The foundations of programming: the approach of the developer behind Quantonia and QRACE

General explanation of the operating principles of the AMQIC International Scientific and Economic Consortium

Author: Aleksei Sergeevich Musienko

Scientific, Technical and Organizational Account

Bishkek, Kyrgyz Republic

16 November 2026

Participating scientists, developers and consultants

Aleksei Sergeevich Musienko — Quantum Information & Artificial Intelligence & Mathematical Physics Scientist, Chief Scientist & Head of AMQIC, Q-Developer, SEC Architect (Blue Team; defensive-focused Purple Team), PQC Architect.

Georgy Davidovich Latugov — Quantum Information & Artificial Intelligence & Mathematical Physics Scientist, AMQIC Staff, Scientist of AMQIC, Q-Developer.

Anton Konstantinovich Dobrylko — AMQIC Staff, Q-Developer, Quantum Quality Assurance Engineer (Q-QA).

Mikhail Bubnov — consulting and conceptual development in SEC, DevOps and DevSecOps.

Abstract

This article sets out the author's approach to building Quantonia and QRACE, defines the technical limits of the results presented, and explains the organizational and ethical principles of the AMQIC Consortium. It brings together an account of exact quantum-state simulation on a classical processor, the results of internal checks, information about the research materials used, and principles for subsequent application of the developments. It examines the M16 implementation for 1–16 qubits, exact algebraic arithmetic, checks of Grover's algorithm and the quantum Fourier transform, and separate studies of finite transformations and optimization of the VM computational counter. It distinguishes implementation, internal testing, a planned public evidence base, and independent external verification. The results described apply to the specified test conditions and do not imply universal quantum advantage. The organizational discussion explains the approach to scientific entrepreneurship, confidentiality, product accessibility, voluntary remuneration and the planned structure of operations. The article combines a technical description with the author's account of the Consortium's operating principles.

Keywords: Quantonia; QRACE; exact arithmetic; quantum computation simulation; Rust; finite monoids; scientific entrepreneurship; AMQIC; cybersecurity.

Preamble

This material was written by a human, with subsequent editing and clarification using artificial intelligence technologies. The final review of the content was conducted by a human.

The position presented is subjective. Some programmers and researchers may not endorse it: it is my personal position and philosophy, which I invite you to explore.

The article uses scientific and technical language and assumes familiarity with certain concepts in programming and mathematics. Readers who are not yet accustomed to this depth may use artificial intelligence to explain individual passages in simple terms and with everyday examples. Those explanations should be checked against the original text: simplification must not alter the meaning. Scientists and programmers can refer directly to the technical account and the cited sources.

I do not reinvent the wheel; I use and improve on humanity's best solutions.

1. Subject of the article and evidence base

This article concerns the relationship between Quantonia, QRACE and the Consortium's applied solutions, as well as the principles governing their development and use. Technical results are presented within the limits of the internal research and testing performed. An evidence base is being prepared for public release through GitHub and subsequent independent verification.

Publishing verifiable evidence does not imply consent to disclose the source code of proprietary technologies or internal documentation. A register of filenames and SHA-256 checksums for confidential materials appears at the end of the document. A checksum identifies the specific bytes of a document; by itself, it does not prove the truth of its contents, authorship, creation date or independent verification.

The quantitative figures below each measure a specific property. The precision of an algebraic representation, the probability of a quantum-state measurement outcome, computational counter consumption and actual execution time are different characteristics and must not be substituted for one another.

1.1. What the prefix “Q” means in our technologies

In our technologies, the prefix “Q” means that the underlying logic includes elements of quantum mathematics generated by QRACE (Quantum Reversible Adaptive Computer Engine) vM16, our quantum computation emulator. All modules in this technological system are written in a proprietary programming language designed to integrate the emulator into the standard Quantonia software environment; the computational engine itself is implemented in Rust [55].

In the Q-Calculator application, the control layer is implemented in Python and connects to QRACE for the relevant computations (Appendix A.3).

2. The heart of QRACE

QRACE — Quantum Reversible Adaptive Computer Engine — is the computational core of the system described here. The M16 implementation, also designated vM16(Q16), simulates quantum states on an ordinary processor and supports 1 to 16 qubits. For 16 qubits, the full vector contains 65,536 complex amplitudes. Internal source: Q4.

The relevant operations are based on exact algebraic arithmetic. Clifford+T uses a cyclotomic representation over Q(ζ8); supported fine-grained phases allow an extension up to Q(ζ65536). Coefficients are stored as integers with a common denominator of 2^k. Checked integer operations, BigInt representations and sparse structures are used; decimal values serve for output and numerical comparisons.

This approach avoids the accumulation of rounding errors in the supported exact operations. The canonical representation makes it possible to check state restoration byte for byte. Computations take place within the supported model, operation set and specified resource limits.

Operating constraints include limits on the number of operations and coefficient sizes, as well as execution-time limits for individual checks. The configuration considered here uses, among other constraints, a limit of 100,000 operations, a 4096-bit limit for the relevant coefficients and the exponent k, and separate limits for numerical analysis and the VM. A run that reaches a limit must be treated as resource-limited, rather than as a successfully completed computation.

3. Research sources and our own work

3.1. Materials on the Diophantine definability of integers in the rationals

One important research source is the ninth version of the manuscript “$\mathbb{Z}$ is Diophantine in $\mathbb{Q}$”, hosted in Elliot Glazer's repository [1]. The research context considered here also mentions Stefan Barańczuk, Jose Brox, Gabriel Conant, Sohail Farhangi, James Freitag, Jennifer Park and Caroline Terry; links identifying the researchers precisely are provided at the end of the document [3–10].

The external researchers mentioned in this section are not affiliated with the Consortium and have not reviewed this work.

The results of our additional research into these materials were used in developing QRACE's computational logic. The specific ways in which they were applied belong to the confidential implementation.

For bibliographic reproducibility, the exact version used is recorded, rather than the repository's changing current branch.

3.2. PELA and the representation of computations

Another source of research ideas is the VSA-TOE material, particularly the publication “PELA and Hilbert Space” [2]. These materials informed our own research into representations, finite transformations and the applicability of the corresponding constructions to software implementation.

The full vector of a 16-qubit state contains 65,536 amplitudes. In one representation, four arrays of i64 coefficients occupy 2 MiB, but this accounts for only part of the implementation's memory. Metadata, temporary structures and coefficient growth require additional memory. Peak logical memory volumes recorded by internal accounting for individual tests were 29,429,760 bytes for H16, 448,937,984 bytes for Grover201 and 159,426,768 bytes for QFT16. These figures do not measure process RSS or actual cache residency.

3.3. Separate Native-chain3 optimization

Internal source: T3. A separate two-qubit Native-chain3 experiment compared variants F and D for checking matrix representations. Consumption of the deterministic VM computational counter fell by 6.7824%, 7.4099% and 7.3592% in three configurations. Testing comprised 96 trials, 24 blocks and 1536 calls; word depths were 2, 4 and 4.

These figures characterize a specific experiment and a specific work counter. They do not represent a reduction in M16 quantum gate count or a demonstrated speedup of all programs. The criterion for a sustained improvement in actual execution time was not met; in configuration W2, variant F was approximately 5.3079% slower than variant D by the median. Further improvement is treated as engineering work, with each improvement tested separately.

