QRAND

A multi-platform and multi-protocol quantum random number generator for arbitrary probability distributions

Pedro Rivero – April 30, 2021

Unitary Fund

Table of contents

The need for *random numbers**

Applications

- Information/communications security *(e.g. crypto-systems, IoT)*
- Computation *(e.g. algorithms, physical simulations, Monte Carlo)*
- Financial systems *(e.g. transactions, credit card chips)*
- Statistical sampling *(e.g. scientific experiments, social studies)*
- Testing *(e.g. randomized benchmarking)*
- Legal and trust sensitive processes *(e.g. jury, lotteries, Kleroterion)*

**Randomly generated numbers*

Building blocks of a crypto-system

Migration to quantum safe solutions

- These can all be performed quantumly:

HIDING MECHANISM

Algorithm

UNPREDICTABLE SECRET

Random key generation

SECRET DISTRIBUTION

Key exchange (BB84)

What we mean by random

A slippery concept

- The attribute of being random applies more correctly to a sequence of numbers: It is strictly related to **lack/impossibility of predictability**.
- The impossibility of **compression** can be considered as the defining trait of randomness (*Chaitin & Kolmogorov*).
- **Shannon Entropy** (H) → Measure of disinformation/uncertainty

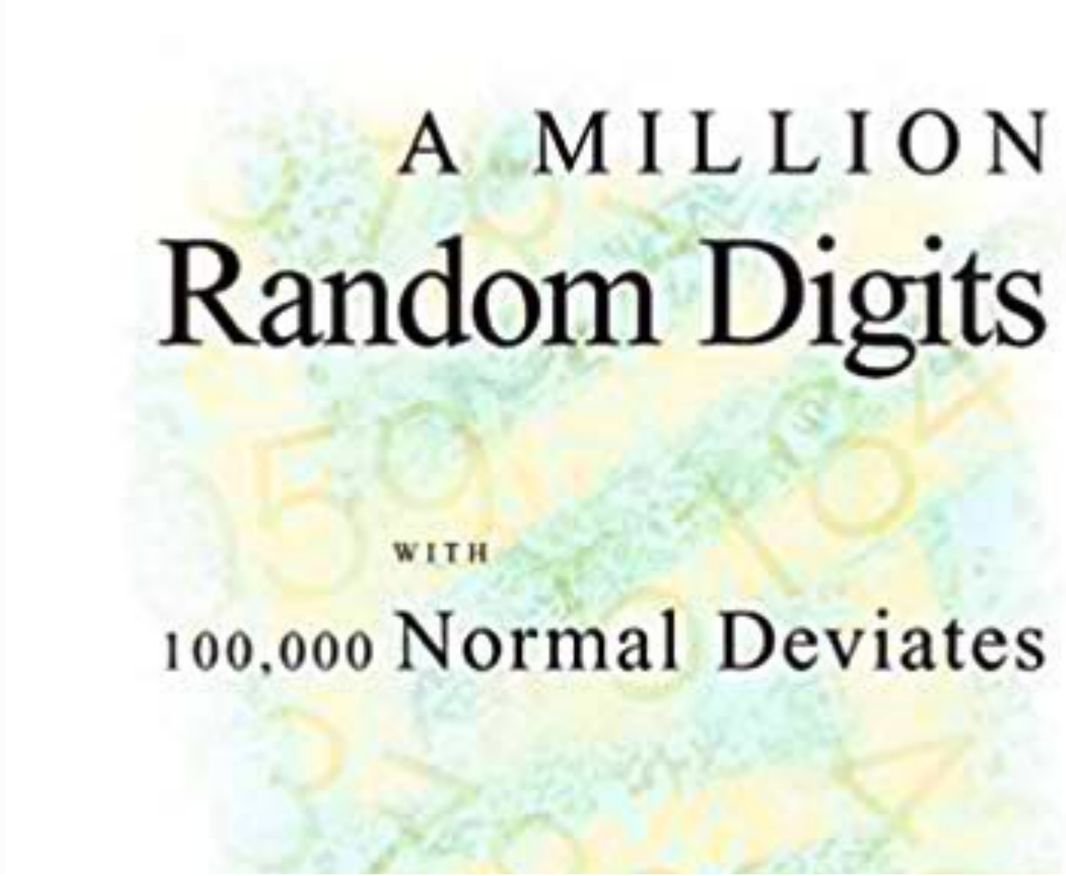
> *The minimum number of yes/no questions that need to be answered about something to be able to guess it correctly with 100% confidence.*

- Bit strings which are not random have limited entropy and can therefore be described with **fewer bits than the string itself**.

Randomness quality

Output tests

- It is nearly impossible to establish whether a source is truly random.
- We can analyze the general properties of output strings (e.g. weight)
- A passing output will always pass (the tests): an adversary might *predict* or *fabricate* the output.
- Numbers are not inherently random *per se:* R(NG) ≠ (RN)G



TDB

★☆☆☆☆ **Not really random**

Reviewed in the United States on September 26, 2012

I bought two copies of this book. I find that the first copy perfectly predicts what the numbers will be in the second copy. I feel cheated.

141 people found this helpful

Random number generation

Properties

- **Uniformity** → all sequences have the same probability of occurring
- **Scalability** → any test applicable to a sequence must must apply to subseq.
- **Consistency** →the behavior must be consistent across all outputs
- **Forward unpredictability** → the next output should not be predictable
- **Backward unpredictability** → previous outputs should not be predictable

Random number generation

The challenge

- Generating random numbers is hard → you need to get your hands dirty and start experimenting with physical hardware.

