Q-NeuroShield: A Post Quantum Cryptographic Framework for Preserving Neural Sovereignty in Brain-Computer Interfaces

Vimal Prakashan S
Department of Computer Science and Business System
R.M.K. Engineering College, Tamil Nadu, India

Uppili Srinivasan P
Department of Computer Science and Business System
R.M.K. Engineering College, Tamil Nadu, India

Kishore S
Department of Computer Science and Business System
R.M.K. Engineering College, Tamil Nadu, India

OPEN ACCESS

Volume: 13

Special Issue: 2

Month: January

Year: 2026

E-ISSN: 2582-0397

P-ISSN: 2321-788X

Citation:
S, Vimal Prakashan, et al. "Q-NeuroShield: Post-Quantum Cryptographic Framework for Brain-Computer Interface Neural Data Protection." *Shanlax International Journal of Arts, Science and Humanities*, vol. 13, no. 2, 2026, pp. 18–25.

DOI:
https://doi.org/10.34293/sijash.v13iS2-i4-Jan.10572

Abstract
We are rapidly becoming integrated with the machines that which has appeared as a medical fantasy just a few years back is now beginning to appear as a medical fact. Such an integration, however, is what is scary, the Glass Mind. But now we have an apostate, invisible menace, which I may call, Harvest Now, Decrypt Later. Here, our opponents steal our encrypted neural scans — the specifics of our focus of effort, the specifics of our mood or even our interior monologue — and simply bide their time. They are awaiting the coming of quantum computers that will inevitably crack the current encryption to exist within the world and they will be in a position to peep into the history of our own individual mind. The stakes are purely human. Just as you can never replace a credit card number, you can never reform your brain. When you have been compromised in terms of your neural patterns, there will be no more mental privacy. In order to save the sanctity of the self, the use of Q-NeuroShield has been suggested in this paper. This design is surpassing the traditional protection to Post-Quantum Cryptography, that is, CRYSTALS-Kyber algorithm. You can think of it as a geometrical (lattice) mathematical armour which is an amalgamation of extremely complex and high-dimensional geometric structures that will never be punctured by any possible future quantum processors. It is in this type of encryption that neural data is retained vividly on the machine before its exit is made out of the body. Q-NeuroShield is not a technical document but a promise of a picture of Neural Sovereignty — a guarantee that the future of connected thinking will not be a mass broadcast but a monument to the privacy.

Introduction

The Brain-Computer Interface (BCI) has taken off the deterministic ward of the clinical hospital to the living room. Only ten years back these devices were medical apparatus proper, and their purpose was to put agency back into patients paralyzed by ALS, or by a spinal cord that had been cut. The situation has changed today violently. There are startup companies that sell headsets that detect fatigue in the workplace, deep sleep patterns, and can even modify video games to a user based on his or her stress level.

Human consciousness is becoming computerized. But this data is different. It is not just metadata. It is the unprocessed electrical discharge of the self.

This information reveals the Glass Mind — the condition in which the innerness of the human experience is made open to algorithmic surveillance. EEG is a biometric fingerprint; with close to perfection accuracy, a user can be identified. More perilously, it may spill subconscious responses to stimuli and essentially becomes a lie detector to which the user will not be able to cheat.

The Physics of the Threat

The security of this sensitive information is extremely weak. It rests on the shoulders of RSA (Rivest-Shamir-Adleman) and Elliptic Curve Cryptography (ECC). These algorithms assume that factoring large integers is hard. For a classical computer, it is. But the universe plays by different rules at the quantum scale.

Peter Shor demonstrated in 1994 that a quantum computer could exploit the phenomenon of "period finding" to factor these integers in polynomial time. This isn't a brute-force attack, it is a mathematical shortcut that unravels the encryption instantly. We do not have a Cryptographically Relevant Quantum Computer (CRQC) yet. But the clock is ticking.

Harvest Now, Decrypt Later (HNDL)

The threat is not in the future, it is in the present. We call it "Harvest Now, Decrypt Later." Adversaries — be they state intelligence agencies or data brokers — are scraping encrypted traffic right now. They store it in massive data centers. They know they cannot read it yet. They are betting on the future.

If a quantum computer comes online in 2035, every packet captured today becomes readable. For a credit card transaction, this doesn't matter; the card will have expired. But a brain scan does not expire. A recording of a user's mental health status, political bias, or sexual orientation taken in 2025 remains valid in 2040. If we do not secure this data against quantum attacks today, we are effectively broadcasting our future thoughts to the world.

