CEN/CLC/JTC 22/WG 3 "Quantum Computing and Simulation"

Convenor: **PAUL Alexandra MME**

CryoSolidState_ControlElectroncs

Document type	Related content	Document date	Expected action
Project / Draft	Meeting: VIRTUAL 11 Feb 2026	2026-02-10	**INFO**

Description

Dear members,

Please find attached a contribution about Cryogenic Solid State QC.

Kind regards,

Text for Control Electronics Layer

Date of submission:	2026-02-10
Submitted by:	Rob F.M. van den Brink (Rob.vandenBrink@Delft-Circuits.com) Juan Boschero (TNO)
Expected action:	Vote
Expected date:	2026-02-11 (JTC22/WG3 meeting)
WG3-Project:	Cryogenic Solid State Quantum Computing

1. Abstract

Document N175 gives the latest Draft for a Technical Specification (TS) about Cryogenic Solid State Quantum Computing. Most of that document contains mature text, but text on control electronics is still lacking.

This contribution proposes literal text for inclusion into chapter 7 of that draft, which is dedicated to the Control Electronics layer. It has sufficient detail to give guidance on the functionality of this layer and is sufficiently vague on purpose to prevent that it blocks future developments.

2. Literal text proposal

Start of literal text proposal

7. Control Electronics

Hardware layer 3 covers all electronics for generating, receiving, and processing microwave, RF and DC signals. Implementations may make use of routing/switching and/or multiplexing of control signals at room temperatures.
The control electronics is typically a plurality of hardware functionalities, ranging from oscillators, arbitrary waveform generators (AWG), voltage and current sources, mixers, switches, filters, signal detectors, voltage and current meters, attenuators, etc. These hardware functionalities may be distributed over multiple standalone devices, sometimes from different vendors, and may also be grouped together into hardware modules, which may be wired together into a coherent system for controlling qubits.

As shown in Figure 4.1, this hardware layer includes a thin sub-layer for *Command Transformation* in order to command the control hardware in a uniform manner. This sub-layer

transforms these common commands into implementation-specific (proprietary) commands, which are fully tailored to the involved control electronics.
Figure 7.1 illustrates how multiple hardware modules can interface with a shared *Command Transformation* sub-layer. Each module may rely on its own proprietary software driver, which integrates into the sub-layer to provide a uniform interface to the higher layers of the stack.

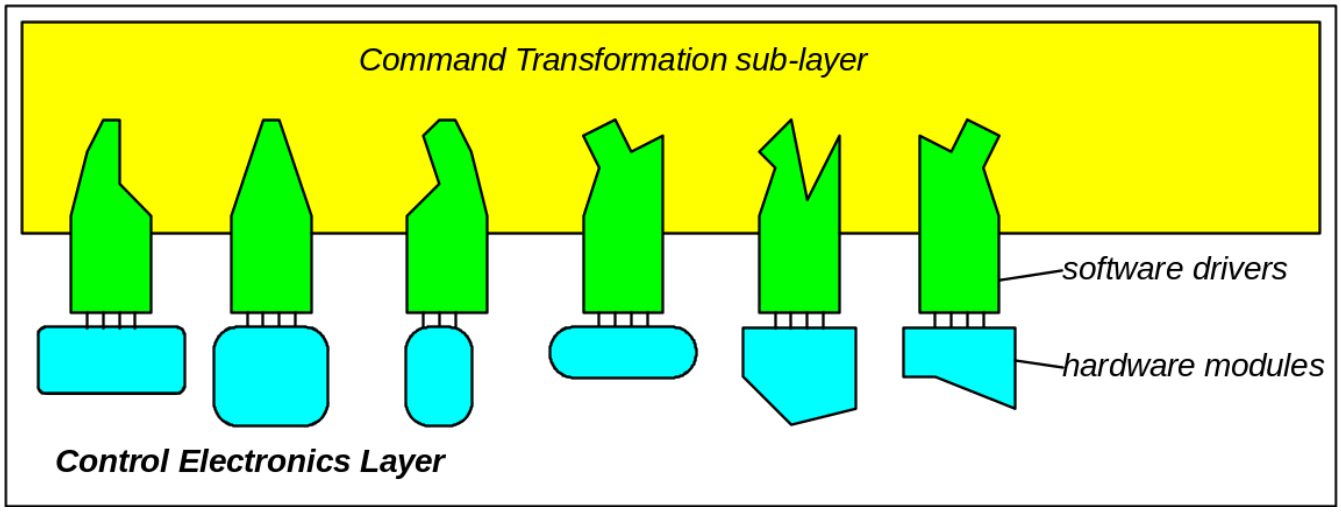

Fig 7.1 *Hard and software within the control electronics layer*

The functionality of this *Command Transformation* sub-layer goes far beyond a simple translation of commands. Since a lot of signals are to be generated and detected in parallel, and since many hardware devices may be involved in the handling of a single signal, a more intelligent approach has to be followed.
Higher layers will therefore communicate with the hardware modules via so called *sequencers.* These are local processing units for sending a sequence of proprietary commands to a variety of hardware devices, in order to fulfill a single task.
Each sequencer operates independently, comparable with a core in a CPU. Similar to how a CPU processes binary instructions, a sequencer executes *micro-programs* written in a proprietary low-level instruction language. Each sequencer has memory to store micro-programs and data. These micro-programs can be stored directly in binary format or compiled to binary via a dedicated (proprietary) assembly language.

The operation of sequencers usually follow the same pattern:

- prepare a sequence definition
  - allocate memory for acquisition
  - define/generate desired waveform
  - define the micro-program
- upload sequence definition to a target sequencer
- bind the sequencer to the desired inputs and outputs
- play the sequence
- read acquisition
- close and free the sequencer

A full system has many sequencers, and they all have much in common. But additional functionality can be build-in into more dedicated sequencers, such as:

- *Control sequencers*, to govern the generation of signals.
- *Readout sequencers*, to govern the generation of signals and acquisition of these signals.
- *Timetag sequencers*, to govern the output and acquisition of trigger events.

In principle, sequencers may be emulated to run their micro-programs directly within the command transformation sub-layer on a common CPU. But it is recommended for performance reasons that each sequencer runs its micro-program on a dedicated FPGA (Field Programmable Gate Array), integrated within the various hardware modules.

Since the qubits and transmission characteristics of the setup will gradually drift over time, the pulses that are to be fed to the qubits are to adapt regularly to the changes in the setup. Therefore the waveform, attenuation, predistortion with filters, etc. has to be adapted accordingly. Higher layers above the control software layer must therefore be offered with the flexibility of reprogramming sequences through a common interface.
This interfacing with higher layers may be implemented by means of a stream of commands, which allows for remote controlling the hardware via the communication unit of the quantum computer. An alternative way of interfacing is by means of function calls. This is simpler to implement but set restrictions on the programming languages being used for implementing higher layers. Both approaches can offer a standardized interface.
The specification of this interface to higher layers is left for future standards.

End of literal text proposal