RANDOM PROCESS

To conveniently sample from for getting entropy

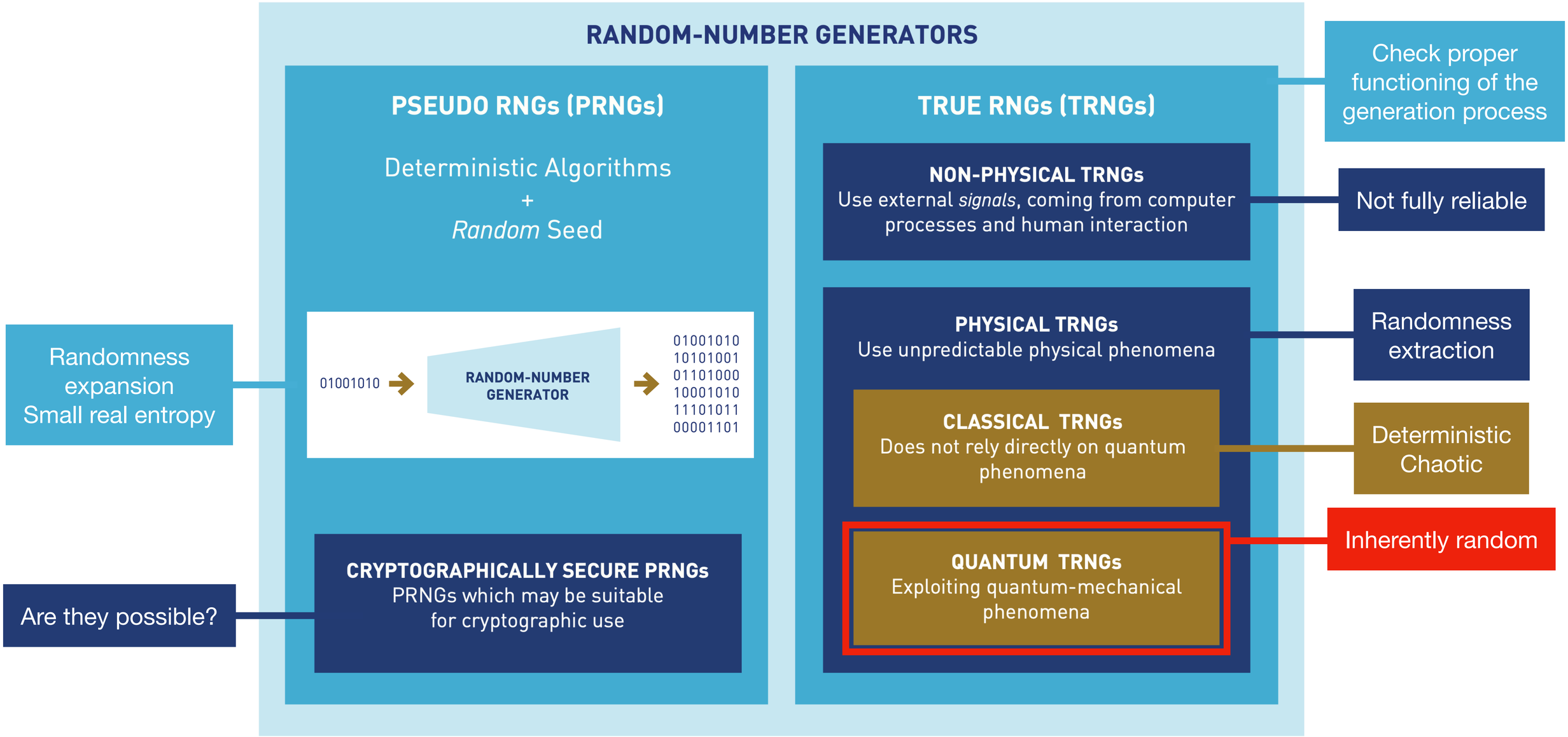
NO SECURITY BY OBSCURITY

Transparent entropy sources

RANDOM NUMBER DELIVERY

Keep the entropy pure (no leaks)

Random number generation

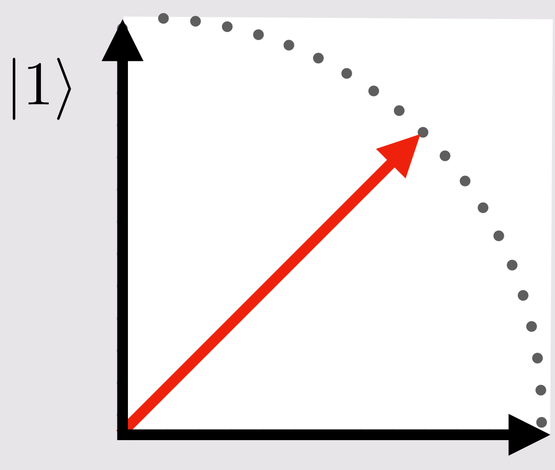


Quantum random number generators

Quantum mechanics primer

$$\text{Bits: } \{0,1\} \quad \rightarrow \quad \text{Qubits: } \{|0\rangle, |1\rangle\}$$

- **Qubits** → quantum system with two possible (distinguishable) states
- **Superposition** → a quantum system can be in a state which is a mix of classically distinguishable states.
- **Inherently random measurements**
- **Entanglement** → classically inexplicable correlations between subsystems



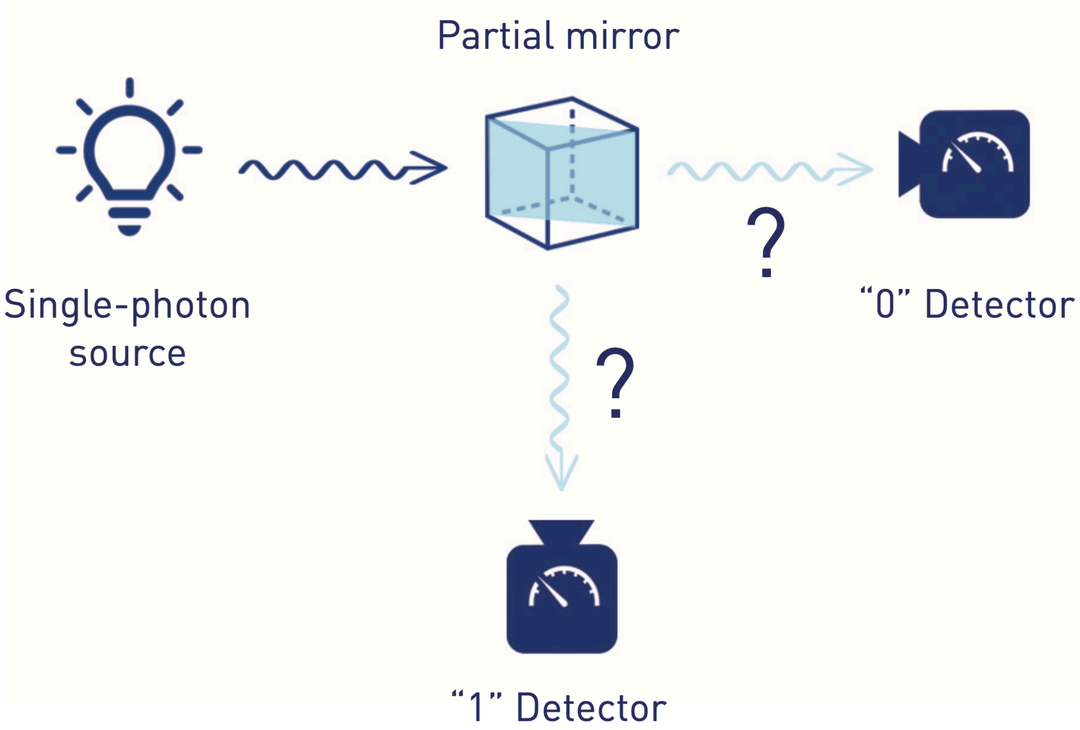
$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$$\equiv |+\rangle$$

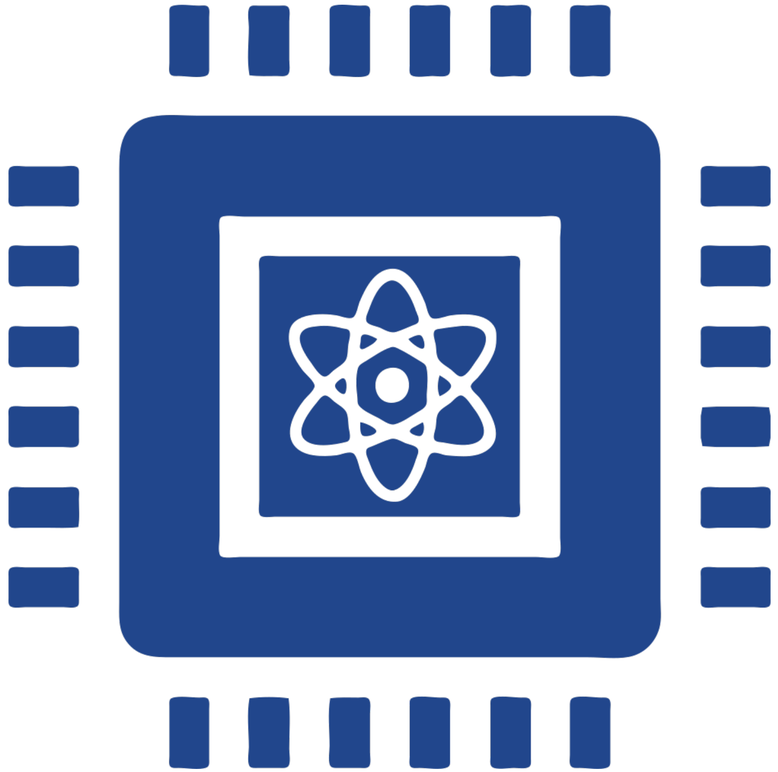
$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Quantum random number generators