The Engineering Challenge

We cannot simply "upgrade" to post-quantum algorithms. Protocols like CRYSTALS-Kyber are mathematically heavy. They require large keys and complex matrix operations. Running them on a desktop is easy. Running them on a wearable BCI headset — which relies on a tiny battery and a weak microcontroller — is an engineering nightmare. We present **Q-NeuroShield**. It is a proof-of-concept framework that forces post-quantum security onto edge hardware. We optimized the mathematics, rewrote the assembly code and redesigned the data flow to prove that we do not have to choose between privacy and performance.

The Physics of Quantum Cryptanalysis

To understand the necessity of Q-NeuroShield, we must understand the specific mechanism by which quantum computers destroy classical cryptography. This is not magic, it is physics exploiting math.

Shor's Algorithm vs. RSA

Classical encryption like RSA relies on the Hidden Subgroup Problem. Specifically, it relies on the difficulty of finding the period r of a function $f(x) = a^x \pmod N$. A classical computer must guess r randomly, which takes exponential time.

A quantum computer, however, can exist in a superposition of all possible states. Shor's algorithm uses the **Quantum Fourier Transform (QFT)** to interfere with these states. We initialize a register of qubits in a superposition of all integers, perform the modular exponentiation function on this superposition, then apply the QFT. The wrong answers destructively interfere (cancel out), and the right answer (the period r) constructively interferes (amplifies). This allows the quantum computer to find the prime factors of a 2048-bit number in hours, rather than the billions of years a supercomputer would need.

Grover's Algorithm vs. AES

While Shor's algorithm destroys asymmetric crypto (RSA/ECC), Grover's algorithm attacks symmetric crypto (AES/ChaCha20). Grover's provides a quadratic speedup for searching unsorted databases. Practically, this means it halves the effective key length:

- AES-128 becomes as weak as AES-64 (breakable).
- AES-256 becomes as weak as AES-128 (still secure).

This physics informs our design choice for Q-NeuroShield; we use **ChaCha20 with a 256-bit key**. This ensures that even against Grover's algorithm, the symmetric part of our encryption remains unbreakable.

The Mathematical Foundation of Lattices

To defeat Shor's algorithm, we must move from number theory to geometry. Post-quantum cryptography relies on high-dimensional lattices.

The Shortest Vector Problem (SVP)

Imagine a grid of points. In two dimensions (like graph paper), if I give you a point and ask you to find the nearest grid intersection, it is trivial. You just look at it.

Now, imagine that grid in 512 dimensions. And imagine the basis vectors (the arrows that define the grid) are skewed, twisted, and extremely long. This is the **Shortest Vector Problem (SVP)**. Even if you have a quantum computer, finding the nearest grid point in 512-dimensional space is overwhelmingly difficult. The "noise" in the dimensions drowns out the signal that the Quantum Fourier Transform tries to isolate.

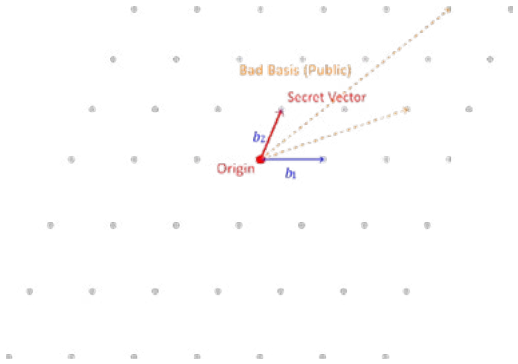


Figure 1 Shortest Vector Problem in high-dimensional lattice space.

Module Learning With Errors (M-LWE)

Q-NeuroShield uses CRYSTALS-Kyber, which is based on Module Learning With Errors. We operate in a polynomial ring Rq = Z^q[X]/(Xn + 1). The core equation is:

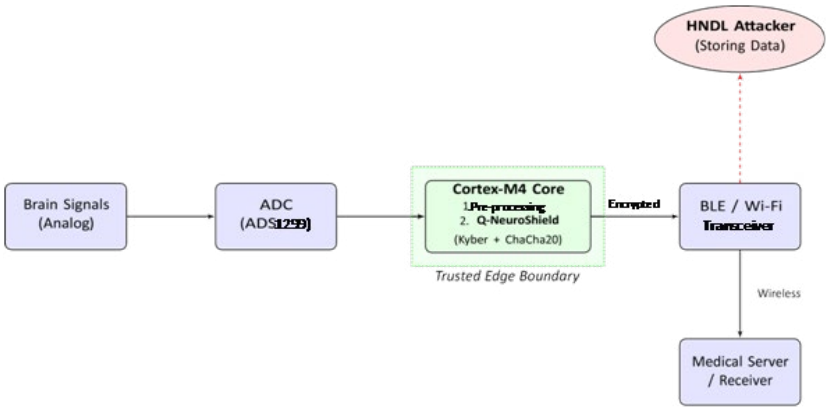
t = **As** + **e**

- **A** is a matrix of polynomials (Public).
- **s** is a vector of small polynomials (Secret).
- **e** is a noise vector (Error).