3.4. A finite transformation space

Here, R4 is a five-element retract of the finite structure under study (the index 4 denotes its number in the classification of retracts, not its number of elements); End⁴(R4) is the result of four successive applications of the endomorphism-monoid operation, and C15 is its quotient by the ideal of maps whose image has at most 14 elements. Internal sources: M2–M5. Another research direction concerns finite endomorphism monoids and minimal generating sets. For the top quotient C15, obtained from End⁴(R4) by quotienting by the ideal of maps whose image has at most 14 elements, the order was established as 353,368 and the minimum cardinality of a generating set as 145,737.

For the family of inclusion-minimal generating sets investigated, the following polynomial was obtained:

P(z) = C · z^145737 · (1 + z), C = 2^5434 · 3^44 · 5^9 · 7^21.

The corresponding cardinalities of inclusion-minimal sets are 145,737 and 145,738. Minimum cardinality and inclusion-minimality describe different properties. These results must not automatically be extended to the full family, which has not yet been established, for the complete End⁴(R4), containing 105,923,880 elements.

The results were accompanied by internal computational verification and a separate internal examination. Independent external verification remains a separate stage in preparing the evidence base.

4. M16 checks

Internal source for the tests described: Q4.

4.1. Grover's algorithm

The Grover search test used 16 qubits, a single marked basis state with index 42301 and 201 iterations. All 65,536 amplitudes were checked against an independent recurrence formula.

The probability of the marked outcome was approximately 0.999988259646166562, or 99.998826%. The corresponding amplitude was approximately 0.99999412980585369134. These are the measurement-outcome probability and the state amplitude after the selected number of iterations.

At the interface level, the test includes 201 oracle calls, 201 diffusion operations and 16 H operations: 418 operations at this level in total. These cannot be treated as a count of elementary hardware gates without a separate decomposition. A continuous run and a run checkpointed after 100 iterations and resumed for another 101 iterations produced matching canonical representations of 56,033,355 bytes.

4.2. Quantum Fourier transform

The QFT input was the basis state with index 37. All 65,536 complex amplitudes were checked against the expression ζ65536^(37j)/256. After the inverse transform, restoration of the original state's canonical bytes was checked. This is a complete check of the output vector for the selected input, rather than an exhaustive check of the entire transformation matrix for every possible input.

4.3. Entanglement and ancillary qubits

Checks across the 8|8 cut produced the expected entropies: 1 bit for the GHZ state, 8 bits for the selected set of Bell pairs and 7 bits in the case with a cleared ancillary qubit. Additional checks covered zero amplitudes at the 32,768 odd indices for q15, correct ancilla return within the specified initialized subspace, and negative controls.

Taken together, these tests provide evidence of correctness for the specific scenarios checked. Preparing external reproduction requires recording the inputs, versions, expected results and run conditions.

5. Forms of QRACE use and the role of Quantonia

5.1. Executable tool

QRACE can be used through an executable command-line tool implementing the supported computational-core commands. On Windows, this form may be distributed as an .exe executable. A practical example is Q-Calculator, a local mathematical tool written in Python that connects to QRACE for selected computations. It combines classical symbolic and numerical operations, computation of π involving QRACE, and exact execution of Q16 circuits. Appendix A.3 provides a detailed description.

5.2. Library integration

qrace-sdk version 0.1.0 implements in-process integration with Rust: compilation, verification and execution in the VM; use of State, Budget, Gate and Poly objects; and supported checkpoints. Windows and Linux checks were performed for the corresponding implementation scope.

The Rust SDK is used for embedding. Cross-language wrappers and distributions with a stable external C ABI require separate interface definition and compatibility testing.

This foundation makes it possible to investigate adding new logic to applications and integrating it with other components. Use on amqic.ai and effects on the performance of individual paths must be assessed through specific implementations and measurements; the properties of the core do not imply an automatic speedup of the website as a whole.

5.3. Quantonia

Quantonia is the author's software layer for interaction with QRACE and the Rust infrastructure, described by the author as a programming language. Its role is to express computational logic and connect it to the core's capabilities. The language version, implemented constructs and availability of particular tools should be considered in light of their own documentation and current delivery status.

6. Possible applications and development directions

The mechanisms developed are intended for exact computational experiments, research into mathematical constructions and development of applied solutions within available resources. Classical simulation makes it possible to study ideal quantum circuits without a physical device's noise; however, the size of the full vector grows exponentially with the number of qubits. It therefore does not automatically replace quantum devices at every scale.

A second direction is software combining algorithms, representations and verification tools in new ways. Practical value depends on the problem being solved, result correctness, resource consumption and comparison with suitable classical methods. A third direction is protective mechanisms and cryptographic integrations, including the QCQM, QSM and Q-LIGENOR family. Quantum computation simulation does not, by itself, certify a cryptosystem's post-quantum security: that requires separate justification and analysis.

I regard the following tasks as directions for long-term work. The first is to build and investigate brain models that could be useful to specialists. Drawing clinical conclusions would require biological data, independent verification, medical research and the relevant authorization; no ready-made method of precise treatment is claimed here. The second is new controllers and software for technological industries, including space research, with mandatory examination of physical, computational and operational constraints. The third is combining logic and combinatorics to reduce the risk of technology misuse and protect the integrity of infrastructure and software.

Within the last direction, protecting creative products, including video games, is of particular interest, with a view to improving developers' chances of receiving fair remuneration. Technical mechanisms can help protect rights and make unlawful copying more difficult, but they do not make piracy absolutely impossible or replace the legal establishment of copyright.

7. Who we are and why we call ourselves scientists

We are ordinary people. We do not claim academic degrees, titles or specialist education that we do not have, nor do we attribute institutional appointments to ourselves. Our path involves independent learning, research, development and verification of results.

In our subjective view, being a scientist is a profession defined by the work performed. Its three main pillars are thinking, researching and creating. For us, this means asking questions, studying existing knowledge, building and testing solutions, recording results and stating the limits of their applicability.

The work we perform is the basis for our self-identification as independent researchers and scientists. Our public materials reflect the directions of our work; specific scientific and technical claims must be accompanied by an appropriate evidence base.

8. Legal status of the technologies

The proprietary developments are trade secrets and know-how of the Consortium, specifically its development division in the Kyrgyz Republic. They are subject to confidentiality and access controls. The public repository described in Section 15 is distributed separately under the MIT License.

The source code of the proprietary technologies is not intended for open publication. The chosen approach is to deliver finished solutions and introduce them progressively into industry products. Public verifiability of the results claimed and confidentiality of implementation methods must be reconciled in advance, with clear boundaries on auditor access.

9. Scientific entrepreneurship and the principle of mutual benefit

The Consortium focuses on scientific entrepreneurship. Before implementation, we seek to assess the economic consequences and choose approaches that benefit different participants. The Win/Win approach expresses an intention to consider the interests of partners, existing companies and workers, rather than regard their losses as a desirable means of achieving our own gain.

Our principle is to choose implementation methods that consider the interests of other sectors and do not aim to bankrupt companies or deprive people of jobs. To this end, we carry out economic analysis and apply the Win/Win approach.