Practical realizations



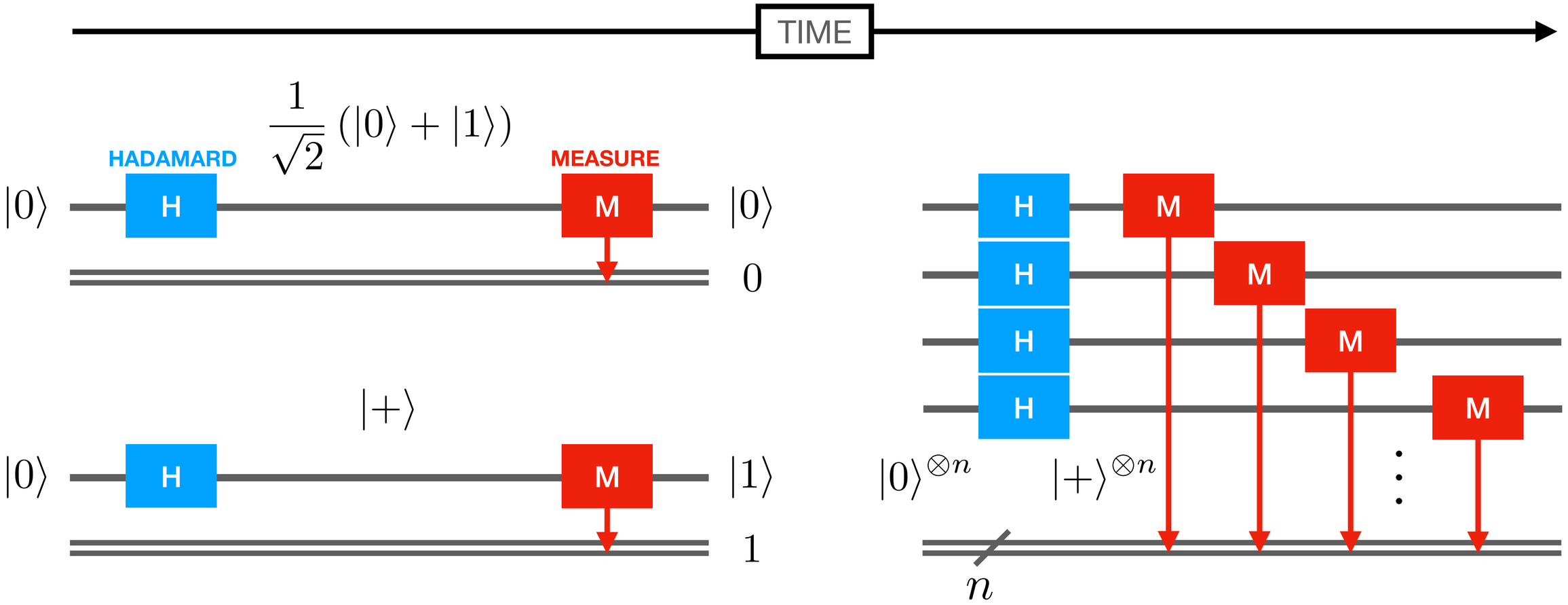
Analog quantum hardware



Digital quantum computer

Hadamard Protocol

The naive approach using quantum circuits



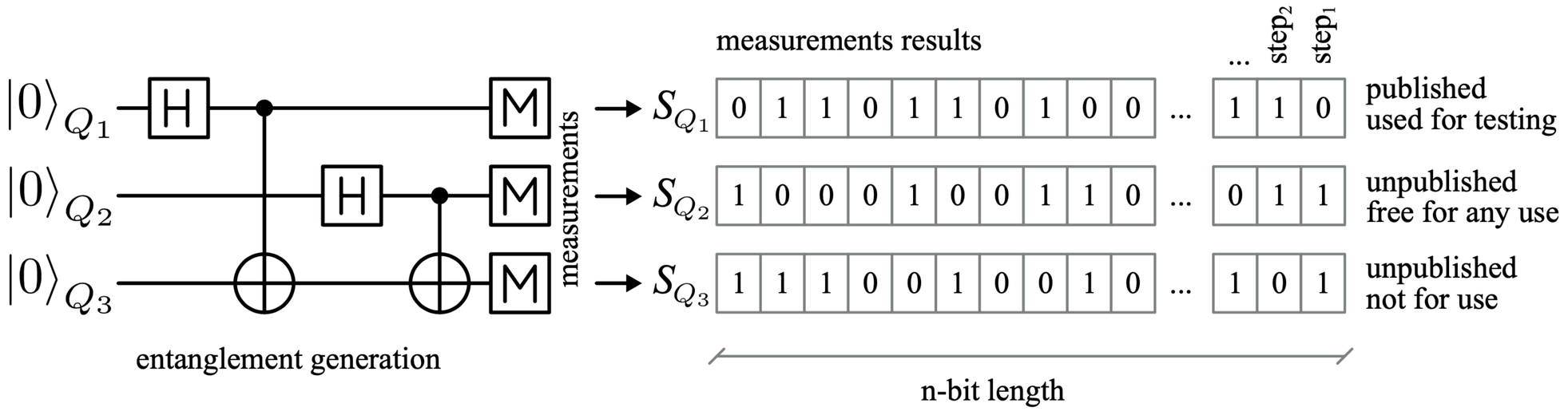
Entanglement Protocol

Validation outsourcing

- Subsequence validation → Exponential overhead

$$|\psi_{Q_1Q_2Q_3}\rangle = \frac{1}{2}\left(|000\rangle + |011\rangle + |101\rangle + |110\rangle\right)$$

