Mark-1: Prototyping a Quantum Hash Function Using Parameterized Circuits

Bhvyadhirr Bharadwaj

In the face of an approaching quantum future, cryptography is undergoing one of the most disruptive paradigm shifts since its inception. From post-quantum algorithms to quantum key distribution, we're entering uncharted territory. As part of this evolving narrative, I've built a prototype quantum hash function named **Mark-1**—a simulated experiment that explores how quantum mechanics can be harnessed to build the cryptographic primitives of tomorrow.

What really is Mark-1?

Mark-1 is a prototype hash function built atop **parameterized quantum circuits**, designed to accept classical inputs and output deterministic hash-like signatures derived from quantum state manipulations.

It simulates a 16-qubit, 3-layer entangled quantum circuit that:

- Encodes classical input into rotation parameters,
- Entangles qubits in alternating block layers,
- Outputs signatures from the final statevector of the circuit.

The hash is derived from the real/imaginary parts of selected amplitudes in the final quantum state.

Note: This is a simulated model, not yet deployable on live quantum hardware. But it's a step forward in thinking how quantum-native cryptographic functions might work.

Hash Quality Evaluation

To test the robustness of the prototype, I ran it through four classical hash evaluation benchmarks:

Entropy Test

- **Avg entropy per byte (100 samples):** *~1.74 to 2.31 bits*

- **Max possible:** 4 bits/byte (32-byte hash)

Collision Test

- **0 collisions** across **1000 randomly generated 32-byte inputs**

Avalanche Effect

- Flipping a single bit in the input caused **72 out of 128 bits** to flip in the output

Bit Independence

- **Avg deviation from 50% bit distribution:** *6.21 bits*
- **Max deviation:** 23 bits

These results, while still improvable, showcase early evidence of desirable cryptographic properties like diffusion and randomness—even at the simulation level.

Project Structure

```
quantum_hash_project/
│
├── analysis/                  # Contains all hash analysis scripts
│   ├── test_entropy.py        # Entropy preservation tests
│   ├── test_collisions.py     # Collision checks
│   ├── test_avalanche.py      # Avalanche effect analysis
│   ├── test_bit_independence.py  # Bit-independence criterion
│
├── quantum_hash/              # Core implementation
│   ├── __init__.py            # This should exist (even if empty)
│   ├── circuit_builder.py     # Builds the parameterized quantum circuit
│   ├── input_encoder.py       # Compresses and encodes input into parameters
│   ├── hash_core.py           # Ties everything into a working hash function
│
├── main.py                    # Input/output runner
├── requirements.txt           # Qiskit + numpy + matplotlib
└── README.md                  # Project overview
```

Visual Results

To support the findings, I generated graphs and diagrams:

- A full **quantum circuit diagram** used in the hashing process
- **Entropy distribution histograms**
- Internal test results visualized for reporting

Technologies Used

- **Qiskit** for quantum circuit simulation
- **Python** (3.11) and **NumPy** for analysis
- **Matplotlib** for data visualization

Way Forward

Mark-1 is a **first prototype** in a growing space. Here's what's next:

- Move from statevector simulations to **measurement-based outputs**
- Add **post-processing layers** to improve statistical uniformity
- Run experiments on real **IBM quantum hardware**
- Extend to **Mark-2**, incorporating dynamic circuits and quantum randomness

Conclusion

Mark-1 may not be battle-ready for secure blockchain protocols today—but it opens the door to how we might **build cryptography for a quantum-first world**.

If you're passionate about quantum computing, cryptography, or even the intersection of classical and quantum tech—feel free to connect or collaborate. Let's build what's next.