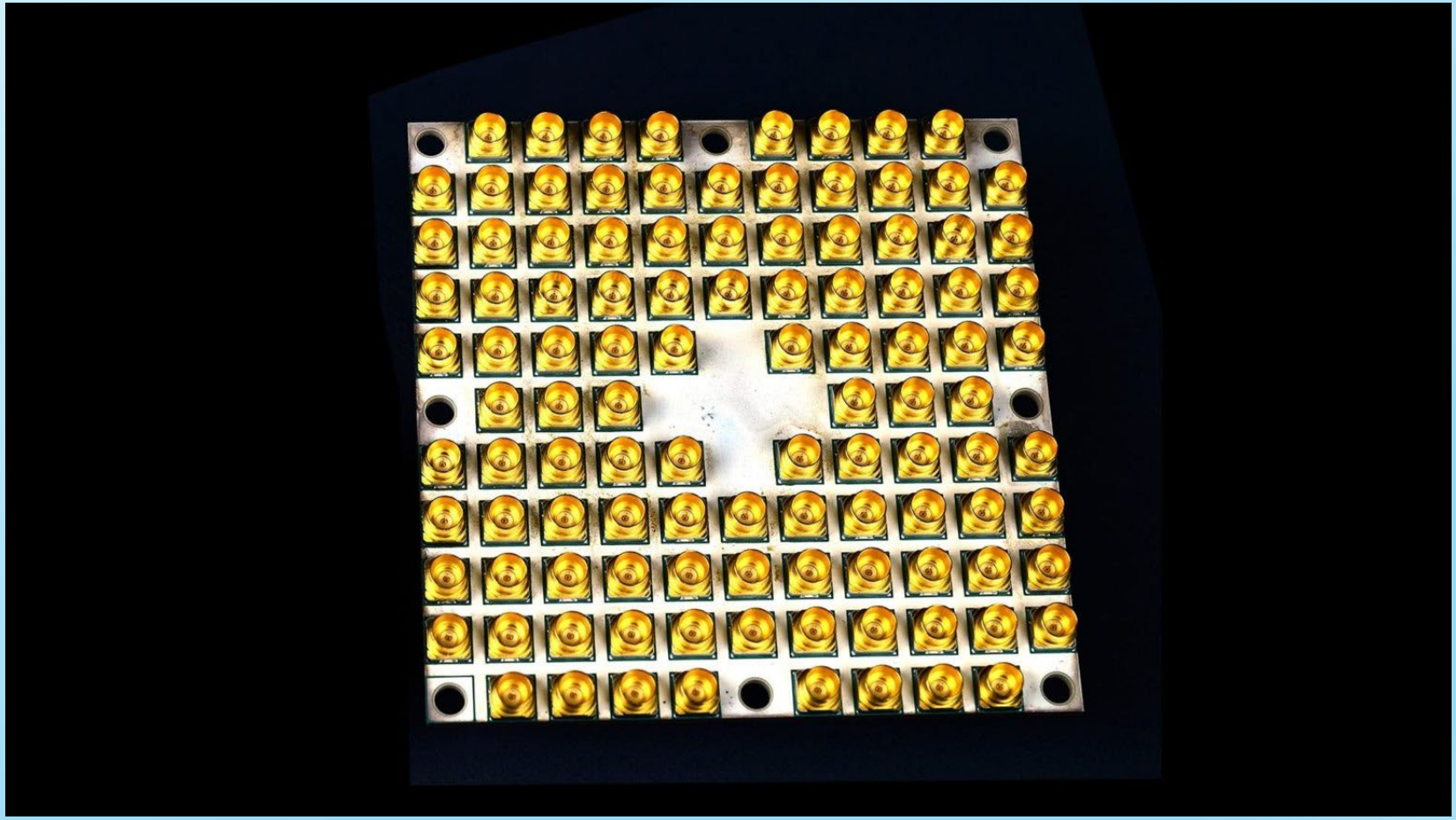
Calculatoarele cuantice bazate pe porți pot avea diverse implementari fizice.
Cu toate acestea, orice realizare trebuie să îndeplinească criteriile DiVincenzo:

