A Double Full-Stack Architecture for Multi-Core Quantum Computers

A thesis submitted for the degree of

Ph.D. in Computer Architecture

by

Santiago Rodrigo Muñoz

Advisors:
Dr. Eduard Alarcón Cot
Dr. Sergi Abadal Cavallé

September, 29th 2023

Nanonetworking Center in Catalunya (N3Cat)
Departament d'Arquitectura de Computadors
Universitat Politècnica de Catalunya

Acknowledgements

First of all, I would like to thank Eduard Alarcón and Sergi Abadal, the best pair of advisors a Ph.D. candidate could ever dream of. They are able to combine ambitious ideas with close and patient guidance. Without them, such a thesis would have not been possible. And this is not just another complimentary phrase. I cannot forget all the help and inspiration that Carmina G. Almudéver has given me since the very beginning of this endeavor. The initial impulse and encouragement from Albert Cabellos were also instrumental for my adventure into the *quantum weirdness*.

To all the ones keeping up the work on quantum at N3Cat (Sahar and Pau), and the "multi-core team" from TU Delft and Valencia (Hans, Medina, Sebastian, Anabel, Laura...), goes my most sincere gratitude: team, we survived a pandemic! Each one of you would deserve a full paragraph here.

Calm and quiet times for coding and writing at Viaró and La Cadiera were essential. Thanks to everyone there, as well as the rest of my family and friends who have supported me along this way. Every smile and encouraging word helped this thesis to become a reality.

Abstract

Despite its tremendous potential, it is still unclear how quantum computing will scale to satisfy the requirements of its most powerful applications. Continued progress in the fabrication and control of qubits is certainly required. However, there are hard limits to the number of qubits that can be integrated into a single chip. Multi-core quantum computing has been identified as a solution to this scalability problem and included in the development roadmaps of the leading industry. Nevertheless, interconnecting quantum chips is not trivial, as quantum communications have their share of *quantum weirdness*. Quantum data cannot be copied, and decoherence is an unforgiving noise source for every qubit transfer, where every extra nanosecond counts and retransmission is physically impossible. Despite all these challenges, a comprehensive approach to quantum computer design based on multi-core architectures that leverages all the potential of quantum communications is crucial to unlocking the scalability issues.

In this context, the present thesis aims to lay the foundations of such a communications-enabled multi-core quantum computing architecture, as a proposed vision for the ultimate success of quantum computing. Our goal is to design a multi-core architecture that entangles computing and communication with a complete understanding of their intertwining requirements. In this way, while putting together dozens of quantum cores (i.e. thousands or millions of qubits collaborating), we alleviate the requirements for control circuits and improve qubit isolation.

In order to achieve this goal, we tackle three main tasks. First, we propose a layered approach for a double full-stack comms-enabled many-core quantum computer architecture (chapter 2), aligned with our vision. We aim to provide the basis for an architecture that may be technology-agnostic and intertwine quantum computing and communications.

Secondly, using design space exploration, we carry out a scalability and feasibility study of multi-core quantum architectures. The first approach used analytical formulations (chapter 4), while at the end of the thesis, leveraging all the acquired knowledge, code-base analysis, and fully-fledged network simulations are employed, run within a framework developed also for this work (chapter 7). The results of the exploration let us also compare different existing qubit and quantum communication technologies (section 4.4.2). This work might facilitate future work for providing design guidelines and optimal operation ranges for efficient and scalable multi-core quantum computers.

Finally, all this work needs to be backed by a study on short-range quantum communications. In particular, we have developed a model of quantum teleportation as a fitting candidate for inter-core communication technology (chapter 5). Moreover, we perform a thorough qubit traffic analysis on several algorithms and architectures that helps us see the bottlenecks and inefficiencies of such a network (sections 6.1 to 6.3). This leads to a latency and throughput analysis with real traffic together with the dimensioning of networking resources, completed by using a fully-fledged simulator developed for this thesis that models with high fidelity the different parts of a multi-core quantum computer (section 7.1). In addition, we have started the development of an efficient MAC protocol specific to our use case

(section 6.4), which we believe will complete the architecture design and communications modeling.

With the results of this thesis, we hope to contribute with design guidelines that may enable multi-core quantum architectures to unleash the potential of quantum computing.

Contents

List of Figures

List of Tables

List of Acronyms

QEC	Quantum Error Correction
QECC	Quantum Error Correction Code
ESM	Error Syndrome Measurements
QCCD	Quantum Charge-Coupled Device
QKD	Quantum Key Distribution
NoC	Network-on-Chip
DSE	Design Space Exploration
FoM	Figure of Merit
MAC	Medium Access Control
QC	Quantum Computing
NMR	Nuclear Magnetic Resonance
NISQ	Noisy Intermediate-Scale Quantum
FT	Fault Tolerant
MUSIQC	Modular Universal Scalable Ion trap Quantum-Computer
ADC	Analog-to-Digital Converter
NV	Nitrogen-Vacancy
QV	Quantum Volume
EPR	Einstein Podolsky Rosen
QI	Quantum Internet
QNoC	Quantum Network-on-Chip
SAP	System Architecting Problem
QAOA	Quantum Approximate Optimization Algorithm
MOO	Multi-Objective Optimization
CLOPS	Circuit Layer Operations Per Second
QoS	Quality of Service
FIFO	First-In First-Out
QFT	Quantum Fourier Transform

Chapter 1

When both expectations and stakes are high

QC (i.e computing platforms based on nanoscale quantum mechanical properties as opposed to classical electronics based on transistors) has been for some decades limited to theoretical developments and algorithms. However, in recent years it has started to become a reality [1–6]. By taking advantage of unconventional properties such as superposition and entanglement (details on this in Section 1.1.1), quantum computer implementations of time-consuming algorithms can be exponentially faster than their classical counterparts. For instance, Shor's algorithm [7] allows factoring large numbers in polynomial time: while a classical computer would take a million years to factor a standard 1024-bit RSA key, a quantum computer could do the same task in hundreds of seconds [8] (see Fig. 1.1). Therefore, fields as important as internet security, pharmacology, complex combinatorial and optimization problem solving, big data analysis or AI could make a leap when fully-fledged quantum computers become available.

However, extraordinary performance requires an extraordinary environment: any interaction with other particles or forces causes a qubit (the *alter ego* of a classical bit in the quantum world) to rapidly lose the information it contains. Preserving qubit entanglement and quantum state superposition –the key enablers of the quantum processing power– implies maintaining the quantum information in qubits intact: see Section 1.1.1 for more details. This, being trivial in classical computing, is in fact one of the most challenging issues for building quantum computers. Therefore, quantum processors must be kept at very low temperatures (close to absolute zero) and isolated from the outside world, something which greatly hinders the external control and computation, for operations on the qubits and measurements of their values.

These demanding requirements make building quantum computers a challenging task and compromise quantum computing scalability. Although during these last years we have seen remarkable sustained advances in quality and number of qubits in working prototypes, the existing realizations of quantum computers are too small-scale and error-prone yet to be able to experimentally demonstrate the theoretical results and proven algorithms that show these impressive speed-ups [2]. It is predicted that millions of qubits will be required in order to run practical quantum algorithms [9].

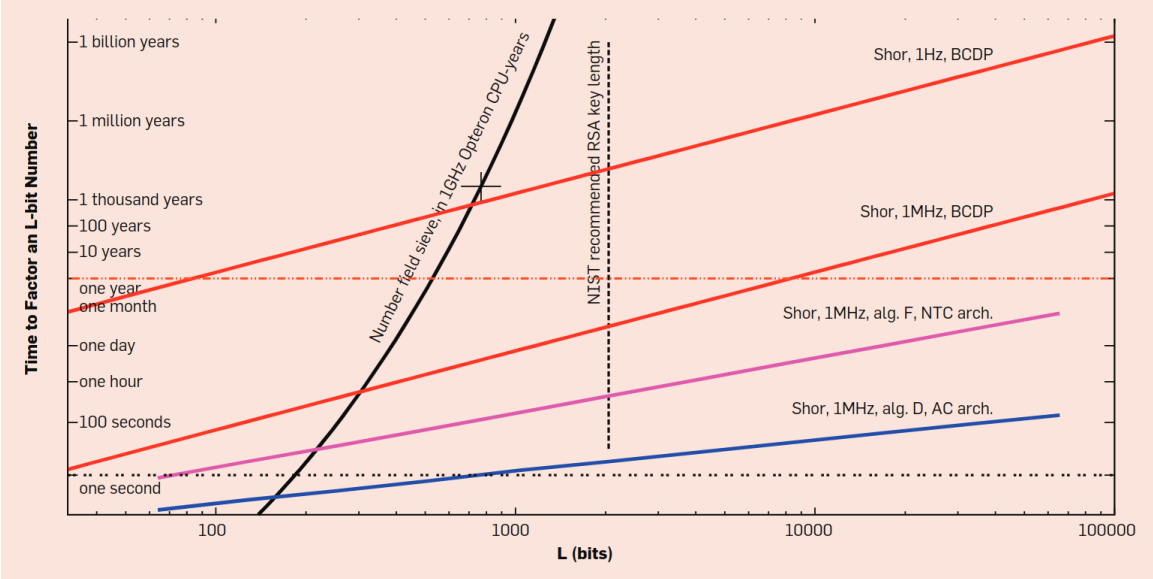


Figure 1.1: Comparison between classical and quantum facorization, as taken from [8]. L stands for the length of the number to be factored. Color lines represent the predicted performance of various quantum platforms, whereas the black line follows the execution time of a classical computer.

Despite all the ongoing research, the largest experimental quantum computers at the time of writing have only recently reached several hundreds of qubits in a single chip [10–13], and monolithic single-chip approaches are expected not to scale past the thousand-qubit barrier, due to impracticality of control circuits integration, per-qubit wiring, prohibitive quantum decoherence and severe qubit operation errors [1, 2, 14].

The *quantum leap* –doubly quantum, we could say– there exists from today's prototypes to fully-functional and useful quantum computers has such a breadth and depth as to require additional support from other disciplines related to processor design. Such an approach to the problem demands a system-wide optimization, the foundations of which may be laid on audacious proposals for the design and architecture of the quantum computer as a whole.

In this introductory chapter to the thesis, we review the fundamentals of QC, for the reader to have enough background to fully grasp the different advantages and disadvantages of such a groundbreaking technology. In Section 1.1 we present the most important properties and challenges when working with quantum computations, and how the different pieces work together: quantum programs, compilers, processors, communications, and networks. With these tools at hand, we describe the complexity of scaling quantum computers in Section 1.2, and present the most promising approach to solve it in Section 1.3. This will let us motivate the present work leading the way into Chapter 2, where the motivation, scope, and goals of this thesis are put forward.

1.1 Quantum background: on qubits, gates and circuits

Reviewing the main concepts of multi-core quantum computing and communication lets us set the stage for our work Basic notions of quantum computing and communications

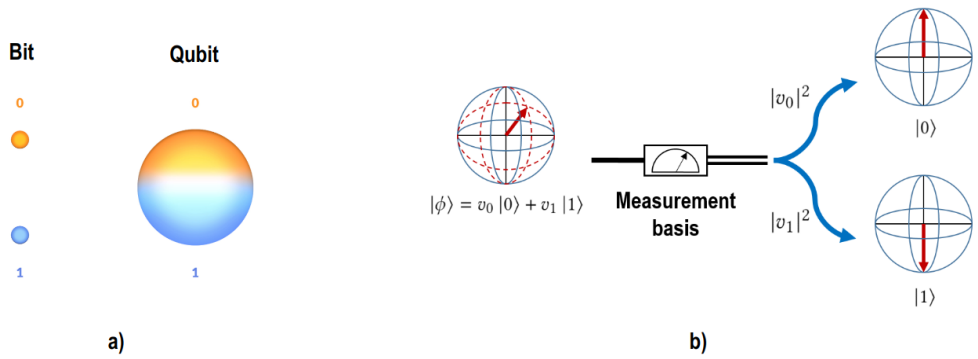


Figure 1.2: Qubits VS bits and "the observer effect" in quantum measurement. a) While bits can only represent two discrete values, qubits hold a probabilistic state that exponentially increases its computing capacity, b) When measuring a qubit a collapse of the quantum state occurs leading to information loss if the qubit is not holding a pure quantum state coherent with the chosen measurement basis

are needed in order to fully understand the implications of QC performance, as well as to identify correctly the particularities that might help in designing a quantum computer. For a deeper look into quantum computing and communications, the interested reader may refer to [2,15]. However, if the reader is familiar with QC, it might be helpful to hop directly into Section 1.2.

1.1.1 The power of qubits

The qubit constitutes the basic unit of computation in the quantum world, as an *alter ego* of a 1-0 classical bit. In the most commonly used model of quantum computation, i.e. the unitary circuit model [16], the quantum information contained in a qubit (a quantum state) can take, as in the classical world, the logical values of 0 and 1[1]. These are usually represented as $|0\rangle$ and $|1\rangle$, also called *ket notation*.

The *quantum weirdness* of a qubit (and its power) might be very well summarized in three key concepts:

- First, **quantum superposition**, i.e. the ability of a qubit to hold a state which might be a pure $|0\rangle$, a pure $|1\rangle$, or just a linear combination of both, as in:

$$|\phi\rangle = v_0|0\rangle + v_1|1\rangle\,, \tag{1.1.1}$$

  where $v_0, v_1 \in \mathbb{C}$ and $|v_0|^2 + |v_1|^2 = 1$. That is, when two qubits are superposed, the quantum state becomes a combination of $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$ states, in a process that can be extended to an arbitrary number of qubits. In other words, a quantum computer with N superposed qubits is operating over 2^N states simultaneously, which provides an exponential increase in performance for certain applications.

[1] Other qubit models (such as quantum annealing [17], measurement-based [18] or quantum walks [19]) assume some variations, such as continuous quantum states or even compress several discrete values in a single d-state qubit, also called *qudit*.

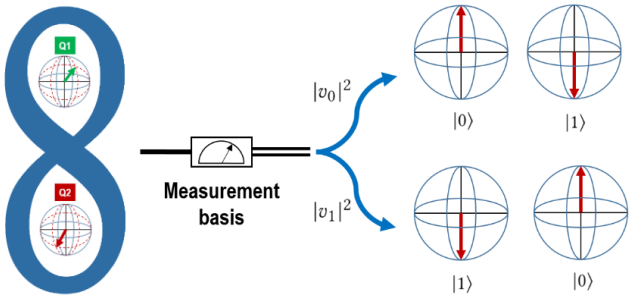


Figure 1.3: Behavior of a pair of entangled qubits. In the depicted case, the qubits collapse always to opposed base states, no matter how far apart they might be.

- Second, we have **quantum measurement and the "observer effect"**. Extracting the information from a qubit into the the macro world means measuring its physical quantum state. Due to quantum mechanics' postulates, this gives us only a partial view of it, as the $|\phi\rangle$ state collapses into the measurement basis, e.g. either $|0\rangle$ or $|1\rangle$. In other words, the measurement leads to either $|0\rangle$ with probability $|v_0|^2$ or to $|1\rangle$ with probability $|v_1|^2$, in a process that *destroys* the quantum state of the qubit. With two qubits A and B, the quantum state before measurement is the superposition of four possible values v_{00}, v_{01}, v_{10} and v_{11} corresponding to the relative probabilities of the qubits taking the $|0\rangle$ or $|1\rangle$ states after measurement. This can be generalized to any number of qubits.

- Third, **quantum entanglement**. Two or more qubits can also be *entangled*, i.e. whenever any of them is measured, all of them collapse into a definite state, with a non-zero correlation of the global result. This happens independently of the existing distance between them. In other words, the state of each of those qubits cannot be described independently of the state of the other(s). For instance, two entangled qubits could be such that either both collapse to 0 or both collapse to 1, as in:

$$|\phi\rangle_{AB} = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{1.1.2}$$

  Observe that the probability of $|01\rangle$ and $|10\rangle$ states is 0. That means that there is no need to measure both: knowing just one of the measurement's values lets us know both of them. In fact, Einstein called this "spooky action at a distance", as this seemed to imply information was traveling faster than light, thus violating the Relativity Theory. However, sharing a state at a distance is just a quantum mechanical property, involving specific operations in the entangled qubits creation phase, with no magic powder participating in the matter.

1.1.2 Adding quantum noise

These powerful properties are the foundations of quantum computing but also imply some difficulties. In particular, the *no-cloning theorem* (a derivative of the "observer effect")

states that it is not possible to create an identical copy of any given quantum state. As a consequence, qubits are not only an abstract unity of information but also the physical entity containing the information: if the qubit is measured or corrupted, the quantum state is lost. This is an issue because qubits nowadays are noisy and prone to *quantum decoherence*, which arises from the interactions of a qubit with the environment and other qubits (i.e. unsought entanglement due to cross-talk), imperfect qubit operations, and qubit leakages. In any case, these undesired (but unavoidable) effects lead, over time, to unwanted modifications of the qubit's state [20].

The **coherence time** τ_c corresponds to the amount of time the qubit is able to maintain its quantum state unchanged. Following one of the most used noise models, measuring qubit decoherence can be done in two different ways, usually referred to as amplitude damping (or T_1) and phase damping (or T_2). Amplitude damping is the average transition time from the excited state to the ground state, mainly due to dynamic coupling. Phase damping determines the amount of time the qubit is able to keep a superposition state. Furthermore, if we want to measure the decoherence of a qubit ensemble instead of a single qubit, we use T_2^*, which takes into account an additional decoherence source, i.e. the uncertainty in the relative phases among different qubits due to the spatial dissimilarity. This means usually that $T_2^* \leq T_2$, having that T_2^* may deviate notably from T_2 [3, 21, 22]. Although these two metrics represent different decoherence phenomena, given that the energy relaxation (amplitude damping) does disturb also the qubit phase, the coherence time T_2 is affected by both decoherence processes, and thus it is widely used in the literature as the standard qubit decoherence metric.

Being the main *quality* metric for a qubit, the decoherence times have been continuously improved in the different existing qubit technologies. Although it is not the only challenge for the scalability of quantum computers, the values hitherto reached are still far from allowing qubits to run successfully representative quantum algorithms without Quantum Error Correction (QEC) [23]. You can find more details on these advances in Section 3.1.

The quantitative metric for the distance between the actual quantum state of a qubit and the theoretical quantum state it should be holding if there was no decoherence affecting it is called **fidelity**, and it is usually expressed as [24]:

$$F(\rho, \sigma) = \left(\mathrm{tr} \sqrt{\sqrt{\phi}\rho\sqrt{\phi}} \right)^2, \tag{1.1.3}$$

where ϕ and ρ are the two quantum states being compared, and thus fidelity represents the (symmetric) probability that either state would pass a test to identify as the other.

1.1.3 Quantum Error Correction

Quantum decoherence and the "observer effect", leading to the *no-cloning theorem*, impose hard limitations on the computing capabilities of a quantum platform: the quantum information is fated to disappear sooner or later without any option to recover the decoherence and operation errors. However, the development of QEC let some optimism in. With QEC, quantum information is protected by encoding a quantum state into several entangled qubits using a specific Quantum Error Correction Code (QECC). Usually, it also implies a continuous monitorization to detect and correct existing errors [8]. Error detection is done through

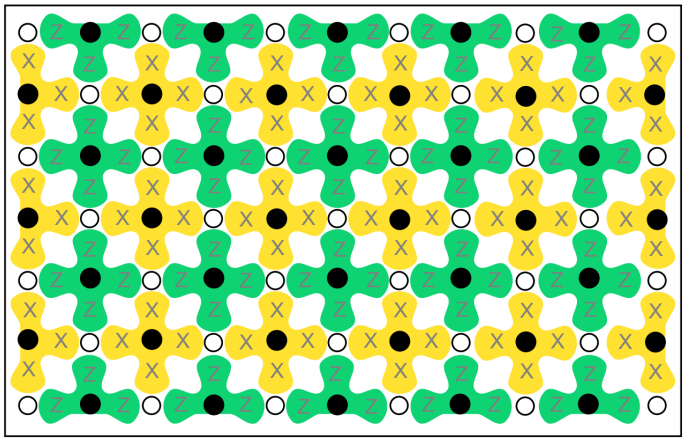


Figure 1.4: A 2D lattice implementation of the surface code, as extracted from [25].

parity check measurements called Error Syndrome Measurements (ESM), which avoid direct measurement of the qubits during these checks, thus preserving the qubit states. The extra qubits used for this encoding overhead are usually called *ancilla qubits*.

One of the most well-known QECC is surface code [25, 26], convenient due to its relatively high error threshold ($\sim 1\%$) and its low-complexity 2D-grid structure that matches with most qubit platforms topologies. In surface code, qubits are arranged in a regular 2D lattice (see Fig. 1.4). Within the array both data (white-filled circles) and ancilla qubits are arranged in an alternate fashion: observe that there are two types of ancilla qubits: Z (green) and X (yellow), used to detect errors on each axis (bit-flip and phase-flip, respectively). When errors are identified after a ESM, the corresponding corrections are applied.

1.1.4 From a quantum circuit to a quantum computation

The power of quantum computing comes from the operation of qubits and their probabilistic states. In the quantum computation unitary circuit model, quantum logic circuits are used. These circuits, as their classical counterpart, contain (quantum) logic gates that operate either on single or several qubits. In this way, they are capable of altering the quantum state of qubits by affecting the values v_0 and v_1 with single-qubit operations, or by applying gates that combine the quantum state of two or more qubits. A controlled-NOT (CNOT), shown in Fig. 1.6, is a clear example of a two-qubit gate: a NOT is applied to a qubit B only when the control qubit A is $|1\rangle$, thus modifying the values of v_{00}, v_{01}, v_{10} and v_{11} accordingly. Quantum algorithms can be therefore described as *quantum circuits*, i.e. a sequence of quantum gates applied to the qubits in the computer: see the example in Fig. 1.5.

Quantum algorithms are usually expressed as quantum circuits that are agnostic of quantum hardware, i.e. it is assumed that all qubits can interact with each other or that quantum gates can be performed in parallel as long as their dependencies are respected. However, quantum processors suffer from several constraints that must be satisfied when executing a quantum algorithm on them [27]. Qubits are arranged on a specific topology, and although all-to-all qubit connectivity is possible for trapped-ion processors [28], in most

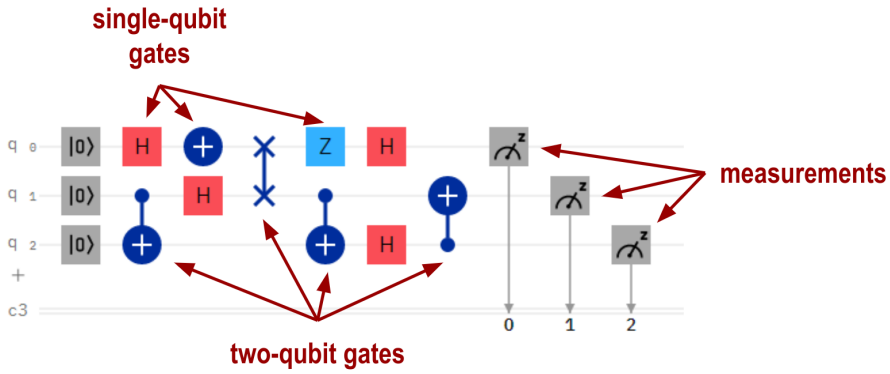


Figure 1.5: An example of a quantum circuit with single- and two-qubit gates together with measurements.

of the quantum devices one of the most stringent constraints is the reduced connectivity between qubits, limiting their possible interactions to, for instance, only nearest-neighbour. In fact, one should differentiate *physical qubits* (the material holders of the quantum data, whether they are movable, e.g. photons, or not, e.g. spin qubits) and *virtual qubits* (the abstract quantum information which is operated on, swapped or teleported around the processor, measured, etc.).

Accordingly, for compiling a quantum circuit into specific hardware, different mapping approaches have been proposed for realizing quantum algorithms in connectivity-constrained, and in more general resource-constrained, single-core quantum architectures [29]. The main steps in this process are:

1. Initial placement of qubits in which the virtual qubits (qubits in the circuit) are assigned to the physical ones (qubits in the quantum chip).

2. Routing of the qubits to adjacent positions whenever they need to interact (using specific quantum operations such as SWAP gates)

3. Scheduling of quantum operations to leverage their parallelism and reduce the overall circuit latency (i.e. the time it takes to complete the execution). This process results in an increase in the number of gates and circuit depth (number of steps in the circuit) that in turn decreases the success rate of the algorithm [30].

1.1.5 Fundamentals of Quantum Communications

In order to improve their isolation and minimize decoherence, qubits are operated and read out using quantum gates which, as opposed to what is done in classical computing, are applied *in-place*. However, when two qubits are required to interact by means of a two-qubit gate, their quantum state needs to be moved to adjacent positions in the computer. For instance, if we want to apply a CNOT gate between qubits A and C of Fig. 1.6, we will have to exchange (using SWAP gates) the position of qubits B and A or C. This movement, which we refer to as qubit communication, is a key operation in QC and its efficiency in

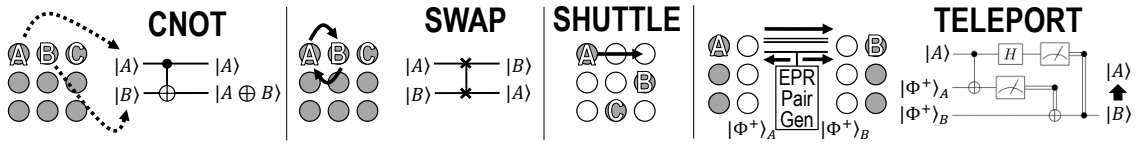


Figure 1.6: Some fundamental operations of quantum computing and communications. From left to right: controlled-NOT gate between qubits A and B, swap between qubits A and B, shuttling of qubit A, and teleportation of qubit A to the position of distant qubit B (whose state becomes that of A after completing the teleportation) via an entangled pair.

space and time is crucial, as it involves a resource and time overhead that may be critical for the overall execution performance.

