CEN/CLC/JTC 22/WG 3 "Quantum Computing and Simulation"
WG Secretariat: **AFNOR**
Convenor: **Lefebvre Catherine Mrs**

LayerModel_Draft01

Document type	Related content	Document date	Expected action
Meeting / Other	Meeting: VIRTUAL 31 Aug 2023	2023-08-31	**INFO**

CEN/TC XXX

Date: 20YY-XX

prEN XXXXX:20YY

Secretariat: XXX

JTC 22 WG3 Quantum Computing

Layer Model

Draft 01, 2023-08-31

CCMC will prepare and attach the official title page.

Contents

Page

[NOTE to the drafter: To update the Table of Contents please select it and press "F9". To recreate the Table of Contents, select *Custom Table of Contents – Options* and choose the appropriate headings/titles to display. For further instructions, see the *CEN Simple Template Quick Start Guide*.]

European foreword

This document (prEN XXXX:20YY) has been prepared by Technical Committee CEN/TC XXX "Title", the secretariat of which is held by XXX.

This document is currently submitted to the CEN Enquiry/Formal Vote/Vote on TS/Vote on TR.

This document will supersede EN XXXX:YYYY.

EN XXXX:YYYY includes the following significant technical changes with respect to EN XXXX:YYYY:

This document has been prepared under a Standardization Request given to CEN by the European Commission and the European Free Trade Association, and supports essential requirements of EU Directive(s) / Regulation(s).

For relationship with EU Directive(s) / Regulation(s), see informative Annex ZA, ZB, ZC or ZD, which is an integral part of this document.

[NOTE to the drafter: Add information about related documents or other parts in a series as necessary. A list of all parts in a series can be found on the CEN website: www.cencenelec.eu.]

Introduction

A layer model for quantum computing is an abstract description of a quantum computing system via a common stack of layers. The layer model slices down the overall complexity into two main layer models of addressing the whole quantum computing system. The lower of the two main layer models addresses mainly hardware, and it is dependent of the physical platform. The upper main layer model addresses mostly software, and it aims to be hardware agnostic. By agnostic we mean that the system works on different quantum computing hardware platforms such as: solid state quantum computing; atoms, ions and molecules optical quantum computing and topological quantum computing. Each of these two main layer models comprises many inner layers.

The first purpose is to define a common language that will be used to describe the features and functional requirements for each layer of the stack of a quantum computer and their interfaces. Another purpose is to analyse and describe the interaction between the layers. These are essential steps towards interworking between modules from different origins through well-defined interfaces. The functional description of each layer should offer sufficient guidance on where a desired functionality should be described, and what kind of exchange is needed with other modules through the interfaces. The boundaries between the layers are natural locations for such interfaces. Correctly defining such boundaries requires careful analysis of the interaction between the layers.

[NOTE to the drafter: If patent rights have been identified during the preparation of the document, the following text shall be included:

"The European Committee for Standardization (CEN) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning (...subject matter...) given in (... subclause...) and which is claimed to be relevant for the following clause(s) of this document:

Clause(s)...

CEN takes no position concerning the evidence, validity and scope of this patent right. The holder of this patent right has assured CEN that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with CEN. Information may be obtained from:

Name of holder of patent right ...

Address ...

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. CEN shall not be held responsible for identifying any or all such patent rights."]

1 Scope

This document describes a layer model that covers the entire stack of a quantum computer. The group of lower-level (hardware) layers are organized in different hardware stacks tailored to different hardware architectures, while the group of higher-level (software) layers are built on top of these and expected to be common for all quantum computing systems. The higher-up in the stack, the more agnostic it will be from the underlying hardware. Reducing the dependencies between higher and lower layers is a crucial point for optimised quantum computations.

This document is limited to a high-level (functional) description of the layers involved. Additional details of the individual layers will be described in other, future, CEN/Trs.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

[NOTE to the drafter: The Normative references clause is compulsory. If there are no normative references, add the following text below the clause title: "There are no normative references in this document."]

EN XXXX, *Title of document*

EN XXXX-1:20YY, *General title of series — Part X: Title of part*

EN XXXXX (all parts), *General title of series*

[NOTE to the drafter: If a dated reference is impacted by a standalone amendment or corrigendum, list the main standard and include a footnote as follows:

EN XXXX:20YY[1], *General title*]

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply / the terms and definitions given in… and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp/

— IEC Electropedia: available at https://www.electropedia.org/

[NOTE to the drafter: The Terms and definitions clause is compulsory. If there are no terms and definitions, add the following text: "No terms and definitions are listed in this document."]