These constraints make funding more difficult to obtain. We prefer to develop useful products in accordance with the Consortium's Fundamental Doctrine, Code of Ethics, Manifesto and Charter [12–15], rather than pursue profitable activities that conflict with our principles.

10. Funding, accessibility and remuneration

For the Consortium's public products, an intention has been formulated to direct 100% of the distributable profit attributable to the Consortium to science, volunteering and the development of computer technologies, including protective mechanisms. This refers to our own share after partners' shares and other obligations, rather than all revenue or third-party funds.

Procedures for determining the calculation base, covering expenses, meeting obligations, distribution and reporting remain to be developed. We plan to complete this work after the appointment of the Consortium's Authorized Representative for Administrative Management, jointly with that representative. Until the corresponding documents are approved, this provision expresses a direction for our activities, rather than an established financial procedure.

CERNOTIS solution services are provided free of charge: we deliver the report without payment, and the organization or developer may independently determine a voluntary remuneration amount according to their means. Accessibility matters to us for large companies and independent developers alike; differences in financial means should not determine respect for a recipient or the right to cooperation in good faith.

We seek to avoid discrimination, including on the basis of nationality. At the same time, any provision of solutions must comply with mandatory restrictions under applicable law. Individual terms of cooperation, scope of work and permitted uses are recorded separately.

We do not position the activities described as exchange trading, currency creation or an IPO. We do not take the place of financial institutions and operate within applicable rules. Similarly, we do not assume the powers of accreditation and certification bodies: developing a product and having an authorized organization formally confirm conformity are different actions.

11. Openness to audits

The Consortium is open to external audits and regards transparency as a foundation for lasting cooperation. The subject of an examination, the examining party's competence, the materials available and the means of publishing conclusions are agreed before procedures begin.

When trade secrets are involved, contractual confidentiality terms and examination boundaries are established. The procedure must allow the agreed claims to be checked while preserving protected

information.

12. The rule of law and jurisdictions

My interest in the foundations of jurisprudence includes the study of Roman and Ottoman law. This knowledge informs my legal philosophy; however, specific activities are assessed under current applicable legislation. If we identify conflicts, we seek clarification before taking the relevant actions, and if those conflicts cannot be resolved, we refrain from proceeding.

The Kyrgyz Republic has been chosen as the primary jurisdiction for the development division. Articles 41 and 42 of the Constitution of the Kyrgyz Republic concern economic freedom and freedom of work respectively [11]. They form part of the legal foundation but do not remove requirements concerning registration, contracts, taxes, intellectual property or other conditions of a specific activity.

The roadmap provides for a separate company in the State of Delaware, USA, to work with participants and undertake subsequent distribution. As of this edition, that company has not been established. The plan envisages contractual interaction between the development division in the Kyrgyz Republic and the future company, taking account of the distribution model chosen for each solution. This model does not provide for equity participation in the development division itself.

13. Freedom of choice and the boundaries of cooperation

For us, freedom of choice is mutual: we may decide whether to cooperate or decline within the law, and we recognize the other party's equal right to do so. We do not regard a partner's consent as an unconditional obligation to participate in every future project. Freedom of contract operates alongside lawful obligations already undertaken and other applicable rules.

We respect the sovereignty of states and institutions and choose forms and jurisdictions of cooperation in accordance with current legislation and agreed terms.

14. What we do not work on

The principal document defining our ethical and legal boundaries is the Fundamental Doctrine of Lawful and Responsible Use [12]. Its preparation drew on 663 legal sources from international law and national legal systems.

We exclude actions intended to harm humanity, offensive military development, assistance to unlawful activities and cybercrime. The Consortium intends to refuse such requests and, where required, act in accordance with the law.

15. Our contribution to accessible cybersecurity

We believe that basic cybersecurity tools should be available to developers regardless of team size or budget. The Consortium's practical contribution in this area is the free public AMQIC Base Cybersecurity for Development repository [40]. It contains reusable mechanisms, executable examples, infrastructure templates, release-preparation tools, laboratory checks and skills for working with artificial intelligence. This is a set of independent components for embedding in applications, web services, APIs, MCP integrations and container systems; it does not require adoption of the entire architecture as a single application.

The repository is distributed under the MIT License [41]. It permits use, copying, modification and distribution, including commercial use, provided the required copyright and license notices are retained. The copyright holder named in the license is amqic.ai. The software is provided “as is”; the license does not replace testing a specific integration. The account below describes the contents of the published version recorded when this edition was prepared, rather than claiming that every template is automatically installed for every user.

15.1. Login, authorization and sensitive operations

The server example organizes a login sequence: username, password, CAPTCHA, TOTP and session issuance. It includes one-time stages, attempt limits, authorization revocation and session handling. A login controller independent of the client framework protects the interface against duplicate submissions and late responses. Actual integration requires connecting the relevant providers and procedures for factor enrollment and access recovery [42].

Machine authentication uses Ed25519 signatures bound to the request, recipient, time, message body and nonce. A persistent replay store rejects requests presented again. A sensitive operation requires a current TOTP check and a one-time authorization bound to the specific action and arguments. Idempotent handling of repeated requests is distinguished from replay protection and from establishing the actual outcome of an external action. For API and MCP use, there is a fixed tool catalog, authorization checks and transport-integration contracts [42].

15.2. Input, database and field encryption

Web forms use a strict server-side schema, constrained Unicode input and parameterized data persistence. User values must not become SQL structure, shell commands or dynamically executed code; browser output is treated as text. These measures are considered alongside application permissions, network boundaries and container restrictions [43].

The PostgreSQL materials describe row-level access policies, tenant-bound database accounts, column permissions and explicit parameterized data selection. For sensitive fields, they provide AES-256-GCM envelope encryption with an interface to a KMS provider, authorization checks before key release, and authenticated binding to the tenant, row and field. Atomic nonce admission, consistent expiration and shared trusted state across service instances are described separately [43, 50, 51].

15.3. Network, server and containers

Infrastructure templates cover HTTPS, Host/SNI validation, blocking private paths, a register of sensitive URIs, Content Security Policy and rate limiting with 429 responses. The initial-deployment guide addresses proxying before public launch, closing unnecessary ports, key-based SSH and firewall rules based on administrative CIDRs. Allowlists distinguish the immediate network peer from the end-client address, which is reconstructed only through an explicitly trusted proxy [44].

Container materials describe a minimal image, an unprivileged UID, a read-only filesystem, and capability, network and resource restrictions. VPN materials cover WireGuard/WireSock and Tailscale, administrative-access restrictions and source-address validation. The Cloudflare guide describes an additional external layer: CDN, selective safe caching, cache bypass for sensitive responses, WAF, request limits, DDoS context, strict TLS to the origin and proxy trust. Neither a VPN nor an external proxy replaces application checks or configuration of the server itself [44].

15.4. File uploads, antivirus scanning and request admission

Upload handling separates the untrusted intake area, quarantine, antivirus scanning, transfer of approved content and serving that content. Different roles and volumes restrict access: a trusted transfer component checks the scanned bytes, while the serving service reads only approved content. The ClamAV materials address updates, scan coverage and failure behavior. Separate examples use private Redis, ACL/TLS, shared limiter state and checks that this state is preserved [45].