In fact, transferring a quantum state in any fashion is a complex task: it cannot be done using classical communications, and, due to the *no-cloning theorem* (i.e. an arbitrary unknown quantum state cannot be copied), qubit retransmissions are impossible. Even more importantly, communication latencies have to be as low as possible, to minimize the effect of the constant degradation on the qubit to be transmitted due to quantum decoherence.

Quantum state communication can take place via the physical movement of the qubit or by the transfer of its quantum state. Aiming at overcoming the described obstacles, different quantum interconnect techniques that enable quantum state transfer are employed. Here, we describe three methods depicted in Figure 1.6: SWAP gates, qubit shuttling, and quantum teleportation.

SWAP gates. The most basic form of communication in quantum computers is the SWAP gate. A SWAP gate can only be applied to two physically adjacent qubits, which exchange their state with one another. Thus, to move a qubit state to an arbitrary position, a chain of SWAPs can be applied. However, this implies interacting with every qubit along the way.

Qubit shuttling. A technique where qubits are physically moved using electromagnetic fields across a chip space intentionally left devoid of qubits (shown as blank positions in Fig. 1.6) in order to place together the ones that need to inter-operate. This technique is used in a specific implementation of qubits, the ion traps. Using multiplexed architectures such as the Quantum Charge-Coupled Device (QCCD), some experiments have shown coherent shuttling of ion qubits through 2D junctions over millimeter distances in microsecond timescales. However, with today's technology, its latency and complexity do not scale well for more than $\sim$ 100 qubits, and optical interconnects are needed to scale to larger platforms [31]. For a deeper look into the state of the art of this technology, see e.g. [32]).

Qubit teleportation. A more versatile yet indirect quantum communication technique is quantum teleportation. This technique exploits the property of *quantum entanglement*, which refers to the ability to have two or more qubits containing states that cannot be described independently of each other. For two qubits X and Y, being in an entangled state $|\Phi\rangle^+$ is defined as:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}\Big(|0\rangle_X \otimes |0\rangle_Y + |1\rangle_X \otimes |1\rangle_Y \Big) = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \tag{1.1.4}$$

implying that, when measured, both qubits collapse to the same state, either $|0\rangle$ or $|1\rangle$. That is, if we measure one qubit, we can be completely sure about the other qubit's state no matter

how far apart they are. Therefore, qubit teleportation is applicable for communication at any distance, from the chip to planetary scales.

Qubit teleportation uses a pair of these entangled photons, also called Einstein Podolsky Rosen (EPR) pair or Bell states [33], and a classical channel to transfer the quantum information of a qubit without moving it physically. For that, as shown in Fig. 1.6, both transmitter A and receiver B are sent one qubit out of a pair that shares an entangled state, which we name $|\Phi\rangle_A^+$ and $|\Phi\rangle_B^+$. These are completely independent of the state $|A\rangle$ to be transferred.

Then, some basic operations involving the qubit to be transmitted and the entangled qubit are applied, followed by a measurement. The result (a binary value) is then sent via a classical channel. With that information, the receiver can reconstruct the original transmitted quantum state by applying some corrections, turning $|B\rangle$ into $|A\rangle$. Note that by being measured, the original state of qubit A is lost and hence the *no-cloning theorem* is respected. Although it is still in a nascent stage, quantum teleportation has been demonstrated experimentally at different scales [5, 34].

The most notable advantages of this technique are the distance-independent latency, the decoupling of transfer into two different channels (entangled qubit pair and classical) for better protection of quantum data, and the compatibility with different qubit technologies. On the other hand, the efficiency of entangled pair distribution and network integration pose hard challenges. As a whole, quantum teleportation is generally accepted as a fit candidate for quantum communication networks, both at large and small scales. The interested reader can read further on the topic in [35].

1.1.6 Quantum networks

In the context of quantum technologies, the challenges of quantum communications are several. First, a dedicated physical infrastructure is required to realize the quantum channels. For instance, EPR generators and optical fibers capable of transmitting entangled photons are required to perform quantum teleportation. A second challenge, posed by the *no-cloning theorem*, is the need to minimize the noise as qubit retransmissions are not possible. Hence, quantum networks are extremely latency-sensitive, since qubits tend to decohere as time passes, which clearly affects protocol design.

These challenges are being addressed in large-scale networks with solutions that already reached the industry. Quantum cryptography, and more specifically Quantum Key Distribution (QKD), provides an unconditionally secure way to encrypt communication [36]. In this case, a string of qubits is directly transmitted using photons in dedicated optical networks and used to produce random secret keys for secure communication. Thanks to the *no-cloning theorem*, this key distribution protocol is able to detect an eavesdropper in the channel. Such a process has been deployed at a city-wide scale [37] and even using free-space optics with a satellite as the secure middle-point producing the qubits [38].

Along these lines, the concept of Quantum Internet (QI) [39] has been conceived as a large-scale quantum network capable of working in parallel with the classical internet to enhance its applications (starting with QKD) [5], as well as of interconnecting distant quantum computers to distribute computation [40]. This has opened a wide and fertile research field, where the networking challenges are being tackled. In particular, the need for quan-

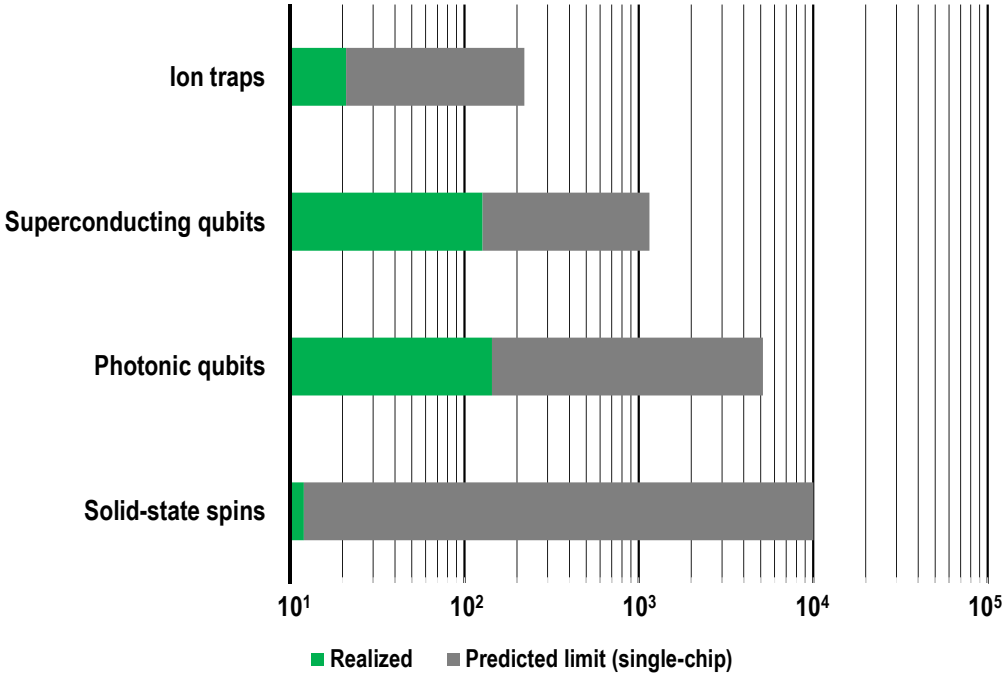


Figure 1.7: Achieved and predicted scalability on a single chip for the most widespread qubit technologies. Data obtained from [1, 10, 11, 44–47]

tum repeaters to allow long-distance quantum teleportation [41], quantum communications protocol stack design and implementation [42, 43], and modeling of communications at this scale [15] are being investigated.

1.2 The scalability problem

Due to the already mentioned challenging issues of quantum decoherence, control errors, qubit crosstals, *no-cloning theorem*, etc., scaling up quantum computers to fully-fledged computing platforms capable of achieving the theoretical exponential speed-ups for turning the Second Quantum Revolution into reality is proving a really hard challenge. An image will explain that much better than a thousand words: in Fig. 1.7 the scalability potential for the different existing qubit technologies is summarized. Observe how none of them exceeds 10,000 qubits, well below the millions of them we need to effectively get into the high-performance QC range.

What are the solutions at hand to this problem? An extended review can be found in Chapter 3, but the technology improvements that are looking into control issues and decoherence rates are not expected to unlock this issue. For that reason, the industry has recently turned into distributed architectures, composed of state-of-the-art-sized quantum chips that work together in order to sum up to larger amounts of total qubits. For instance, IBM updated its roadmap to include a evolving strategy into multi-core/distributed quantum

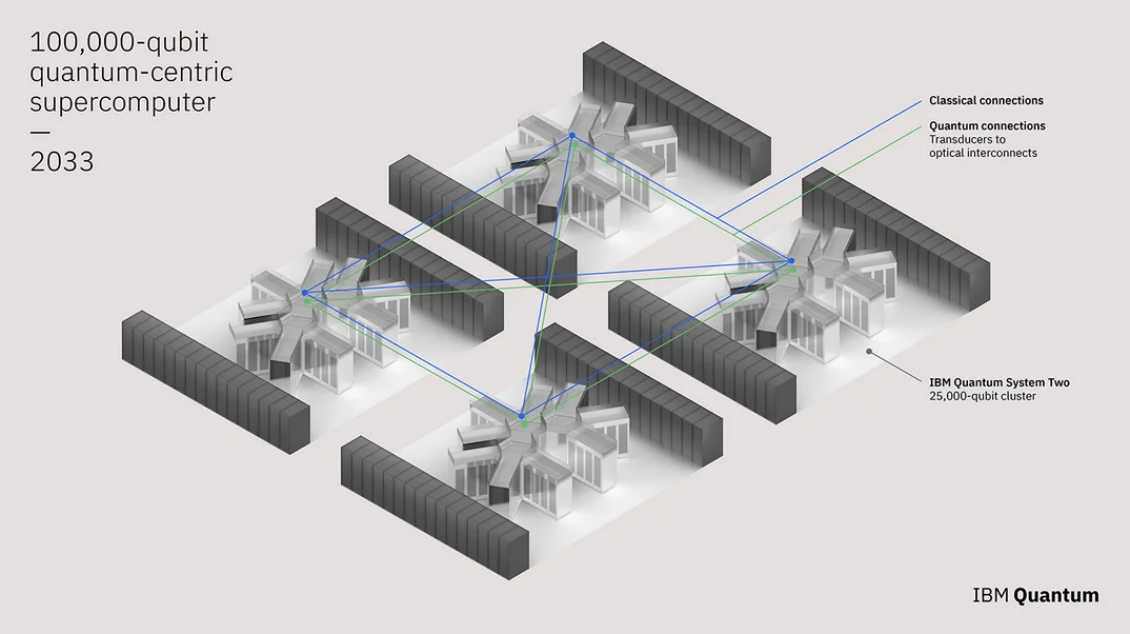


Figure 1.8: IBM's vision of their 2033 100,000 quantum supercomputer, as extracted from [50]

architectures, starting with classical communication among chips, entanglement forging, quantum circuit knitting and fully-fledged distributed QC [1, 48, 49]. Most recently, IBM announced their goal to reach 100,000 qubits by the next decade using such a distributed approach [50]. But how are these architectures envisioned for actually enabling such a wide quantum leap?

1.3 Divide and conquer

Multi-core quantum architectures represent an effort to utilize currently existing limited-size quantum processors into a large structure by interconnecting them and making the whole platform work jointly. It may consist of dozens of quantum chips, containing hundreds to thousands of qubits each, communicated both by a classical network (intended for signaling, message passing, and measurement results communications) and a quantum backbone (to allow quantum state sharing).

Connecting several quantum chips in a multi-core fashion has been proposed as a modular approach that may enable the scaling of quantum computers (see section 3.3.2 for a detailed review of existing proposals). This not only simplifies control circuit requirements but also reduces crosstalk errors and other impairments derived from a densely packed group of qubits integrated in a single-chip quantum processor. Existing works on these and similar architectures [31, 40, 51–56] use different qubit technologies (ion trap, quantum dots or impurities in solids) and *module* interconnects (ion shuttling, photonic switches, quantum teleportation).

However, multi-core quantum computers come with their own set of challenges, which have to be overcome. Particularly, communicating quantum chips is far from being a simple

task. Sharing quantum data across cores is hindered by a variety of issues such as that qubits cannot be copied due to the *no cloning* theorem or that qubits have a limited lifetime rendering communication extremely latency sensitive. Moreover, this quantum-coherent network needs to work in parallel with a chip-scale network transporting classical data, intended for assisting quantum transfers with control and synchronization messages.

Nevertheless, the previously described proposals remain at a theoretical plane without entering into details on the actual implementation, lacking completeness in how to solve all these communication issues. In the present work, we have aimed to fill this void by going further and studying thoroughly the implications of quantum communications in quantum multi-core computing in order to produce analysis tools and architecture proposals that might serve as strong foundations for such a promising technology.

Chapter 2

Intertwining Communications and Computation to Unlock Quantum Computing Power

The promising features and computing power of quantum computers let us dream of a revolution in crucial research fields. However, the existing challenges and bottlenecks cannot be understated. Although current prototypes let us glimpse the theoretical potential of this technology, their production and maintenance costs, together with the existing difficulties to scale them up to fully-fledged computers, struggled with over a long time, can lower the expectations over such a revolution. However, research is not about expectations or predictions. Many of the "prophecies" on new technologies have been quickly proved wrong[1]: proving a hypothesis right or wrong requires a thorough study on the matter, with model, data and lots of testing.

After around forty years of research in QC and billions of USD invested into this technology, a firm grounding effort including great industry players such as Intel, Google, or IBM has facilitated a rapid evolution and growth, particularly in the last decade. We may very well say that QC is at that precise turning point where it may definitely become either a *too-good-to-be-true* theoretical development or a historical milestone. At this critical period, research in QC must focus on the somewhat sophisticated and even sometimes counter-intuitive nature of its roots, digging deep into quantum mechanics.

Therefore, promising efforts to scale up quantum computers by interconnecting several processors (such as those presented at the end of the previous chapter) should be built from the ground up, avoiding any pretended parallelism with classical computing or mimicking classical networks. In fact, none of the existing works on the matter has deeply analyzed whether this approach is effectively enabling an architecturally scalable quantum computer, and which are the resource overheads and computational costs of such architectures.

This thesis' postulate is that a comprehensive approach to the quantum computer design based on multi-core architectures, as opposed to current densely-packed monolithic models, is crucial to unlock the scalability issues. Such an endeavor implies not only proposing an

[1] One of the most famous in computer history would be the one coming from the then-IBM Chairman, Thomas Watson, who said in 1943 "I think there is a world market for maybe five computers".

architecture design (i.e. specifying the organization, functionality, and design rules), but also doing so by taking into account the specific challenges and virtues of qubit operation inside a core, and all the *quantum weirdness* of entanglement and superposition in inter-core quantum communications.

The nature of quantum computing makes it to be very much related to communicating, and quantum communications (see more on this in Chapter 5) are crucial for executing even small quantum programs. This indeed is more present in the case of the multi-core architectures, where the described complexity of quantum communications is inserted into the already constrained quantum computing environment.

This implies that in order to lay firm foundations for multi-core quantum computer architectures, a deeply entangled design between computation and communications is essential. We need to minimize overheads and inefficiencies in order to leverage the full potential of quantum mechanics: avoiding long communication latencies, buffering waiting times, or data losses (something which may be easily overcome in classical multi-core computing) may be crucial for the QC success.

Finally, this work is to be done with a multi-disciplinary perspective: quantum physics, computer science, electrical engineering, and computer networks are bound together. Being at the dawn of large-scale QC, abstracting out any of the involved research fields can lead to a partial and sub-optimal design. In particular, this thesis is grounded on the basics of qubit technologies and operations, communications design, and network protocols, looking forward to covering the existing gap between work on single-core quantum chips and that on large-scale distributed QC and the quantum Internet [5].

In the present chapter, we are going to delve into the entanglement between quantum computing and quantum communications, explore the full QC stack while discovering how the communications stack is already implicitly present, and summarize the goals of the present thesis which develop its multidisciplinary vocation.

2.1 The blurry frontier between quantum computing and communication

The basics of computing with qubits have been reviewed in the previous chapter. From there, four distinctive elements could be highlighted: qubits are operated in place, the topology of quantum processors typically forces two-qubit gates to be executed on contiguous targets, quantum states steadily corrupt with time, and qubits cannot be copied.

Let us make a simple metaphor in order to extract some insights from these characteristics: operating with qubits can be compared to cooking with delicate and perishable ingredients, while classical computing is closer to the job of a traditional scribe.

Both processes have a clear outcome that comes from a series of dependent tasks that should be performed in a specific order: chopping, peeling, boiling some water... and also preparing the ink, working out a structure, looking for the most fitting wording... However, many elements make both tasks differ.

First, the scribe might write some parts of the text on different pieces of paper, and later on he can just copy all of them in order to obtain the final result. On the other hand, the

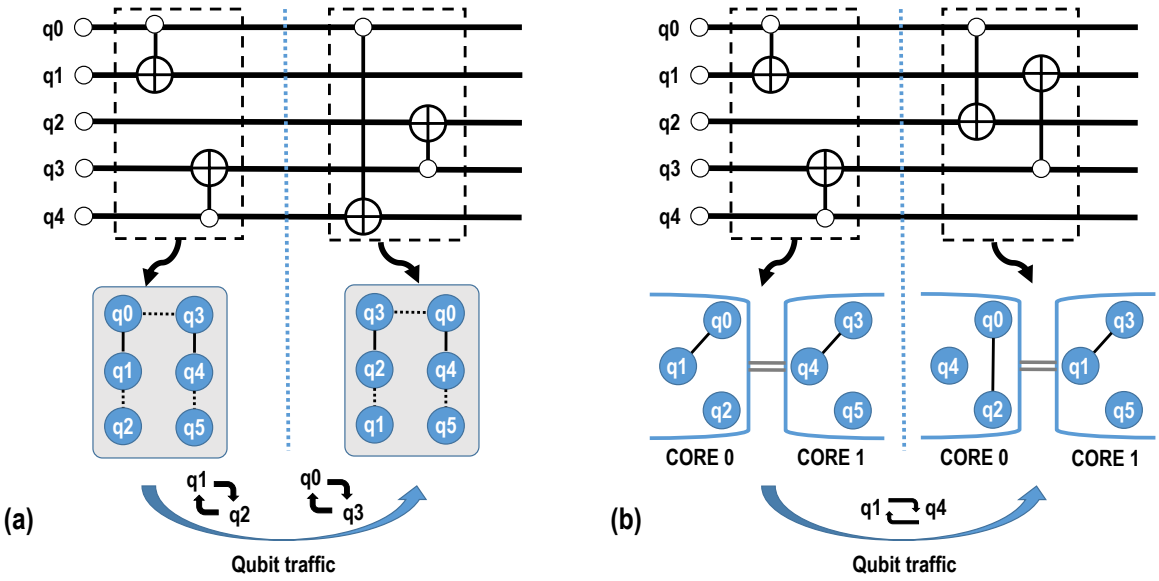


Figure 2.1: Quantum communication implied by computations with qubits. a) Single-core scenario. b) Multi-core scenario.

cooker cannot chop one carrot and boil in water another carrot, because he cannot combine the output of both tasks afterward: (qubit) operations must be performed in place.

Second, whenever two paragraphs have to be combined together, the scribe can simply make a new copy with the final combination, without changing the first copies of both texts. Meanwhile, the cooker needs to put all ingredients into the blender whenever he needs to mix them up: ingredients (qubits) need to be moved close together to "interact".

Third, the passing of time affects very differently both professionals. The paper and ink couple can produce a legible text lasting decades. Both the ink and the paper, if properly kept, can be used for writing texts for a long time. However, the cooker needs to be careful with the conservation of food and cooking timing, as some of the ingredients might corrupt: the dish (the quantum state) might be contaminated if much time passes before eating it.

Finally, even in case the scribe finds out that a copy of one of the chapters of his book is corrupting (because it fell into the river or the ink was low-quality), he only needs to copy it again into another paper. On the other hand, the cooker has to be very careful with the processes and corruption of ingredients: if he realizes any ingredient or process has altered the whole dish, he is not able to simply "copy" it in order to fix the error. The whole receipt (quantum circuit) must be carried out from the beginning, with newly acquired ingredients (qubits).

Let us now take a step further and analyze how these characteristics make quantum computing (cooking) and communicating (carrying food) much entangled: qubits are constantly moving around, latencies are critical, and communication losses equals computation failures. And multi-core quantum architectures, as in the case of precooked meals (with several "kitchens" interconnected and doing separate processes on the same food), imply even tougher requirements (specialized freight carriers, dehydrating processes, etc.).

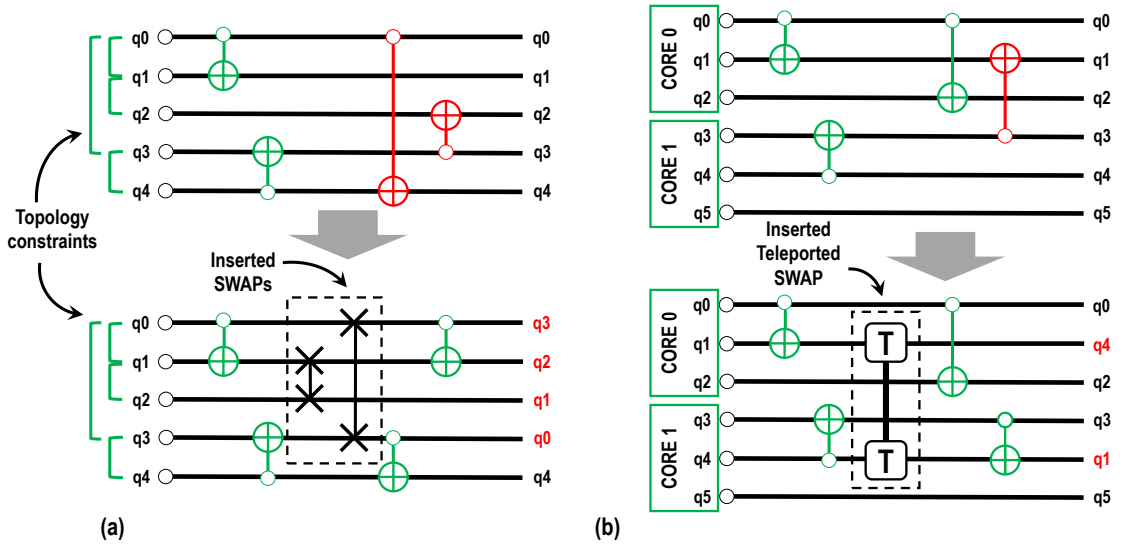


Figure 2.2: Inserting extra communication gates in compilation time. a) SWAP gates. b) Inter-core teleported SWAP.

Indeed, whenever two distant qubits are to be operated by means of a two-qubit gate, they must be moved to adjacent positions. This happens even for simple circuits. This implies that quantum circuits involve constant qubit traffic. See Fig. 2.1a for a simple example of how qubit movement is generated by swapping qubits in order to be able to execute a simple two-qubit gate. When going to a multi-core environment, this problem extends into inter-core operations requiring two qubits in different cores to interact (see Fig. 2.1b).

When moving qubits we are in fact making the execution last longer, as we introduce extra SWAP or inter-core communication operations. And, having perishable ingredients (corruptible qubits), this overhead affects the overall computing performance. In the example of Figs. 2.2a and 2.2b, where a couple of SWAP gates and a teleportation operation are inserted in compilation time, we are elongating the circuit duration by a factor of two or more. Understandably, the multi-core case is even more affected, as inter-core communications are slower than SWAP operations: latencies are from $5\times$ to $100\times$ longer [14, 51, 57].

Finally, communications should be carried out in a way that best keeps the coherence of qubits (quality of ingredients), together with a fine optimization on the timing for them to avoid inefficiencies such as idling or waiting queues that will affect the overall latency and performance. See for instance the example shown in Fig. 2.3, where in the left-hand execution the communication has been started when the channel was not cleared, thus incurring a useless wait that only allows for decoherence of the qubit being moved. In the right-hand execution, this inefficiency is fixed in the schedule, thus achieving a Just In Time approach.

Observe that these requirements do not apply to classical computing: bits always stay in cache/memory, and the operations are performed with duplicates; latencies may affect the qualitative perception of the computing performance, but not necessarily the quantitative result; very poor communication environments and network scheduling may equally affect

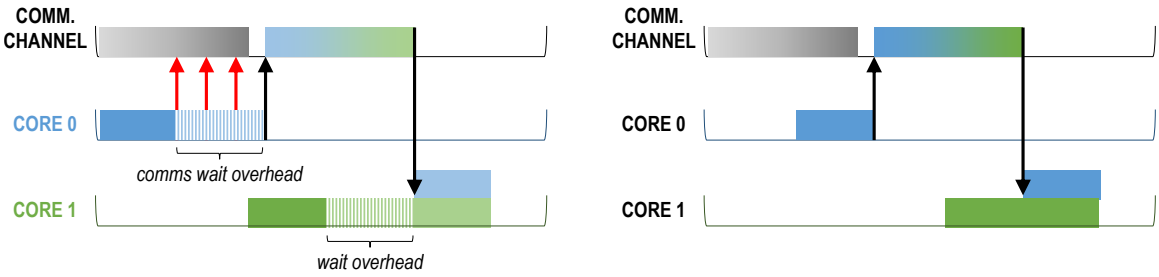


Figure 2.3: Effect of tight schedule on computation performance and fidelity.

the performance in terms of perceived responsiveness, but only extremely harsh environments will not be able to provide the correct result.