3.1
term
text of the definition

[1] As impacted by EN XXXX:20YY/A1:20YY.

3.2
term
admitted term
text of the definition

Note 1 to entry:

[SOURCE: EN XXXX:20YY, definition XX]

[NOTE to the drafter: If applicable, a list of 'Symbols and abbreviated terms' can be included as a subclause under Clause 3 or added as a separate Clause 4.]

4 Overview

Quantum computing is an area covering many different implementations. A convenient way of specifying its requirements is via a stack of layers, as shown in Figure [*]. The layers are chosen in such a manner that the functionality of each layer can be described in an independent manner. This causes that the interworking between these layers can be described through well-defined interfaces at the boundaries of these layers. Note that such an interface can be virtual (hidden internally within the implementation of the same origin) or real (between implementations of different origin).

The stack covers both hardware and software layers, while some layers are a mix of both. The software layers are drawn in Figure [*] above the hardware layers with another colour. The aim is that higher software layers are more agnostic to differences in hardware architectures. These software layers are described in further detail in succeeding chapters.

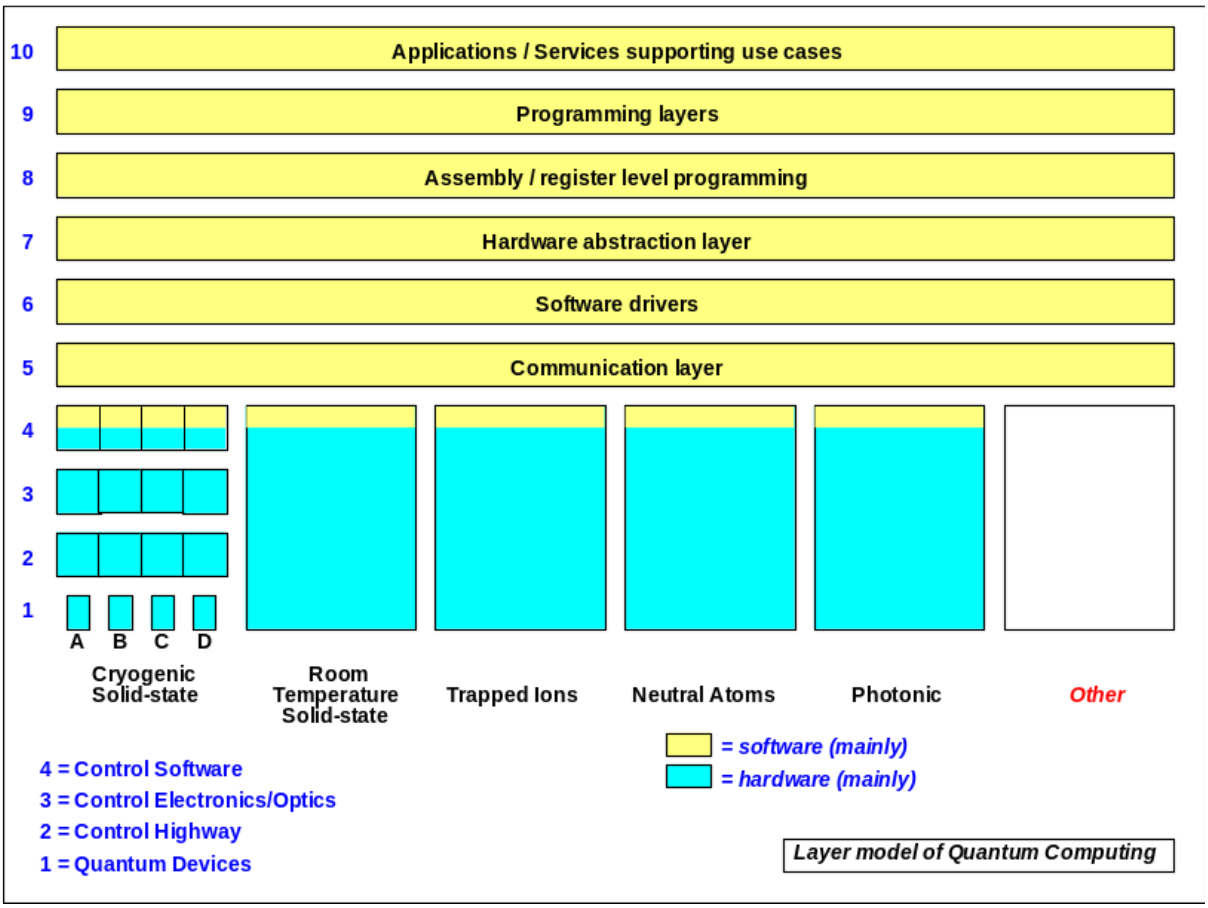

Figure [*] [This figure is based on the results from the Roadmap document, produces by FGQT of CEN-CENELEC. This figure will be updated when more details and or refinements have been elaborated]