1. Un sistem fizic scalabil cu qubiți bine caracterizați
2. Abilitatea de a inițializa starea qubiților într-o stare cuantică simplă care poate fi reprodusă în mod fiabil cu variabilitate scăzută
3. Timpi relativ lungi de decoerență
4. Un set „universal” de porți cuantice
5. O capabilitate de măsurare specifică qubitului

Calculatoarele cuantice analogice au nevoie de toate cele de mai sus, cu excepția elementului 4, deoarece nu folosesc porți pentru a-și exprima algoritmii.
Cu toate acestea, decoerența joacă un rol foarte diferit în calculul cuantic analogic decât în modelul de poartă.
De exemplu, o oarecare decoerență este tolerabilă în “recoacerea cuantică” și un anumit interval de relaxare energetică este necesar pentru ca “recoacerea cuantică” să reușească.

Calculatoarele cuantice au fost implementate folosind modele NISQ cuantice analogice și digitale.
Sistemele complet corectate de erori sunt mai dificil de realizat și sunt încă în curs de dezvoltare.

Research at Intel Labs has led directly to the development of Tangle Lake, a superconducting quantum processor that incorporates 49 qubits in a package manufactured at Intel's 300-millimeter fabrication facility in Hillsboro, Oregon.
This device represents the third-generation quantum processors produced by Intel, scaling upward from 17 qubits in its predecessor.



CANDIDATES FOR QUANTUM COMPUTERS

- Superconductor-based quantum computers (including SQUID-based quantum computers)
- Ion trap-based quantum computers
- "Nuclear magnetic resonance on molecules in solution"-based
- “Quantum dot on surface"-based
- “Laser acting on floating ions (in vacuum)"-based (Ion trapping)
- "Cavity quantum electrodynamics" (CQED)-based
- Molecular magnet-based
- Fullerene-based ESR quantum computer
- Solid state NMR Kane quantum computer

QUANTUM COMPUTING PROBLEMS

- Current technology
  - ≈ 40 Qubit operating machine needed to rival current classical equivalents.
- Errors
  - Decoherence - the tendency of a quantum computer to decay from a given quantum state into an incoherent state as it interacts with the environment.
    - Interactions are unavoidable and induce breakdown of information stored in the quantum computer resulting in computation errors.
  - Error rates are typically proportional to the ratio of operating time to decoherence time
    - operations must be completed much quicker than the decoherence time.

Algoritmi cuantici

ALGORITMUL SHOR

Algoritm de factorizare a numerelor intregi

Exemplu de factorizare a unui număr întreg impar N (să alegem 15):

1. Alegem un intreg *q* astfel incat $N^2 < q < 2N^2$ **hai sa alegem 256**
2. Alegem *x arbitrar* astfel incat cmmdc(*x*, N) = 1 **hai sa alegem 7**
3. Creem doi registri cuantici (acesti registri trebuie sa fie “entangled” astfel incat colapsarea registului de intrare sa corespunda cu colapsarea registrului de iesire)
   - Registrul de intrare: trebuie sa contina suficienti qubiti pentru a stoca numere la fel de mari ca q-1. **pana la 255, este nevoie de 8 qubiti**
   - Registrul de iesire: trebuie sa contina suficienti qubiti pentru a stoca numere la fel de mari ca N-1. **pana la 14, este nevoie de 4 qubiti**

SHOR'S ALGORITHM - PREPARING DATA

4. Incarcam the registrul de intrare cu o superpozitie, de ponderi egale, a tuturor intregilor from 0 to *q*-1. 0 to 255
5. Incarcam registrul de iesire cu zero peste tot.

The starea totala a sistemului in acest punct va fi:

$$\frac{1}{\sqrt{256}} \sum_{a=0}^{255} |a, 000\rangle$$

Input Register

Output Register

Nota: virgula semnifica faptul ca registrii sunt "entangled"

SHOR'S ALGORITHM - MODULAR ARITHMETIC

6. Aplicam transformarea $x^a \bmod N$ fiecarui numar din registrul de intrare, stocand rezultatul rezultatul fiecarui calcul in registrul de iesire.