In a nutshell, a tight co-design of quantum computing processes and qubit communications is not only desirable but instrumental for leveraging the power of this approach and reducing inefficiencies for optimized performance. Moreover, multi-core quantum architectures, with their promising approach to QC scalability, will benefit from such an entangled computation-communications design by controlling the fragility of qubits in a better-isolated environment while benefiting from the surprising properties of quantum communication.

2.2 Revisiting the computer stack to allow quantum entanglement in

Layered stacks are a powerful tool to tackle complex systems in computing architectures and communications. This type of hierarchical conceptualization has already been used in some existing proposals of layered architectures for QC [58–60]. However, all of them focus on single-core quantum computers, lacking a communications perspective.

We introduce a general-purpose (i.e. no specific qubit or interconnect technology is assumed) layered stack specific to multi-core QC. We call it a *double full-stack* as it merges the traditional computing stack (application, runtime/compiler, micro-architecture, hardware) with the communication stack (routing qubits among cores, qubit reservation and swapping, etc.). As explained in the previous section, intertwining both will facilitate a tight co-design and may provide further insights on how to efficiently leverage the specifics of quantum computing and communications.

The full-stack layered architecture vision for multi-core quantum computers that we propose is presented in Fig. 2.4. The different abstractions of the quantum computer at each of the layers are included in the *stairway*: the step treads correspond to elements that configure that specific layer and the step risers its key functions. The whole network layer and the elements included in the red "wedge" correspond to the multi-chip implementation-specific kernel of the stack. Quantum data transfers in multi-core quantum computers affect all the way from the high-level code to the physical operations performed for measuring a qubit.

In the following pages, we do a quick overview of the different layers of the presented full-stack architecture in order to show how communications play a fundamental role not only in a specific part of the computation but also in the computer as a whole. Without the

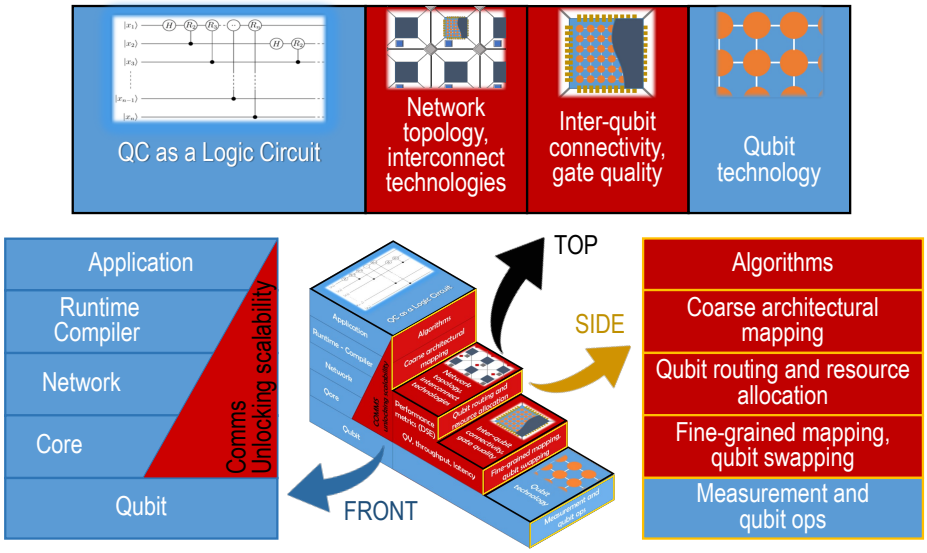


Figure 2.4: A double full-stack multi-core quantum computer vision.

communications block (in red), the stack of Fig. 2.4 is unstable. But, the question arises of whether this key block would really unlock quantum computer scalability.

2.2.1 Qubit layer

This layer is the foundation of the quantum computer, composed of each one of the qubits that can be individually controlled and read out. It could be further divided into logical and physical sub-layers when the processor implements a reliability system where several physical qubits act as a single logical qubit using QEC techniques [61].

At this level (controlling and operating single qubits) no quantum communications are involved. However, the performance of this layer is key as it will impose some limits on latencies and qubit rates of upper layers communication processes. This is particularly relevant when we consider that quantum communication is closer to "transporting physical qubits" rather than to "sending quantum information": whatever affects the physical qubit (and hence the quantum state in it) will affect the entire computation performance, and any communication involving that qubit should act consequently.

Decoherence processes as well as measurement and gate performance are the main aspects here (see Section IV in [22]). They are highly dependent upon the qubit technology (e.g. ion traps, superconducting qubits, or quantum dots) and its maturity stage (see, e.g. Section 5 of [4]).

The relationship of these parameters with the upper layer communication processes is indirect but real, e.g., the coherence time (τ_c) sets a fundamental limit on the maximum time we can operate, read out the state, or transfer the qubit before the quantum information (see corresponding subsection in Section VI of [22]) is degraded irremediably due to decoherence. Also, long gate latencies (i.e. small quality factors) have a similar effect. Qubits supporting long travels will not withstand too many operations on their already worn-out quantum state. Finally, low qubit gate fidelities are equivalent to the inverse of classical communications

error rates; the higher the better for accurate quantum information transmission. On top of that, the fact that the qubit errors and decoherence are accumulated asks for accounting for their effects on the optimization in communications and also on upper layers.

2.2.2 Core layer

The core layer is formed by a set of qubits integrated into a single chip capable of inter-operating using one and two-qubit gates. If the core is incorporated into a multi-core architecture, some of its qubits will be responsible for interconnecting the core with one or several cores, acting as transducers or communication ports (see e.g. *Linking atomic qubits with photons* subsection in [31]).

This layer performs the fine-grained qubit mapping inside the core as well as inter-core communications control (in a multi-core architecture). In a single-core quantum computer, the core layer reaches the whole physical environment of the quantum computer.

The qubit interconnection graph might follow a certain topology, whether it is all-to-all, a ring, or a regular 2D lattice. Together with the intra-core communication technology, it characterizes the *intra-core connectivity*, as well as the intra-core communication latencies and qubit transportation capacities. Finally, the control wiring and qubit technology determine a minimum *qubit-to-qubit distance*, which will impose restrictions on the minimum area occupied by the core and affect the communication latencies.

Communications are also here remarkably entangled with computations, as two-qubit operations inside a quantum core are usually constrained to contiguous locations (see e.g. *Qubit plan organization* subsection in Section VI of [22]). Therefore, qubit movement or swapping –the most basic form of quantum communication– is a constant for almost every two-qubit operation. Also, in the multi-core case, the core receives and sends quantum states from and to other cores. Two-qubit gate quality metrics are hence key for communications performance, similar to what we have just seen on the qubit layer with single-qubit gates. First, two-qubit gate latency: the time spent in performing a certain quantum operation (such as a SWAP gate for intra-core communication). And second, two-qubit gate fidelity, which represents the accuracy of a given quantum operation. Long gate latencies and low gate fidelities will affect the time and number of transfers a qubit may be able to support before losing the quantum information it stores.

Note also that the performance of this communication process will be affected by the qubit interconnection topology, the number of qubits per core, and the inter-qubit spacing e.g. a large processor with an uneven topology may need on average longer travels. In other types of communication, such as qubit shuttling, the inter-qubit spacing (which depends on qubit technology and control wiring) will determine the travel distance (and duration).

2.2.3 Network layer

This layer is fully responsible for interconnecting cores, both with a classical network (for exchanging control messages and qubit measurement results) and a quantum network, capable of transferring quantum states using any of the existing techniques (qubit shuttling, quantum teleportation, etc.). Also, it carries out the control and optimization of this inter-core communication, by means of medium access protocols, entanglement distribution, etc. That is resource reservation and network scheduling. This will strongly depend on the technology

employed. Interested readers may look for more details in Chapters 5 and 6. In case the qubit mapping is disaggregated in two steps, this layer would implement the qubit-to-core mapping defined by the upper layer.

The specific inter-core topologies and interconnect technologies (e.g. ion shuttling, qubit teleportation...) that define the connection among cores are key. These will determine the *inter-core connectivity* in terms of core-to-core distances, inter-core communication latencies and qubit transfer rates, along with other technology-specific parameters such as e.g. number and output fidelity of EPR generators for qubit teleportation, or trapped voltage and segment size for ion shuttling [32, 35].

Quantum communications at this layer imply inter-core qubit transportation for local two-qubit gates (i.e. qubit routing), although some proposals on remote gates [62] would imply an even tightened relationship with ongoing computations. Therefore, the network policies and protocols should be efficiently designed taking into account the harsh requirements in computation on the lower layers.

2.2.4 Runtime/Compiler Layer

This layer is the first logical layer i.e. abstracting out physical elements of the multi-core quantum computer. It is in charge of compiling the code to quantum assembly and coordinating the execution of the instructions together with the coarse architectural mapping (i.e. partitioning of the algorithm among the existing cores, in analogy with the *mapping* process in classical many-core computer architectures), always in pursuit of optimized processing.

This implies applying *offline* optimizations to the code (naturally for both computation and communications) by taking into account the limited resources and characteristics of the specific architecture: capacity and flexibility of the inter-core network, topology, and features of the cores, qubit technology employed, etc. At this layer, we *see* the quantum computer as a set of connected quantum cores (i.e. "processing units").

Therefore, inter-core communications, as well as some details on the capabilities and topology of the multi-core platform, are implied in the compilation and coarse mapping process. Observe that qubit traffic is almost deterministic by nature, because of its direct relationship with the computation: hence, qubit transfers are indeed loosely scheduled at this layer, though they are ultimately controlled by the network layer.

2.2.5 Application Layer

The uppermost layer corresponds to the code description of the quantum algorithm to be run on the quantum computer. This layer is hardware agnostic, meaning that low-level architectural details or constraints are not considered. There is no reference to any explicit communication operation unless it is performed within a large-scale quantum network such as the Quantum Internet.

In any case, the code might include some compiler directives enabling optimized qubit distribution and instructions execution, as it is already done in multi-core classical computing.

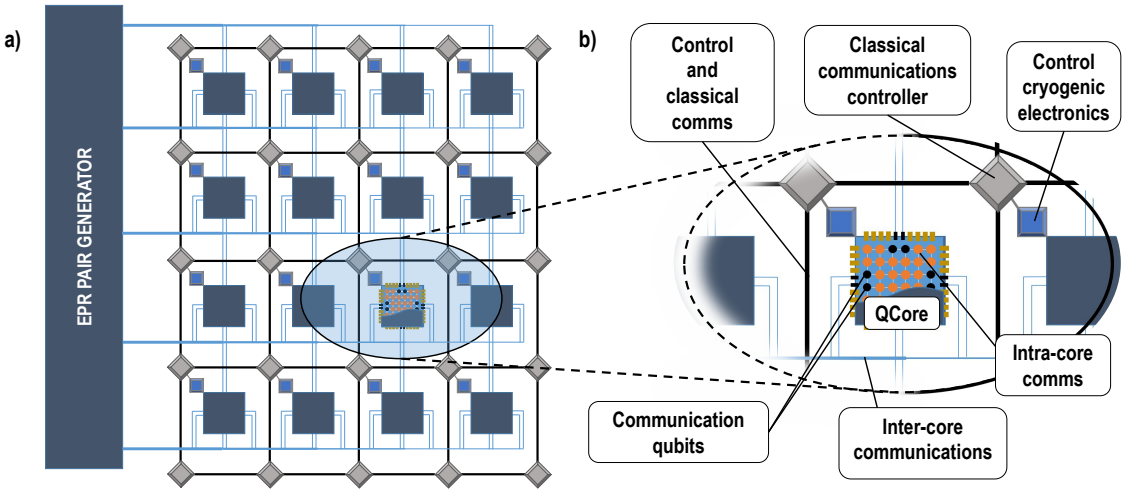


Figure 2.5: Multi-core quantum architecture vision. a) 2D diagram of a multi-chip architecture. b Detailed view of a single core and the different elements involved in computation and communication.

2.3 Developing a multi-disciplinary roadmap for double full-stack multi-core quantum architectures

Being still unclear whether QC will break current scalability limits, the vision of this work is taking multi-core quantum computers not just as an intermediate step before having fully functional large-sized quantum monolithic chips integrating millions of qubits, but as the enabler of the full power of QC. Instead of proposing a "black-box" interconnection of Noisy Intermediate-Scale Quantum (NISQ) chips, designing a multi-core architecture that entangles computing and communication with a full understanding of their intertwining requirements has to pay attention to its specific limitations while preserving its properties.

Rooted into that vision, the main goal of the thesis is to lay the foundations of communications-enabled multi-core quantum computers as a proposed enabler of the ultimate success of QC. This main goal can be decomposed into several objectives focusing on different outcomes, as detailed in Section 2.4.

This multi-core quantum computer, presented in Fig. 2.5, will cluster together dozens of NISQ cores (with tens to hundreds of qubits), connected through a quantum communications network (for core-to-core qubit transport, such as quantum teleportation or photonic switches) and a control classical network (for core coordination and job distribution), mapping the quantum algorithm among them to boost performance. In this way, we alleviate the requirements for control circuits and improve qubit isolation, while leveraging all the advantages of quantum parallelism.

This vision is sustained by three main pillars:

- **An architecture designed bottom-up from an entangled communications-computation perspective:** Some layered architectures for QC have already been proposed [58–60], but all of them focus on single-core quantum computers, lacking for a communications perspective. The presented vision goes however beyond adding mere

interconnects, encompassing instead communications and computing in a consolidated layered architecture itself –*a là* Network-on-Chip (NoC) [63]–. Although there exist some stack proposals extending quantum computers to connected environments, these approaches come from a Quantum Internet perspective, i.e. do not integrate the quantum computation process with communications: they are network stacks rather than computer architecture stacks [5, 42, 43].

- **Short-range quantum communications specific design**: While recent years have seen an explosion of research in quantum communications and networking for the QI, less attention has been placed on chip-scale communications for the scaling of quantum computers. On the one hand, for its role as an interconnect among computing cores within or across chips, quantum interconnects may recall the classical concept of NoCs. On the other hand, for their quantum nature and the need to transfer quantum data using mainly quantum teleportation, they may be compared with its *big brother*, the QI. However, the radical differences between quantum and classical communications (i.e. relevance of keeping latencies as low as possible, inability to retransmit lost qubits, etc.), and the resource limitations, together with the distance and temperature ranges, call for a specific analysis and modeling of quantum communications at a multi-chip scale.
- **Lasting design principles:** the complexity of the task, the variety of challenges, and the need to design without being able to validate experimentally a fully-fledged multi-core quantum computer is at the risk of falling into a theoretical model with too many assumptions that would hardly be implemented into an actual machine. Therefore, accurate models are needed together with a complete exploration of the design space. Moreover, instead of looking for optimal and very specific designs, this work aims at doing a first validation on the main issue (scalability of these architectures) and giving out design trends that may guide future experimental research.

2.3.1 Three big questions to be answered

"*Research is formalized curiosity*", said the american anthropologist Zola Neale Hurston. That is, doing good research is all about formulating the right questions... When looking at solving scalability issues in QC using a multi-core approach, which are the questions that may help us in arriving earlier to the most useful conclusions? We believe they can be summarized into three:

1. *Will the multi-core approach unlock the current monolithic single-core quantum computers' scalability bottlenecks?* This implies proving the main postulate of the thesis.
2. *For a given technology used to implement (or a specific application to be executed on) a multi-core quantum architecture, which is the optimal architectural configuration?* That is, providing design guidelines that allow us to foresee the requirements for multi-core quantum computers as well as how the computing power is better leveraged for different architectures.
3. *How could we improve inter-core communications in order to have fast/efficient multi-core quantum architectures?* Being communications the obvious bottleneck to any type

of distributed computing architecture, it is key to determine which physical/design parameters of the communication systems affect most to the overall computing performance.

Each of these questions opens a research line itself. In the present thesis, we lay the foundations of the three of them, willing to open the research path for subsequent works deepening in them for the advancement of multi-core QC.

2.4 Summary of this thesis' contributions

The aim of the proposed thesis is to lay the foundations of communications-enabled multi-core quantum computers as a proposed vision for the ultimate success of QC. According to the main pillars of this vision, we set as objectives of this thesis the following three main tasks/research lines:

I. Multi-core quantum architectures scalability study

Multi-core quantum architectures are proposed as an approach to unlock the scalability of QC. However, this has not been demonstrated yet. Till we have experimental resources available to build the first multi-core models, we need to confirm the viability of this approach using analytical models and simulations. In order to do that, we use DSE, evaluating the scalability of multi-core architectures in terms of the number of cores, and number of qubits per core together with qubit technology and performance, among others. This methodology allows us to:

- Explore the entire design space without being limited by the "intuition" and designer's previous experience that might hinder the way to the optimal (but maybe not intuitive) solution.
- Provide not just a single optimal analytical solution, but rather design trends and guidelines extracted from the exploration.
- Remain valid also for early design decisions when there are no analytical models or computer simulations for the performance metrics of the system.

Moreover, the results of the exploration let us compare different existing qubits and quantum communication technologies. We plan to tackle this study as an "unfolding" of models, adding more complexity layers to the study in order to focus better the exploration:

- **Analytical model:** Together with the literature review on the topic, we have developed a behavioral model describing the performance of a multi-core quantum architecture. In order to do so, we first studied how the different elements in a quantum chip and the interconnection network may interact, to then select the variables and parameters that provide valuable information for the exploration. As we are validating whether a multi-core architecture can make the difference in terms of processing power scalability, we must particularly take into account the elements of the quantum computer that may be affected by the architecture paradigm (single-core *versus*

multi-core). Most importantly, we have defined a Figure of Merit (FoM), that aggregates metrics on performance, cost, and qualitative attributes [64]. It is important to note that, not having experimental fully-fledged quantum computers, measuring the progress of QC is, right now, more about calibrating its *maturity* and overcoming individual obstacles rather than benchmarking computational power [65]. This is further developed in Chapter 4.

- **Communications overhead analysis:** The previous study (a preliminary approach whose validity is limited to the FoM accuracy) has been completed afterward with an analysis of what we expect to be the hardest bottleneck in implementing multi-core architectures: the inter-chip communications overhead. This implies studying the trade-off between communications overhead (both in terms of execution time and consumed resources) and the gain in size of the algorithms that can be executed on them. Using well-known quantum algorithms with various inter-qubit traffic profiles also gives an idea of how multi-core architectures may be adapted to specific algorithms or rather be prepared to run generic quantum algorithms. This helps not only to determine the viability of multi-core computing but also to characterize the *decision threshold* where the communications cost of distributing quantum computation among several cores pays off.

  Studies on this trade-off are available [14, 66–69], although none of them allows us to answer any of these key questions, namely: how fast should inter-core communications be in order to allow multi-core architectures to supersede traditional single-core quantum processors? For a given interconnect technology, which is the *optimal* architectural configuration that the communication costs pay off?

  This analysis is presented in Chapter 7.

- **Network Performance Scalability:** The final analysis of this "layered" exploration consists of a complete study of a fully simulated multi-core network running real quantum programs (at a scaled size). We have done so by adapting existing simulators for QC. The aim is to provide a list of design guidelines and optimal design spaces where multi-core may effectively scale QC, for a variety of applications and technologies. This study also contributes to dimensioning the proposed architecture (Objective II), by enabling the exploration of different topologies, resources, etc. The results of these full simulations are presented in Chapter 7.

II. Architecture proposal: a double full-stack comms-enabled many-core quantum computer

Although remaining mostly at the theoretical plane, several architectures for different approaches of modular quantum computers have been proposed [31, 40, 51–56]. Though we are not an experimental group and hence have not been able to develop a functioning prototype, we provide a scalable proposal (presented before, see Fig. 2.4) validated via simulation (as explained in Objective I). In particular, our aspiration is to generate an architecture that may be *i)* technology-agnostic, and sufficiently generic so as to be valid not only during the present NISQ stage, but also for large numbers of cores and qubits working together,

and *ii)* a combination of quantum computing and communications, a conception which we believe will unlock both layers and is rooted deeply inside the most unique characteristics of quantum. The outcomes of this task are as follows:

- **Entangling communications and computation into a single quantum stack:** As a way to showcase our vision of a multi-core quantum computer where computing and comms are fused together, we present a layered stack where both worlds are considered. This implies studying the state of the art in a vertical fashion, covering from the physical research on qubit technology and operation to quantum algorithms and coding, in order to be able to understand the entire process of describing and executing a quantum program. This work has also been a valuable input for the DSE, as it facilitates a structured analysis of the variables and parameters present at a multi-core quantum computer, and how they affect performance. The layered stack has been already presented in this chapter.

- **Dimensioning a many-core quantum computer:** Proposing an architecture implies not only providing an understanding and organization of the operation flow but also giving out design guidelines and decisions following that theoretical conception. In our case, after having developed the layered architecture, we have been able to explore the design space (along with the scalability analysis of Objective I) in order to find the limitations, bounds, and trade-offs in the architecture. We provide the best operation ranges for e.g. number of cores, the number of qubits per core, communication resources, qubit technologies, etc. This exploration is developed using different starting points in Chapters 4 to 7.

III. Short-range quantum communications model

As the key technology enabling multi-core QC, quantum communications at the chip scale are an important part of this thesis. Understanding the particularities of this type of communication means getting to know how quantum communications work, and what are its main characteristics when operated at such a scale. To the best of our knowledge, the research on quantum communications has been focused on large-scale communications, and hence in order to provide a valid scalability analysis (Objective I) and a complete multi-core quantum architecture (Objective II) we need to adequately model short-range quantum communications:

- **Multi-core scale communications model and simulation:** Following the literature on quantum communications at large-scale and existing experimental work on chip-to-chip quantum networks, we have developed a model of a multi-core quantum network, looking for a fit technology among the existing ones, and analyzing it from a pure communications perspective. Interesting outcomes have been latency and throughput analysis with real traffic, dimensioning of networking resources and interaction with the runtime controller and the computation flow, adapting an existing quantum communications simulator (NetSquid) [70]. The model and results from these analyses are presented in Chapters 5-7.

- **Preliminary MAC Protocol Development** To the best of our knowledge, the few publications existing yet on multi-core quantum computers have not tackled chip-to-chip communications as far as designing the needed protocol set. We have aimed at completing the architecture design and communications modeling with a first simple Medium Access Control (MAC) protocol for this environment. We envision that an efficient MAC protocol will be a key element in multi-core quantum networks, due to the prohibitive cost of qubit losses and communication latencies. This work is presented in Chapter 6.

Chapter 3

State of the Art

Connecting several quantum chips in a multi-core fashion has already been proposed as a modular approach that may enable the scaling of quantum computers. As we have mentioned in the previous chapter, this not only simplifies control circuit requirements but also reduces crosstalk errors and other impairments derived from a densely packed group of qubits integrated into a single-chip quantum processor.

Existing works on these and similar architectures [31, 40, 51–56] use different qubit technologies (ion trap, quantum dots or impurities in solids) and module interconnects (ion shuttling, photonic switches, quantum teleportation). However, all of them remain on a theoretical plane without entering into details on the actual implementation and offer a limited technology-specific approach. You can go forward to Section 3.3.2 for more details on these existing proposals.

By looking at the multi-core scale building a NoC on top of NISQ chips, our work is focused on a realistic approach that leverages a tight co-design between communications and computation to maximize the efficiency of the overall design. In this way, we are able to provide detailed analysis of the different layers (qubit parameters, NoC performance, EPR pair generation specifications, communication scheduling guidelines, etc.) for covering the existing gap, while enabling experimentalists to take benefit from these design guidelines for specific technology implementations.

Nevertheless, multi-core quantum computing as a scalability enabler for unlocking the full power of QC is just the tip of a research iceberg more than 40 years old. In order to fully understand the complexity of this QC paradigm and the existing interplay between the involved technologies, it might be of help to go through a broad overview of the research conducted during the past decades.

QC, which is now living the "Second Quantum Revolution", was born around the 80s, although to find some of its theoretical foundations we have to go to some years before and even to the dawn of the twentieth century, with the postulation and discovery of quantum mechanics.

Realizing that modeling quantum systems is out of reach for classical computers, Feynman, Deutsch and others put the foundations of universal quantum computers [71, 72]. Still in the theoretical realm, the development of paper-and-pencil quantum algorithms came afterward. In 1994, Peter Shor published what would become a turning point for QC: he demonstrated a quantum algorithm that could factor integers in polynomial time [73]. This

translated into a rise in the interest in QC as it proved the potential of this technology. It also puts at a theoretical risk all the modern cryptography, mainly based on the complexity of factoring long integers [74]. The qubit decoherence problem was also tackled early with the development of the first quantum analogs to error correcting codes [75, 76].

