The Math behind
Quantum Computing

From Boolean algebra and the half adder, through von Neumann and Backus, to vectors, matrices and the tensor product — a guided route from what we learned at school to the mathematics quantum computers actually run on.

Dipl. Inf. Ramon Schweiss · QWeb © Research · qweb-quantum.com

Where this is going

six steps, each one built on the last

I — **The classical prologue**

Boolean algebra, adders, shift registers, von Neumann — and Backus's 1977 challenge

II — **The break**

what replaces Boolean algebra: linear algebra over complex vector spaces

III — **The tensor product**

how several qubits are combined into one description — and why it grows as 2^n

IV — **Where entanglement lives**

a state is entangled exactly when the tensor product cannot be undone

V — **Boolean algebra returns**

at measurement, and in reversible embeddings of classical logic

VI — **Coda**

a quantum circuit is the functional program Backus asked for

I What we already learned at school

Boolean algebra — the whole of classical computing rests on it

The five gates everything is built from

Boolean algebra: two values, a handful of operations, and every computer ever made

NOT	AND	OR	XOR	NAND
0 → 1	00 → 0	00 → 0	00 → 0	00 → 1
1 → 0	01 → 0	01 → 1	01 → 1	01 → 1
	10 → 0	10 → 1	10 → 1	10 → 1
	11 → 1	11 → 1	11 → 0	11 → 0

NAND alone is universal — every other gate can be built from copies of it.

Two values, five operations, and a set of rules for combining them. George Boole published this algebra in 1854, eighty years before anyone had a machine to run it on. Every processor ever manufactured is, at bottom, an enormous arrangement of these operations.

I

From logic to arithmetic

adders and shift registers — where gates become numbers

From gates to arithmetic

two gates make a half adder; two half adders make a full adder

HALF ADDER

- XOR → sum
- AND → carry

adds two bits
no carry in

FULL ADDER

- half adder
- half adder + OR

adds three bits
carry in and out

And multiplication? A shift register.

1 0 1 1	**1 0 1 1 0**	**1 0 1**
the number 11	shift left = × 2	shift right = ÷ 2

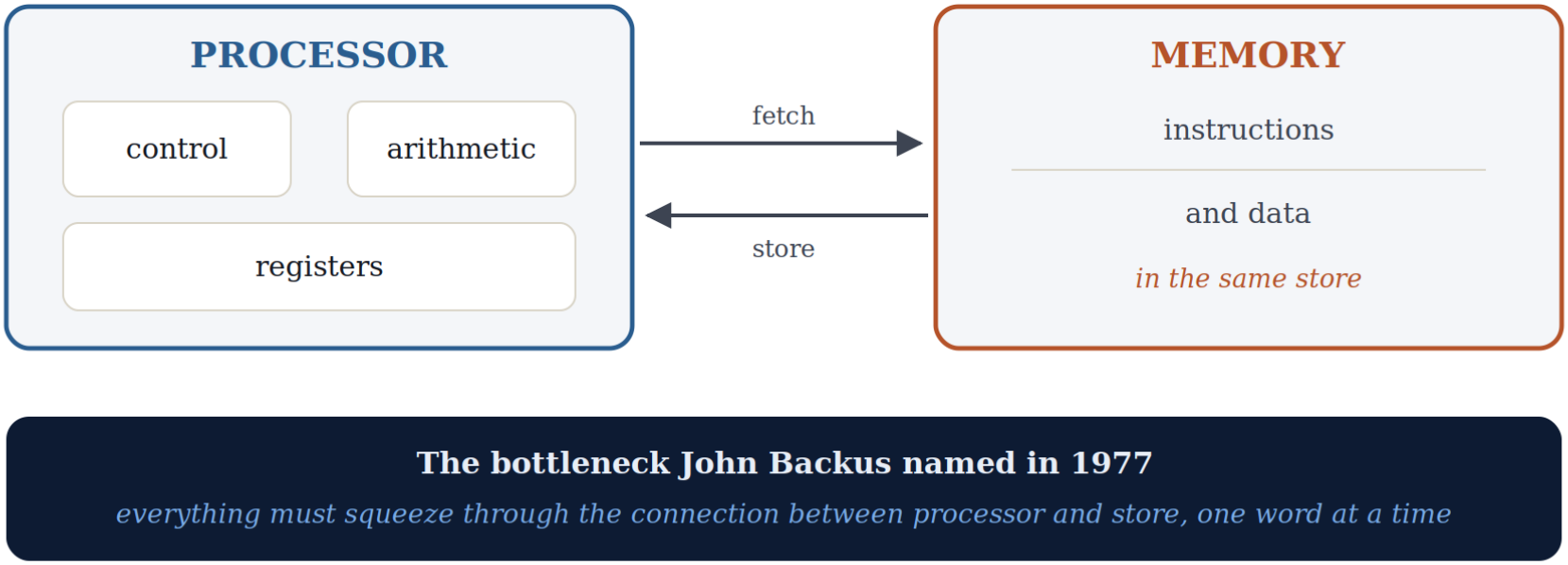
The half adder adds two bits; chain full adders together and you can add numbers of any width. Multiplication and division by two need no gates at all — only a shift of the bit pattern left or right. This is the entire arithmetic vocabulary of a classical machine.