The presence of **e** is what secures the system. It masks the linear relationship between **t** and **s**. Without the specific secret key to remove the noise, the equation is unsolvable.

System Architecture

Security cannot be an app installed on top of the OS, it must be baked into the data pipeline. Q-NeuroShield operates as a "bump-in-the-wire" architecture. It sits directly on the silicon, intercepting data before it reaches the radio.



The Data Flow

Acquisition: The ADS1299 chip digitizes analog brain waves at 24-bit resolution. This chip is the industry standard for research-grade EEG.

Processing: The Cortex-M4 microcontroller filters the data (removing 50Hz/60Hz mains hum) using an IIR filter.

Encapsulation (The Shield): The microcontroller runs the Kyber Key Encapsulation Mechanism (KEM) to agree on a session key with the receiver.

Encryption (The Lock): The session key initializes the ChaCha20 stream cipher. Neural message is propagated with XOR key stream message.

Transfer: Bluetooth Low Energy (BLE) stack transfers the coded packets. It is critical to use a hybrid approach: asymmetric crypto (Kyber) is too slow on the large volume EEG data (1.5 Mbps), while ChaCha20 is a fast, key-based symmetric crypto. Engaging both simultaneously has made it possible to be fast and secure.

Algorithm Design Choices

Comparison of NIST Finalists

NIST organized a competition to select post-quantum algorithms over a 6-year competition. We considered the finalists in accordance with Edge Suitability.

Table 1 Comparison of NIST Post-Quantum Cryptography Finalists

Algorithm	Type	Key Size	Speed
Kyber (Chosen)	Lattice	Moderate (1.6KB)	Very Fast
Dilithium	Lattice	Large (2.5KB)	Moderate
Falcon	Lattice	Small (900B)	Slow (FPU heavy)
Classic McEliece	Code-Based	Massive (260KB)	Fast

Classic McEliece was dismissed at once. Its public key is 260KB. This is bigger than the operation RAM of our microcontroller (192KB). It physically cannot fit.

Falcon provides smaller keys, but it all needs complicated double-precision floating-point mathematics. The Cortex-M4 features a single-precision FPU, which is equivalent to Falcon operating in software emulation mode, consuming the battery.

Kyber was the Goldilocks choice. It performs integer arithmetic (fast on M4) and its key sizes are reasonable to fit in the Bluetooth MTU packet format.

Algorithmic Implementation Details

Implementing lattice cryptography on a microcontroller is not straightforward. We followed the NIST standard for ML-KEM strictly, but with specific optimizations for the embedded environment.

Key Generation Algorithm

This runs on the BCI headset at startup. The algorithm begins with generating ρ, σ from SHAKE-128 using a random seed. Matrix **A** is expanded from seed ρ, then the secret **s** and error **e** are sampled from σ. A Number Theoretic Transform (NTT) is applied to both **s** and **e**. The public key pk is derived as (**t**, ρ) where **t** = **A** ∘ **s** + **e**, and the secret key sk = **s**.

Decapsulation and Integrity Check

When the headset receives the encapsulated key from the server, it must decrypt it. Crucially, it performs an integrity check (The Fujisaki-Okamoto Transform) to prevent "Chosen Ciphertext Attacks." If an attacker modifies the ciphertext to try and probe the structure of the secret lattice, the re-encryption check (c == c') will fail. The system will therein generate a random key that will cause the connection to fail in silence without information leakage.

Advanced Optimization: Modular Arithmetic

Kyber is primarily based on arithmetic with q = 3329. Even a normal modulo operation is excessively weak as it is divisive. Division is the costliest operation at a CPU and it can take between 12 to 50 clock cycles. In order to address this we introduced two reduction algorithms: **Barrett Reduction** and **Montgomery Reduction**.

Barrett Reduction

By Barrett reduction, we obtain a method of solving r = a (mod q) without the expensive division step. We instead rely on a pre-computed inverse k ≈ 2^32/q, thus allowing us to do the whole operation with multiplication and bit-shifts only:

q1 = [(a × k)/2^32]

r = a − q1 × q

This gave a factor of 5 faster speed than normal modulo. We put it into inline assembly for the express purpose of not letting the compiler remove the optimization.