The laboratory request-admission example implements a bound, single-use proof of work, an SQLite replay store and a fixed Redis limiter. This is a specific admission mechanism, not a promise to eliminate every attack on availability. Production activation of these tools requires separate integration [45].

15.5. Secrets, dependencies, backups and recovery

Secret handling provides for hidden human input, encrypted storage and transmission of metadata alone to artificial intelligence. Isolation from a process with the full rights of the same operating-system user requires a separate trusted boundary. The KeePassXC guide covers KDBX, separate storage of factors and access recovery [46].

Dependency inspection separates package acquisition from subsequent offline examination, includes artifact inventory and makes decisions about specific bytes. Version-currency checks distinguish the installed version, pinned dependencies and dated official information: current branch fixes, support lifetime and available information about withdrawn releases. Private package names are not automatically sent over the network [46].

Backup tools operate on explicitly selected files, create a manifest, use AES-256-GCM and verify decryption and the contents of restored data. Restoration takes place in a new directory. A running database requires a consistent dump or snapshot, rather than arbitrary copying of its open files [46, 50, 51].

15.6. Release preparation, protection and verification

Production delivery preparation creates a separate clean directory, preserves the source tree and excludes Git/Graphify service data, internal instructions, configurations, tests, caches, backups and known secret filenames. Supported text formats are checked against known secret patterns, and a manifest is created. Such checks do not replace manual review or separate treatment of binary artifacts [47].

The binary-release protection materials distinguish symbol stripping, obfuscation and retention of private diagnostics. They provide version-bound examples for ELF, Go and Garble/R8, and .NET-specific considerations; a signature must match the final bytes. Obfuscation increases the effort required for analysis but does not make an embedded secret safe [47].

For official releases, the materials describe verifying the signed SHA256SUMS register through the AMQIC service, followed by comparison of each file's SHA-256. The VirusTotal scan register binds observations to exact hashes and distinguishes completed scans from pending ones. These observations do not guarantee the absence of every threat. Release and deployment guides cover delivery contents, versions, upload and readback verification, migrations, phased rollout, recovery and rollback; release publication and activation in the operational environment are tracked separately [47].

15.7. Laboratory, evidence and skills for artificial intelligence

The toolkit includes a laboratory action plan in JSON, a fixed local runner, a scenario catalog, a protocol schema and a description of host requirements. The reproducible laboratory combines an HTTP form, server-side validation of synthetic input, parameterized storage, file quarantine and transfer, the ClamAV protocol, a browser client and Redis limiter testing. A coverage inventory and claim register connect specific mechanisms with checks and limitations. External or manual provider checks are distinguished from local tests; the presence of a CI configuration does not, by itself, mean that CI has actually run [48].

The bundled skills cover protected input, backup with restoration verification, dependency examination, production-release preparation, version-currency checks, binary obfuscation and laboratory execution. A separate skill provides for consulting the external SandHive corpus to analyze threat scenarios. This is a recommendation of an independent resource, rather than an embedded executable SandHive component [48].

The documentation includes a mapping to selected OWASP ASVS 5.0.0 requirements and OWASP Top 10:2025 categories, as well as information on observations of protection of AMQIC's public website. The mapping is neither OWASP certification nor confirmation of an ASVS level; operational observations do not automatically establish the effectiveness of every integration [49].

Our position is to provide developers with accessible source tools, understandable explanations and verifiable examples that help them progressively strengthen their own systems. Free distribution and the ability to modify the materials make this contribution publicly available, while the documentation preserves the distinction between an implemented example, a configuration template and a measure that still needs to be integrated into a specific application [40, 41].

16. Conclusion

We support international scientific cooperation within applicable law and with mutual respect for participants' interests.

We are considering joint work on scientific and legal solutions that help reconcile security, accessibility and participants' interests. Our approach combines freedom of research, verifiable results, concern for the consequences of implementation and respect for the law.

If these ideas resonate with you, we are open to discussing cooperation within the boundaries outlined.

Appendix A. Product specifications

A.1. Quantonia

A software stack for developing and executing bounded computational tasks, including a language and related computational components. Rust components have been implemented and their integration tested. Capabilities depend on the specific execution profile. Quantonia is not a physical quantum computer. Internal sources: Q1–Q3.

A.2. QRACE / M16(Q16)

QRACE combines compilation, verification and execution tools for a typed language; M16 is a separate exact CPU backend for bounded quantum circuits. CLI and in-process Rust SDK 0.1.0 are supported. M16 operates over 1–16 qubits with explicit resource limits. The designation Q16 in this context must not be understood as a fixed-point format. Support for a stable DLL/C ABI and wrappers for every language is not claimed. Internal sources: Q2–Q4, T1–T2.

A.3. Q-Calculator

Q-Calculator is a local command-line mathematical tool combining symbolic and numerical operations, computation of π using QRACE, and execution of exact Q16 circuits. It runs on an ordinary processor. The control layer is implemented in Python; its mathematical dependencies are SymPy 1.14.0 and mpmath 1.3.0. A selected QRACE executable is connected for the relevant computations.

Its mathematical toolkit includes expression evaluation and simplification, factorization and expansion, equation solving and numerical root finding, differentiation, integration, limits, sums, series and matrix operations. Exact or symbolic results are distinguished from numerical approximations. If an operation remains unevaluated, that status is retained; finding a numerical root does not mean that all roots have been found.

For π computation, QRACE computes the leaf factors of the Chudnovsky algorithm, while Python combines large integers, estimates integer bounds and performs an independent check using Machin's formula. Q16 functionality supports circuits of 1–16 qubits, with up to 4096 gates per request. The result is accompanied by an exact-state file and its checksum; an abbreviated amplitude representation does not replace the full vector.

The tool provides a bounded request format, resource controls, records of run inputs and results, and a local library of worked mathematical problems with integrity checks for saved materials. The documentation identifies local dependencies and does not describe this copy as a portable public release. Internal sources: QC1–QC3.

A.4. AMRGIS

AMRGIS — Artificial Multiagent Recursive Generation Intelligence System: organization of recursive multi-agent work on bounded tasks. AMRGIS forms part of DOCTENSA's technological foundation. Related author abstract: “AMRGIS: Recursive Task Decomposition with Preserved Boundaries” [36].

A.5. AEGIS

AEGIS — Autonomous Evolution Generation Intelligence System: adaptation of task execution based on check results, current state and specified constraints. AEGIS forms part of DOCTENSA's technological foundation. Related author abstract: “AEGIS: Adapting the Agent Process Based on Check Results” [37].

A.6. DOCTENSA

DOCTENSA is a tool for managing artificial intelligence behavior based on AMRGIS and AEGIS. It organizes agent work, instruction transfer and result verification within the connected software environment and its authority.

DOCTENSA binds a task to instructions, permitted actions and result checks within the connected software environment and its authority. Its principles of use are defined in the Fundamental Doctrine of Lawful and Responsible Use [12]; the doctrine has been implemented in DOCTENSA, and post-quantum protection algorithms of the Vallatum QCQM module have been applied.

DOCTENSA engages specialists, combines their results and organizes verification in the context of the task. Related abstracts: description of the Fundamental Doctrine [34], DOCTENSA [35], AMRGIS [36] and AEGIS [37]. Internal source for the additional technical description: D1.