TWO BITS OF ENTROPY

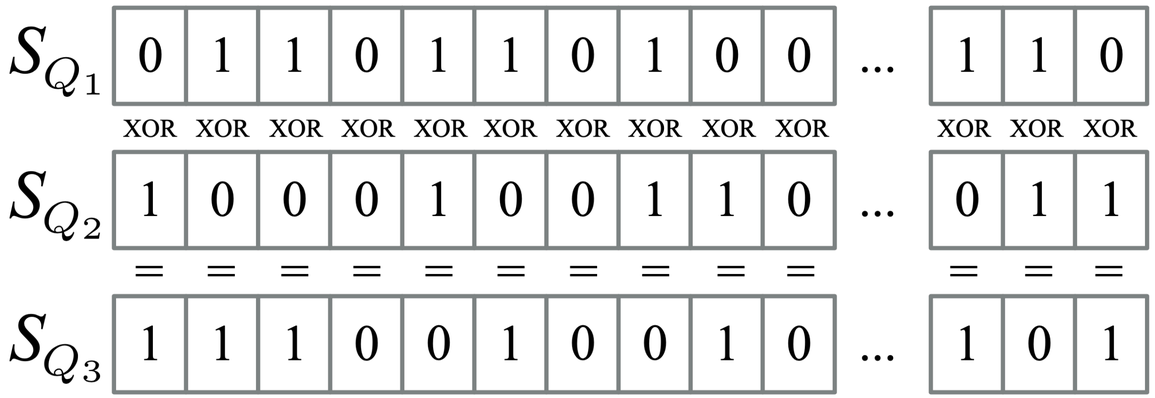


PARITY

Entanglement Protocol

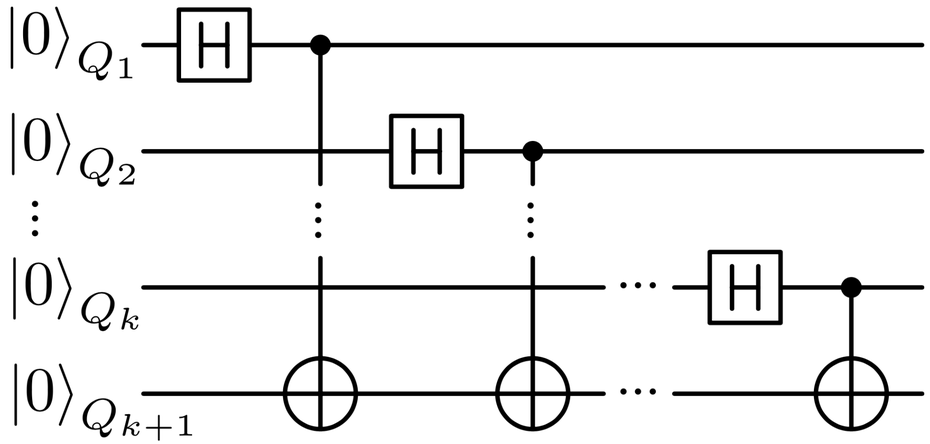
Error correction and scaling

- Parity bit



PURIFY ENTROPY

1 SEQUENCE TO VALIDATE

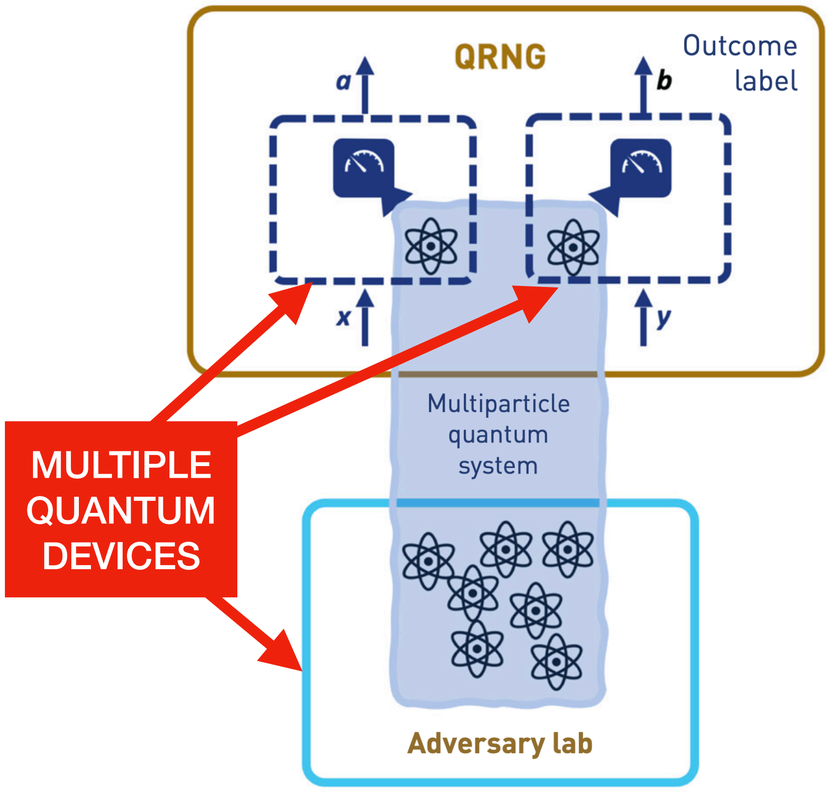


K-1 USABLE SQUENCES

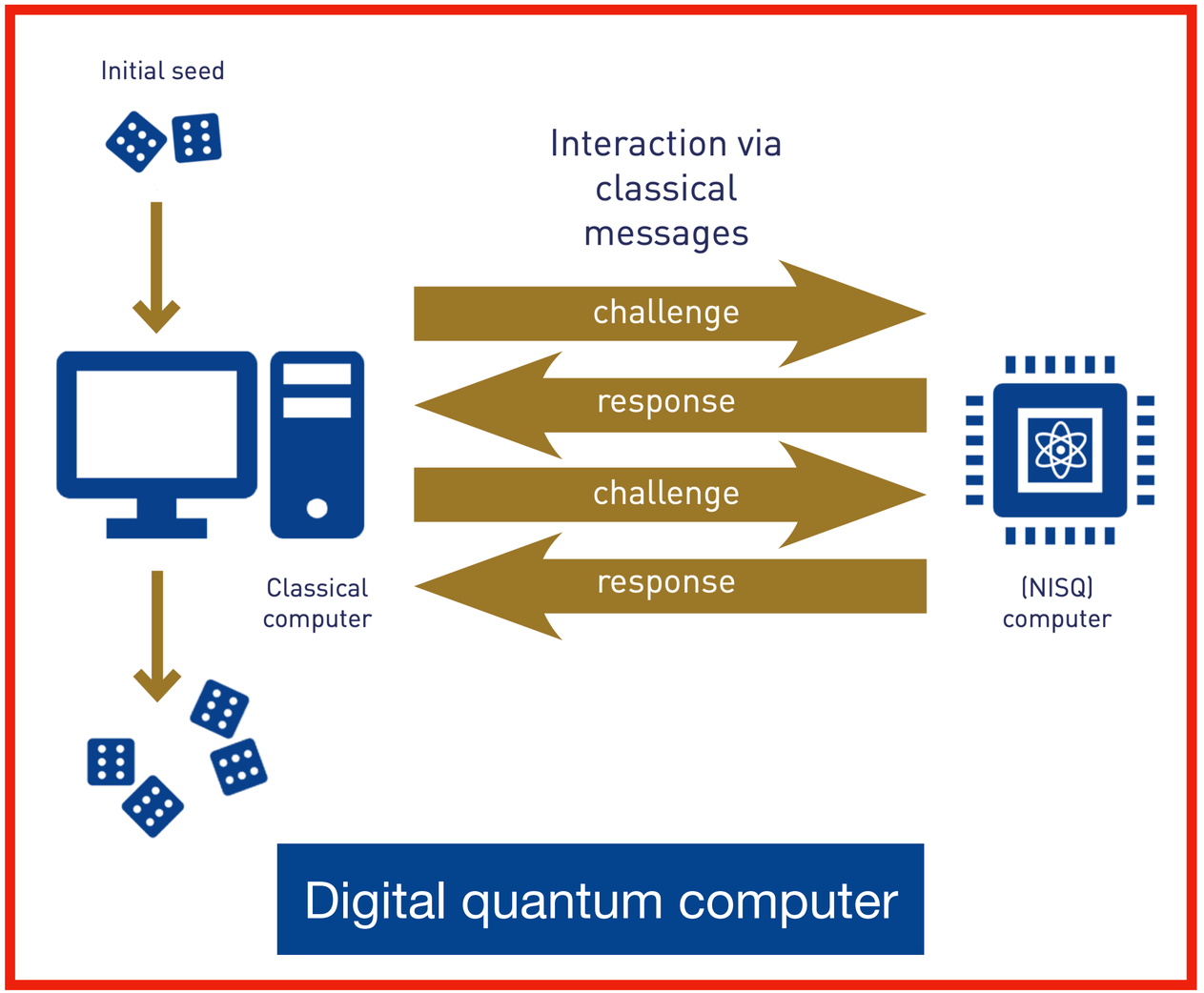
PARITY

Verification strategies

Device independency



Locality proofs