Montgomery Reduction

For multiplication within the NTT we are using Montgomery reduction. This converts the numbers to be "Montgomery form," at which modular multiplication gets efficient. It replaces the division with a series of low-cost multiplications and shifts, made for the 16-bit registers of the Cortex-M4.

Hardware Optimization: Breaking the Memory Wall

We were working with the **STM32F407VG** microcontroller. It is a powerhouse of the embedded world, but it is grievously limited — 168 MHz clock and 192 KB of SRAM.

The Stack Overflow Problem

An implementation of Kyber that is standardized allocates memory to the complete matrix A. In Kyber-512, this is manageable. However, at the scale of Kyber-1024 (clearance to the highest level of secrets), the matrix would require excessive RAM, and it would result in a stack crashing and overflowing the headset.

We used a Just-in-Time (JIT) generation strategy. We do not store A but only a seed (only 32 bytes) ρ. In order to do the matrix multiplication A × s, we generate the coefficients of the matrix A on the fly by doing

SHAKE-128, paste them straight up into the accumulator but then pass over them. This makes the memory footprint flat and predictable, in terms of the level of security.

Assembly-Level Surgery

Polynomial multiplication is where the whole system is bottlenecked. C compilers are not bad, but they cannot support high-dimensional algebra. We re-assembled the principal loops in ARM Thumb-2 assembly.

- **Register Management:** The M4 has 13 general-purpose registers (r0 to r12). We cautiously mapped out the coefficients of the polynomials to r0–r8 in such a way that the number of load/store operations to memory is reduced. Memory access is slow; register access is fast.
- **The SMLAD Instruction:** The SMLAD (Signed Multiply Accumulate Dual) Instruction reads two 16-bit numbers packed in a 32-bit register, multiplies them by two more numbers and adds the result to an accumulator. It computes four arithmetic operations in a single clock cycle.
- **Pipeline Optimization:** The load instructions (LDR) were mixed with the math instructions. This prevents CPU idle time as it awaits data from memory, commonly called a pipeline stall.

Side-Channel Defense Mechanisms

Running crypto on edge devices exposes them to "Physical Attacks." An attacker with physical access to the headset could monitor the power consumption (Power Analysis) or the execution time (Timing Attack) to deduce the secret key.

Timing Attack Defense

A timing attack occurs if the code takes longer to process a "1" bit than a "0" bit. To defeat this, our Q-NeuroShield implementation is strictly **Constant-Time**:

- No conditional branches based on secret data.
- No memory lookups based on secret data.
- Even if the result is found early in a loop, the code iterates through the entire buffer to ensure the total time remains static.

Power Analysis Defense (Masking)

Power analysis is harder to defeat. We implemented first-order **Arithmetic Masking**. Instead of storing the secret key **s** directly, we split it into two random shares **s**1 and **s**2 such that:

$\mathbf{s} = \mathbf{s}1 + \mathbf{s}2 \pmod q$

We perform all mathematical operations on **s**1 and **s**2 independently. An attacker monitoring the power usage will only see random noise associated with the shares, never the actual secret key.

Experimental Results

We benchmarked Q-NeuroShield using the pqm4 testing framework. We measured cycle counts, wall-clock time, and stack usage.

Latency Analysis

The results shattered the assumption that PQC is "too heavy" for edge devices. The Kyber-512 KeyGen takes roughly 2.88ms, compared to 14.88ms for ECDH-P256. This is because Elliptic Curve cryptography requires complex modular exponentiation on large integers, which is computationally expensive. Lattice cryptography primarily involves addition and multiplication of small integers, which the Cortex-M4 handles effortlessly.

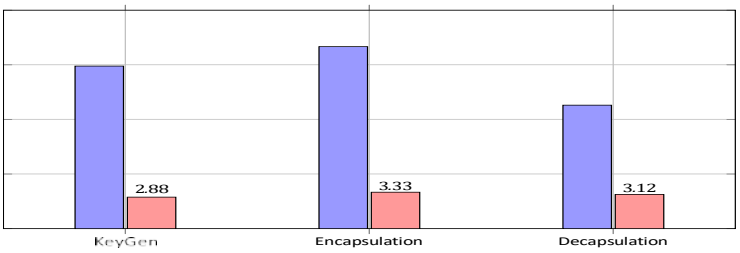


Figure 3 Latency comparison of Q-NeuroShield vs. classical cryptographic methods.