The layered approach allows for using different hardware stacks for specifying the requirements of different architecture families. Each architecture family can have multiple members (A, B, C, ...) and the description of its hardware layers (1,2,3,4) may account for differences between these members. The diagram in Figure [*] has illustrated this symbolically by drawing different “boxes” in the layers for different members of the same family.

So far, the following architecture families have been identified (in arbitrary order):

- Cryogenic solid-state based;

- Room temperature solid-state based;
- Trapped ions;
- Neutral atoms;
- Photonic quantum computing;
- Other architectures that may be identified in future

These architectures are described in further detail in succeeding chapters.

A module is an implementation that may be constructed from (smaller) modules and components. It could offer the functionality of a single layer, of multiple layers, or just of a fragment of a layer. A module may also support different operating modes, such that it complies with the requirements of multiple members and/or multiple architecture families. As such, the functionality of a module may cover multiple layers and/or families and/or members.

4.1 Subclause title

4.2 Subclause title

4.2.1 Subclause title

4.2.1.1 Subclause title

4.2.1.1.1 Subclause title

4.2.1.1.1.1 Subclause title

Text of subclause.

5 Low level Hardware layers

5.1 Cryogenic Solid State

The members of this architecture family have in common that they all make use of a cryostat, where the quantum devices in a holder are controlled from outside the fridge by room-temperature electronics. Consequently, a huge amount of control channels is required to interconnect those two, especially when many qubits are to be controlled in a single fridge.

The following members have been identified within this architecture family:

Superconducting:

- Transmons;
- Flux qubits;
- Semiconductor spin qubits;
- Topological qubits;
- Artificial atoms in solids.

Four hardware layers have been identified for this architecture family.

5.1.1 Layer 1 - Quantum Devices

The quantum devices in hardware layer 1 are modules with qubits that are typically operating at cryogenic temperatures and may be implemented as chip and/or on PCB. They may have though requirements on shielding, operating temperature, magnetic aspects, etc.

5.1.2 Layer 2 – Control Highway

Hardware layer 2 covers all infrastructure needed for transporting microwave, lightwave, RF and DC signals (via electrical and/or optical means) between the control electronics at room temperature and the quantum devices at cryogenic temperatures. It is usually a mix of transmission lines, filtering, attenuation, amplification, (de)multiplexing, as well as means for proper thermalization. A huge number of control channels are required to control many qubits in a single fridge (which clarifies the name) and this can easily become very bulky. It could have tough requirements on aspects like heat-flow, thermal noise and vacuum properties.

5.1.3 Layer 3 – Control Electronics

Hardware layer 3 covers all electronics for generating, receiving, and processing microwave, RF and DC signals. Some implementations make use of routing/switching and/or multiplexing of control signals at room temperatures. It may have some firmware on board to guide the signal generation and signal processing.

5.1.4 Layer 4 – Control Software

Hardware layer 4 covers a mix of hardware and low-level driver software for instructing the control electronics and means for performing calibration. It has a software interface to higher layers for receiving sequences of instructions about when, where and what pulses are to be generated, and how to process and read-out the response.

Placed on top of quantum hardware, control software delivers high-performing qubit operations to higher level of abstraction in the quantum stack with minimal user intervention. It may include calibration means, low-level code to translate instructions from higher software layers into commands for guiding the control electronics/optics, and comprises the techniques used to define error-robust physical operations and associated supporting protocols designed to tune-up and stabilize the hardware.

Control software for quantum hardware is typically stored on digital computers, i.e., there is a very strict separation between the place where the control software is stored and the quantum registers. In the long term, control software may work in concert with Quantum Error Correction (QEC), which is supposed to lay at the assembly / register level programming layer, to provide broad coverage of various error types. More specifically, control software could improve the efficiency of QEC, i.e., reduce resource overheads required for encoding, by homogenizing error rates and reducing error correlations.