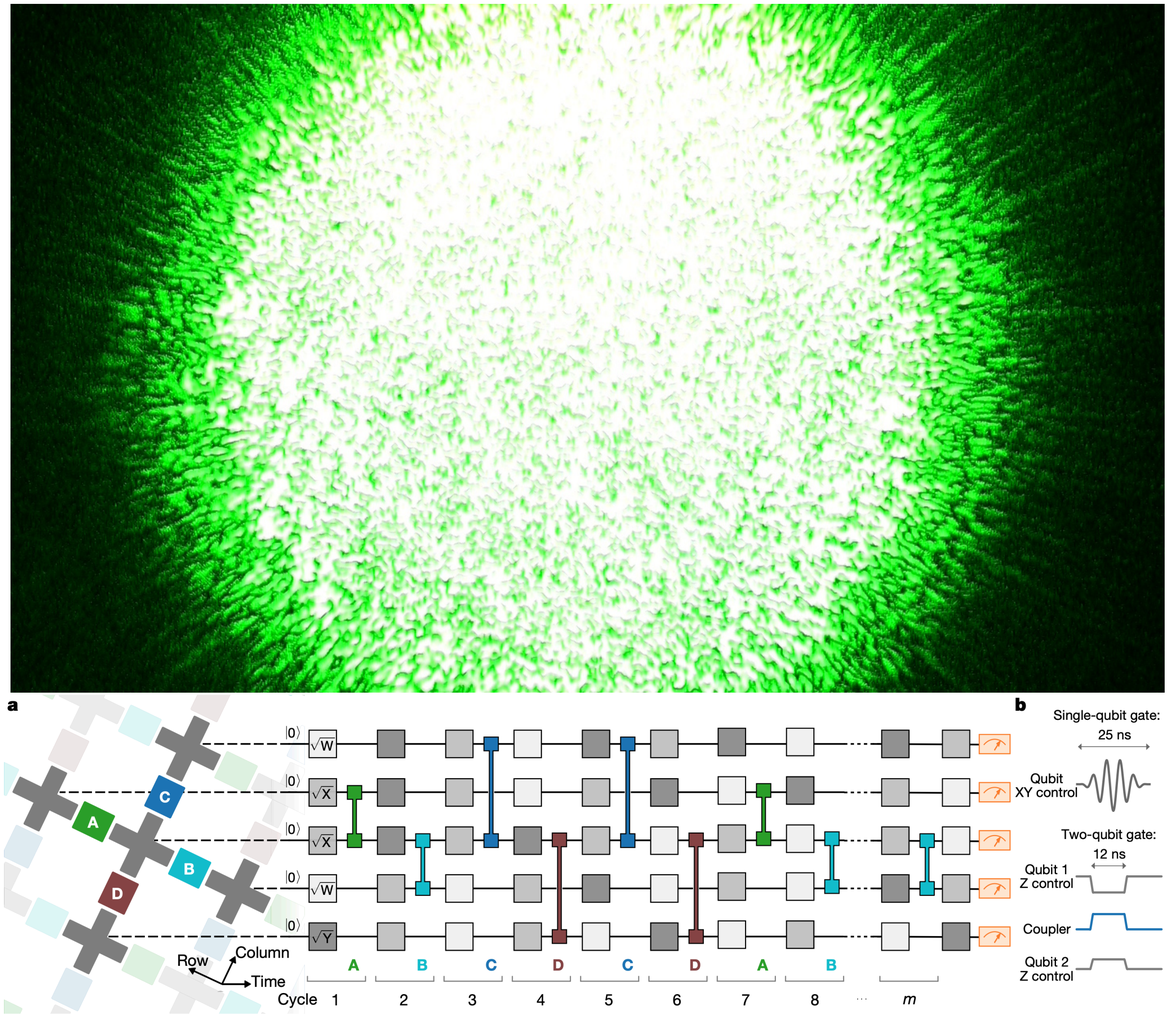


Digital quantum computer

Sycamore Protocol

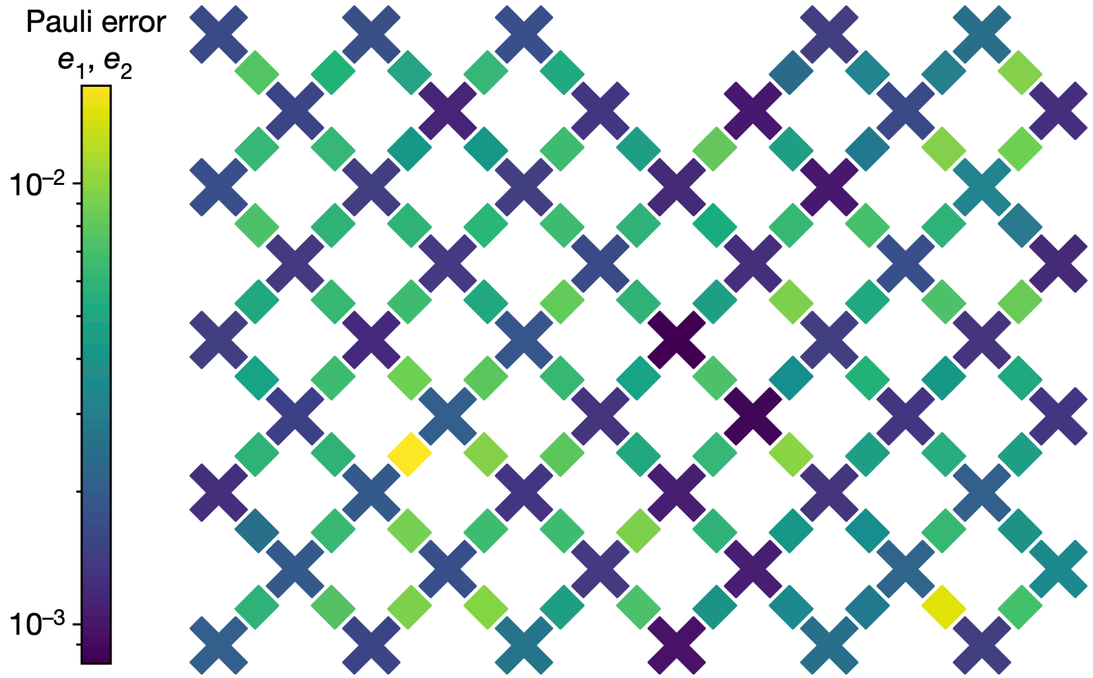
Scott Aaronson

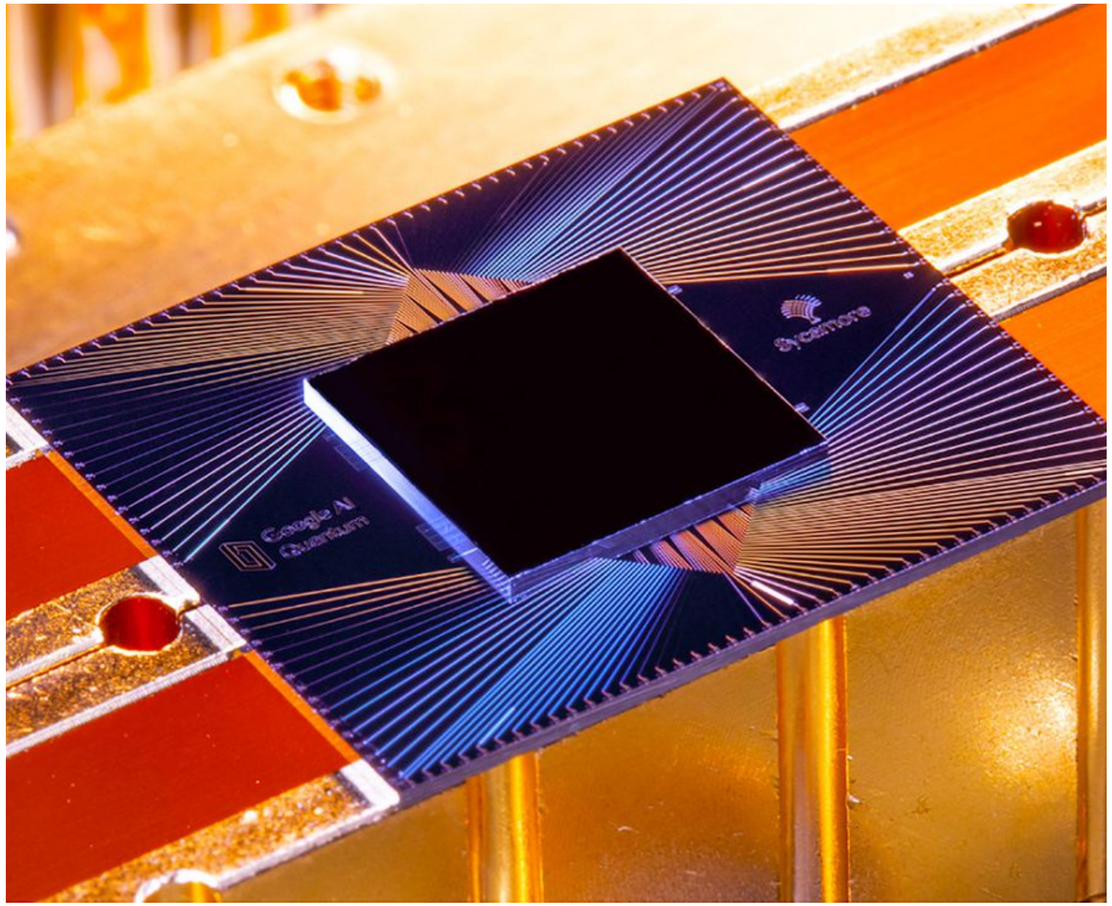
- Process:
  1. *Run a randomly chosen (known) circuit.*
  2. *Make a "small" number of measurements.*
  3. *Simulate results classically and see if they agree.*
- Computational hardness assumption.