A.7. Vallatum QCQM

QCQM (Quantum Crypto QRACE Module) is a software module protecting executable policies and their use in a specified application context. Vallatum QCQM is its modification within Vallatum. The module runs on classical processors and combines cryptographic content protection with binding to a recipient and execution context. The term “post-quantum” describes the cryptographic component used; execution requires no physical quantum hardware [38].

The implemented container profile uses X-Wing (X25519 + ML-KEM-768), HPKE in PSK mode, HKDF-SHA256 and ChaCha20-Poly1305 [17–23]. X-Wing is pinned to Internet-Draft version 11; this is a draft specification, rather than a final RFC. Manifest signing uses the hybrid digital signature Ed25519 + ML-DSA-65 [24, 30]. QCQM provides a protected container and recipient/context binding, while QRACE executes the policy in a native environment. Loading checks the manifest signature and whether the components meet the specified access conditions. Content protection, provenance verification and decision computation remain related but distinct functions. No new cryptographic primitive is proposed. Internal profile sources: Q2, K1.

Fractal-recursive decomposition and reverse reconstruction. The fractal-recursive architecture provides for the following: key material is split into components, each receiving its own QCQM cryptographic protection. The same principle can be repeated within a composite component: decomposition, protection of the resulting parts and verification of their conditions of use. The construction has finite depth, determined by the chosen architecture. Reverse reconstruction of key material forms the reversible part of this process [38].

Admission to reconstruction uses a separate activation key, also protected by QCQM. Its purpose is to authorize access to the necessary components when the specified conditions are met. The activation key's protection relies on independent trusted storage and a separate admission procedure. Reconstruction therefore does not require an unreconstructed key to decrypt its own activation mechanism first.

An authorized protected consumer checks the components' purpose and access conditions, then reconstructs the key material in RAM for the authorized operation. An architectural rule limits its use to that operation and excludes intentional output of key values to logs, reports or ordinary files. Memory-access boundaries, the lifetime of temporary copies and their clearing are verified separately for each specific implementation.

“Fractal-recursive” denotes repetition of one architectural principle at nested levels. It does not assert a proven fractal dimension, infinite recursion or physical quantum hardware. Decomposition and additional levels do not, by themselves, create extra entropy or specify an increase in cryptographic security in bits. Security depends on the algorithms, profile, independence of admission mechanisms and implementation quality. Describing the principle does not mean that all its variants have been implemented, tested or activated in every QCQM delivery; the contents and status of a specific delivery are confirmed separately [38].

Integration functionality was confirmed by a local functional test: the native environment evaluated 55 rules of an existing module in a controlled application-call scenario. The module has been adapted for Windows, macOS and Linux. Ensuring that the interface is called remains the integrating application's responsibility; an administrator can still remove such a call. This test does not establish the cryptographic security of the entire system or replace independent security analysis [38].

A.8. Vallatum QSM

Vallatum QSM is a modification of the QSM product within Vallatum. In the specified profile, both ECDSA-P256-SHA256 and ML-DSA-65 checks are mandatory. Component standards: FIPS 186-5, SP 800-186, FIPS 180-4 and FIPS 204 [22, 24–26]. Source for the specific tested profile: K4. This is an application profile; using standard algorithms is not equivalent to FIPS product certification.

Tools for signing and verifying files and documents, linked to protected delivery. Integration and local checks have been implemented. The tested M16 signed-file profile used mandatory dual P256 and ML-DSA-65 verification; this fact applies specifically to that profile. Public verification of a particular file requires its actual registration and an available validation process. Internal sources: Q2, Q5.

A.9. Q-LIGENOR

The locally tested implementation includes the X25519MLKEM768, SecP256r1MLKEM768 and SecP384r1MLKEM1024 groups; one connection negotiates one group. Sources: TLS 1.3 [27], the hybrid exchange construction [28] and the group specification [29]. Ed25519 and ML-DSA-65 device signatures are considered separately from key exchange [24, 30]. In the profile described, both checks are mandatory through two nested TLS sessions. The raw public keys profile is specified for public-key transmission [31]. Internal sources: K1–K3, K6.

A component for device identification and protected inter-device communication. Locally implemented and tested profiles have been confirmed. Digital signatures and establishment of a shared connection secret perform different tasks and are assessed separately. Internal source: Q2.

A.10. QRAM

A computational component in the Quantonia family for agreed bounded software scenarios. Integration with M16 was checked within the scope of the corresponding report. Internal source: Q2.

A.11. Vallatum ALL55

Vallatum is a modification of ALL55: a module for connecting the Fundamental Security Doctrine to applications using artificial intelligence models. The SDK connects a model's proposal to a separate check against 55 principles before an action is executed and real consequences arise.

Checking an action in the context of a task. The application submits the user's original goal, proposed action, affected objects, possible consequences and participants' authority for examination. When a task is divided into subtasks, the link to the original request is preserved: the meaning of an intermediate operation is assessed together with the overall goal.

A conflict with applicable principles, insufficient material information or conflicting data requires the action to be held while the grounds for a decision are clarified. A current check is performed after circumstances change. Its result applies to the context examined; the application itself checks actual authority and execution conditions.

SDK integration. The integration principle described in the published abstract does not require changing model weights or additional model training. The SDK connects a checking layer to the application process and can be used with different models, provided the application routes every action requiring examination through it.

For example, artificial intelligence proposes changing a document. Before writing, the application submits the goal, the content of the change and the access context for checking. The prepared proposal, check result and change actually made are tracked separately. If the execution outcome is unknown, the operation's state is established first so that repetition does not create an unwanted duplicate effect.

Module protection and traceability. The QCQM cryptographic system protects the doctrine module's contents and its use in a specified context. Provenance verification, protected-policy integrity and assessment of an action's permissibility address different tasks and are considered separately. The QCQM abstract describes the cryptographic system in detail [38].

To support examination of decisions, the relationship between the original goal, context, grounds for checking and actual execution is preserved. The completeness of this control depends on which actions the application actually passes to the SDK and on the accuracy of the information supplied.

Vallatum ALL55 is disseminated and distributed through certified and accredited cybersecurity companies.

Related abstracts: the principle of doctrine integration through an SDK [34], QCQM hybrid cryptographic protection of executable policies [38], and the conceptual traceability model [39]. The doctrine text appears in the list of legal documents [12]. Internal source for the ALL55 technical mechanism: D2.

A.12. CERNOTIS

CERNOTIS is a software tool for structural analysis and quality assessment of software systems, source-code repositories, and server and network infrastructure. Its methodology provides for decomposition of the subject into components, identification of their dependencies and analysis of their interactions. Its aim is to identify potential logical errors, performance limitations and cybersecurity weaknesses, followed by a report containing observations and grounds for further investigation. Combined use with Graphify provides a graph representation of the subject's structure and the relationships between its elements [52, 53].

Main functions:

• Structural visualization of software-system nodes, selected code fragments and dependencies between them.

• Quality assurance (QA) of individual components and their interactions: analysis of execution logic, relationship consistency and factors affecting performance.

• Cybersecurity analysis of source code and configurations, including the presence and use of cryptographic protection mechanisms.