5.2 Room Temperature Solid State

The members of this architecture family have in common that solid-state qubits are all operating at room temperatures. Examples of members in this architecture family are:

— Artificial atoms in solids, such as NV centres;

— Optical quantum dots.

5.2.1 Layer 1 - Quantum Devices

5.2.2 Layer 2 – ...

5.2.3 Layer 3 – ...

5.2.4 Layer 4 – ...

5.3 Trapped Ions

The members of this architecture family can operate either at room temperature or at cryogenic temperatures (e.g. 4K). Quantum devices are controlled by electronics operating either at room temperature or under cryogenic conditions. For a larger number of qubits, the required amount of routing signals becomes bulky, and efficient thermal management, low-noise electrical and magnetic components are required.

Room temperature architectures that are identified are

— Optical qubits;

— Raman qubits;

— Spin (microwave) qubits;

Cryogenic (4K) architectures that are identified are

— Optical qubits;

— Raman qubits;

— Spin (microwave) qubits

5.3.1 Layer 1 - Quantum Devices

5.3.2 Layer 2 – ...

5.3.3 Layer 3 – ...

5.3.4 Layer 4 – ...

5.4 Neutral Atoms

Systems of individually-controlled neutral atoms, interacting with each other when excited to Rydberg states, have emerged as a possible platform for quantum information processing. The two main examples are ensembles of individual atoms trapped in optical lattices or in arrays of microscopic dipole traps separated by a few micrometres. In these platforms, the atoms are almost fully controllable by optical addressing techniques.

5.4.1 Layer 1 - Quantum Devices

5.4.2 Layer 2 – ...

5.4.3 Layer 3 – ...

5.4.4 Layer 4 – ...

5.5 Photonics

These architectures have in common that the quantum information during computing is encoded into photonic properties. We can divide different families of photonic quantum computers in two categories, universal and non-universal quantum computers. Non-universal quantum computers cannot perform every task but can at least perform one task.

Non-universal photonic quantum computing families that are identified are:

— Boson sampling;

— Gaussian boson sampling.

Universal families that are identified are:

— Knill-Laflamme-Milburn scheme (This was a theoretical proof-of-principle, but not practically feasible);

— Measurement based quantum computing using cluster states;

— Continuous variable quantum computing.

5.5.1 Layer 1 - Quantum Devices

5.5.2 Layer 2 – ...

5.5.3 Layer 3 – ...

5.5.4 Layer 4 – ...

5.6 *Other...*

Other architectures may be identified in future, and will then be added to this list

6 Communication Layer

A quantum computer must be provided with an operating system (OS), which is a resource manager for the underlying quantum hardware, provided with built-in networking functions allowing multiple users and applications to use the resources as remote clients. To an application, it appears as if it has its own resources and is protected from other applications. Applications can make use of facilities only as offered by the OS. For example, the OS provides communication primitives (e.g., based on the POSIX standard for the sockets interface) and only by means of these primitives should it be possible to pass messages between client applications and the quantum computer.

7 Software Drivers

In the layered view illustrated in Figure 12, software drivers are components that are plugged into the operating system and allow hardware-abstraction programs to call the control software of the underlying quantum hardware. If the hardware changes, the software drivers must change as well.

8 Hardware abstraction Layer (HAL)

The hardware abstraction layer (HAL) should allow quantum computer users, such as application developers, platform and system software engineers and cross-platform software architects, to abstract away the quantum computer implementation details while keeping the performance. The hardware may change, but the QASM-like programs (belonging to the upper assembly / register level programming layer) should be still able to work.

The hardware abstraction layer provides Application Programming Interfaces (APIs) to the upper layer, decoupling from the different types of quantum hardware technologies

9 Assembly / register level programming layer

This layer concerns quantum assembly languages (QASM) that describe quantum computations according to one specific model (e.g., circuit model, measurement-based model, quantum annealing model), with a per-architecture instruction set.

QASM is not a single assembly language and the syntax may also differ among various implementations. Languages for gate-based quantum computing have in common that they can describe universal circuits with single qubit gates, and entangled gates such as CNOT. Due to the huge diversity of quantum computing architectures, it is not likely that a unique, widely accepted QASM would emerge and later become a standard.

10 Programming layer

The specification of quantum algorithms using QASM languages is not easy for programmers. Indeed, QASM programs are usually generated by a software library, from a piece of code written in a common programming language, such as Python.

In general, the programming layers include all the languages, libraries, and software development facilities (e.g., software development kits, debugging tools, quantum compilers) used by developers for coding quantum algorithms or high-level applications that use predefined quantum algorithms as subroutines.