Sycamore Protocol

Google's quantum supremacy claim

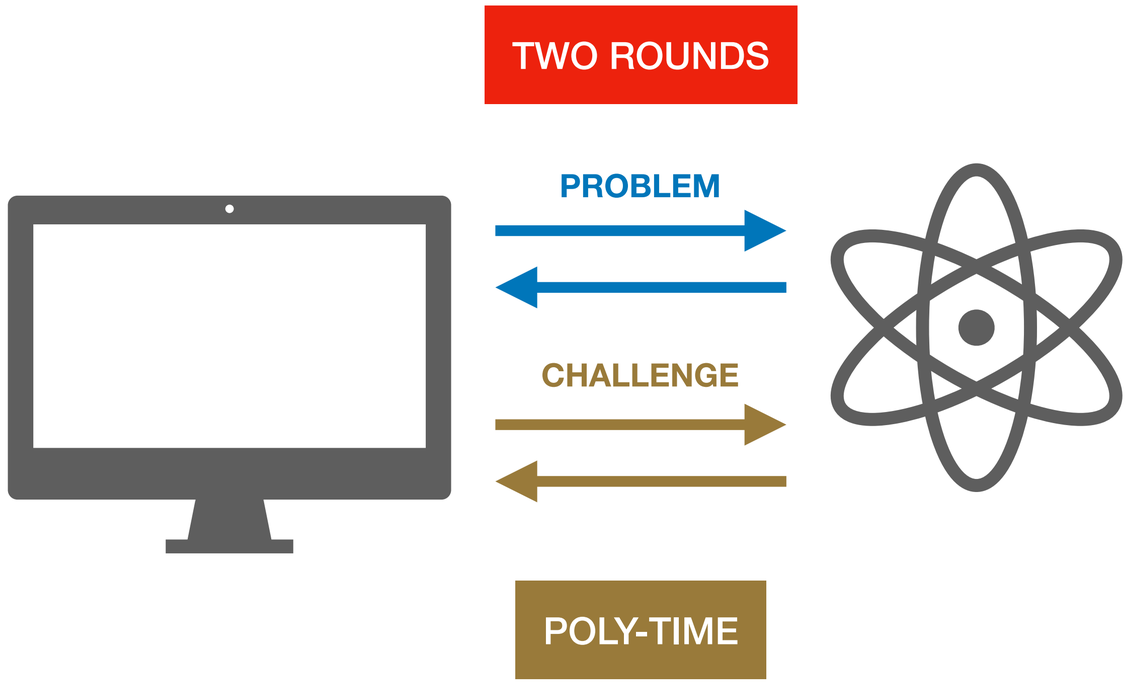




BCMVV Protocol

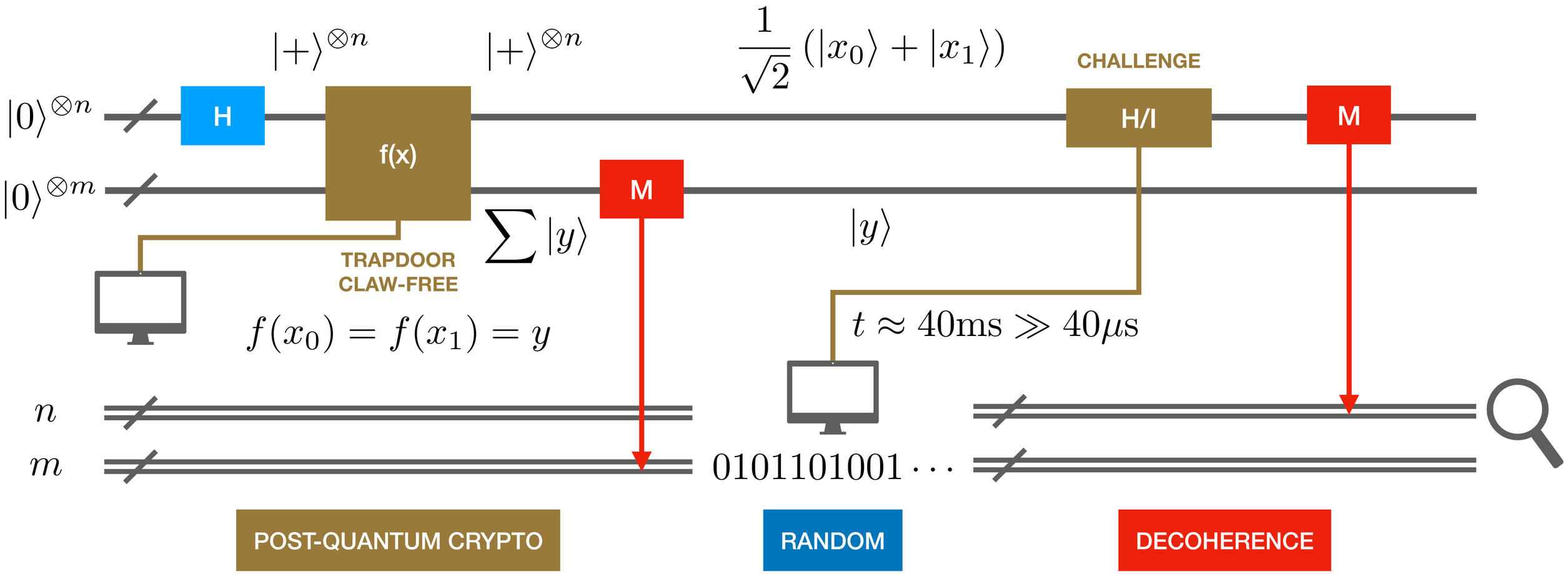
Post-quantum cryptography approach

- Previous approach relied on the classical exponential overhead → *NISQ devices*
- **Post-quantum crypto** (not to be confused with quantum crypto)
- Two rounds:
  - *Problem*
  - *Challenge*