• Server and network infrastructure checks cover more than 800 parameters. These include the operating system, Origin/Nginx configuration, installed programs and commands used.

The Consortium independently examines public repositories. Server and network infrastructure assessments are conducted with the owner's permission and only through certified and accredited cybersecurity companies acting as the principal service providers.

Distribution and remuneration. Solution services are provided free of charge. We deliver the report without payment; the organization or developer may independently determine voluntary remuneration according to their means.

Internal source: C1.

All of the Consortium's operational solutions and products are integrated into all technologies, systems and devices we use and are applied in our daily work. Separately identified development plans are not described as features already implemented.

A.13. Planned OpenPGP adapter

A versioned OpenPGP adapter based on RFC 9980 and RFC 9580 is planned separately [32, 33]. This plan does not establish a completed implementation or tested compatibility. Matching cryptographic primitive names in a container, TLS and OpenPGP do not make their formats interchangeable. Internal source: K5.

A.14. QDMO

QDMO — Quantum Dynamic Memory Obfuscation — is a product in the Vallatum family with a native Rust component for managing protected data in an application's own memory.

Its functions include encrypted data representation between accesses, integrity and associated-context checks, brief controlled access, opaque handles with generation checks, managed relocation of component-owned objects and rotation. It provides memory allocation and locking, clearing of its own buffers and explicit refusal when the chosen profile's mandatory requirements cannot be met. Relocation applies to the component's managed objects, rather than arbitrary address space or other processes. Internal sources: QM1–QM2.

Bounded computational admission through PoW uses a single-use challenge bound to an object, channel and epoch, with a specified expiration and cancellation support. It does not replace authentication and authorization checks. A bounded local queue transfers data between stages; restoration of the original bytes is checked separately.

XChaCha20-Poly1305 is used for encrypted memory representation. The XChaCha extension is described in the expired Internet-Draft draft-irtf-cfrg-xchacha-03 [54]; the underlying ChaCha20 and

Poly1305 are described in RFC 8439 [23]. PoW uses SHA-256 [22], while challenge, expiration and consumption rules are defined by the application profile.

Integration with QCQM. In the tested library adapter, QDMO supports handling of sensitive data within its own execution node, while QCQM provides a protected container and binding to the expected context. The chain includes computational admission, in-memory processing, container protection, ciphertext transfer, context checking and reverse transformation. The QCQM container algorithms are described in A.7 and are not replaced by QDMO's memory-encryption algorithm. The tested adapter preserves the original key identity. Internal source: QM3.

Integration with QRACE/M16 includes a mathematical computation stage on a classical processor and a check of exact restoration. Subsequent M16 integration preserves the QDMO, PoW and channel contracts. The presence of reversible computation is not independent evidence of additional cryptographic security. Internal source: Q2.

Implementation status. The Foundation 0.2.0 test record of 9 November 2026 covers Windows x64 and Linux x86_64: on each platform, 47 library tests, 6 quantum-example tests, 4 native-worker tests and 16 scenarios for the existing Foundation C ABI. A separate QCQM-QDMO 0.1.0 test record reports, for Windows/Linux, 10 protected-interface checks, 2 quantum-state tests, 1 original QCQM test with a missing or incorrect PSK, 6 PoW checks and a successful complete example using X-Wing. These are the results of two separate test records. The library delivery includes Cargo sources and a lock file, documentation, examples and a toolchain-bound Rust rlib; this test record does not establish a new standalone C ABI/DLL for QCQM-QDMO. Internal sources: QM1, QM3.

Protection boundaries. Keys and necessary plaintext data exist in memory during use. No protection is claimed against every privileged read by an administrator, kernel or hypervisor, or against code within a compromised process. The lifetime of temporary copies, clearing and actual protection coverage are confirmed for each specific delivery.

A.15. Preparation of the public verifier

The public verifier is in preparation. A completed public API, final installation and end-to-end acceptance of the client package are not claimed. Artifact signing, local verification and public-service availability are separate states.

Materials for preparing the public verifier distinguish checks of artifact provenance and integrity, expected context and admission, execution of a bounded request, result verification and termination of the task-owned worker. The numerical worker and the separate checking tool have different roles.

The prepared independent Python checking tool reads every line of the canonical state, checks exact normalization, complex amplitudes and properties specified by the particular test. It does not import the proprietary numerical or protective implementation. Its purpose is to verify the result, rather than disclose QCQM's or QDMO's internal mechanisms. The available materials document local and test-environment checks, including recovery of a synthetic QCQM capsule and negative cases. Internal sources: PV1–PV3.

Appendix B. Register of internal documentary sources

Only filenames and the SHA-256 hashes of specific versions are provided. File contents are not published. The identifiers distinguish documents with identical names; local paths, keys and confidential mechanisms are excluded.