Experimentalists did not take much longer to achieve the first working prototype of a small quantum computer. It was in 1998 when the first experimental demonstration of an algorithm was shown, jointly at IBM, Oxford, and Berkeley. They used 2 qubits to perform a fast quantum search using Nuclear Magnetic Resonance (NMR) techniques [77, 78]. The beginnings of the 21st century led to fast advances in qubit technology and other key elements for QC development (quantum interconnects, qubit control circuits, quantum error correction...). Most recently, Google demonstrated *quantum supremacy* (a controversial term coined by John Preskill [79,80]) on their 53-qubit Sycamore processor, i.e. performed a computation that is practically impossible to execute on the most powerful classical computer available [81].

As explained in Chapter 1, although the number of qubits successfully integrated in quantum computer prototypes has been steadily increasing, some concerns on QC scalability have risen, leading research in the fields of physics, materials, electronic, and quantum engineering. Much of the research being carried out now on scalability of QC aims at extracting the most performance out of the current NISQ computers[1] by optimizing compilers [82–86], qubit mapping and routing [27, 87–92], and very importantly, QEC, i.e. techniques developed for qubit error detection and correction [61, 93, 94]. Closer to the physics are the works on improving control wiring, as well as signaling and circuits that may work at near-zero temperatures [95–99]. Moreover, several technologies for implementing qubits, with different performance advantages, have been proposed [100–104].

Modular QC, represented by proposals of large-scale distributed QC on the Quantum Internet (e.g. [40]) and short-scale efficient multi-core quantum architectures (such as the work in [56]), is hence sitting on top of a large quantity of past and ongoing research. Therefore, in order to better contextualize and understand the work in this dissertation, we are orderly presenting in this chapter the state of the art in QC as a whole. We will structure this survey by visiting the quantum computer stack layers one by one. The reader can use Table 3.1 for an overview of the state of the art and pointers to the different sections in the chapter where each layer can be found.

3.1 Quantum bits

Quantum bits are the foundation of QC, and hence research on qubit technologies, looking forward to improving their quality and control, is key. As of the time of writing this thesis, there is no dominating qubit technology yet, although some are preferred over the rest for their recent experimental performance results. However, each technology is at a different maturity stage and presents advantages and disadvantages on the various qubit quality attributes. This, far from being a problem, may also make certain qubit technologies more fitting for specific roles: e.g. multi-core quantum architectures are in need of both static,

[1] NISQ is a term coined by John Preskill that encompasses the small-sized and constrained (yet fascinating) computers built nowadays [9]

Table 3.1: Layered overview of QC state of the art

Layer	Main research focus	Challenges	Details and refs.	Section
Qubit	Qubit technologies	Qubit coherence time	Ion trap [47, 105–107], superconducting [12, 103, 108–110], solid-state [101, 102, 104, 111–117] and photonic [118–123]	3.1
Core	Qubit interaction	Qubit count (in a single chip), qubit control & isolation, gate fidelity, Fault Tolerant (FT) QC	Platforms: ion trap [46, 47, 124–127], superconducting [1, 10, 81, 108, 128–130], solid-state [45, 131–133], photonic [11, 13, 122, 134]; cryogenic control circuits [97–99, 135, 136] and QEC [137–140]	3.2
Network	Quantum interconnects	Qubit transfer rate and fidelity	Ion shuttling [32, 141], mw photon-mediated coupling [142], entanglement generation and distribution [143–147], quantum teleportation [34, 35, 148], remote quantum gates [62, 149], light-to-matter [51, 150, 151] and hybrid qubit platforms [152–155]	3.3.1
	Multi-chip proposals	Fully-fledged experimental demonstration, HW/SW and computation/communication co-design	Ion trap-based [28, 31, 51, 54–56, 69, 156–162], spin-based [52], superconducting based [1, 163–165] and photonic-based [44]	3.3.2
Runtime/ compiler	Compilers: mapping and routing	Algorithm complexity due to architectural constraints, optimal routing, mapping in multi-core QC	Circuit synthesis [166–168], QEC integration [91, 169, 170], qubit mapping optimization [87, 89, 92, 171], scheduling [27, 88, 90], multi-core targets [14] and distributed QC on QI [172].	3.4
Application	Quantum languages	Plethora of available options, dependency on low-level languages	Functional languages [173, 174], imperative languages [67, 82, 175] and assembly languages [176–180]	3.5

memory-focused qubits (i.e. long coherence times and high fidelity control and read-out), and mobile qubits (i.e. low interaction with the environment, kept at room temperature), intended for communications, with the ability to transfer quantum states from one another.

Proposals for quantum computer implementations include largely developed candidates such as ion traps or superconducting qubits, promising candidates such as quantum dots and other solid-state proposals (such as Nitrogen-Vacancy (NV) centers in diamond and silicon-based nuclear/electron donor spins), and many others, including photonic and topological qubits [2, 3, 8, 181, 182].

Being the most prominent technologies for now, we will review Ion traps, Superconducting qubits, and three types of Solid-state spins: quantum dots, donor spins in silicon and NV centers in diamond, and photonic qubits.

A visual comparison among qubit technologies in terms of their main quality attributes is presented in Fig. 3.1. Photonic qubits are not included due to being conceptually different and thus not comparable using these metrics.

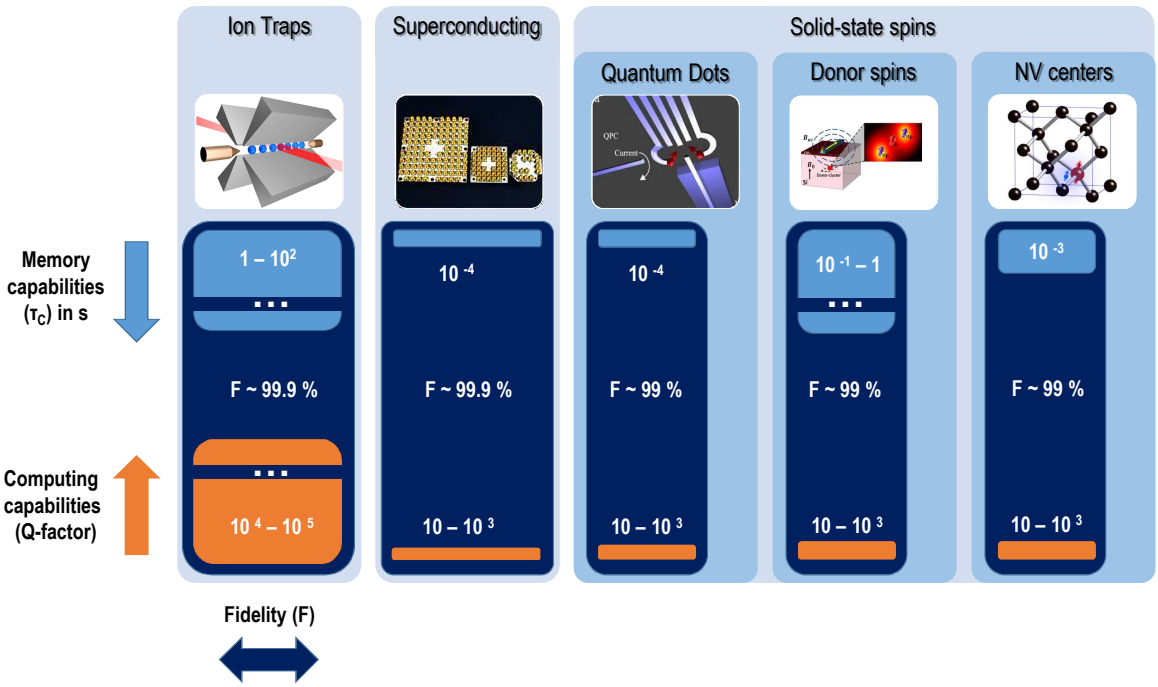


Figure 3.1: A visual comparison among different quantum technologies. Fidelity percentages and quality factors correspond to two-qubit gate values.

In the figure, three different characteristics are depicted for each technology: coherence time, quality factor, and gate fidelity. Instead of just providing a table with figures and metrics (which have been extracted from the refs. above), these parameters are represented geometrically: the height (or width) of each box is proportional to the value of the parameter for that technology. As the coherence times of both donor spins in Si and ion traps (as well as quality factor in ion traps) are out of scale, their corresponding boxes are represented as a "discontinued" surface. In this way, we can easily compare the performance of the different technologies (i.e. of the most recent experimental results using each of them). During the last years, all technologies have been able to demonstrate fidelities greater than 99%, being the bare minimum fidelity that QEC protocols are able to correct. However, some of them have reached that quality only in two-qubit lab experiments.

Ion Traps. In this implementation, qubits are represented by the energy levels of electron spins in single ions, which are trapped together in a vacuum (Paul traps) using electromagnetic fields. Ion traps are reliable and present low error rates (being charged atoms, they can be manipulated with high precision): these qubits have showcased coherence times of tens of seconds and single- and two-qubit gate fidelities of 99.9999% and 99.9% respectively [47]. However, operating with them is slow and complex [105–107]. As they led the way to high-fidelity operation, they are also pioneering experimental fault-tolerant logical qubits with impressive results [137, 138].

Superconducting qubits. Using well-known CMOS technology, these qubits are "artificial atoms" implemented in superconducting circuits at cryogenic temperatures as resonant circuits translated into discrete energy-level systems thanks to the non-linear behav-

ior of Josephson junctions. The information, encoded in different ways (charge, flux, or phase qubits) can be electrically controlled via microwaves, voltages, magnetic fields, or currents [103]. Leading tech companies such as IBM and Google are betting on this technology. While they started lagging behind ion traps' qubit lifetimes and error resistance, they have been rapidly progressing: average lifetimes are still below the millisecond [12, 108], but the gate fidelity has rocketed recently from 99% up to 99.91% [108–110]. They are also the technology of choice at Google for demonstrating a complex development of quantum computers such as QEC [140]. All these advances imply that superconducting qubits are currently leading the quantum race.

Solid-state spins. To avoid the issues found in ion traps with cooling and controlling a large number of atoms in a vacuum, several qubit technology proposals are based on "artificial atoms" integrated into solid-state hosts, such as quantum dots (nano-structures of trapped electrons, which could be said to be the extreme one-electron case of a transistor) [101], NV centers in diamond (a platform which works well at room temperatures and is a fit candidate for interconnecting quantum processors for its good optical properties) [104], or donor spins in silicon (e.g. P:Si) [102]. Although less consolidated than superconducting qubits or ion traps, they provide fast operating and long-lived qubits. Moreover, the silicon-based implementations benefit from the know-how of the global silicon manufacturing experience. Most recently, the three mentioned technologies have well surpassed the 99% fidelity barrier [111–113]. The research community behind solid-state spins is dynamic with several open ongoing efforts for better substrates and techniques [114–117].

Photonic qubits. Encoding the quantum state into polarization, time-bin, or path, photons provide a room-temperature noise-resistant platform for qubits, which is by nature ideal for transporting quantum information [118]. On the other hand, its inherent mobility and the resistance to interacting with the environment make it difficult to operate with other qubits and to store them, leading to important research on light-to-matter transducers [119, 120]. In the last years, research on programmable integrated photonic circuits has come to bear fruit, using high-quality single-photon sources and detectors, and also advocating for an alternate form of quantum computing called measurement-based QC [121–123]. However, it adds constraints on working at low temperatures.

The interested reader can refer to [6, 47, 130, 183, 184] for a deeper dive into existing qubit technologies.

3.2 Qubits operating with other qubits: interaction VS isolation

At this point, we are certain about the fragile nature of quantum computing, as any electromagnetic disturbance from the environment may affect the quantum state being operated. That is the reason for having quantum computers running completely isolated inside cryogenic freezers, at almost absolute 0 temperatures. In fact, the ideal scenario would be having every single qubit perfectly confined, getting never-seen-before coherence times. However, making a quantum computer implies getting several qubits to interact, breaking therefore qubit isolation. Finally, we cannot be satisfied by just having a perfectly isolated quantum

chip, that is able to run a program if we are not able to control and read the final solution out of it.

That is, QC is born as a contradiction itself: the more isolation for the qubits, the better; but it also holds true that the faster and simpler the qubit control and read-out (i.e. the access from outside), the better. Such a design problem has been present since the dawn of experimental QC, and is still demanding the hardest work from material engineers and physicists: we need qubits that operate efficiently with other qubits and can be easily controlled while having long coherence times.

According to the commonly used gate model, quantum algorithms are described by quantum circuits composed of one- or several-qubit gates, and results are read out using measurement operations. Also, qubits are initialized using specific preparation operations. Though the measurement operations collapse the quantum state of the measured qubit and hence cause an irreversible information loss, all operations on the quantum system always aim at minimizing its disturbance on the system. The physical implementation of these operations requires independent control circuitry, which depends very much on the specific qubit technology employed. In the following, we will detail the state of the art per qubit technology, in order to give a general perspective on the hard work done and open challenges for integrating as many qubits as possible in a single processor for QC:

- **Ion Traps.** Usually operated at room temperatures, ions have been typically controlled using laser fields, though lately the use of microwave pulses is gaining momentum, as they promise unique scaling benefits and fidelities beyond 99.99% [124]. Though showing high-fidelity operations, these have been demonstrated in small lab experiments, rather than in multi-qubit devices. Advanced prototypes with $\sim$ 20 ions are available [46], but experimental physicists are struggling with scaling up to more than 50 qubits [47]. In particular, scalability-enabling miniaturization is a challenge, though some micro-fabricated 2D RF-trap arrays have already been successfully demonstrated [125], and even a compact rack-mounted demonstrator computer has been presented [126]. On the other hand, qubit-to-qubit connectivity is not limited to nearest-neighbor interactions, which simplifies two-qubit gate control. Lately, some proposals on using human-made ions that may have better properties for housing qubits and integrating with photonic devices have been announced [127].

- **Superconducting qubits.** Superconducting qubits can be controlled using microwave pulses. Although there exist fixed frequency approaches, tuneable superconducting chips allow to selection of specific qubits to operate on by changing the frequency of the pulse [108]. The possibility of reusing existing classical chip technology to build them, as well as their fast operation, has raised the interest of industry players, such as Google or IBM. Such investment has already largely paid off, as this technology is already leading the qubit count and integration complexity (433 qubits in a single chip at the time of writing [10], with a multi-layered architecture including qubits, readout resonators and control lines on different layers together with integrated filtering for noise reduction and greater stability) and the quantum supremacy demonstration [81]. Measurement and classical control electronics are also continuously progressing, while struggling with the scalability challenge of avoiding control

crosstalk, spurious cross-coupling, imperfections in fabrication reproducibility of device parameters [128, 129]: it is predicted that the limit on qubit count for a single chip will be on the thousand qubits [1, 130]. One of the latest proposals uses superconducting circuits also for control, providing great flexibility and a bright future for scaling [136].

- **Solid-state spins.** Donor spins and quantum dots, exploiting the semiconductor industry's well-known technologies, are also controlled electrically, in a similar way to transistors in digital electronics. Nevertheless, global microwave fields have also been recently proposed for driving gates, as in some trapped ion experiments, due to their potential scaling advantages [131]. On the other hand, impurities in solids, such as NV centers, are typically controlled by a combination of techniques from liquid state NMR and optical manipulation. These systems can be operated at room temperature, though fidelity improves at lower temperatures. Though they show outstanding performance on qubit lifetimes and single-qubit control and are predicted to scale over 10^5 qubits in a single chip [132], it is still hard to go beyond a dozen qubits employing these technologies [45, 133]. Some of the reasons behind this are the only recent high-fidelity in two-qubit operations, as well as poor fabrication reproducibility of qubits.

- **Photonic qubits.** Miniaturization and integration of photonic platforms have allowed for higher programmability and scalability while leveraging nano-feature accuracy that industry fabrication processes can produce. This theoretically implies being able to integrate thousands of detectors, sources, and waveguides on each wafer. It also calls for going into cryogenic temperatures, and following the measurement-based QC model, which is equivalent to the well-known gate-model [122]. This model comes with its own share of challenges and has yet to experimentally prove its validity for large-scale processors, although it promises to hold thousands of qubits in a single chip [44]. Of course, moving away from almost absolute zero operating temperatures is also desirable. While crosstalk is now not an issue for photonic qubits, it might enter the scene when stretching qubit count up. However, photonic qubits are in good shape, having been able to compete with Google's quantum supremacy results, first with a 50indistinguisable single-mode squeezed states [134], later by means of a 144-mode photonic platform [13], and most recently with an impressive publicly available photonic platform from Xanadu with 216 squeeze-state qubits [11].

We have already summarized the scalability potential of these qubit technologies in Fig. 1.7. Photonic qubits, with a recent explosive growth and promising potential, substantially contrast with solid-state spins (high potential but still stuck in a growth bottleneck) and ion traps (once they led the race but single-chip scalability has made them hop quickly into modular alternatives – see Section 3.3.2).

Though the complexity existing on qubit control techniques is high, as we have checked through the literature review above, it is only one of the several challenges in integrating thousands of qubits in a chip. Let us move on to the next one: classical control circuitry, in charge of translating logic commands to analog inputs (electric or microwave pulses, laser fields, etc.) and quantum-classical interfacing for readout and processing of measurements. When applicable, it can also perform QEC-related computations.

Up to now, only a few off-the-shelf quantum processors are available, and we can safely consider them prototypes for their qubit count and performance limitations. Most of the cutting-edge experiments showcasing record-breaking fidelities are performed using qubit platforms which are controlled using expensive and full-sized equipment such as microwave sources, Analog-to-Digital Converter (ADC), and high-rate waveform generators. They work at room temperatures, present high power consumption rates (around 1kW per qubit [22]), and have direct high-quality connection lines to the chip: scaling these platforms up to 10^6 qubits would mean thousands of racks and an unacceptable amount of cabling connecting the chip (typically at cryogenic temperatures) and the control system, resulting in an unprecedented challenge for the refrigerator to keep the temperature with such high heat dissipation. Moreover, we would face highly demanding transfer rates for individually controlling such an amount of qubit devices.

To this end, integrated, miniaturized, and multiplexing-based control platforms are desirable [96, 129]. However, that is not sufficient: allocating local control circuitry closer to the qubits, in a fully integrated design approach, would alleviate some of these harsh requirements [52]. This implies, for most of the qubit technologies, having cryogenic control circuits, which comes with a big challenge of having to work with really low power budgets to minimize heat dissipation and thus keep the system within the typical capabilities of cryogenic refrigerators. On top of this complex but desirable control device, we should look for highly demanding requirements for realizing qubit control according to the different qubit technologies: stable DC voltages with very high resolution (of μV to mV), baseband AC control displaying bandwidths ranging from DC to a few hundred MHz, waveform generators with up to 1 GS/s sample rates [46], control currents reaching over 1mA at zero resistance, or up to 20 GHz microwave carrier frequencies for qubit control [22].

Research on such a complex endeavor is still ongoing, with exploratory experiments working below 4K looking for the performance and limitations of this approach [97–99]. More recently, the industry has also shown some impressive results on cryogenic control chips, for spin qubits (Intel and QuTech Horse Ridge chip [135]), superconducting qubits (using superconducting circuits also for control [136]) and close future prospects on achieving similar approaches on photonic platforms.

The last challenge at this qubit interaction layer, but not least, comes the need for integrating FT QC, by means of QEC techniques (you may revisit Section 1.1.3 for refreshing the main concepts behind it). Even if qubit technologies ever evolve up to really low error rates (around $10^{-12} - 10^{-15}$ is said to be required [25]), QEC provides an extra layer of performance, but at a high cost in number of qubits and computation time: in [25] an overhead ratio of around 5×10^4 is calculated for an example involving surface-code-based QEC on Shor's algorithm for factoring a 2000-bit number.

Although most of the current prototypes are focused on increasing the qubit count, coherence time, and gate fidelities, some advances have already been presented on experimentally implementing logical qubits for fault-tolerant qubit platforms, as well as small working implementations of QEC. Ion traps and superconducting qubits are leading these demonstrations, with several logical qubit platforms [137,138], and implementations of color codes (from Honeywell's trapped ions technology [139]) and repetition codes (from Google's superconducting qubits, on their way to full 2D surface codes [140]).

3.3 Interconnecting quantum chips

Even though some qubit technologies are rapidly advancing and leading the QC endeavor to quantum supremacy and fully-fledged computers, the review of the previous section has clearly stated some scalability problems for all of them. While large-scale QC needs several magnitudes more qubits, we are still struggling with integrating only a few more than a hundred. And controlling issues are adding up to the complexity.

Therefore, modular approaches where several chips are interconnected to jointly operate are being included in every quantum industry developer roadmap. The difficulties reviewed in Section 1.1.5 regarding the fragility of transferring quantum states through quantum coherent links are being tackled by the research community by studying and refining existing qubit interconnects as well as presenting some proposals for such architectures.

3.3.1 Quantum Interconnects

Quantum interconnects are devices that are able to transfer quantum states between different physical media. Modular QC proposals basically aim at joining separate chips using these interconnects (i.e. quantum coherently connecting them in order to enable qubit transfers among them). Two main approaches exist: direct qubit transportation and entanglement distribution.

While the first has been experimentally demonstrated in several qubit technologies [144, 148], channel losses provide poor performance even in highly-curated experiments. However, approaches such as ion trap shuttling and microwave photon-mediated coupling in superconducting circuits feature higher fidelities for short communication distances involved [32,142]. Most recently, a team from the University of Sussex and Universal has shown a proposal of ion transport with large improvement in rate and transfer fidelity (2400 per second and 99.999995%) [141].

Entanglement distribution uses a pair of previously entangled flying qubits (typically photons) to facilitate distant operations between modules, using remote quantum gates [62] or quantum teleportation [35]. Key performance metrics of this technique are the rate and fidelity of the generated entanglement. There have been several experimental demonstrations obtaining reliable entanglement for different technologies: trapped-ion systems [143], neutral atoms [144, 145], NV centers in diamond [146] and quantum dots [147], with only moderate entanglement generation rates yet (the most recent on ion traps showcases 82 per second with fidelity 94%). However, the higher flexibility provided by entanglement-mediated communication puts high expectations of substantial increases in these metrics. Moreover, quantum teleportation enabling distant gates between separate modules has also been demonstrated [34, 149]. You will find more details on quantum teleportation and entanglement distribution in Section 5.3.

In modular QC architectures, quantum interconnects may be used in different scenarios:

- **Light-to-matter**. For their capabilities as qubit carriers with low interaction with the environment, use of standard optical components for control as well as high-speed low-loss transmissions, photons provide the best solution for qubit transportation for all distance ranges [51, 150]. In modular QC, having short distances (inside a room or in a chip), low frequency (mm-Wave or microwave) photons may be used. These

photons may be used for direct qubit state transfer or entanglement distribution, but in any case these interconnects need to provide a high-fidelity coupling. For such an endeavor, optical cavities or waveguides behave best in the single-photon regime. High-fidelity requirements must be met: operating at short wavelengths, low insertion loss, and compatibility with qubits' mK operation temperatures [151]. Deterministic and active light-to-matter teleportation has been realized with light pulses for atomic ensembles, photonic qubits, and cavity quantum electrodynamics for various types of trapped ions, and quantum dots for solid-state systems, with increasing fidelities up to $\sim 90\%$ (although for small scenarios, with 2-3 qubits) [35].

- **Matter-to-matter**. The wide variety of qubit technologies, together with the contrasting advantages that each one presents, may also lead to task-driven modularity, where different qubit technologies are used for executing tasks with disparate requirements. For example, qubit technologies presenting long coherence times, such as ion traps, may be used for memory purposes, and superconducting qubits could be fitting for processing, for their fast operation [152]. Interconnecting such diverse modules is a hard challenge itself, but some experimental results are already available [153, 154]. A promising approach is that of superconducting resonators and cavities, for their wide compatibility with atoms, ions, quantum dots, and others [22]. A qubit technology that also may provide high flexibility for small modules is NV centers in diamond: it combines a memory qubit, able to both keep quantum states for a long time, and a communication qubit, with single-photon emission properties. Most recently this technology has proved to keep three nodes entangled at 4K, at QuTech [155].

In summary, interconnect technologies are still a cutting-edge research topic, looking to achieve fast quantum communication rates that may be competitive with the current decoherence times. This is hard yet at the short-scale (communications within a chip), but future scalability might also require coherent communication in a room, among separate computers [163].

3.3.2 Multi-chip proposals

Despite the technical hurdles in demonstrating experimental quantum interconnects, proposals on multi-chip quantum computers as scalability enablers are not new. Since the early 2000s, some theoretical architectural developments have been presented.

In [156], authors propose an architecture under the name of QCCD for ion trap technology, using ion shuttling to transport qubits between separate arrays, specialized into memory and interaction regions. Remarkably, this architecture has been recently demonstrated on a programmable trapped-ion quantum computer with low error rates comparable to those of the individual qubits [157]. The authors have been able to realize a teleported CNOT gate achieving a Quantum Volume (QV) = 64.

QLA ion trap architecture was presented by Metodi *et al.* [158], where each module is able to perform a two-qubit gate, together with ancilla generation and teleportation with neighboring modules. This concept was further developed with greater flexibility with CQLA [69], Qalipso [159] and Requp [160] architectures.