Quantum compilation is the problem of translating an input quantum circuit into the most efficient equivalent of itself, considering the characteristics of the device that will execute the computation and minimizing the

number of required two-qubit gates. The most advanced quantum compilers are noise-adaptive, i.e., they take the noise statistics of the device into account.

11 Applications / Services supporting use cases

To effectively support industrial and research use cases, quantum applications must be executed in suitable environments. Currently, some vendors provide access to quantum devices via user-friendly cloud platforms. The quantum programs must be locally compiled for a specific device and submitted for batch processing to the remote platform. However, other paradigms are emerging. For example, the quantum internet will enable networked quantum applications, whose execution will involve multiple quantum nodes and will be characterized by interleaved digital classical and quantum message passing.

Annex A
(informative)111

Title of Annex A, e.g. Example of a table, a figure and a formula

Annex BClause title

B.1.1 Subclause title

Annex CSubclause title

Annex DSubclause title

Annex ESubclause title

Text of the annex.

Annex FExample of a table

Table A.1 — Table title

Table header[a]			
Table text	Text[b]		
NOTE Table note.			
[a] Table footnote. [b] Second table footnote.			

[NOTE to the drafter: For indented text, it is recommended to create new cells instead of using tabs. Similarly, when aligning text to the right or center, use Word alignment buttons rather than tabs.]

Annex GExample of a figure

Dimensions in millimetres

Insert and Link Figure

Key

X definition for X

Y definition for Y

NOTE Figure note.

Figure text.

Figure A.1 — Figure title

Annex HExamples of formulae

A + B = C (1)

where

A is … ;

B is … ;

C is … .

[NOTE to the drafter: For simple formulae, the keyboard can be used. For more complex formulae, it is recommended to use MathType, if available, or MS Word Equation Editor.]

$$D_1 = 5,77 \times 10^{-13} \frac{C_1 \rho_1}{4\pi} \sum \gamma_i \left(\frac{\mu_{en}}{\rho} \right) E_i \int B_i(1) \frac{e^{-\mu_i(1) s_1}}{\ell^2} dV \quad (2)$$

where

$B_i(1)$ is …

D_1 is…

…

Annex ZA
(informative)

Relationship between this European Standard and the [essential]/[interoperability]/[…] requirements of [Directive]/[Regulation]/[Decision]/[…][Reference numbers of the legal act] aimed to be covered

[NOTE to the drafter: This is the Generic Annex ZA template. For some Directives/Regulations, specific templates need to be used and these can be found on the CEN BOSS: https://boss.cen.eu/reference-material/FormsTemplates/Pages/]

This European Standard has been prepared under a Commission's standardization request [Full reference to the request "M/xxx"/"C(2015) xxxx final"] to provide one voluntary means of conforming to [essential] / [interoperability] / […] requirements of [Directive] / [Regulation] / [Decision] / […] [Reference numbers of the legal act] [Full title].

Once this standard is cited in the Official Journal of the European Union under that [Directive] / [Regulation] / [Decision] / […], compliance with the normative clauses of this standard given in Table […] confers, within the limits of the scope of this standard, a presumption of conformity with the corresponding [essential] / [interoperability] / […] requirements of that [Directive] / [Regulation] / [Decision] / […], and associated EFTA regulations.

Table ZA.1 — Correspondence between this European Standard and [Annex … of] / [Article(s) … of] [Directive] / [Regulation] / [Decision] [Reference numbers of the legal Act]]

[Essential]/ [interoperability]/[…] Requirements of [Directive]/[Regulation]/[Decision] […]	**Clause(s)/sub-clause(s) of this EN**	**Remarks/Notes**

[NOTE to the drafter, to be removed before publication:

This table can be used to accommodate all possible cases and independently how detailed correspondence is established or is possible to give:

- to declare the correspondence with a general statement 'all requirements are covered' by complying 'all (or indicated) clauses' (then the table would contain only one row);

- to declare more detailed correspondence (then the table would contain as many rows as needed).]

WARNING 1 — Presumption of conformity stays valid only as long as a reference to this European Standard is maintained in the list published in the Official Journal of the European Union. Users of this standard should consult frequently the latest list published in the Official Journal of the European Union.

WARNING 2 — Other Union legislation may be applicable to the product(s) / [service(s)] / […] falling within the scope of this standard.

Bibliography

[1] EN XXXX, *General title*