Q1. README.md

`SHA-256: 01ae4b1bcba09b31def9e71a8bd9bde26b581141cb14d14768a50ef31f6c8269`

Q2. RESULT-RU.md

`SHA-256: 7c26c3cf6215dfcabb260ab3f5c1025fedd1170af82869755580fe04027652c5`

Q3. SDK-READINESS.json

`SHA-256: 4ac936894f8b0c29e8610b871c1eead64ccf2b0c408e7d6cdb9c4c5974010764`

Q4. RESULTS-RU-EN.md

`SHA-256: aa55b3d867c1a34a35494790e4d8d2c13f2c08a3745602df7ed0713d9f1f1d7b`

Q5. QSM-COMPATIBILITY.md

`SHA-256: 2772587c7947aafd9d5567139b13cefe71c1f93432e1f4873392eac9fa9397c8`

D1. DOCTENSA-GUIDE.md

`SHA-256: 621fc80071d67a2f2f98c738d556284ae4abcae7123857ed4f9f9e823e872871`

D2. ORDINARY-ALL55.md

`SHA-256: 1e51a460bae2436a9aa61d40cd08cbb436ad7c50b783e45823e64daaa0c0ffe3`

C1. README.md

`SHA-256: b0c0a07dd606824c3c8b3f29448399201c5bc64d61aab3671a4208251d4c9930`

M1. V9-AUDIT-RU.md

`SHA-256: 085b9c56fe13a537abee8fa8306aa1a0cd2a3ea648dbbb2b2daeb46e9188abfd`

M2. RESEARCH.md

`SHA-256: 445cb46d1e4b759ba4b4cbcd6d38770b23a29ebbdcfc8114190a27c3b88a0e0e`

M3. README.md

`SHA-256: a7b13d935670ca4496e9f29290078c90a820880b2526c8f5b3b3c308a3d9b9f5`

T1. QRACE-SDK-RU.md

`SHA-256: c78927a784eff38622c0490d8bfa22e5c9c11142d65223dda8af1b36a46157d2`

T2. QRACE-CLI-RU.md

`SHA-256: 2cdba9e89787acf208138d2a3cf9546e822938699bb9574ec0c22642e8760ad9`

T3. NATIVE-CHAIN3-RESULTS-RU-EN.md

`SHA-256: 2d86de5ce031b414d589f3deea543398a55af7344cae9369ec14426799bd19ff`

T4. PELA-APPLICABILITY.md

SHA-256: 7e990f58561b8954c80e08ef8af7d480ec22183037e6910bb6b867ea9ee1c190

K1. ABSTRACTS-RU-DRAFT.md

SHA-256: 7b8de13403344c46d81012860b7b24f619df83487b8e7bbe2d7cbb60de4ba456

K2. IMPLEMENTED-RU.md

SHA-256: f756411da52dbb82198f505882336ecb10cd12473dd177b59f6a5ffebef98433

K3. CURRENT.json

SHA-256: 78a42152c1e9f45961b68312dc64fbf10a3d6a814de8f17477588cffc17b30d3

K4. ROOT-34-PROOF-READBACK-g1.json

SHA-256: b6a05140f0fe2296c12d204bc12d90f691c8ff5dcc07f9a1389d163b21a9db24

K5. RFC9980-SPEC.md

SHA-256: 1eb3e5976e745611854c11ba3cd4bd45cfce51d409ffb164f9bfe27bcb75b7c1

K6. transport.rs

SHA-256: 8dde5e9aef8ae26826d213530f0850040febda018c963c8ac0fbb72389985d04

QC1. README.md

SHA-256: 1edbc09afc5da9512fe5315db5a94d495a1a1b0a2791c705533c838c29db4ba3

QC2. backends.json

SHA-256: a409dcd4bfad945dd6bee35551db3f3c1dd6510e250da07d8fa782c3833fcc48

QC3. requirements.txt

SHA-256: 47fd19021144e071cc43b6e265aafada0056abe89724ad95bd6e3a8e37995b94

QM1. RESULT-RU.md

SHA-256: 349a5a0e47af8c25d8a200a1569bd35a43a8fe89f190a0f39bfde1ef0fdbb3d7

QM2. ARCHITECTURE-RU.md

SHA-256: df650a0cf8ccf6cedad76a42d0745eff80a649d9e3dc9c2f40d2845afa6c43e8

QM3. RESULT-RU.md

SHA-256: 16892fef89d43c840c3bfbf74796524a7d3615cc6280ed9304e858b35731ecce

PV1. README.public.proposed.en.md

SHA-256: 9d8217fe037c5e6aa9fd8560845293dbe0ac474950fa6b26ac875e343e610a08

PV2. VERIFICATION-PROTOCOL.public.proposed.en.md

SHA-256: eac796dda780ea24516da95c5cf3c115dee6bb2ad7c293782bd31b6aad70675e

PV3. OWNER-HANDOFF-r3.md

SHA-256: 8b2e525563b2cabbea836ed532d61344d92b04473634725a0309890a9e466f8d

M4. INDUCED-RETRACTS.md

SHA-256: fc6807a3b548d7a1070e1ab01b27224ff2269f421abd000769aeed057e179ecf

M5. FOURTH-END-MONOIDS.md

SHA-256: c4faf7e546ec71fa8ae4d6e8ad479d7ecc1cf2cb01f3ba5b4aeb49df5f3e5d7d

References

Public sources. Accessed: 16 November 2026. Internal documents are listed separately in Appendix B.

[1] Z is Diophantine in Q. Version 9. 7 November 2026. Repository elliotglazer/diophantine-z-q; commit 036cebe4f1a4aa7ca89820a9b48f3629b4c364d6; manuscripts/quartic_v9.tex. The author field in the historical source file is blank. URL: https://github.com/elliotglazer/diophantine-z-q/blob/036cebe4f1a4aa7ca89820a9b48f3629b4c364d6/manuscripts/quartic_v9.tex

[2] VSA-TOE. PELA and Hilbert Space. Cognitive's Substack. The author uses a pseudonym; no publication date is given. URL: https://vsatoe.substack.com/p/pela-and-hilbert-space

[3] Elliot Glazer. Public profile and repository of research materials. URL: https://github.com/elliotglazer/diophantine-z-q

[4] Stefan Barańczuk. Personal page on the Adam Mickiewicz University website. URL: https://stefbar.faculty.wmi.amu.edu.pl/

[5] Jose Brox. Research profile at CMUC, University of Coimbra. URL: https://cmuc.mat.uc.pt/rdonweb/person/ppgeral.do?idpessoa=1373

[6] Gabriel Conant. Personal page at the University of Illinois Chicago. URL: https://homepages.math.uic.edu/~gconant/

[7] Sohail Farhangi. Personal research page. URL: https://sohailfarhangi.com/

[8] James Freitag. Personal page at the University of Illinois Chicago. URL: https://homepages.math.uic.edu/~freitag/

[9] Jennifer Park. Profile at the Department of Mathematics, The Ohio State University. URL: https://math.osu.edu/people/park.2720

[10] Caroline Terry. Profile at the University of Illinois Chicago. URL: https://mscs.uic.edu/profiles/caterry/

[11] Constitution of the Kyrgyz Republic. Official website of the Cabinet of Ministers of the Kyrgyz Republic; Articles 41 and 42. URL: https://www.gov.kg/ru/p/constitution

[12] AMQIC. Fundamental Doctrine of Lawful and Responsible Use. Version 2026-10-25.1. URL: https://amqic.ai/api/v1/public/files/legal-documents/fundamental-security-doctrine/file?locale=en

[13] AMQIC. AlexMus Quantum Information Consortium Code of Ethics. Version 2026-10-22.3. URL: https://amqic.ai/api/v1/public/files/legal-documents/ethics-code/file?locale=en

[14] AMQIC. Consortium Manifesto. Version 2026-10-24.2. URL: https://amqic.ai/api/v1/public/files/legal-documents/consortium-manifesto/file?locale=en

[15] AMQIC. Charter of the Multifunctional Scientific and Economic Consortium. Version 2026-10-23.1. URL: https://amqic.ai/api/v1/public/files/legal-documents/consortium-charter/file?locale=en

[16] VSA-TOE. The Axiomatization of The Prism Echo Lattice Algebra (PELA) / The Canonical Axiomatization of the Prism Echo Lattice Algebra. 4 October 2026. Cognitive's Substack. URL: https://vsatoe.substack.com/p/the-canonical-axiomatization-of-the

[17] X-Wing: general-purpose hybrid post-quantum KEM. Internet-Draft, draft-connolly-cfrg-xwing-kem-11. 23 September 2026. URL: https://datatracker.ietf.org/doc/html/draft-connolly-cfrg-xwing-kem-11

[18] RFC 7748. Elliptic Curves for Security. URL: https://www.rfc-editor.org/rfc/rfc7748.html

[19] NIST FIPS 203. Module-Lattice-Based Key-Encapsulation Mechanism Standard. URL: https://csrc.nist.gov/pubs/fips/203/final

[20] RFC 9180. Hybrid Public Key Encryption. URL: https://www.rfc-editor.org/rfc/rfc9180.html

[21] RFC 5869. HMAC-based Extract-and-Expand Key Derivation Function. URL: https://www.rfc-editor.org/rfc/rfc5869.html

[22] NIST FIPS 180-4. Secure Hash Standard. URL: https://csrc.nist.gov/pubs/fips/180-4/upd1/final

[23] RFC 8439. ChaCha20 and Poly1305 for IETF Protocols. URL: https://www.rfc-editor.org/rfc/rfc8439.html

[24] NIST FIPS 204. Module-Lattice-Based Digital Signature Standard. URL: https://csrc.nist.gov/pubs/fips/204/final