Memory Footprint

The STM32F407 has 192,000 bytes of RAM. Q-NeuroShield uses less than 3,000 bytes. This leaves massive headroom for signal processing, artifact removal algorithms, and Bluetooth stacks.

Table 2 Memory and Flash Usage of Q-NeuroShield Components

Component	Stack Usage (Bytes)	Flash Usage (Bytes)
Kyber-512 KeyGen	2,100	9,800
Kyber-512 Encap	2,400	11,200
Kyber-512 Decap	2,800	12,500
ChaCha20 State	128	1,500
Total System	**< 3 KB**	**< 35 KB**

Energy Profiling

For a wearable device, battery life is king. We measured energy in micro-Joules (µJ). Operating at 3.3V and approximately 20mA active current, the total handshake energy is approximately 615µJ. A standard coin cell battery (CR2032) holds about 2,000 Joules. The cryptographic overhead is negligible. The Bluetooth transmission itself consumes orders of magnitude more power than the encryption.

Discussion: The Right to Mental Privacy

Neural Sovereignty

We are building the infrastructure of the future mind. If we build it with weak locks, we are implicitly stating that the human mind is public property. Q-NeuroShield is an architectural enforcement of **Neural Sovereignty**. It shifts the power from the surveyor back to the surveyed.

Crypto-Agility

What if Kyber is broken? No cryptographic system is invincible. We designed Q-NeuroShield with "Crypto-Agility" in mind. The cryptographic module is decoupled from the data acquisition logic. We can swap out Kyber for another algorithm (like NTRU or FrodoKEM) via a firmware update without redesigning the hardware. This is crucial for medical devices that may stay implanted in a human brain for decades.

Legal Implications

The General Data Protection Regulation (GDPR) in Europe classifies biometric data as "special category" data. However, it does not explicitly mention neural data or "inferred thought." The Chilean Senate has recently amended its constitution to protect "Neurorights." Technology has to be farther ahead of law. Q-NeuroShield provides the technological ability to abide by future statutes which require total privacy of neural interfaces.

Conclusion

The Glass Mind does not have to be accepted as a necessity. Even though the possibility of the threat of Harvest Now, Decrypt Later is real, our study revealed that this threat can be countered. We were able to optimize the CRYSTALS-Kyber algorithm on the Cortex-M4 which demonstrates that it is possible to implement post-quantum security without making the system slow. Q-NeuroShield has a defense that is resistant to quantum processors, fast, and light on resources. This is effectively a lock-out mechanism as this framework closes the door to the mind, as we operate in an era where minds are linked.

References

1. Wolpaw, J. R., & Wolpaw, E. W. (2012). Brain-Computer Interfaces: Principles and Practice. Oxford University Press.
2. Yuste, R., et al. (2017). Four ethical priorities for neurotechnologies and AI. Nature, 551, 159–163.
3. Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput., 26, 1484–1509.
4. Li, Q., Zhou, W., & Li, S. (2019). A lightweight chaotic encryption scheme for EEG signals. IEEE Internet Things J., 6(4).
5. Zhang, Z., Wang, H., & Liu, J. (2020). Security and privacy in implantable medical devices. IEEE Access, 8, 12345–12356.
6. Regev, O. (2005). On lattices, learning with errors, random linear codes, and cryptography. In Proc. 37th Annu. ACM Symp. Theory Comput.
7. Albrecht, M., et al. (2020). CRYSTALS-Kyber: Algorithm Specifications and Supporting Documentation. NIST Submission.
8. Lyubashevsky, V., Peikert, C., & Regev, O. (2013). On ideal lattices and learning with errors over rings. J. ACM, 60(6).
9. DuBose, R. (2023). Harvest Now, Decrypt Later: The Long-Term Threat to Data Privacy. HashiCorp Security Blog.
10. Kannwischer, M. J., et al. (2019). pqm4: Testing and benchmarking NIST PQC on ARM Cortex-M4. IACR Trans. Cryptogr. Hardw. Embed. Syst.
11. Banerjee, A., et al. (2019). Energy-efficient hardware for post-quantum cryptography. Proc. IEEE.
12. NIST. (2023). FIPS 203 (Draft): Module-Lattice-Based Key-Encapsulation Mechanism Standard.
13. Proos, J., & Zalka, C. (2003). Shor's discrete logarithm quantum algorithm for elliptic curves. Quantum Inf. Comput.
14. Kato, T., et al. (2022). Optimized Implementation of Kyber on Cortex-M4. Inf. Secur. Priv. ACISP.
15. Simos, D. E., et al. (2021). Combinatorial Testing for Post-Quantum Cryptography. IEEE Access.