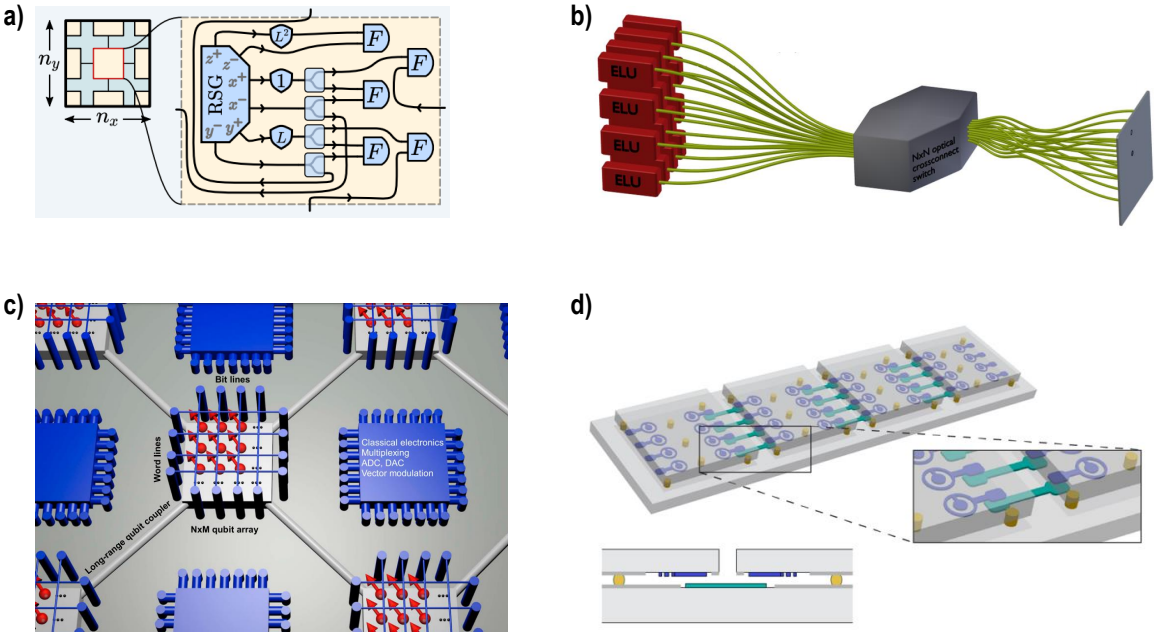


Figure 3.2: Some modular architecture proposals for QC. a) Interleaved modular architectures for quantum photonic platforms [44] b) Optically switched ion trap modules in Modular Universal Scalable Ion trap Quantum-Computer (MUSIQC) proposal [51] c) Sparse qubit arrays with integrated local control electronics for spin qubit technology [52] d) Modular architecture for superconducting qubits as proposed by Rigetti [164].

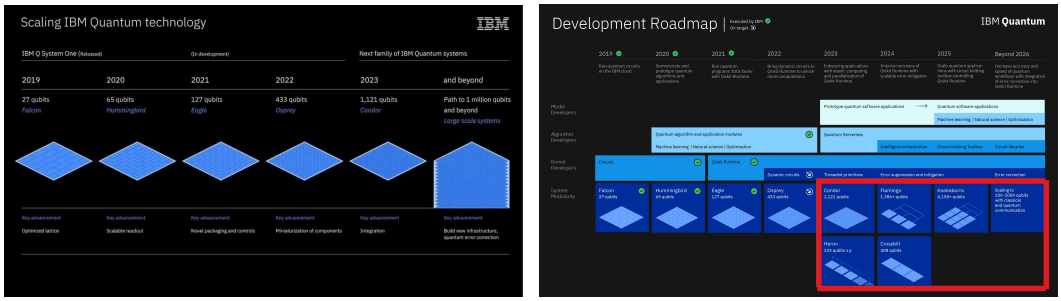
Also assuming ion trap technology, researchers at UC Berkeley presented early exploratory designs for teleportation-based distributed architecture [56], highlighting the influence of communications on overall fidelity and performance, and the amount of resources required for such a system.

Another work from that time [54] presents a distributed architecture connection through photonic links several five-qubit quantum registers (assuming either ion traps or NV centers for their long coherence times), and studies its performance assuming high-fidelity local operations but high error in register-to-register communications.

One of the authors of the original QCCD paper has more recently proposed a scalable modular architecture called MUSIQC, which is based on photonic interconnects and entanglement distribution, and tackles the technological challenges in true scalability of QCCD beyond 100 qubits [31,51]. Linke *et al.* [28] experimentally compared the performance of the base ion-trap module for such architectures with the IBM Quantum Experience superconducting platform, showing how qubit connectivity and quantum hardware-software co-design will be instrumental for QC scalability.

Following up on these ion trap proposals, Ahsan *et al.* combined the MUSIQC architecture with the QLA proposal, to provide more flexibility [161]. Most recently, SAQIP architecture was proposed as an improvement to all previous ion trap architectures derived from QCCD [55].

Even though ion trap, for its longest history and leading performance, is almost monopolizing the multi-chip modular architecture proposals, other technologies are also getting on



(a) IBM Roadmap as of 2020 [185] **(b)** IBM Roadmap expansion of 2022 [1]

Figure 3.3: Evolution of IBM Roadmap to include multi-chip approach past year. The red rectangle marks the added multi-chip prototypes.

board. For instance, regarding spin qubit technology, authors in [52] proposed sparse qubit arrays with local control electronics in between to allow for better scalability. Following their bet on fusion-based QC [122], scientists at PsiQuantum have also proposed a modular architecture for scaling their photonic chips to million qubits [44]. Most recently, LaRacuente *et al.* [165] have proposed a distributed architecture based on superconducting processors interconnected through microwave links, providing valuable insights on the promising prospects of the multi-core approach. Their focus on bridging network and physical layers follows our call for co-designing hardware and software at all computing and communications stack layers (explained in Section 2.2), while in that contribution they limit the analysis to experimental analysis of a given qubit and interconnect technology.

Very importantly, all this research interest on multi-chip architectures has also reached industry, and not only for ion trap technology, with IonQ, Rigetti, and IBM including this approach in their roadmaps [1, 162, 164]. In the case of IBM, as shown in Fig. 3.3, the addition of the multi-core approach as a scalability enabler was made public as an extension to their previous roadmap in 2022.

3.4 Mapping, routing and translating code: quantum compilers

The task of the quantum compilers is broader than that of their classical counterparts: partially because current technology limitations require much more detailed optimizations and hardware-tailored implementations. The tight resource constraints that NISQ systems impose in terms of the number of qubits available (circuit width) and coherence time (limiting the total execution time of a single circuit or its depth) require aggressive optimizations, that must be applied during the compiling process. In this way, the few resources available are completely squeezed.

These optimizations can range from algorithm simplifications (hardware independent), such as gate transformations and decompositions that lead to further reductions, to gate-level qubit-technology-related operations. This need for deep optimization leads to increasing computational resources being consumed in the compilation process. Research on these

optimizations for the very specific realm of QC algorithms is being carried out by a large community. In the following, we will briefly review the main lines for optimization in quantum compilers: the interested reader can dive into [186] for a larger outlook.

Synthesizing quantum circuits involves decomposing and translating them into a series of operations pertaining to a universal set, fitted to a FT implementation available in the target hardware [166–168]. With that translated version, if QEC is implemented in the platform where the program is going to be executed, logical qubits are encoded into multiple physical qubits, and the instructions are thus expanded [91, 169, 170].

With that hardware-adapted code, the next step is mapping the virtual qubits in the circuit to the physical qubits [87]. The most constraining requirement to be met in this process is the topology of the quantum platform: not all operations can be performed in all qubit positions, and typically two-qubit gates have to be performed in adjacent positions. Thus, SWAP and MOVE operations need to be introduced to *route* the qubits into the required places. This process, leading to a latency overhead, is also to be precisely optimized in order to minimize the degradation of the whole circuit performance [89, 92, 171]. Within the mapping process, the scheduling of operations is also performed, which aims at exploiting the available parallelism while meeting qubit control constraints as well as operation dependencies [27, 88, 90].

As multi-chip QC architectures are still in their nascent stage, only a few compilers have been designed for this scenario. Some of the proposed modular architectures include a specific compiler [55, 67], but also there are recent works devoted exclusively to compilers for distributed/modular QC, such as [14] and [172], with diverging approaches.

In [14], authors tackle the compilation of a quantum circuit for a multi-core target architecture by leveraging partitioning algorithms to allocate qubits to the different nodes for each timeslice of the algorithm and inserting afterward the required communication overhead operations in between. This approach is hardware-agnostic, though without considering typical requirements, such as intra-module topology constraints.

Cuomo *et al.* [172], however, focus on a large-scale distributed platform and take for granted an underlying entanglement distribution layer that provides continuous inter-node communication. Moreover, they make use of remote gates [62]. On top of that system, they develop a highly curated solution for what they call *the distributed quantum compilation problem*.

3.5 Speaking primitive quantum languages

Because of the complexity behind current NISQ quantum computers, existing quantum programming languages exhibit a low level of abstraction. Therefore, most of them follow the circuit model, sometimes providing libraries or high-level definitions for embedding or abstracting complex well-known routines.

Although the development of such languages is still ongoing, there is a wide variety of available options already, including functional and imperative languages. Current examples of the first group include Quipper [173] and LIQUi|> [174], whereas others such as Scaffold [67], ProjectQ [175] or OpenQL [82] fall into the second one. Most of them are in fact extensions of current classical programming languages and include complete tool flows for programming and compilation.

Right below these rather high-level languages, there exist some low-level assembly languages, also specified at the gate level. The variety and rapid development of existing software, together with the already mentioned need for curated optimization of the final code, make these assembly languages much more familiar to quantum researchers than it would be desirable. These include QASM [176] and its derivatives, such as cQASM [177], eQASM [178] and OpenQASM [179, 180].

At the time of writing, to the best of our knowledge, still there is no tool flow or programming language specifically developed for multi-core quantum architectures. Although the reference to the modular nature of the computer could be hidden from the programmer, developing such a tool will be of interest for *i)* the need in current NISQ QC to optimize the code with the most details of the target hardware, and *ii)* provide the programmer with some control over the distribution of the code, as it is already done in classical distributed computing, for leveraging any information on the algorithm structure for further optimization.

Chapter 4

On a Design Space Exploration for Double Full-stack Communications-enabled Quantum Computers

Having now fresh in mind the colossal research efforts on QC and their unstoppable advance in the last decade with commendable results, it might be easier to bet on the success of such promising technology, assuming scalability-enabler approaches such as multi-core quantum architectures.

However, we should not forget the *quantum leap* there remains from today's prototypes to fully functional and useful quantum computers[1]. Moreover, multi-core quantum computers come with their own set of challenges. Certainly, putting together currently available small-sized computing nodes as opposed to packing more qubits into monolithic quantum chips alleviates the requirements for control circuits and improves qubit isolation. Nevertheless, interconnecting quantum chips is far from being a simple task (more details on this in Chapter 5).

Therefore, in order to lay firm foundations of scalable multi-core QC, there is a need for a systemic approach. In particular, we are looking for a well-reasoned answer to the first of our three questions (if curious or in need of a refresh, check Section 2.3.1). That is, analyzing whether multi-core architectures can effectively enable a scalable quantum computer, which are the resource overheads and computational costs of such architectures, and even which qubit type or communication technology will behave best. In other words, substantiating that quantum computing scalability (and ultimately, quantum computing culmination) may not be possible without multi-core architectures enabled by communications, in a Quantum Network-on-Chip (QNoC) fashion.

We have aimed at doing such a systematic approach in this thesis by means of a "V"-like procedure. We have started with a top-bottom scaffolding of the whole architecture with the double full-stack layered architecture proposal (see Section 2.2 in Chapter 2). Second, we

[1]Remember from the previous chapter that most advanced QC prototypes reach a few hundred qubits, while more than a million are required for executing algorithms of interest.

analyze specific elements of the stack for optimizing them (Chapters 4 to 6), with a specific focus on communications, and finally, we go up again into full network simulation (Chapter 7), having thus a complete view of the full system. In these comprehensive analyses, we aim to find out the device and architectural parameters that are optimally enabling multi-core QC. For such a complex endeavor, we have employed DSE for its virtues on system-wide optimizations.

In this chapter, we set the framework for this exploration and start diving into it. After introducing the DSE technique, we present a DSE formulation to compare the multi-core and monolithic single-core approaches, based upon the double full-stack layered architecture from Chapter 2. This helps us to find a sweet spot (or region of the design space) where the design performs better by examining the parameter space (i.e. qubit technology, qubit and core interconnects, number of cores...) and evaluating it with performance metrics.

The chapter is structured as follows. Section 4.1 is devoted to explaining DSE. Section 4.2 details how this technique can be applied to QC, with a brief state-of-the-question on performance metrics for QC. Then, in Section 4.3 the modeling and results of the first analytical scalability exploration and quantum technology gap analysis are shown, before drawing some conclusions.

4.1 The Design Space Exploration technique

DSE is a structured design methodology that allows optimizing a system by maximizing a given cost function –or FoM– based on some parameters of interest that describe the performance, quality, or overall cost of the solution [187,188]. Like any other structured design process, this optimization relies on modeling the interdependencies among the different performance metrics and the variables describing the system. This modeling process might include analytic/theoretical expressions, behavioral models, computer-based simulations, or their zone-wise combinations.

However, it is important to note that DSE is used to *design*, not just to *optimize*. Performance metrics optimization is in fact just one of the DSE use cases, as it is also useful for rapid prototyping or system integration with no need for analytical metrics [188]. Indeed, DSE uses the optimization framework to control the design process by looking for trends and guidelines in the system performance when varying the available parameters. Whatever the design problem is, if the analysis is correctly prepared, DSE will not blindly look for "the extreme-case highest-performing scenario", which could be unpractical or ignore sub-optimal options that may suffice for the actual context or the resources constraints. Rather, the main virtue of DSE is to be able to consider system-wide trade-offs and different metrics that may also affect the design optimality.

For example, a DSE analysis of a network deployment will not optimize the average throughput of the entire network but will take into account deployment costs and qualitative characteristics such as network reliability or flexibility. DSE achieves this by letting the designer concurrently sweep all the open variables in the design space, instead of "manually" tweaking them in a one-by-one approach and consolidating several performance/cost metrics into a single FoM, which is then optimized.

DSE can be applied to any design problem, from a very specific single-variable local problem to a System Architecting Problem (SAP), such as the one we are facing. In either case, DSE optimization can be generically formulated as in:

$$\begin{aligned} \max_{\boldsymbol{v}} \quad & \Gamma = f\left(\boldsymbol{J}\left(\boldsymbol{v},\boldsymbol{p}\right)\right) \\ \text{s.t.} \quad & \boldsymbol{i}\left(\boldsymbol{v},\boldsymbol{p}\right) \leq 0 \\ & \boldsymbol{e}\left(\boldsymbol{v},\boldsymbol{p}\right) = 0 \\ & \boldsymbol{v} \in D \end{aligned} \tag{4.1.1}$$

where Γ is the FoM and $\boldsymbol{J}\left(\boldsymbol{v},\boldsymbol{p}\right)$ is the vector containing the different optimization metrics considered. Each point in the solution space is represented by the pair $\langle \boldsymbol{v},\boldsymbol{p} \rangle$, where $\boldsymbol{v}$ is the vector containing the decision variables that determine the system design, and $\boldsymbol{p}$ the vector of fixed parameters that specify the environment/scenario the system is placed on. The equalities in vector $\boldsymbol{e}$ and inequalities in vector $\boldsymbol{i}$ must be satisfied in order for the solution to remain within the feasible domain D.

Note that the FoM Γ is obtained as a function f of the set $\boldsymbol{J}$ containing the selected objectives/metrics, and may take any form. These three elements are thus interrelated. However, they should be readily differentiated.

The multiple metrics in $\boldsymbol{J}$ are selected to convey the performance/quality/utility-related attributes of a given solution to the problem (i.e. $\langle \boldsymbol{v},\boldsymbol{p} \rangle$). For instance, an industrial production flow could be evaluated through several metrics: energy employed, worker hours needed, resulting spare material, etc. Each of these objectives is related to different stakeholders' interests: overall cost, perceived quality, product lifespan, etc.

These metrics depend on a variety of components of the solution: dimensioning, architecture, technology, applications, etc. For instance, the power supply for a network node will depend on the number of ports that are included, the average latency of a NoC is conditioned by the chosen inter-nodes topology, the total weight of a road bike is determined by the materials used, the perceived quality of a knife is based on the applications for which it has been designed... Certainly, there will be crossed dependencies and trade-offs: e.g. worst-case bit rate and latency will improve if a network is dimensioned for heavy usage, while the overall cost will be higher.

Naturally, when facing these design problems, we are not going to evaluate each solution in the explored design space by means of an experimental prototype and measurement. To enable such an extensive exploration, metrics are extracted for each solution through previously obtained models. They might come from analytical models (either theoretical or behavioral), simulations, or experimental data from a subset of the solution space.

However, the complexity of optimizing a problem with several objectives (in our case, those contained in $\boldsymbol{J}$) is self-evident: such a Multi-Objective Optimization (MOO) problem with usually multiple optima, discrete variables, and often a non-linear combinatorially-exploding nature is typically solved either by means of aggregating those objectives into a single objective function or using Pareto methods [189]. Although the latter provides a rich technique for telling apart the effects of the different attributes, its complexity explodes with the number of involved metrics. In MOO problems where the attributes can be said to

contribute to the overall performance/quality/utility of the solution, defining a FoM Γ as a single unitless attribute combining all of them is usually a more efficient approach.

Therefore, although Γ is a numeric value and can be used to compare two different solutions, it is not possible to relate its numerical value to a specific cause. It simply (and powerfully) provides a way to transform a MOO problem (which could turn highly complex and hence computationally challenging) into a single *golden* figure optimization problem.

Consequently, the f function is the chosen form of numerically aggregating the metrics in $\boldsymbol{J}$ into a single figure, Γ. The simplest definition for f would be that of a weighted sum, meaning that each metric in $\boldsymbol{J}$ contributes independently to the overall FoM Γ. This could be expressed as in Eq. 4.1.2:

$$\Gamma = \sum_i w_i \cdot J_i \tag{4.1.2}$$

where:

$$w_i \in (0, 1], \ \forall i \tag{4.1.3a}$$

$$\sum_{\forall i} w_i = 1 \tag{4.1.3b}$$

This method, despite its simplicity, might not be useful for contexts where the overall value of the solution implies that all attributes are contributing to it: the weighted sum could mark a solution where some high-scoring metrics make up for others presenting low values as equivalent to another solution with higher-than-average values on all metrics.

Therefore, defining the f function as a non-linear cross-combination of the different metrics is usually closer to the overall stakeholders' satisfaction with a given solution. Therefore, f typically takes the form of a utility function $U(\cdot)$, which can be bounded in the interval $[0, 1]$ for convenience. Many different definitions have been proposed (the interested reader can dive into ref. [189] and the references therein for a wider review), but the multiplicative approach [190] is the most commonly used (see Eq. 4.1.4):

$$U(\boldsymbol{J}) = \frac{1}{W}\left[\prod_i (1 + W w_i J_i)\right] \tag{4.1.4}$$

where w_i are the weights for each objective $\boldsymbol{J}_i$, and the parameter W is used to ensure that $U(\boldsymbol{J}) \in [0, 1]$. The relation between W and w_i follows:

$$1 + W = \prod_i 1 + W w_i \tag{4.1.5}$$

The flexibility of this framework is what makes DSE a powerful approach for complex design optimization problems. In summary, this methodology facilitates the task by:

- Exploring the entire design space without being limited by the "intuition" and designer's previous experience that might hinder the way to the optimal (but maybe not intuitive) solution.

- Providing not just a single optimal analytical solution, but rather design trends and guidelines extracted from the exploration.
- Being valid also for early design decisions, when there are no experimental data sets, computer simulations, or even analytical models for the performance metrics of the system.

4.2 Applying DSE methodology to QC

DSE provides a resilient design flow that fits the characteristics of any design problem within QC: the largely unexplored design space, with still many design decisions left open, the high cost and current unfeasibility of experimenting with physical implementations, and the novelty of the quantum realm (the *quantum weirdness*), requiring multi-disciplinary collaboration [191].

For that reason, DSE has been already used for different problems in the field of QC: optimal quantum arithmetic reversible circuit synthesizing [192]; optimizing the parameters of the Quantum Approximate Optimization Algorithm (QAOA), a quantum-classical hybrid technique to solve NP-hard problems in NISQ quantum computers [193]; and mapping, either by reducing circuit overhead for specific target hardware [194], or by benchmarking existing mapping approaches and deriving optimal strategies for specific quantum algorithms and quantum processors [195, 196].

Most interestingly, DSE has already been applied also for architecting QC systems. Focusing on NISQ ion trap QCCD architectures, in [30], Murali *et al.* gave out some optimal parameters, operation implementation of gates and communications, as well as topology choices for improving scalability in the near term. This would be a qubit-technology-specific case study covering a very small space in our overarching approach on scalability and communications analysis for going beyond NISQ through modular architectures.

When applying DSE to a given problem, one needs to determine the solution domain (i.e. the variables and parameters we can tweak in order to optimize the design), the performance/quality metrics, and the aggregation function that groups them into the FoM. While the latter is a critical task, it is less dependent on the specific nature of the problem (in our case, QC). Before getting into our specific design exploration (in Section4.3), where we describe the different variables and parameters involved, let us first review the actual state-of-the-art performance metrics and benchmarks in QC. This will help us in choosing the adequate set of metrics $\boldsymbol{J}$ and reasoning on the best fitting FoM Γ for our problem.

4.2.1 Existing metrics for QC

Being QC still a developing *research* field, exploring it by means of DSE has to be careful on which performance metrics are chosen. Measuring the progress of QC is, right now, more about calibrating its *maturity* and overcoming individual obstacles rather than benchmarking computational power [65]. To calibrate how this fact could impact our exploration, let us analyze the metrics proposed in the existing literature in the context of QC (the interested reader may consult [20] for a complete review on this topic).

Current quantum computers are closer to experimental prototypes than to commercial products. Therefore, cost metrics are not a priority: for the moment, the interest lies in achieving a certain goal, almost at any cost (in dollars or Watts). Similarly, some system attributes such as design flexibility or reliability are not taken into account either: these characteristics are for the moment luxuries that remain secondary actors in this initial era of QC. Including them in the design decisions will be for sure part of the transition from a *research* technology to a *production* one.

The early stage QC is at involves also the tools used to measure performance. Whenever a new prototype of a quantum computer is presented, aspects such as its number of qubits, lower gate errors, longer coherence times, or the demonstration of a new technology fill the headlines. Of course, all of these aspects are necessary for QC development and contribute to the overall performance of the quantum computer. However, consensus on a generic qubit-technology-agnostic metric is essential in order to establish a common framework for a fair comparison among existing prototypes and technologies.

Few system-level metrics have been defined for quantum computers. Rather, qubit- and gate-level metrics related to their error performance (such as coherence time, gate fidelities, and latencies) have been widely used, and several methods for measuring them have been developed (e.g. direct fidelity estimation [197] and state gate set tomography [198]).

Algorithms used as benchmarks in experimental QC are, thus, adapted to their current limited size and capabilities, and interdependencies among single-qubit errors make extrapolating benchmark results from small computers to larger ones impractical. Because of that, existing metrics and benchmarks are still far from capturing all the limitations of a quantum computer design [20, 65]. Nevertheless, their validity for driving progress in current research is beyond a reasonable doubt. In fact, quantum tomography [199], miniature versions of key algorithms [200, 201] or randomized benchmarking [202, 203] are extensively accepted for technology demonstrations, and new approaches are continuously proposed and improved [204–207].

This necessity for a global performance metric valid for comparing quantum computer prototypes in the NISQ era, regardless of the qubit technology employed, was first answered by a proposal stemming from IBM's quantum research team [204, 208]: QV. This performance metric takes a step forward from qubit counts, aggregating the most important factors affecting performance, such as the number of physical qubits, the number of gates that can be operated before errors make results useless, qubit connectivity, and the number of operations that can be run in parallel. The QV is computed using the largest random square circuit the quantum computer is able to execute successfully. Authors in [209] highlighted the limitations of QV and generalized the concept by presenting the volumetric benchmarks.

More recently, the same team has also introduced Circuit Layer Operations Per Second (CLOPS), which measures the number of QV layers that can be executed per second (averaged over 100 samples) on a given processor. This helps with knowing about the computational speed aggregating gate, read-out, and control latencies [210].

These metrics are in fact adding up to the quality metrics (gate fidelities, qubit coherence times, qubit count...) providing further information on computational power (QV) and speed (CLOPS). These involve the first efforts to give an architecture-neutral figure for evaluating the useful amount of quantum computing performed by a given system.

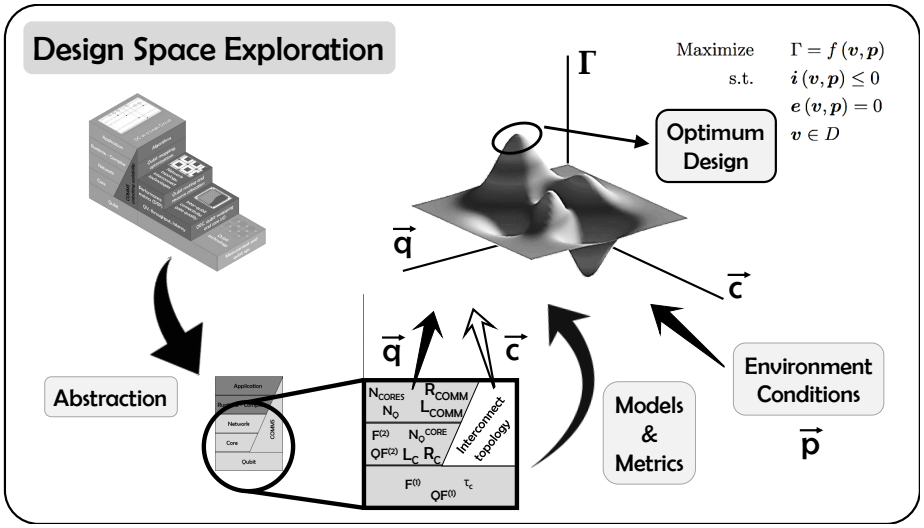


Figure 4.1: A DSE for Multi-core Quantum Computers

4.3 An analytical approach to exploration of Multi-Core Quantum Computers

In the present section, we start the DSE study of this thesis with an exploratory analytical approach. That is, based upon figures and numerical parameters describing the quantum multi-core computer architecture and a model that predicts its performance, we delve into the scalability limits and prospects of the multi-core approach.