[25] NIST FIPS 186-5. Digital Signature Standard. URL: https://csrc.nist.gov/pubs/fips/186-5/final

[26] NIST SP 800-186. Recommendations for Discrete Logarithm-Based Cryptography: Elliptic Curve Domain Parameters. URL: https://csrc.nist.gov/pubs/sp/800/186/final

[27] RFC 8446. The Transport Layer Security (TLS) Protocol Version 1.3. URL: https://www.rfc-editor.org/rfc/rfc8446.html

[28] RFC 9954. Hybrid Key Exchange in TLS 1.3. Informational. July 2026. URL: https://www.rfc-editor.org/rfc/rfc9954.html

[29] RFC 10024. Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3. Standards Track. August 2026. URL: https://www.rfc-editor.org/rfc/rfc10024.html

[30] RFC 8032. Edwards-Curve Digital Signature Algorithm (EdDSA). URL: https://www.rfc-editor.org/rfc/rfc8032.html

[31] RFC 7250. Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). URL: https://www.rfc-editor.org/rfc/rfc7250.html

[32] RFC 9980. Post-Quantum Cryptography in OpenPGP. Standards Track. June 2026. URL: https://www.rfc-editor.org/rfc/rfc9980.html

[33] RFC 9580. OpenPGP. URL: https://www.rfc-editor.org/rfc/rfc9580.html

[34] AMQIC. Fundamental Doctrine for the Safe and Responsible Use of Artificial Intelligence Models. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-354712f2-7bef-49ca-9424-f80ecbea233a

[35] AMQIC. DOCTENSA: From Logical and Combinatorial Models to Observable Artificial Intelligence Execution. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-de9681ba-9007-4b5e-b974-fd20cd90df11

[36] AMQIC. AMRGIS: Recursive Task Decomposition with Preserved Boundaries. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-3fd79b1e-4d82-499f-a019-a712d52fde02

[37] AMQIC. AEGIS: Adapting the Agent Process Based on Check Results. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-ed0aa328-6d5b-4070-b429-dec6597de981

[38] AMQIC. QCQM: Hybrid Cryptographic Protection of Executable Policies. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-0abdfb9b-3dfd-4b37-9315-aa55aaaf6ae8

[39] AMQIC. From Normative Doctrine to Verifiable Governance of Scientific and Artificial Intelligence Technologies: A Conceptual Traceability Model. Author abstract. URL: https://amqic.ai/?lang=ru#content-heading-a1ad6301-7e29-4bd8-b3ad-01ece8fe1850

[40] AMQIC Base Cybersecurity for Development. Public repository; pinned revision a78dc703c117810607a76dcf9714a0bfcddef926. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_de

velopment/tree/a78dc703c117810607a76dcf9714a0bfcddef926

[41] MIT License. Copyright (c) 2026 amqic.ai. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/LICENSE

[42] Authorization, Sessions, Machine Authentication and API/MCP: Example Documentation. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/reference/README.md

[43] Database and Sensitive-Field Security. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/DATABASE-SECURITY.md

[44] Infrastructure and Server Boundaries. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/INFRASTRUCTURE.md

[45] Service Protection, Scanning and Shared Request Admission. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/SERVICE-PROTECTION.md

[46] Dependencies, Secrets and Backups. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/OPERATIONS.md

[47] Production Release Preparation. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/PRODUCTION.md

[48] Complete Local Laboratory. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/ALL-LAB.md

[49] OWASP Mapping. URL: https://github.com/MrMarshalRF/amqic_base_cybersec_for_development/blob/a78dc703c117810607a76dcf9714a0bfcddef926/docs/OWASP-MAPPING.md

[50] NIST FIPS 197. Advanced Encryption Standard. URL: https://csrc.nist.gov/pubs/fips/197/final

[51] NIST SP 800-38D. Galois/Counter Mode. URL: https://csrc.nist.gov/pubs/sp/800/38/d/final

[52] Graphify Labs. Graphify. Official website. Accessed: 16 November 2026. URL: https://www.graphify.com

[53] Graphify Labs. graphify. Official source-code repository. Accessed: 16 November 2026. URL: https://github.com/Graphify-Labs/graphify

[54] XChaCha: eXtended-nonce ChaCha and AEAD_XChaCha20_Poly1305. draft-irtf-cfrg-xchacha-03. Expired Internet-Draft, not a final RFC. URL: https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-xchacha-03

[55] Musienko A. S. Explanation of the “Q” Prefix, the QRACE/Quantonia Architecture and the QCQM Module. Author publication on LinkedIn. Accessed: 16 November 2026. URL: https://www.linkedin.com/feed/update/urn:li:activity:7527642298713600000/

Appendix C. Complete list of published legal documents

According to the Consortium's public API, the following 10 English-language documents were published as of 16 November 2026. The titles and versions returned by the API and links to the English files are provided.

Consortium Manifesto. Version 2026-10-24.2. File: consortium-manifesto-en-2026-10-24.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/consortium-manifesto/file?locale=en

Notice of data processing when you contact us. Version 2026-10-23.2. File: en-intake-notice.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/intake-notice/file?locale=en

Charter of the Multifunctional Scientific and Economic Consortium. Version 2026-10-23.1. File: en-consortium-charter.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/consortium-charter/file?locale=en

Fundamental Doctrine of Lawful and Responsible Use. Version 2026-10-25.1. File: en-fundamental-security-doctrine.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/fundamental-security-doctrine/file?locale=en

AlexMus Quantum Information Consortium Code of Ethics. Version 2026-10-22.3. File: en-ethics-code.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/ethics-code/file?locale=en

Consent to personal data processing for a specific enquiry. Version 2026-10-23.2. File: en-personal-data-consent.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/personal-data-consent/file?locale=en

Contacts and legal details. Version 2026-10-22.3. File: en-contacts.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/contacts/file?locale=en

Cookie Policy. Version 2026-10-23.1. File: en-cookies.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/cookies/file?locale=en

Personal Data Processing Policy. Version 2026-10-23.2. File: en-privacy-policy.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/privacy-policy/file?locale=en

User Agreement. Version 2026-10-22.3. File: en-terms.pdf. URL: https://amqic.ai/api/v1/public/files/legal-documents/terms/file?locale=en

List source: https://amqic.ai/api/v1/legal-documents?locale=en

QSM Signature Information

Information page. Cryptographic verification is performed separately.

This page was added before cryptographic signing. Original document pages remain in their original order. Visual appearance alone does not establish signature validity.

Aleksei Sergeevich Musienko

Head of the AlexMus Quantum Information Consortium

Kyrgyz Republic

Sender address: mail@amqic.ai

Document ID: faa87bec-1af2-4a2b-bae4-595defde0b4a

Certificate: 3d07da6d-a42d-41c9-9d5d-5fc6b3fabc6d

Signed at: 2026-11-16 02:34:04 UTC

Document addressee: For public reference

Verification portal

Verification portal: https://amqic.ai/pages/request-verification

Use the final PDF with its .qsig file or QPDF container for verification. The QR code opens the portal and is not a document verification result.

The Consortium’s corporate verification tool. It does not replace official electronic signatures. Intended for use by Consortium employees only.