I The machine that runs it

one processor, one store, one instruction at a time

The von Neumann architecture

one processor, one memory, one instruction at a time — and a bus between them



Instructions and data share the same memory, and both travel across the same narrow connection. Whatever else changes between a smartwatch and a supercomputer, this shape does not.

I

The first objection — 1977

John Backus, who gave us Fortran, turned on the style he had helped create

“Can Programming Be Liberated from the von Neumann Style?
A Functional Style and Its Algebra of Programs”

Turing Award Lecture, 1977 · Communications of the ACM, August 1978

The complaint

Programs spend their effort shuttling single words to and from a store. The assignment statement is the bottleneck made into a language feature.

The proposal

Build programs by composing functions instead — and give that composition a proper algebra, so programs can be reasoned about like equations.

Why it matters here

Backus was describing, in 1977, almost exactly the shape a quantum circuit would turn out to have. We return to this at the end.

II

What replaces Boolean algebra

the answer is linear algebra

What replaces Boolean algebra

the answer is linear algebra over complex vector spaces

	classical	quantum
state space	the set $\{0,1\}^n$	**a complex vector space**
a state	a string of bits	**a normalised vector**
a gate	a truth table	**a unitary matrix**
running a gate	look up the row	**multiply the matrix**
combining systems	write them side by side	**tensor product**

Boolean algebra does not vanish — but it moves to the edges: measurement, and reversible embedding.

A qubit is not a value but a direction: a vector in a complex space. A gate is not a lookup table but a matrix, and running it means multiplying. Nothing here is an extension of Boolean algebra — it is a different mathematics underneath.

II

Why the matrices must be unitary

the one constraint physics imposes on every quantum gate

Reversible

Every quantum gate can be undone. Multiply by the inverse matrix and you are exactly where you started — no information is ever discarded.

Length-preserving

The vector's length always stays 1. Since the squared entries are the probabilities of each outcome, they must keep summing to 1.

The consequence

There can be no quantum AND gate. AND throws information away, so no matrix can perform it. Classical logic has to be rebuilt reversibly.

This is the whole reason 1 + 1 is harder on a quantum computer

A classical half adder uses AND for the carry, and AND is irreversible. A quantum machine must instead borrow blank helper qubits and use a three-input gate that keeps every input intact — which is then itself assembled from roughly six smaller operations.

III

The tensor product

how two qubits become one description

The tensor product: how qubits combine

not placed side by side — multiplied together

one qubit — two numbers

$$|0\rangle = (1 , 0)$$

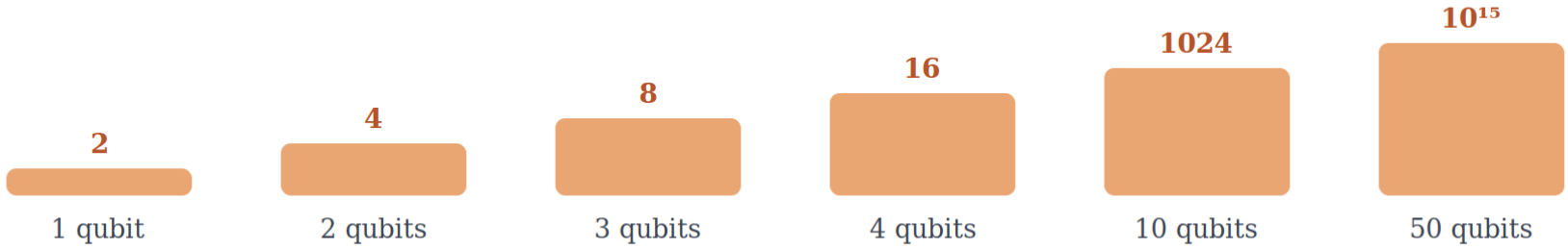
$$|1\rangle = (0 , 1)$$

two qubits — four numbers

$$|0\rangle \otimes |1\rangle = (0 , 1 , 0 , 0)$$

$$|1\rangle \otimes |1\rangle = (0 , 0 , 0 , 1)$$

Every entry of the first vector is multiplied by the whole second vector.



n qubits need 2^n numbers to describe. That exponential is not a metaphor — it is the dimension of the space.