Applying DSE to our system-wide optimization problem implies identifying each one of the generic components of the formulation above (Eq. 4.1.1) in the multi-core QC environment (see Fig. 4.1). That is, choosing one or more performance metrics (which encapsulate the computational power of the resulting design) to evaluate the whole multidimensional design space. Nevertheless, we must first determine this solution domain, by specifying the set of variables and parameters that have to be taken into account for our problem (i.e. the pair $\langle \boldsymbol{v}, \boldsymbol{p} \rangle$).

4.3.1 Compressing a quantum computer through models: distilling its essence

When describing a problem in order to explore the design space with a performance optimization purpose, one must carefully select the elements in the design that are crucial from the point of view of the optimization problem and discard the rest of them, as long as they do not influence the model from the chosen perspective.

In our case, we are exploring whether a multi-core architecture can make a difference in terms of processing power scalability. Therefore, we must particularly take into account the elements of the quantum computer that may be affected by the architecture paradigm (single-core *versus* multi-core).

For this reason, based upon the layered stack described in Chapter 2 (see Section 2.2), we will analyze the three lower layers that come into play when considering multi-core archi-

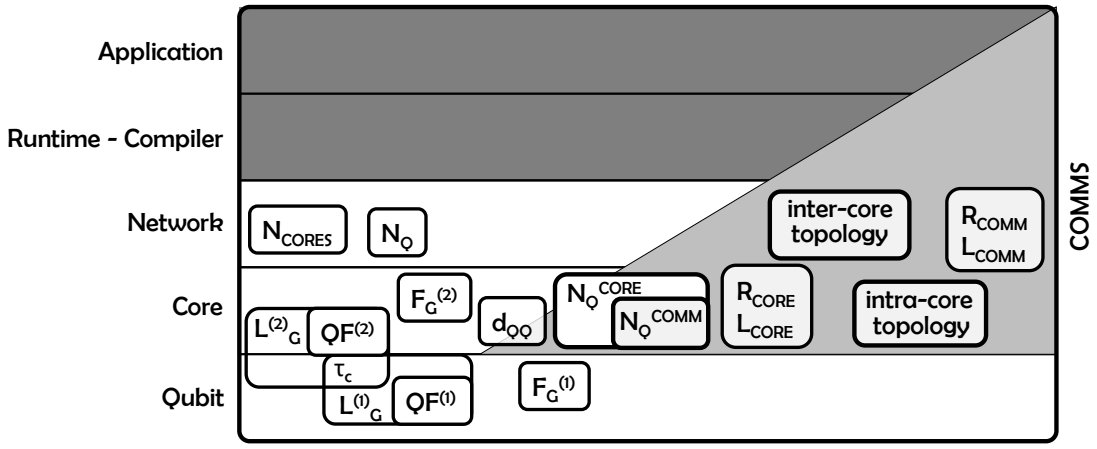


Figure 4.2: Parameters in the multi-core QC DSE, placed within their respective layers in the stack

tectures and thus are most affected by intra-core and inter-core quantum communications, namely: qubit, core, and network layers (see Fig. 4.2).

Let us now go through each layer separately and identify the main parameters to take into account for the subsequent DSE, with special attention to their effect on communications.

4.3.1.1 Qubit layer

When looking at an individual qubit for the main features that may affect the performance of the quantum computer as a whole, we should recall that it plays a double role: it acts both as a memory position (storing a quantum state) and as a computing unit (one can perform operations/gates on it). Therefore, we could make use of three main parameters. With respect to memory capabilities, we can use coherence time. Regarding computing capabilities, we have the gate fidelity and the quality factor. Both are also applicable for two-qubit operations (which we will review next).

- **Coherence time.** The coherence time τ_c (you can go back to Section 1.1.2 for more background on this concept) sets a fundamental limit on the maximum time we can operate and read out the state of the qubit. Thus, τ_c is proportional to the *qubit's memory capabilities*. Short coherence times mean short-lived variables (which in turn implies that complex algorithms are not supported), considering that, as stated by the *no-cloning theorem*, there is no way to replicate quantum information [211], [212]: only QEC is able to somewhat extend variables lifetime, generally at a notably high cost in space (qubits used) and time. Being the main *quality* metric for a qubit, the decoherence times have been continuously improved in the different selected qubit technologies.

- **Quality factor - single qubit gates.** Computing capabilities on qubits are governed not only by the qubit characteristics themselves but also by the control operations

quality. The quality factor QF is a unitless parameter derived from the coherence time τ_c and the gate latency L_G (i.e. the time spent in performing a certain quantum operation, such as a Hadamard gate or a CNOT). It is an estimate of the number of gates (quantum operations) that can be applied to a qubit while it contains coherent information (see for instance [3]) and hence is related to the qubit's computing capabilities. The quality factor is computed as:

$$QF = \frac{\tau_c}{L_G} \tag{4.3.1}$$

It is clear to see the relationship between coherence time and the memory capabilities of that qubit technology. For instance, ion traps and quantum dots are technologies well suited to building up quantum memories. On the other hand, we have chosen the quality factor instead of the gate latency L_G because it conveys more intuitively the computing capabilities of the qubit: a high-quality factor allows operating many times on the qubit. Note that it is not necessarily related to large coherence times (τ_c); a good *memory* qubit may constitute a lower quality *processing* qubit, see e.g. quantum dots. This relationship τ_c – QF is quite interesting and may affect quantum processors' design and claim for hybrid technologies approaches.

- **Single-qubit gate fidelity.** The single-qubit gate fidelity (F_G) is a simplification of the complex quantum error models, and represents how likely a quantum operation will not introduce errors in the system (i.e. inverse to the gate error rate). It is usually displayed as a percentage value, where 100% means that the qubit state has not experienced any deviation from the ideal value it is representing. Low fidelity values will render a qubit useless, no matter how long the coherence time might be, as each operation on the qubit will corrupt its information and affect the entire computation.

4.3.1.2 Core layer

We have just introduced τ_c, QF, and F_G. Although the coherence time is directly related to a single qubit, the quality factor and the gate fidelity are usually computed separately for one-qubit gates ($QF^{(1)}, F_G^{(1)}$) and two-qubit gates ($QF^{(2)}, F_G^{(2)}$). Therefore, as they involve operations among more than one qubit, they are the first parameters of the core layer.

- **Two-qubit gate fidelity.** $F_G^{(2)}$ is the application of the concept of fidelity to the case of the two-qubit gates.
- **Quality factor - two-qubit gates.** We can use τ_c and the two-qubit gate latency to compute $QF^{(2)}$ to obtain the quality factor extended to the case of the two-qubit gates.
- **Number of qubits per core.** We will use N_Q^{CORE} for the *total number of qubits* forming each core in the multi-core system.
- **Number of communication qubits per core.** Out of the N_Q^{CORE} qubits, N_Q^{COMM} of them will be responsible for interconnecting the core with one or several (identical)

modules, in the multi-core case (i.e. they would act as input/output ports of the core). In monolithic single-core architecture, no other module exists, and hence $N_Q^{COMM} = 0$.

- **Intra-core topology.** The qubit interconnection graph might be all-to-all, a ring, or a regular 2D lattice... hence constraining computation among qubits in the same core.
- **Quantum intra-core communication latency.** Average qubit transfer time within a core (L_{CORE}).
- **Qubit intra-core transfer rate.** Together with L_{CORE}, R_{CORE} characterizes the performance of the intra-core connectivity network.

4.3.1.3 Network layer

At this layer, we can see the whole quantum communications network perspective. Parameters interesting to our analysis include:

- **Number of cores.** Monolithic quantum chips count in a given multi-core processor (N_{CORES})
- **Total number of qubits.** Sum of the number of qubits integrated into the cores forming the processor (N_Q).
- **Inter-core topology.** Disposition of the nodes and their connectivity within the multi-core network.
- **Quantum inter-core communication latency.** Average qubit transfer time between cores (L_{COMM}).
- **Qubit inter-core transfer rate.** R_{COMM} can be defined as the average qubit rate in the multi-core network.

With this, we have a first description of the system parameters and open variables in our system design, layer by layer. That is, they integrate the vectors $\boldsymbol{p}$ and $\boldsymbol{v}$ in the DSE exploration. Depending on the aspects to be analyzed, a given element will be considered in the analysis as a fixed parameter (part of the scenario under study, i.e. part of $\boldsymbol{p}$) or as an open variable (being part of $\boldsymbol{v}$ the DSE will explore different values for that element looking for the one that may contribute to the optimal solution). As an example, let us take N_{CORES}. If $N_{CORES} \in \boldsymbol{p}$, we have fixed its value for our exploration, otherwise, $N_{CORES} \in \boldsymbol{v}$, and DSE will sweep over different values of N_{CORES}. Accounting for our computing-communications double-stack approach, $\boldsymbol{v}$ is divided into two different axes: $\boldsymbol{q}$ (variables coming from the pure computing stack, such as gate latencies or fidelities) and $\boldsymbol{c}$ (those that are part of the communications).

4.3.2 A behavioral model for a DSE scalability analysis of multi-core quantum architectures

Now that we have described the parameters that model the proposed approach of a double full-stack communications-enabled quantum computer, it is time to delve into the significant metrics $\boldsymbol{J}$ and the development of the aggregated FoM Γ for our scalability analysis.

Table 4.1: Notation and symbol definitions for Sections 4.3 and 4.4

Notation	Meaning
τ_c	Coherence time
$L_G^{(1)}, L_G^{(2)}$	One and two-qubit gate latencies $[s]$
$F_G^{(1)}, F_G^{(2)}$	One and two-qubit gate fidelities
$QF^{(1)}, QF^{(2)}$	One and two-qubit quality factors
N_Q	Total number of qubits in a quantum computer
N_Q^{CORE}	Total number of qubits in a quantum core
N_Q^{COMM}	Number of qubits dedicated to inter-core communications in a core
N_{CORES}	Number of cores in a multi-core quantum computer
L_{CORE}	Average quantum communication latency within a core
R_{CORE}	Average qubit rate within a core
L_{COMM}	Average quantum communication latency in inter-core communications
R_{COMM}	Average qubit rate in inter-core communications
J_{QF}	Qubit quality metric
J_F	Aggregated fidelity metric
J_{Qb}	Computational power metric
J_I	Qubit integration penalty metric
ϵ_I	Qubit-to-qubit disturbance error
N_Q^{MAX}	Qubit technology limit to the number of qubits integrated into the cores forming a multi-core quantum computer
N_Q^{LIM}	Qubit technology limit to the number of qubits integrated into the same chip
d	Exponential factor on the overhead qubit ratio
J_C	Inter-core communications performance metric
ϵ_C	Error factor due to inter-core communications overhead
N_{USED}	Number of cores that contain active qubits (i.e. are being used in the current configuration)
$\tilde{J}_i$	Normalized value for metric J_i

For the sake of the completeness of the FoM, it should aggregate metrics on performance, cost, and qualitative attributes [64]. Applying existing proposals (see Section 4.2.1) is hence not straightforward nor comprehensive, as the elements considered in these metrics do not

include communication latencies or qubit rates at different layers. They do not differentiate either the communication time or the computational time.

Having this in mind, we could conclude that an optimal performance metric for multi-core quantum computers should be:

- **Communications-oriented**. Adding communication overheads and other considerations related to the multi-core nature of the proposed double-stack architecture claims for accurate modeling of communication processes inside the quantum chip and possibly designing specific benchmarks.
- **Adaptive**. The non-universality and expiration time of current benchmarks (including QV and CLOPS, which are also intended for short-term NISQ devices) implies Γ should evolve along with quantum computation to avoid misleading designs. Its definition should take into account qualitative attributes.
- **A multidisciplinary effort**. A full-fledged analysis requires very refined models (for elements as diverse as qubit crosstalk, fidelity degradation, quantum communication technologies, etc.) and a complete definition of Γ, something that cannot be obtained without the collaboration of all the fields involved (physicists, material engineers, electrical and computer engineers...).

Therefore, in this first exploratory analysis, we have tailored intuitive yet useful performance metrics and models. Not being possible to obtain experimental data from large-scale multi-core computers (as it is only a theoretical proposal yet) nor even from large single-core chips, we should resort to other modeling approaches. Although in subsequent sections and chapters, we will get into benchmarking different aspects of multi-core architectures by means of simulating their performance, we aim in the present study to widely explore the parameters space. For that, the most fitting approach is to use a behavioral model, based on existing results and literature. Moreover, this approach suffices for showing all the possibilities that DSE has to offer.

For each layer, we have focused on the main elements affecting the overall quantum computing performance. In the qubit layer, we have included qubit quality metrics based on the coherence time together with gate fidelities and latencies. Regarding the core layer, we have incorporated the effects of qubit crosstalk and control issues, as well as the gain in computational power when putting together many qubits. Finally, we have enclosed the overhead of inter-core communication in the network layer. In the following, we describe each metric with the proposed analytical formulation.

4.3.2.1 Qubit quality

As summarized in Section 4.3.1, the qubit implementation, depending on its technology and the fabrication quality, affects its own computational and memory capabilities. These are related to the gate latencies and coherence time, however, there is an inherent coherence-controllability trade-off among these two: qubit isolation allows for longer coherence times, while usually making qubit interactions more difficult, hence affecting the gate performance. Therefore, as suggested in [3, 16], we have chosen single-qubit quality factor QF (i.e. an estimate on the number of quantum operations that can be performed, which is also related

to the maximum size of the quantum algorithms we could run on an ideal platform formed by these qubits) as the *qubit quality metric* J_{QF}:

$$J_{QF} = QF^{(1)} = \frac{\tau_c}{L_G^{(1)}} \tag{4.3.2}$$

4.3.2.2 Aggregated fidelity in a quantum core

The qubit performance does not depend only on coherence time and gate latencies: however big the ratio among them (QF) is, if the gates themselves are quickly destroying the quantum state, the computing performance will be highly degraded. That is, we need to take into account gate fidelities. In accordance with the literature (see e.g. [3, 206]), two-qubit gate fidelities model better large systems performance than their single-qubit counterpart.

In order to improve this performance indicator in large ensembles of qubits sharing the same chip, we have added an exponential degradation factor related to the qubit count, due to the crosstalk and controllability issues [22]. This is coherent with the fact that the overall fidelity of a sequence of gates results from the product of the fidelities of each one of them. Therefore, we have defined the *aggregated fidelity metric* as:

$$J_F = \left[F^{(2)}\right]^{N_Q} \tag{4.3.3}$$

Observe that, being $F^{(2)} < 1$, the greater the number of qubits (exponent) the lower the aggregated fidelity.

4.3.2.3 Computational power

We could have a system with great *qubit quality* and *aggregated fidelity*, but with a low number of qubits: this would not be anything else than a prototype, not really useful for solving adequate-size circuits. The potential computing power of a quantum chip increases exponentially with the qubit count [16], so we have defined the *computational power metric* as:

$$J_{Qb} = 2^{N_Q} \tag{4.3.4}$$

4.3.2.4 Performance degradation due to qubit single-core integration limit

Qubit crosstalk and controllability issues do not affect performance degradation linearly, especially because of the control scalability issues [52]. As already explained in the previous chapter (see Section 3.2) for each existing technology there exists a predicted qubit integration limit for a single chip.

To model this, we have defined a *qubit integration penalty metric* which works as a step function on this threshold to the number of qubits per core (which we have labeled N_Q^{LIM}): in case we have $N_Q^{CORE} > N_Q^{LIM}$, this metric measures the performance degradation related to this integration qubit overhead. For this, we have derived qubit density from the number of qubits per core $N_Q^{CORE} = N_Q/N_{CORES}$, and a technology-related multiplicative factor which we have termed as qubit-to-qubit disturbance ϵ_I. This latter component comes from

e.g. the physical qubit-to-qubit distance, control frequency overlapping, etc., and is different for each qubit implementation.

To model the steep deterioration, we have used a multiplicative factor related to the overhead qubit ratio and an exponential factor d:

$$J_I = \underbrace{\frac{\epsilon_I N_Q}{N_{CORES}}}_{\text{qubit density}} \cdot \underbrace{\left(\frac{N_Q}{N_Q^{MAX}}\right)^d}_{\text{overhead qubit ratio}} \cdot \underbrace{\mathcal{H}\left(N_Q - N_Q^{MAX}\right)}_{\text{non-linear penalty on max qubit count}} \tag{4.3.5}$$

where:

N_Q^{MAX} is the aggregated maximum of qubits that may be integrated into N_{CORES}:

$$N_Q^{MAX} = N_Q^{LIM} \cdot N_{CORES} \tag{4.3.6}$$

$\mathcal{H}(x)$ is the Heaviside step function:

$$\mathcal{H}(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases} \tag{4.3.7}$$

Observe that, in opposition to the previous metrics, J_I is proportional to the performance degradation (i.e. inverse to the computing performance). It is useful to note also that:
For $N_Q < N_Q^{MAX}$ the degradation is not accounted for:

$$J_I\Big|_{N_Q < N_Q^{MAX}} = 0 \tag{4.3.8}$$

For $N_Q = N_Q^{MAX}$:

$$J_I\Big|_{N_Q = N_Q^{MAX}} = \epsilon_I N_Q^{LIM} \tag{4.3.9}$$

and from there, the penalty follows an exponential function:

$$J_I\Big|_{N_Q > N_Q^{MAX}} = \frac{\epsilon_I N_Q}{N_{CORES}} \cdot \left(\frac{N_Q}{N_Q^{MAX}}\right)^d \tag{4.3.10}$$

4.3.2.5 Inter-core communications performance

Once we have covered the performance limitations coming from both qubit and core layers, we have to take into account the overhead incurred in multi-core quantum architectures as we scale in the number of cores integrated into the platform [56]. When using more than one core, the communication processes (be it quantum teleportation, ion shuttling, etc.) that are needed to operate among qubits placed in different cores might be costly and set a limit for the multi-core approach.

In order to avoid technology-specific metrics, we have preferred to concentrate the different characteristics of existing and future quantum core interconnects into a single parameter ϵ_C which corresponds to the error escalation due to communications overhead when adding cores to the system. For the sake of simplicity, this *inter-core communications performance metric* has been defined as the exponentiation of the overhead factor with the number of cores (which is a rough approximation of the effect of all-to-all traffic among the cores):

$$J_C = (1 - \epsilon_C)^{N_{USED}} \tag{4.3.11}$$

where N_{USED} is the number of cores that contain active qubits (i.e. qubits that are being used in the given configuration), and $(1 - \epsilon_C)$ correspond to the complementary of the communications error. Observe that being $(1 - \epsilon_C) < 1$, the greater the number of cores, the lower the performance.

4.3.3 Aggregating the metrics into a single FoM

In order to adequately design the function f that aggregates the metrics in $\boldsymbol{J}$ into the FoM Γ, we need to establish: *i)* how each metric affects the overall performance of the system, *ii)* how these metrics are interrelated in order to choose the most fitting function to group them, and *iii)* whether we need to normalize or weigh each metric.

First, when we view our scalability problem as a whole, we realize that a high-performing multi-core system should have:

- enough qubits to compute and the required applications (high J_{Qb}),
- a good coherence-latency ratio for keeping the data coherent while the circuit is being executed (high J_{QF}),
- sufficiently high gate fidelity to perform all operations (keeping low the effect on the number of qubits in the aggregated fidelity J_F),
- a good trade-off on fitting a high number of qubits in each core while not suffering from high errors due to qubit crosstalk or controllability issues (low J_I), and
- a low inter-core communications overhead (controlling the impact on J_C).

Choosing a particular f function depends mainly on the interaction between the metrics in their contribution to Γ. In our case, all five metrics should have high values in the optimal solution: low qubit quality is as unacceptable as a low number of qubits, high crosstalk, and controllability issues, or high inter-core communications overhead.

This means that a weighted sum, such as that shown in Eq. 4.1.2, which assumes each metric contributes independently to Γ, does not fit our problem.

The multiplicative approach that can be found in Eq. 4.1.4 already includes the interdependence on the different metrics. However, observe that developing the equation particularizing to $J_i = 0$, for some i does not lead to $U(\boldsymbol{J}) = 0$. This behavior is not useful either for our case: when we have, e.g. $F^{(2)} = 0 \implies J_F = 0$ we should have $U(\boldsymbol{J}) = 0$. In fact, that should apply to every metric.

For this reason, we have chosen an even tighter form for the f equation: the product of the whole sequence of metrics. For a better understanding of how each metric affects the overall Γ value, we have placed the performance-centric metrics in the numerator, and the degradation-centric ones in the denominator (as in Eq. 4.3.12).

$$U(\boldsymbol{J}) = \frac{\prod_i J_i}{\prod_k J_k} \tag{4.3.12}$$

Normalizing the different components of the FoM helps to equalize and bound the effect of the metrics on the overall performance, and is usually done to the $[0, 1]$ interval. However, determining which normalization to apply on each metric or whether to apply it might be different for each one.

The qubit quality metric J_{QF} represents a technology-specific advancement and should be in principle unbounded (the coherence time could always be larger and the gate latency will be asymptotically tending to 0). Therefore, being a fundamental parameter for any quantum computer's quality benchmark, we have decided to keep it as it is, hence not normalizing it:

$$\tilde{J}_{QF} = J_{QF} \tag{4.3.13}$$

On the other hand, the computational power grows also with technology development, but at an exponential rate: without a normalization function, this metric would greatly distort the FoM Γ. Moreover, it is lower-bounded to 1 ($N_Q = 0$), so we have also shifted this bound to 0. Therefore, we have:

$$\tilde{J}_{Qb} = 2^{\tilde{N_Q}} - 1 \tag{4.3.14}$$

where $\tilde{N}_Q$ is N_Q normalized between $[0, 1]$ to the maximum number of qubits in the undergoing DSE exploration, hence having also $\tilde{J}_{Qb} \in [0, 1]$.

Regarding aggregated fidelity, it is naturally bounded within $[0, 1]$. However, as this metric is focused on the effect that putting many qubits together has on the aggregated fidelity, we consider that it depicts a degradation rather than an improvement and we will place it in the denominator of the f function for computing the FoM. Therefore, we have preferred to get the inverse metric (the lower, the better for the measured quantum computer), that is, mapping it into $[1, 2]$ (to avoid zeroes in the denominator):

$$\tilde{J}_F = 2 - J_F \tag{4.3.15}$$

Being also a degradation-centric metric (and hence appearing in Γ denominator), we have also J_I shifted its lower bound to 1. However, we have preferred not to limit the penalty (i.e. the upper bound of it) as it only appears after the N_Q surpasses the predicted qubit limit for a specific technology and hence we want to force that any solution within this part of the solution space is avoided:

$$\tilde{J}_I = 1 + J_I \tag{4.3.16}$$

Only the inter-core communications overhead is left. By definition, it is already bounded within $[0, 1]$. However, as it represents an added overhead, we have included it in the degradation-centric metrics, hence shifting and reversing it to $[1, 2]$ in order to place it in the denominator of Γ.

$$\tilde{J}_C = 2 - J_C \tag{4.3.17}$$

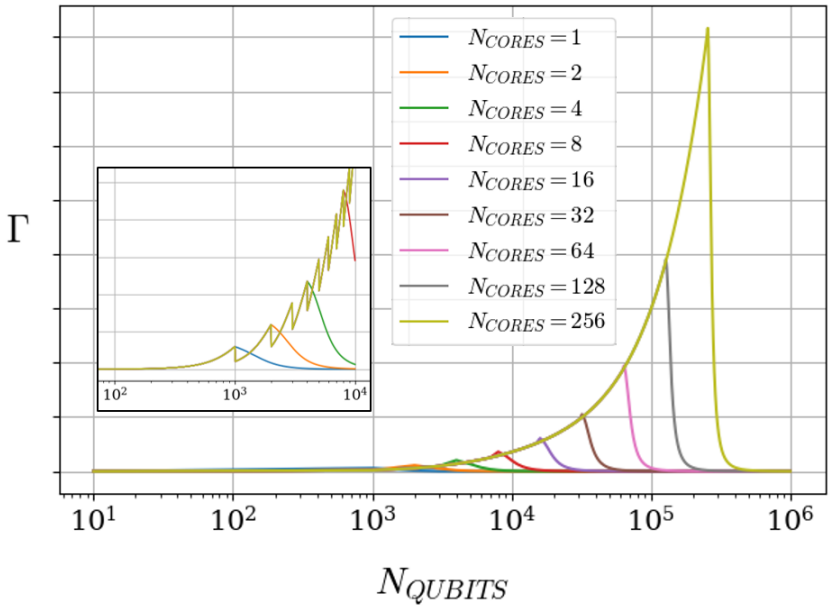


Figure 4.3: Scalability analysis (I) Quantum computer's overall performance is plotted against the number of qubits used in the system, for several configurations in terms of number of cores used and the following parameters' values: $F^{(2)} = 99.9\%$, $\epsilon_C = 5\%$, $\epsilon_I = 0.1\%$, $d = 3$, and $N_Q^{LIM} = 1000$.