Input Register	7 Mod 15^a	Output Register
|0>	7 Mod 15^0	1
|1>	7 Mod 15^1	7
|2>	7 Mod 15^2	4
|3>	7 Mod 15^3	13
|4>	7 Mod 15^4	1
|5>	7 Mod 15^5	7
|6>	7 Mod 15^6	4
|7>	7 Mod 15^7	13

Nota: folosim numere zecimale numai pentru simplitate

SHOR'S ALGORITHM - SUPERPOSITION COLLAPSE

7. Acum aplicam masuratori asupra registrului de iesire. Aceasta va colapsa superpozitia catre unul din rezultatele transformarii, sa numim aceasta valoare ***c***.

Registrul de iesire va colapsa catre una din urmatoarele stari:

|1>, |4>, |7>, or |13>

De dragul exemplului, să alegem **|1>**

SHOR'S ALGORITHM - ENTANGLEMENT

8. Deoarece registri sunt "entangled", masurind registrul de iesire, ca efect va apare colapsarea of partiala a registrului de intrare intr-o **superpozitie egala** a fiecarei stari intre 0 si q-1 care produce c (valoarea din registrul de iesire colapsat)

Deoarece registrul de iesire colapseaza catre $|1\rangle$, registrul de intrare va colapsa partial catre:

$$\frac{1}{\sqrt{64}}|0\rangle + \frac{1}{\sqrt{64}}|4\rangle + \frac{1}{\sqrt{64}}|8\rangle + \frac{1}{\sqrt{64}}|12\rangle, \ldots$$

Probabilitatile in acest caz sunt $\frac{1}{\sqrt{64}}$ deoarece registrul de intrare este acum intr-o superpozitie egala de 64 values (0, 4, 8, . . . 252)

SHOR'S ALGORITHM - QFT

Acum aplicăm transformata Fourier cuantică (TFq) pe registrul de intrare parțial colapsat.

Transformata Fourier are efectul de a lua o stare |a> si de a o transforma intr-o stare data de :

$$\frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} |c\rangle * e^{2\pi i a c / q}$$

SHOR'S ALGORITHM - QFT

$$\frac{1}{\sqrt{64}} \sum_{a \in A} |a\rangle , |1\rangle \longrightarrow \frac{1}{\sqrt{256}} \sum_{c=0}^{255} |c\rangle * e^{2\pi i ac / 256}$$

Nota: A este multimea tuturor valorilor pentru care $7^a \bmod 15$ produce 1.
In cazul nostru A = {0, 4, 8, ..., 252}

Astfel starea finala a registrului de intrare dupa aplicarea TFq este:

$$\frac{1}{\sqrt{64}} \sum_{a \in A} \frac{1}{\sqrt{256}} \sum_{c=0}^{255} |c\rangle * e^{2\pi i ac / 256}, |1\rangle$$

SHOR'S ALGORITHM - QFT

TFq va genera, în esență, amplitudinile de probabilitate ca multipli întregi ai lui q/4 în cazul nostru 256/4 sau 64.

|0>, |64>, |128>, |192>, ...

Deci nu mai avem o suprapunere egală de stări, amplitudinile de probabilitate ale stărilor de mai sus sunt acum mai mari decât celelalte stări din registrul nostru. Măsurăm registrul și va colapsa cu mare probabilitate la unul dintre acești multipli de 64, să numim această valoare p.

Cu cunoștințele noastre despre q și p, există metode de calculare a perioadei (o metodă este dezvoltarea fracției continue a raportului dintre q și p.)

SHOR'S ALGORITHM - THE FACTORS :)

10. Acum că avem perioada, factorii lui N pot fi determinați luând cel mai mare divizor comun al lui N față de x ^ (P/2) + 1 și x ^ (P/2) - 1.
In aceasta etapa calcul se va face pe un calculator clasic.

Calculam:

cmmdc $(7^{4/2} + 1, 15) = 5$

cmmdc $(7^{4/2} - 1, 15) = 3$

Am factorizat cu success 15.

SHOR'S ALGORITHM - PROBLEMS

- TFq poate oferi o perioada greșită.

Probabilitatea depinde de fapt de alegerea pentru q.
Cu cât este mai mare q, cu atât este mai mare probabilitatea de a găsi probabilitatea corectă.

- Perioada seriei ajunge să fie impara

Dacă apare oricare dintre aceste cazuri, ne întoarcem la început și alegem un nou x.

CONCLUZIE

In 2001, o masina cu 7 qubiți a fost construita si programata pentru a executa algoritmul Shor, factorizand cu success numarul 15.