Two systems are not written side by side, as two bits would be. Their vectors are multiplied entry by entry, and the result lives in a space whose dimension is the product of the two — which is where the famous exponential comes from.

III What a tensor is — and is not

clearing up a word that carries two different meanings

Not this

The tensor of general relativity — an object defined by how it transforms when you change coordinates. Rich, difficult, and not what quantum computing means by the word.

But this

The tensor product, also called the Kronecker product: a concrete recipe for building one big vector or matrix out of two smaller ones. Multiply every entry of the first by the whole of the second.

The same recipe applies to gates

To act on the first qubit and leave the second untouched, take the tensor product of the gate with the identity matrix. Two 2×2 matrices become one 4×4 matrix that acts on the pair as a whole. Every circuit you write is ultimately one very large matrix of this kind — which is also why simulating many qubits on a classical computer becomes hopeless so quickly.

IV

Where entanglement lives

not an extra rule — a property of the vector space itself

Where entanglement lives

it is the failure of the tensor product to come apart again

SEPARABLE

$|00\rangle + |01\rangle$

splits cleanly into

$|0\rangle \otimes (|0\rangle+|1\rangle)$

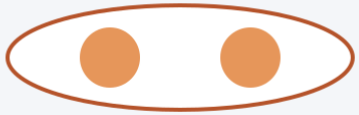
each qubit has its own state

ENTANGLED

$|00\rangle + |11\rangle$

cannot be written as

$A \otimes B$ — for any A, B



only the pair has a state

Entanglement is not an extra ingredient added to the mathematics.
It is what most vectors in the space simply are.

A two-qubit state is separable exactly when it can be written as one qubit's vector tensored with another's. When no such pair of vectors exists, the state is entangled — and neither qubit has a state of its own. Entanglement is therefore not something added to the mathematics; it is what almost every vector in the space already is.

V Boolean algebra comes back

at the two places where the quantum world meets the classical one

At measurement

Running the circuit produces a vector — but reading it produces bits. Measurement collapses the whole apparatus back to a string of zeros and ones, which is then processed by entirely ordinary classical code.

In reversible logic

Classical operations can be embedded in quantum circuits, provided nothing is erased. AND becomes a three-input gate that writes its answer onto a spare qubit and leaves the inputs standing.

So Boolean algebra is not the foundation of quantum computing — it is its interface.

It governs what goes in and what comes out. Between those two moments, the machine is doing something else entirely: multiplying a very large matrix by a very long vector, with no bits anywhere in sight.

VI

Coda: Backus, answered by physics

the functional style he argued for, arriving from an unexpected direction

Backus, answered by physics

what he asked for in 1977, a quantum circuit simply is

THE VON NEUMANN STYLE

- a store full of cells
- assignment as the basic act
- one word at a time
- state changes as you go

A QUANTUM CIRCUIT

- no cells, no store
- composition as the basic act
- the whole state at once
- no assignment anywhere

"Can programming be liberated from the von Neumann style?"

John Backus, Turing Award Lecture 1977 — published August 1978

A quantum circuit has no store, no assignments, and no step-by-step mutation of memory. It is a composition of functions with a genuine algebra behind it — matrix algebra. Nobody set out to satisfy Backus. Physics simply left no other option.

What to take away

1. **Boolean algebra is the interface, not the engine**

   It governs input and output. The computation in between is linear algebra.

2. **A qubit is a vector, a gate is a matrix**

   Running a circuit means multiplying. Unitarity — reversibility — is the one hard constraint.

3. **The tensor product is how systems combine**

   n qubits need 2^n numbers. The exponential is the dimension of the space, not a manner of speaking.

4. **Entanglement is failed factorisation**

   A state is entangled precisely when no pair of separate vectors can reproduce it.

5. **The shape is functional, not von Neumann**

   No store, no assignment, just composition — the style Backus asked for in 1977.

Further reading

Companion articles in the QWeb © Foundations series

The Camshaft Problem: What a Quantum Computer Actually Is

why the machine is deliberately bad at arithmetic — and what it is for instead

Whose Measurement Came First?

entanglement, measurement order, and the relativity of collapse

How Entanglement Lowers the Price of Agreement

quantum communication complexity, and the cost of consistency in distributed systems

The Mermin–Peres Magic Square Game

a task two players win every time with entanglement, and never without it

qweb-quantum.com