Accordingly, the final form of the FoM Γ is that of Eq. (4.3.18):

$$\Gamma = \frac{\tilde{J}_{Qb} \cdot \tilde{J}_{QF}}{\tilde{J}_F \cdot \tilde{J}_I \cdot \tilde{J}_C} \tag{4.3.18}$$

Observe that Γ is not bounded within $[0, 1]$, as it is not a requirement for the DSE [213].

4.4 Results

In order to visualize and clarify the possibilities of DSE, we now present the results of a first analysis on multi-core quantum computers looking for answers to some of the most interesting questions in QC: How will the quantum computer scale in the number of qubits? Will the multi-core approach unlock the current monolithic single-core quantum computers' scalability bottlenecks? How do the existing qubit technologies compare as candidates for multi-core quantum computing? Does the inter-core communications technology affect the performance of multi-core quantum computers? Let us look into the procedure used to perform this first analysis before commenting on the results.

4.4.1 Scalability analysis

The main concern of the present chapter, and thus the main result expected from the DSE analysis, is to determine whether the multi-core approach will effectively supersede current

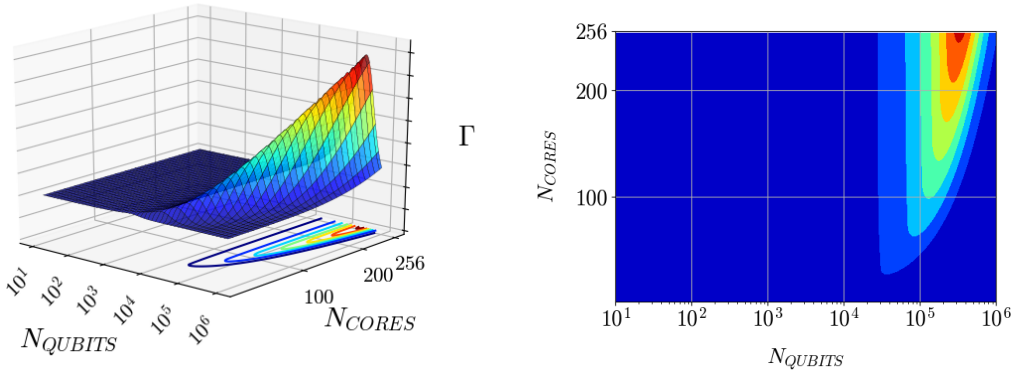


Figure 4.4: Scalability analysis (II) Performance analysis when varying both the number of qubits and the number of cores in the quantum computer. The isolines in the plot let us know different configurations that provide the same performance.

NISQ computers and enable their scalability into large quantum computers. Then, if we take into account a specific scenario or a given set of requirements (i.e. having fixed $\boldsymbol{p}$, e.g. core-to-core communication latency, gate fidelity upper-bound, existing (or predicted) coherence time τ_c range, etc.), we will be able to study the evolution of the design performance (Γ) when sweeping over the number of cores N_{CORES} from 1 (single-core *traditional* quantum processor) to tens or hundreds of them. This is called *scalability analysis*. This type of study allows the detection of scalability trends and bottlenecks, hence obtaining design guidelines on quantum computer configurations for optimal performance.

Using the assumptions and models presented in the previous section, we show such exploration in Figs. 4.3 and 4.4, where we plot the Γ values for a wide range of quantum computer configurations. We have selected N_Q and N_{CORES} as our vector $\boldsymbol{v}$, fixing the rest of the elements. That is, the exploration is restricting the scalability to the increase in the number of qubits and cores, and the relationship among them. The fixed parameters ($\boldsymbol{p}$) have been set to realistic values for ion trap technology (the most evolved yet) taken from [3] and [2], that is, two-qubit gate fidelity $F^{(2)} = 99.9\%$, gate latency $L_G^1 = 5.4 \cdot 10^{-7} s$, coherence time $\tau_c = 2 \cdot 10^{-1} s$, $\epsilon_I = 0.0001$, $d = 3$, $\epsilon_C = 0.05$ and maximum number of qubits per core $N_Q^{LIM} = 1000$.

In Fig. 4.3, a single-core quantum computer is compared to several multi-core configurations, for a total number of qubits N_Q in the system varying from 10 to 10^6 qubits. For each configuration, the performance (Γ) follows a peaky bell-shape trend, with a maximum close to $N_Q = N_Q^{MAX}$ (the optimal configuration for that number of cores). Trying to integrate more than N_Q^{LIM} qubits in a single core causes a steep degradation of performance. The single-core processor is clearly exceeded by multi-core approaches. In the zoom-in, a saw-like profile in the performance curve can be clearly seen. Whenever the optimal qubit distribution requires another core to be used (if available in the configuration), the extra communications overhead causes a steep fall in performance. This implies that the configuration with more cores is not always the best-performing one. The plots in Fig. 4.4

contain a complete input variables sweeping, with N_Q varying from 10 to 10^6 qubits, and N_{CORES} from 1 to 256. The isolines in the plot let us know different configurations that provide the same performance, e.g. 10^5 qubits in a 200-core quantum computer perform the same as twice as many qubits using less than half of the cores. Observe that, when incrementing N_{CORES}, the overall goodness of the architecture does not increase linearly, due to the inter-core communications overhead. Note also that the more cores are present in the system, the narrower is the performance curve, that is, a low number of cores guarantees good relative performance for both low and high numbers of qubits. That may collapse for a sufficient number of qubits, where no matter the number of cores, the performance may be the same or worse (due to communications overheads).

Using this simple model, we can clearly draw three main conclusions:

- The multi-core approach is promising as a scalability enabler,
- for every multi-core quantum computer configuration there exists an optimal working range (Γ over a certain minimum threshold), and
- the N_Q^{LIM} parameter clearly constrains the performance of the configuration and thus we should consider it as a fundamental design variable.

With more accurate data and models, we postulate that this type of analysis will effectively accelerate and optimize the research on QC.

4.4.2 A Qubit Technology Gap Analysis

DSE can also be applied to analyses that go further than stating the benefits of the application of multi-core architectures to quantum computers: it may help quantum engineers optimize the efforts and investments in QC. For instance, with accurate models, we are able to explore the whole design space in order to focus research and experiments only on the most promising materials, technologies, or parameter ranges. As an example, we have performed a simple quantitative technology gap analysis, i.e. a performance comparison of the selected qubit technologies and their evolution in the next years that opens a window to the future, letting us know which technologies may provide higher profitability after a certain research investment. To do so, a common performance metric, such as the previously defined golden metric Γ, is needed, in order to establish a common ground for a fair comparison.

In Fig. 4.5 the comparison among several current qubit technologies (and their projected performance) is shown, both for varying N_Q and N_{CORES}. Each technology is represented using the three most representative physical parameters: τ_c, $L_G^{(1)}$ and $F^{(2)}$. Using actual measurements retrieved from [3] and [181] as the parameters baseline, δ represents a correlative technology improvement on the three of them. In the case of quality factor $QF^{(1)}$, δ represents a constant proportional improvement (i.e. $QF_\delta^{(1)} = QF^{(1)} * (1+\delta)$). The fidelity is set to improve asymptotically to 100 %. Under the used assumptions and models, we could already draw some conclusions on the comparison of existing technologies, e.g. donor spins in Si seem to be the most promising technology, with still much room for growth.

It is important to highlight the non-linear behavior of performance improvement with δ. See Figs. 4.6 and 4.7. In ion traps, for instance, in the interval $\delta = (0.5, 0.6)$ the performance

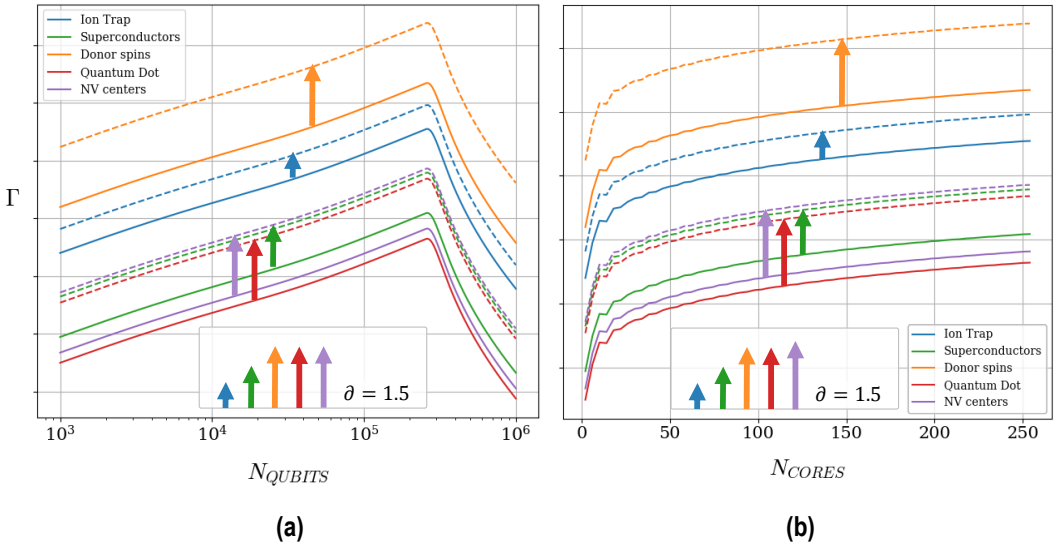


Figure 4.5: Quantitative qubit technology gap analysis a) Quantum computer's overall performance Γ for selected qubit technologies is log-plotted against the number of qubits used in the system. For each technology, current state-of-the-art parameters [2,3] are used to draw the solid line, and a $\delta = 1.5$ evolution/improvement on them is reflected in the dashed line. The number of cores is set to 256, and the rest of the parameters are fixed as in previous figures. The validity of the conclusions drawn from this technology comparison depends exclusively on the accuracy of the models used. b) In this case, Γ is plotted against the number of cores in the system. The performance value assigned to a given number of cores corresponds to the peak performance of that quantum computer configuration (i.e. the number of qubits is set to be optimal for every value of N_{CORES}).

scales exponentially. It corresponds to fidelity values going from 99.999% to 99.9999%, that is, investments to make improvements in this interval will pay off abundantly. Moreover, we can conclude that, even with a 100% improvement, only ion traps can place performance on a range similar to that of solid-state-based quantum computers. Observe also that a δ-evolution on qubit technologies does not only improve peak performance but also the optimal operational margin. For instance, 30.000 "future" ion trap qubits ($\delta = 1$) integrated in a 256-core quantum processor provide the same performance as 200.000 ($5x$ increase) current ion trap qubits. A similar behavior is present in all technologies. When studying the effect of varying the number of cores in the system, we observe as before a clear performance improvement in δ. However, it is to be remarked the "conversion factor" that we find of a δ-evolution and the number of cores needed to achieve the same performance. Observe that, for instance, for all technologies, a $\delta = 0.6$ suffices for having a 70-core processor matching a processor more than twice as large in number of qubits and cores.

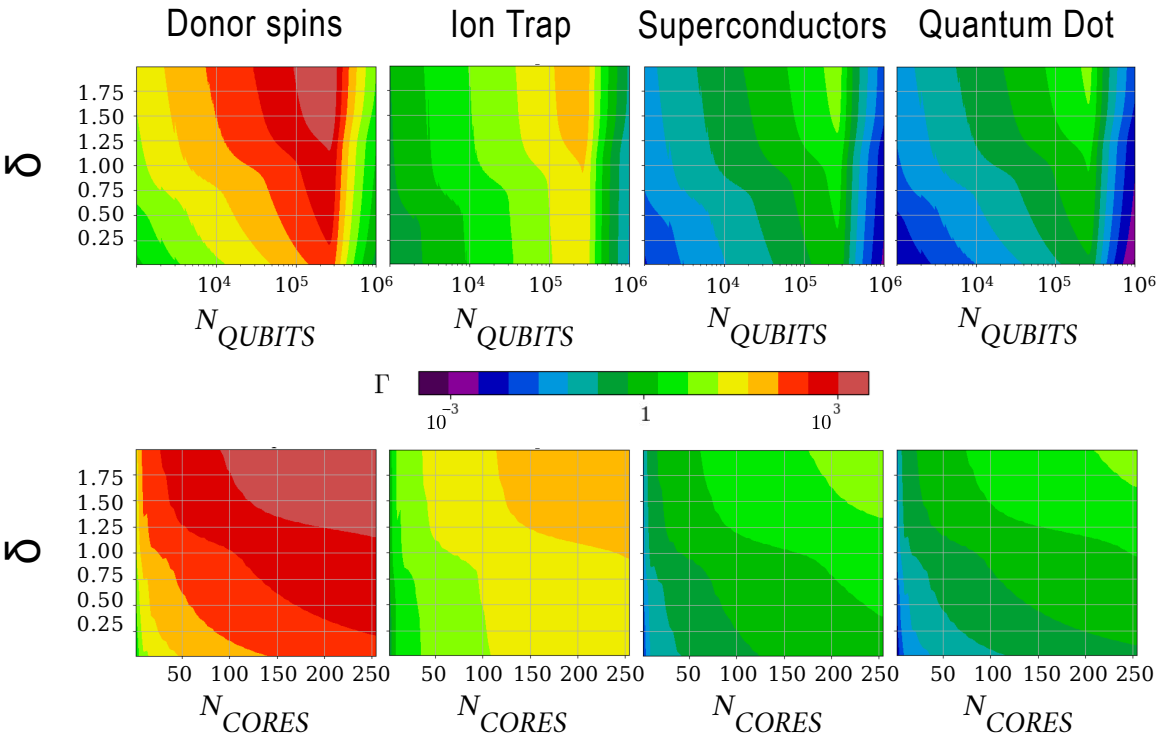


Figure 4.6: Qubit technology gap analysis (log plot) extended for a wide range of δ, for selected qubit technologies, sweeping the number of qubits in the system. Observe that performance does not increase linearly with δ, nor does it behave the same for different technologies. Equivalent performance can be obtained with a notably lower number of qubits if the technology is improved.

4.4.3 Discussion

The results presented allow us to foresee not only the promising effects of applying multi-core architecture to quantum computers but also the possibilities of using such a system-wide optimization proposal (DSE) which might facilitate once-for-all design guidelines unifying the still separated design technologies into a consolidated solution with optimal technologies and parameters for every situation. This type of analysis can be easily reproduced together with more accurate data, FoM and models coming from materials engineers and physicists to draw conclusions such as "among the selected qubit technologies, basing upon model predictions, the technology X is the most promising for building multi-core quantum computers in high-latency environments", or "a lattice topology provides the lowest latency for quantum teleportation-based intra-core networks", and especially "the multi-core quantum computer approach performs better than the monolithic single-core designs when more than N qubits are needed, and/or intra-core latencies are lower than L_{max}, and/or coherence times τ_c are in the range (t_L, t_H)", that will effectively accelerate and optimize the research on QC.

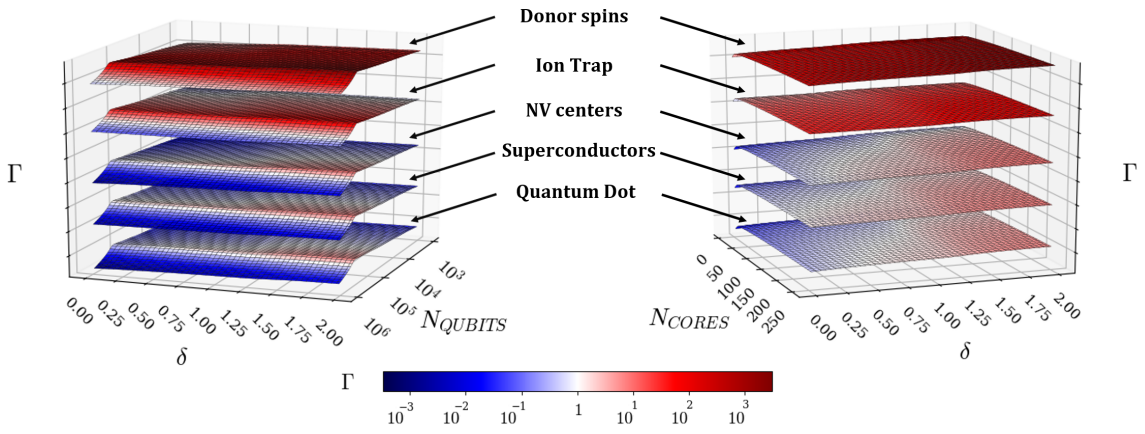


Figure 4.7: 3D version of the qubit technology gap analysis log plot, for selected qubit technologies (they are offset in Z axis for the sake of clarity while keeping the color mapping for Γ values), sweeping the number of qubits and the number of cores in the system. Observe that donor spins performance evolution with δ is really promising when compared to the other technologies, which in comparison remain almost plain. When exploring for an increasing number of qubits, the number of cores is set to 256, while for varying numbers of cores, the number of qubits is determined as the one providing peak performance. The rest of the parameters remain the same as in Fig. 4.3.

Chapter 5

Communications in Multi-core Quantum Computing: the Good, the Bad and the Ugly

Multi-core quantum architectures may recall the revolution of classical multi-core computing. However, while multi-core classical processors enabled the potential of parallelism and solved existing energy and thermal issues, multi-core quantum computing comes as a solution to correlated errors and control issues, which limit its potential even for small computations. Therefore, interconnecting quantum nodes is not a matter of upgrading quantum computers but unlocking their prospects of success.

However, transferring a quantum state among quantum chips is a complex task: it cannot be done using classical communications, and, due to the *no-cloning theorem* (i.e. an arbitrary unknown quantum state cannot be copied), qubit retransmissions are impossible. Moreover, as we have already seen in Section 2.1 when quantum communications are inserted in a quantum computing environment, they have to deal with qubits that are constantly moving around, stringent latencies requirements, and irreparable communication losses. However, the *quantum weirdness* does not only contribute to the complexity and inefficiency of multi-core quantum architectures but also provides some very interesting properties that should be leveraged in order to design and implement powerful quantum multi-core computers.

Therefore, in order to achieve the aim of this thesis, i.e. set the foundations for scalable multi-core quantum architectures, a special focus must be set on quantum inter-core communications. In other words, in order to appreciate the implications of implementing a multi-core quantum architecture, it is key to understand the particularities of quantum communications and how deeply they are intertwined with quantum computing.

In the present chapter, we will leave the global perspective from the last chapter to set the spotlight on the details of chip-level quantum communications. For that, we highlight its unique characteristics and the differences when compared to other similar communication environments. We also dive into the already presented quantum teleportation, and its constant companion, entanglement distribution. Finally, we present a model and several

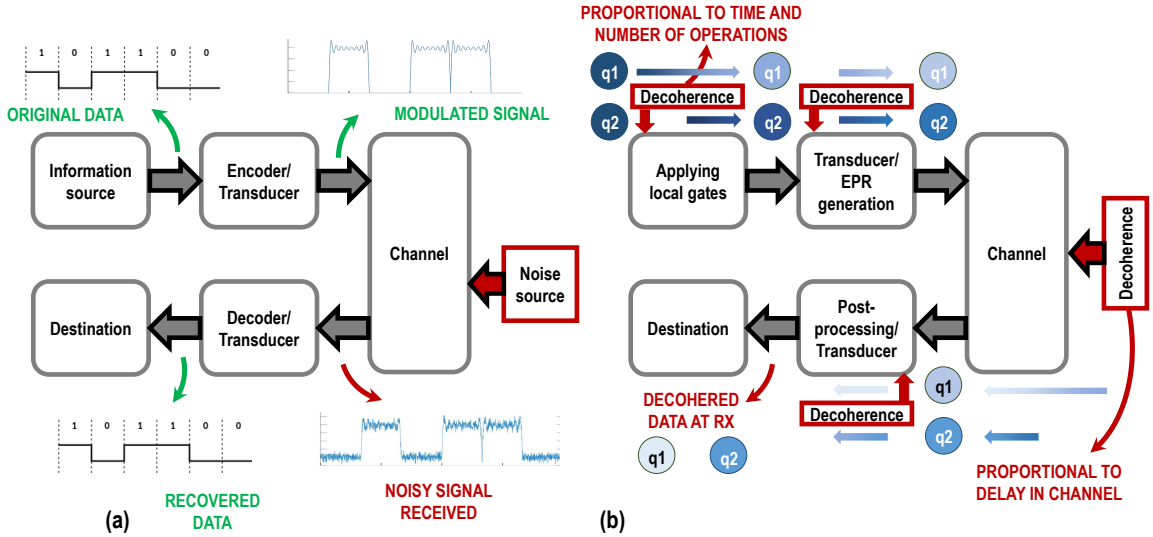


Figure 5.1: Classical noise versus quantum noise in communications. a) Classical noise can be battled with at any stage of the communication, reconstructing the signal: it is not accumulative, b) quantum noise, however, is harder to tackle, and accumulates everywhere (when computing, when communicating, etc.). Qubit's "age" is hence important.

simulation-based studies to stress this quantum communication technique in search of performance boundaries and to explore the design space for quantum multi-core architectures.

5.1 Provably secure (but impatient and merciless) communication at a distance

Quantum mechanics brings some interesting properties not only to computing with quantum information but also to quantum communication. Not even the well-known Shannon's model of communication [214], which is the foundation of all the modern Information Theory, is able to describe fully the transfer of quantum information mainly due to a quantum-specific noise source (*quantum decoherence*) and the measurement problem. The interested reader may dive into this topic in [215, 216].

Indeed, whereas in classical communication the noise is either injected in the channel or at reception/transmission, quantum information carriers are subject to degradation at contact with any external energy while at any point in the communication process. This degradation (quantum decoherence) can be described as a multiplicative (rather than additive) noise, which leads to an effectively constant degradation with time: it affects more to the older qubits, as it accumulates over time (see Fig. 5.1). This translates into making communication latencies (i.e. encoding, travel and waiting times) much more critical than in the classical case: it is not only a matter of timeliness but of validity of the received data. That is, longer latencies are equivalent to larger communication error rates.

On the other hand, qubits cannot be "read out" (measured) without effectively destroying their quantum state. This, in communication terms, implies that no retransmissions are

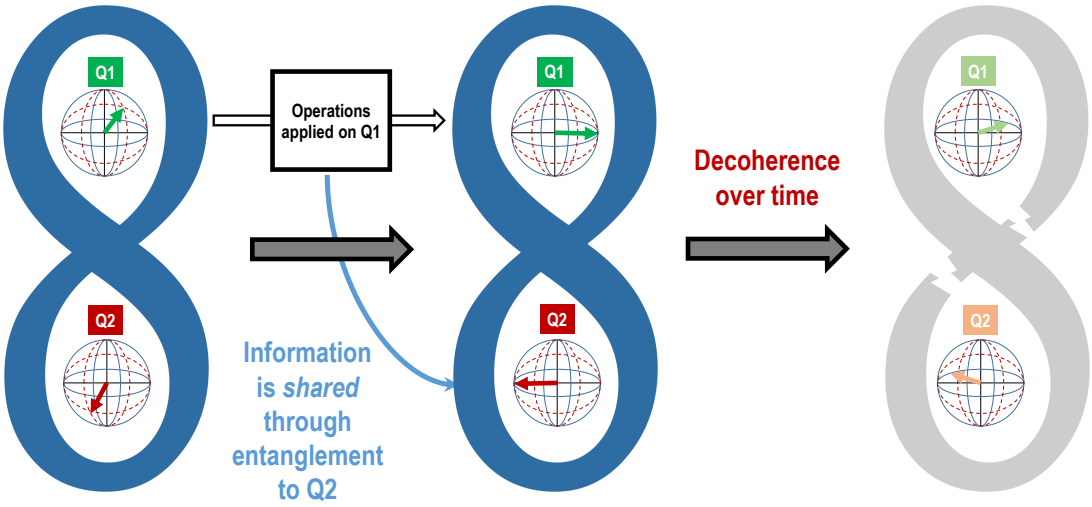


Figure 5.2: Quantum entanglement as a channel, also affected by quantum decoherence, which disentangles the group state

possible, as there is no way to replicate an arbitrary quantum state into another qubit (*no-cloning theorem*). That is, every quantum data exchange is a single-chance transmission, analogous to best-effort classical communications.

The combination of the effects of long communication latencies and the unavailability of retransmissions leaves us with a communication system that needs fast operation and zero tolerance for transmission errors to work properly. This is a perfect storm for designing a communication system: in classical communications, usually these two requirements are never demanded simultaneously. Environments with low tolerance to errors (e.g. distributed computing or storage) implement lengthier error codes and refined networking policies for optimizing Quality of Service (QoS) and make use of retransmissions whenever required. All these policies may imply large latency overheads, which are assumed by design. However, applications requiring ultra-low latencies usually accept in return information loss/errors (think of video calling or streaming). Hence, quantum communications pose a hard problem, especially when used in computing (such as multi-core or distributed architectures) due to their need for high-fidelity data.

Nevertheless, not everything is lost here. The same phenomena that complicate things for retransmissions and information quality build precisely the perfect environment for a sensitive topic: information security and privacy. Not being able to copy or read out quantum data without interrupting the transmission is the best defense against eavesdroppers. This powerful property has been already leveraged for several practical quantum communication applications, such as QKD (used for securely distributing a communication key among two parties [36]), communication among non-trusted nodes (such as quantum coin-flipping [217]), or blind quantum computation (for delegating computations to an untrusted device without compromising data privacy [218]).