BCMVV Protocol

Post-quantum cryptography approach



QRAND

Free and Open Source Software

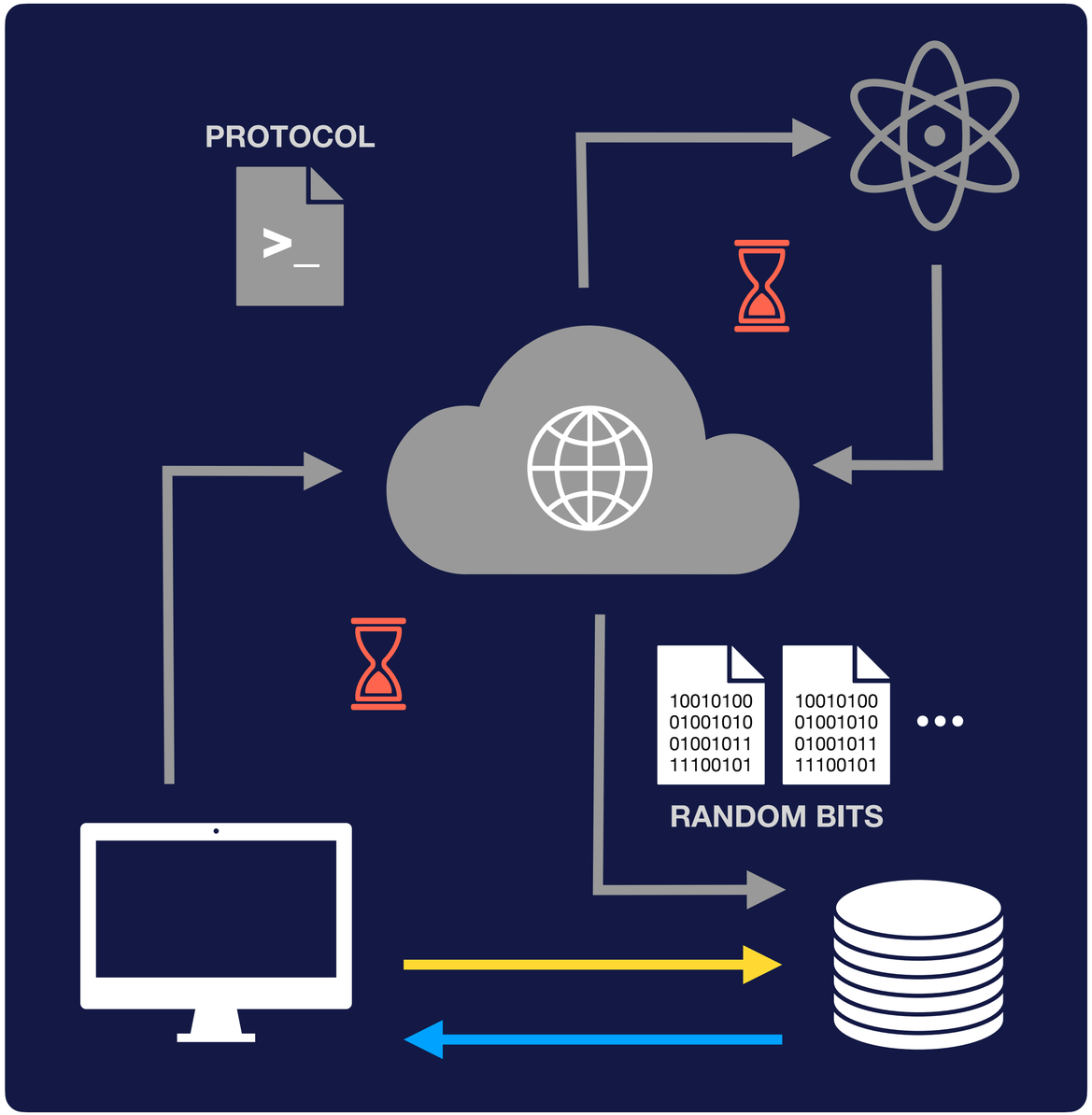
- Multiprotocol → *Hadam, Entang, ...*
- Multiplatform → ***qiskit**, cirq, q#*
- Output formats → *bitstring, int, float, complex, hex, base64...*
- Entropy validation suite
- NumPy interface → non-uniform probability distributions
- Efficiency sampling → *Backend Select. + Multithreading + Caching*

```python
from qrand import QiskitBitGenerator
from qiskit import IBMQ

provider = IBMQ.load_account()
bitgen = QiskitBitGenerator(provider)

s = bitgen.random_bitstring(2)
i = bitgen.random_uint(4)

from qrand import Qrng
qrng = Qrng(bitgen)

x = qrng.get_random_double(0, 0.1)
z = qrng.get_random_complex_polar()

from numpy.random import Generator
gen = Generator(bitgen)

l = gen.logistic()
p = gen.poisson()
```

QRAND pypi

Free and Open Source Software

- Multiprotocol → *Hadam, Entang, ...*
- Multiplatform → ***qiskit**, cirq, q#*
- Output formats → *bitstring, int, float, complex, hex, base64...*
- Entropy validation suite
- NumPy interface → non-uniform probability distributions
- Efficiency sampling → *Backend Select. + Multithreading + Caching*




Unitary Fund
unitary
May 14-30th
HACK
The Unitary Fund is proud to host our first quantum open source hackathon with SWAG and BOUNTIES on May 14-30th!
hack2021.unitary.fund
Digital swag for all participants
>$2k USD in bounties
Random drawings for swag mail

References

(1) Piani et al, *Quantum Random-Number Generators: Practical Considerations and Use Cases.* **EvolutionQ**.

(2) Abellán, *Randomness and quantum entropy w/ Carlos Abellán from Quside.* **Quantum Barcelona**.

(3) Jacak et al, *Quantum random number generators with entanglement for public randomness testing.* **Nature Research**.

(4) Arute et al, *Quantum supremacy using a programmable superconducting processor*. **Nature**.

(5) Brakerski et al, *A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device.* **IEEE 59th Annual FOCS Symp.**

Thanks

Pedro Rivero

ILLINOIS TECH

CHICAGO QUANTUM EXCHANGE

Argonne NATIONAL LABORATORY

Migration to quantum safe solutions

Why it is important to act now

COMPLIANCE WITH REGULATIONS

NIST standards

LONG MIGRATION TIME

Crypto-agile methodologies

PRODUCTS WITH LONG LIFETIMES

Car industry, self-serving

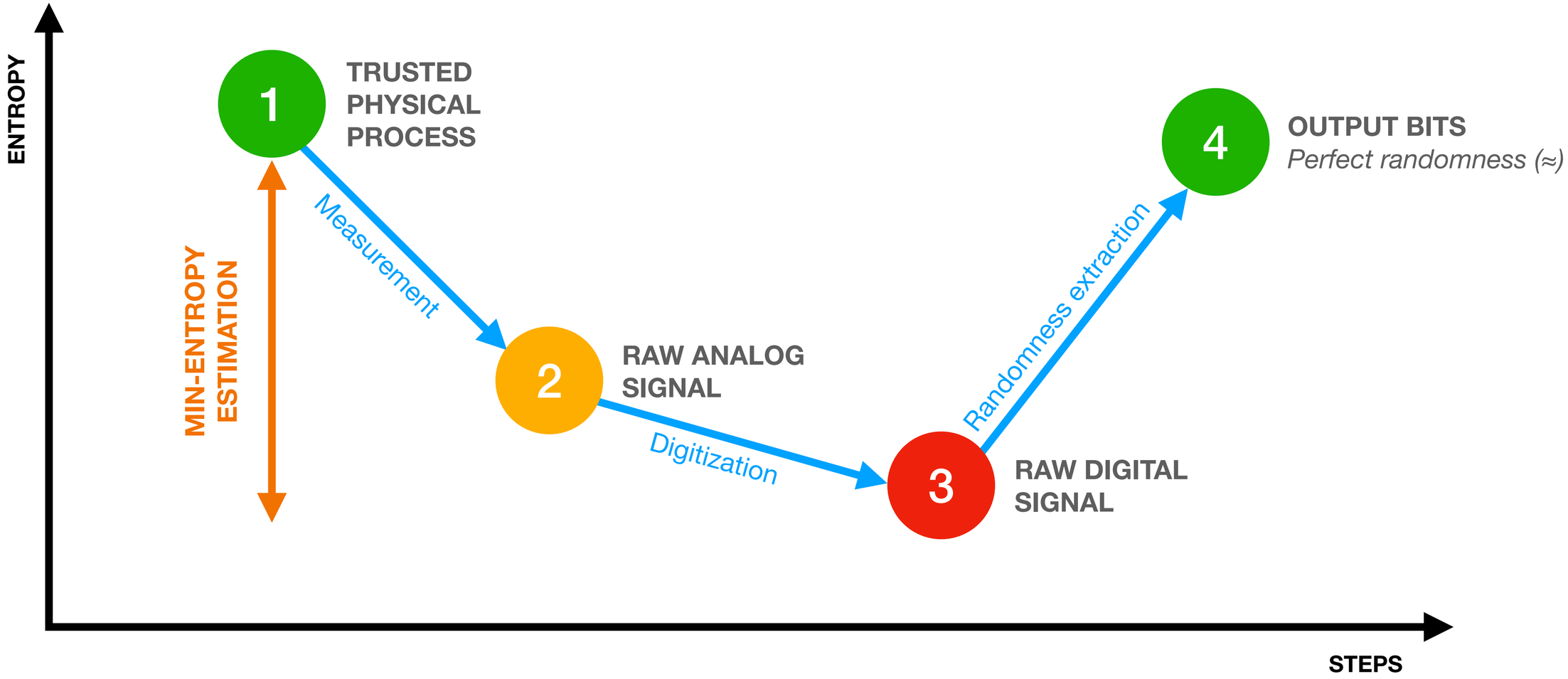
STORE NOW DECRYPT LATER

Information can be sensitive for extended periods of time

Exemplar bit string produced	String probability assuming 0 and 1 equally likely and bits are independent	Fraction of zeros	Displays Pattern?	Is the source random?
0 0 0 0 0 0 0 0 0 0 **H = 0**	2^{-10}	100%	Yes	It is reasonable to suspect it is not; it might well be that the source outputs only 0s
0 1 0 1 0 1 0 1 0 1 **H = 1**	2^{-10}	50%	Yes	It is reasonable to suspect it is not; it might well be that the source output simply switches between 0 and 1
0 0 1 0 0 0 0 0 0 1 $H = -\sum p(x) \log p(x)$ **H ≈ 7.2 < 10**	2^{-10}	80%	No	Potentially yes, but it appears but it appears to be biased; it may indicate that 0 and 1 are not equally likely – that is, the assumption of 0 and 1 being actually likely may not hold
1 0 1 1 0 1 0 0 1 0 **H = 10**	2^{-10}	50%	No	Potentially yes, and it appears also unbiased; 0 and 1 may really be equally likely

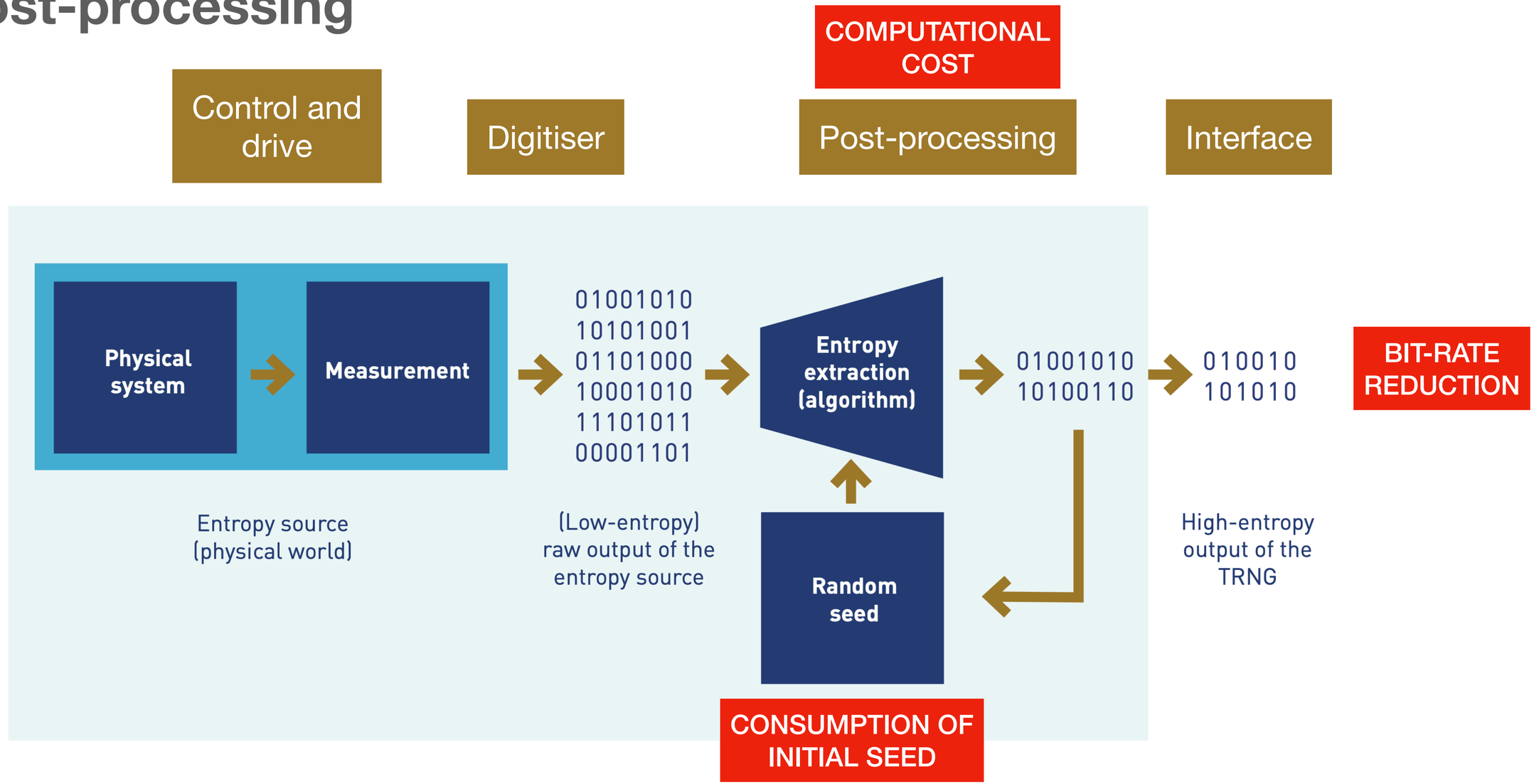
Randomness extraction

Post-processing: randomness metrology



Randomness extraction

Post-processing



Quantum random number generators

Trade-offs of quantum RNGs

ENTROPY SOURCE

Fundamentally random

ENTROPY QUALITY

High from the start

CERTIFICATION

Can validate the underlying process

SECURITY

Built-in with device independent techniques

SPEED

High for its quality
Slower on digital quantum computers

SIZE

Variable, from very small to room size

VERSATILITY

Easily scalable, specially with universal quantum computers

SUSTAINABILITY

Better for universal quantum computers