Still, the complexity of latency and error rate requirements remains. One way to deal with it, circumventing or at least relaxing the requirements, is by using quantum entan-

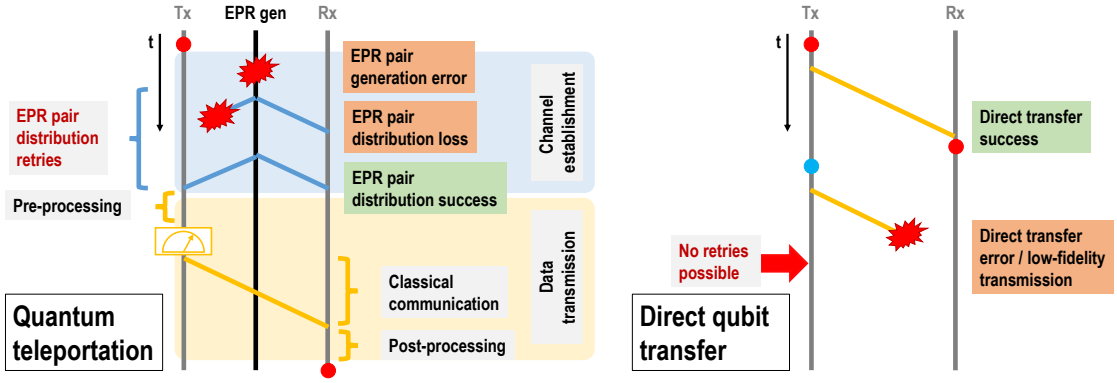


Figure 5.3: Quantum teleportation VS direct qubit transfer. The decoupling of channel establishment (entanglement distribution) and actual data transmission in quantum teleportation allows us to increase the chances of success with the ability to "retry" or retransmit the entanglement distribution process. In qubit direct transfer we are dealing “best-effort” communication paradigm.

glement (you may do a quick refresh on that by reviewing Section 1.1.1 if you need it). From a communications perspective, quantum entanglement works as a sort of instantaneous channel, where the quantum information is shared among several parties. It can also be described as a multi-partite quantum state distributed among several qubits that might be geographically distant, and whichever operation is made on any of the entangled qubits affects the whole state. Of course, quantum entanglement is still vulnerable to quantum decoherence, which *disentangles* the involved qubits (see Fig. 5.2).

5.1.1 Quantum teleportation: introducing channel Quality of Service

We have already introduced quantum teleportation in Section 1.1.5, as a versatile and indirect quantum communication technique that is applicable for communication at any distance, from the chip to planetary scales. It is generally considered a safer alternative to direct qubit shuttling for transferring quantum states on low-quality environments and demanding applications [204, 215]. This includes multi-core quantum communications, hence we assume this technology is the standard for inter-core communications for the rest of this manuscript. Let us see now why it has such an advantage over other alternatives.

As a brief reminder, in the quantum teleportation protocol, there are five main stages in the process: *i)* entanglement generation at the EPR pair generator, *ii)* entanglement distribution, when the EPR pair is generated and shared among the two nodes willing to communicate, *iii)* Tx operations, applied in the transmitter node to entangle the EPR qubit and the data qubit, *iv)* measurement at Tx and classical transmission of the results, and *v)* Rx operations, applied in the receiver node to the target qubit depending on the bits just received, finally obtaining the original data.

The crucial point here is the decoupling of the channel establishment and the data transmission itself. Direct qubit transfer is a single-shot communication attempt, and its success will depend on the environmental conditions at the time of communication. In qubit

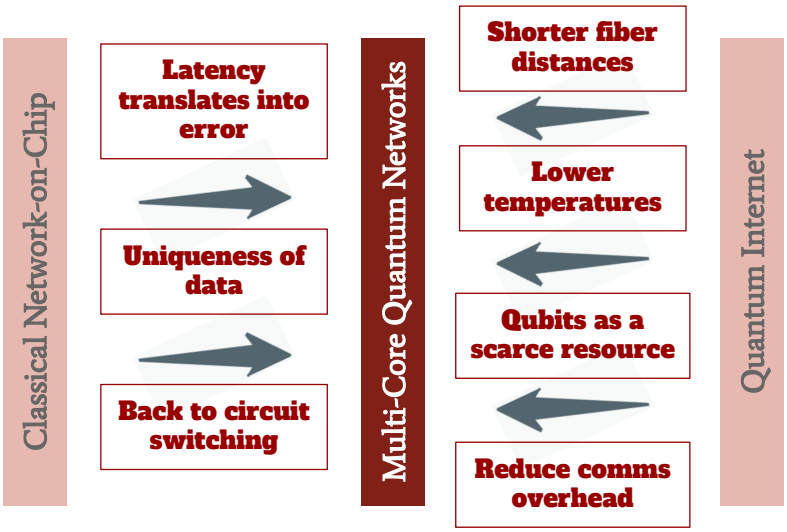


Figure 5.4: Two-way comparison between multi-core quantum inter-core communications and two analogous scenarios: Quantum Internet and Network-on-Chip.

teleportation, however, entanglement generation and distribution provide a reinforcement layer: the EPR pair is a well-known quantum state that, contrarily to what happens with the arbitrary quantum state we want to send, can be easily reproduced if lost or severely damaged when sending it (see Fig. 5.3). Only when we are sure we have a sufficiently high-quality entanglement, do we proceed to stages *iii)* and further, i.e. the actual transmission of the data. Of course, that introduces uncertainty on the global latency of the operation but may guarantee better conditions for the overall operation.

Indeed, quantum teleportation upgrades quantum communications by leveraging the power of quantum entanglement, enabling some sort of channel QoS.

5.2 Between the Quantum Internet and Network-on-Chip

While recent years have seen an explosion of research in large-scale quantum communications and networking for the QI, less attention has been placed on chip-scale communications for the scaling of quantum computers. Although the fundamentals of quantum communication remain the same at both scales, they provide substantially different environments whose specific characteristics shall be considered in the communications system design.

In order to provide a context analysis of the communications scenario concerning this thesis, we will compare multi-core quantum communications with two analogous applications. The comparison is summarized in Fig. 5.4. On the one hand, for its role as an interconnect among computing cores within or across chips, quantum interconnects may recall the classical concept of NoC. On the other hand, for its quantum nature and the need to transfer quantum data using mainly quantum teleportation, it may be compared with its *big brother*, the QI [5].

5.2.1 Comparison with the Quantum Internet

Research on the QI is one step ahead of multi-core quantum computers and its main focus is on quantum teleportation, which is also a strong candidate for the multi-core scenario. Hence, the QI could seem like a good source of inspiration for models and protocols of quantum multi-cores, yet with the following non-trivial differences.

Distance: In the classical communications world, distance is related mainly to attenuation, which can be fixed by deploying amplifiers along the line. In quantum communications, quantum states get decohered with time and interaction with the environment. This means that long fiber lines cause photon qubits to irremediably lose their information. However, due to the *no-cloning theorem*, it is not possible to re-use the classical solution and reconstruct the signal by *amplifying* it. But there is always an exception. The special case of EPR pairs quantum states can be restored with quantum repeaters [5,41]. Therefore, we improve performance by using quantum teleportation instead of physically sending the qubits, and applying this "amplification" technique to entanglement distribution. In the multi-core scenario, however, distances are in the millimeter scale, and hence there is no need (and no desire, due to stringent resource constraints) to deploy repeaters. This may simplify the link and network layer protocols.

Temperature: Going into the outside world along a country-wide inter-network makes it impossible to keep low temperatures to facilitate quantum coherence, and photonic qubits can fairly withstand room temperatures. However, inter-chip networks in multi-core architectures can be kept at cryogenic temperatures, thus increasing the coherence time for communicating qubits and improving the isolation of the whole computer. In order to facilitate this, all the circuitry used to control and communicate the qubits in quantum computers should ideally be prepared for operating at cryogenic temperatures. Moreover, given the limited cooling power of refrigerators, protocols need to be simplified to reduce the energy footprint. Such a constraint is not present in the QI.

Qubits as a scarce resource: Multi-core quantum computers are a resource-constrained environment, both in area and in power consumption, as opposed to the QI. On the one hand, every qubit that is sent to another core is unique: due to the *no-cloning theorem*, the only way to reproduce a qubit state is to repeat the computation all over again. In the QI scenario, we may have the sufficient quantum computational power to recalculate some small parts of the computation or implement large QEC codes as we are dealing with larger amounts of qubits, but in multi-core architectures, every qubit is invaluable and hence all qubit transfers must be operated with the highest of guarantees and ultra-low latency.

Communications overhead: Given the resource-constrained nature of this environment, the number of qubits that we can integrate on each chip is limited. In the QI, in order to protect qubits from decoherence, QEC techniques are applied. This implies that multiple physical qubits encode a logical qubit, which implies a very significant overhead. In multi-core quantum computers, qubits are scarce and, therefore, we may face the trade-off between dedicating qubits to QEC or to computation, having to sacrifice some protection along the way.

5.2.2 Comparison with Network-on-Chip

The NoC paradigm accompanied the rise of multi-core processors in the classical domain. Its divide-and-conquer approach is similar to that proposed in multi-core quantum computing, and both are highly resource-constrained environments: whereas NoC needs to efficiently leverage every microwatt of power while integrating big caches at different levels, multi-core quantum processors must be designed in a highly conservative manner aiming at making the most out of every nanosecond: latency translates into error, hence timeliness means higher network capacity. Of course, both NoC and multi-core quantum inter-core networks may share some design principles, but interconnecting quantum nodes implies some particularities, namely:

Latency as an error: In NoC, latency is critical because it essentially delays the computation [219], but it is generally not tied to a loss of accuracy. In contrast, qubits tend to decohere as time passes, which implies that the communication latency not only delays the computation in multi-core quantum computers but also degrades its accuracy, to the point of completely destroying it when a particular latency threshold is exceeded. This fundamentally impacts flow and congestion control protocols, if any.

Uniqueness of data: Due to the *no-cloning theorem*, data cannot be copied and distributed to multiple cores. Data is physically moved around, and therefore scheduling the communication operations is of critical importance to minimize qubit movements and consolidate interactions with a given qubit in the minimum amount of time. A NoC is generally not bound to scheduling, although efforts in real-time embedded systems or machine learning accelerators also advocate for it in the classical domain [220, 221].

Welcome back to circuit switching: Quantum teleportation uses both a classical channel and a quantum channel to transmit the information: the measurement output at the Tx node (2 bits), and the EPR entangled pair. While the classical channel works as expected (plain information traveling through a wire), the quantum channel is not used to transmit directly quantum information, but a quantum resource: entanglement. This resource is used thereafter to *teleport* the qubit. In this way, we could describe the entangled pair as a channel itself, being the EPR pair generator a shared resource. Similar to what is done in circuit switching technologies, the two communicating parties are interconnecting by means of a channel (entanglement) reservation. This makes multi-core quantum networks depart from the traditionally packet-switched NoC paradigm: the topology among cores is virtually configured as needed, and there is a shared resource (the EPR pair generator, which may also be decentralized) that enables communication.

5.3 A Communications-Centric Model of Quantum Teleportation

In order to adequately explore the implications of quantum communications on the multi-core architecture design, we need to look deeper into modeling the main underlying process of inter-core links, i.e. quantum teleportation.

Quantum teleportation was theoretically demonstrated some decades ago [222] and experimentally shown shortly thereafter [223]. In recent years, this technology has been validated for distances of tens or even hundreds of kilometers via satellite and optical fiber

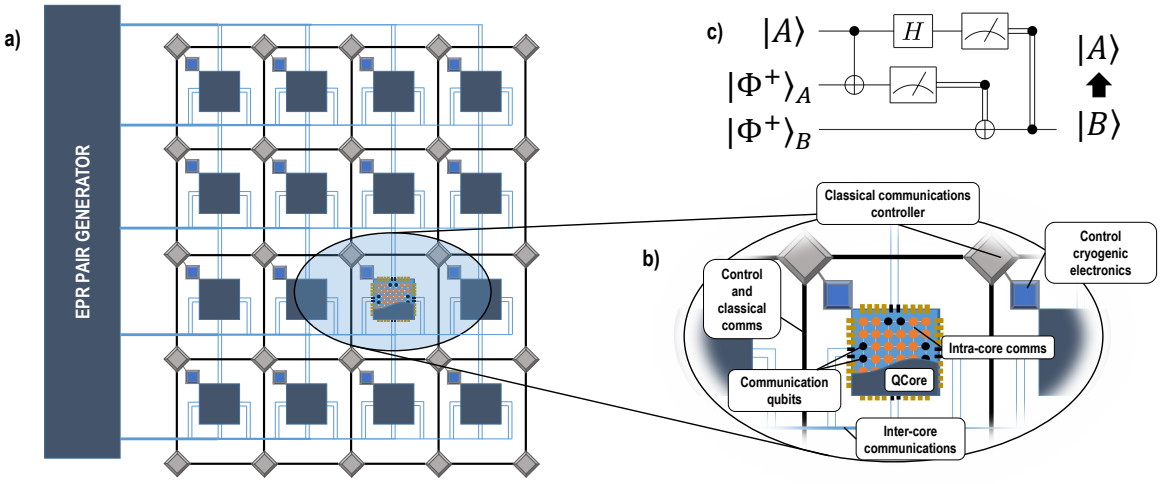


Figure 5.5: Multi-chip quantum computer full view. a) 2D diagram of a multi-chip architecture. The classical network also depicted completes the networking infrastructure, b) Enumeration of the components, including intra- and inter-core communications, and c) Circuit for quantum teleportation.

links [224, 225]. However, most of the research conducted on it is focused on improving link quality and robustness, as well as on developing further supporting technology (EPR pair generators, quantum repeaters, etc.).

In contrast, the importance of modeling quantum teleportation as a communications system has only recently been highlighted. This kind of model enables a deeper understanding of this technology from the communications standpoint, which facilitates and provides guidelines for protocols and system design [43, 226]. Models for quantum teleportation have been mostly developed for the QI scenario [215]. Because of the differences previously mentioned between multi-core quantum communications and QI, specific modeling of the former is needed. This is of crucial importance because the performance of a multi-core quantum computer will depend on its communications.

Let us assume a quantum computer composed of N_{CORES} cores that communicate through quantum teleportation (Fig. 5.5). This implies having three resources available for every core-to-core link:

- A classical network enabling classical data passing between any pair of cores in the computer.
- An EPR "factory" or generator, which produces the EPR pairs, to be then distributed to the end nodes. Using photons as entanglement carriers is widely accepted due to their inherent mobility and low interaction with the environment. Ranging from low to high resource requirements, the EPR generation and later distribution could be implemented as [215]:
  - A single EPR pair generator shared among all cores (1 teleportation at a time), using typically spontaneous parametric down-conversion. In this case, the pair of resulting photons are entangled in polarization in one of the Bell states by means

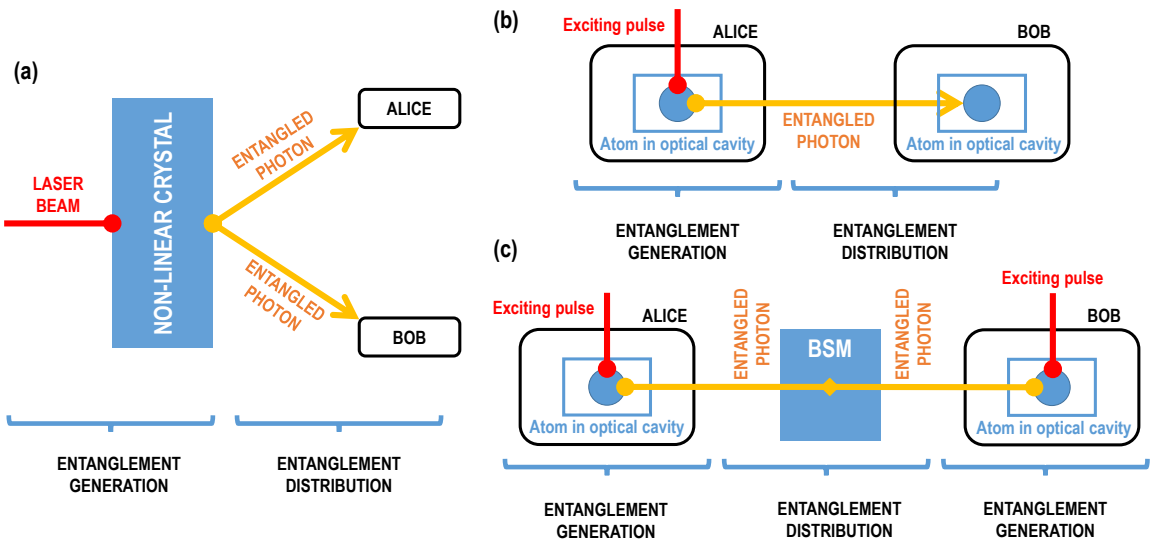


Figure 5.6: EPR generation and distribution techniques. a) EPR pair generator is shared among all nodes, b) The Tx node generates the entanglement and transmits it to the Rx side, c) Both nodes generate photons which meet in the middle to generate entanglement.

of a laser beam which is directed toward a non-linear crystal (see Fig. 5.6a). After this, each photon is sent to one of the communicating nodes.

- Several EPR pair generators are shared among all cores (several teleportations can be performed in parallel), replicating the same process as before.
- An entangling device allocated for every link between a given pair of nodes. This could be practically implemented by means of any of these two techniques:
  - EPR generation at source node (see Fig. 5.6b). The entanglement is in this case generated when an atom in an optical cavity placed at each node is excited by a laser pulse. This generates a photon whose polarization is entangled with some internal state of the atom. The photon is then transferred to the receiving node distributing thus the entanglement.
  - EPR generation at both end-points (see Fig. 5.6c). This third scheme replicates the previous one but this time both nodes produce a photon, which they send to the other node. Both photons meet at a beam-splitter, where they are measured, effectively entangling both nodes at their "communication atoms".

Using any of these configurations and techniques depends on the qubit technology employed and other considerations. Although all of them preserve the same functionality and have similar behavior, for our highly-constrained multi-core environment we assume from now on the first configuration (a single EPR pair generator shared among all nodes).

- A quantum optical network connecting the EPR pair generation facilities and the cores.

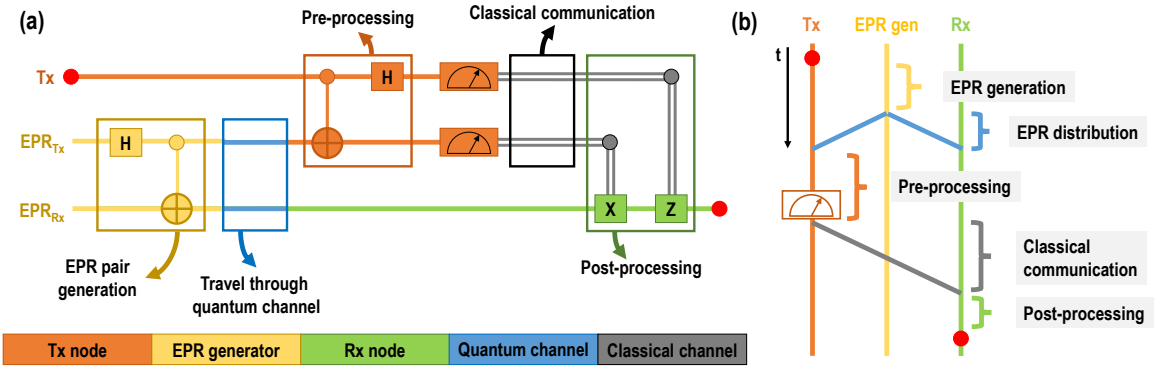


Figure 5.7: Teleportation circuit and time sequence diagrams, color-coded by the locations where each phase is performed.

Each of the cores integrates N_Q^{CORE} physical qubits, and has N_P transducer ports for the quantum interconnection with other cores, allowing for the same number of parallel transfers. Each of the ports has a qubit buffer attached to it. The total number of qubits in the core dedicated to buffer/communication tasks is N_Q^{COMM}, leaving $N_Q^{COMP} = N_Q^{CORE} - N_Q^{COMM}$ for computation.

The quantum teleportation process can be illustrated using the circuit diagram from Fig. 5.5c, which is the most common representation for quantum algorithms. However, this depiction does not show clearly the physical location of each phase, nor the inter-dependencies and operations latencies in the process. In Fig. 5.7a, a "localized" version is shown, where each color represents a different location in the network. Furthermore, Fig. 5.7b shows the corresponding time diagram which breaks down the delays of the teleportation process. Latency is indeed crucial, as it is directly correlated to the decoherence of qubits.

In our model, a qubit transfer proceeds as follows: when a given qubit q holding a quantum state $|\phi\rangle$ has to be moved to a different core, the controller checks whether any of the N_P communication ports are free. If not, the qubit needs to wait in the communication buffer as long as there is any available position, to leave computing space for other parallel operations needing it. In case the buffer is also full, we give up on the operation and assume that the communication of that qubit will not be possible. The control system would then have to decide whether the computation has to restart or the whole operation flow has to stop and wait till the communication subsystem is freed up.

When a communication port becomes available, the controller triggers the entangled pair generator in order to start the EPR pair generation and distribution. This process deserves a little more attention: let us dive into it.

5.3.1 Entanglement generation and distribution

Generating a pair of entangled photonic qubits typically implies a probabilistic photon pair emission and subsequent entanglement by forcing them to interfere with a beam splitter. This process' success can be usually confirmed (a.k.a heralded) via the detection of the two photons at the output [151]. Therefore, the process of entangling two qubits is divided into three main phases:

1. Generating a pair of photons
2. Coupling them together to form a Bell basis (also called EPR pair)
3. Distribute both of them among the two end nodes.

The probabilistic nature of this process calls for a more detailed analysis. Following, for instance, recent results on entanglement generation for ion traps [143, 227] (the process is similar for other qubit technologies) we can first look at how the total operation time is distributed among these three sub-processes:

1. Single-photon generation is a quite mature technology with single-photon lasers that can realize high photon rates at around 100 MHz.
2. Achieving (and detecting) entanglement of the two qubits is a hard task that requires several different processes to concur in order to be successful (e.g. producing the correct Bell states, state preparation, excitation, detection...). Usually, it decreases the final entangled photon rate to 1 in 1000 laser-emitted photons.
3. Successfully sending photons over a fiber can be safely assumed to happen without any loss, i.e. as a deterministic process, especially when working at very short distances, such as in a multi-core environment.

Therefore, the photon coupling sub-process implies the actual bottleneck in the whole EPR generation process.

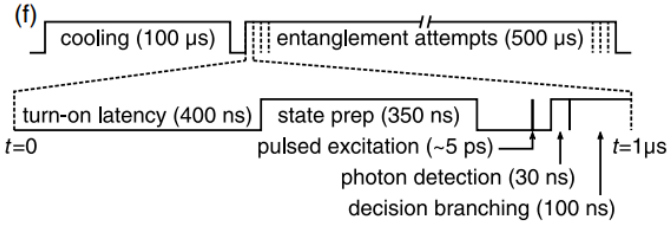


Figure 5.8: EPR generation experimental sequence. Figure extracted from [143]

In order to analyze the probability distribution of the photon entangling process, we can look at a recent example, extracted from a recent article [143]. In Fig. 5.8, we see the working cycle of the EPR pair generator, formed by a 100 microseconds cooling phase and a series of entanglement attempts. Each attempt is divided into:

- A turn-on latency time
- State preparation (followed by a safety delay)
- Pulsed excitation
- A short detection window
- Time for decision branching based on the outcome of the attempt

This is called an entanglement attempt because it does not produce always an entangled pair[1]. This failure may happen for several reasons:

- Only two out of the four possible Bell states are valid for the heralding process (detecting entanglement), and we have failed to produce those
- The state preparation has failed
- The excitation pulse has not been successful
- Other technology-related failures
- The entangled pair was correctly produced, but the detection failed

As all of them are success/failure processes, we may well say that our probability P of having a successful entangled pair generation attempt follows:

$$P = f\Big(\prod_{\forall i} p_i\Big) \tag{5.3.1}$$

where the different p_i correspond to the success probability for the concatenated processes explained above.

Therefore, generating an EPR pair (attempting it until we obtain a success) can be basically reduced to a Bernoulli trial (i.e. success/failure experiment), being P the success probability.

This implies that the time to have a successful attempt follows a geometric probability distribution function, i.e.:

$$P(X = k) = p \cdot (1 - p)^{(k-1)} \tag{5.3.2}$$

, where X is the random variable describing the number of attempts, and k, is the kth attempt, being it the first success. The average number of attempts, till we have a success, will then be:

$$E[X] = \frac{1 - p}{p} \tag{5.3.3}$$

And the probability of having a generation time (i.e. time to success) larger than $n \cdot T_{attempt}$ (i.e. n attempts, being $T_{attempt}$ is the time that a complete attempt takes to be done) is:

$$P(X > n) = (1 - p)^n \tag{5.3.4}$$

That is, the probabilistic nature of the entanglement process leads to an unbounded time to generate the EPR pair, hence affecting the overall waiting time (and as we know, that means more decoherence inserted into the computation). The error probability of this operation will depend on the EPR generator distribution probability distribution function

[1]This scheme remains valid for most of the EPR generator techniques, be it produced in a centralized generator or by coupling photons obtained at the end nodes, no matter the qubit technology.

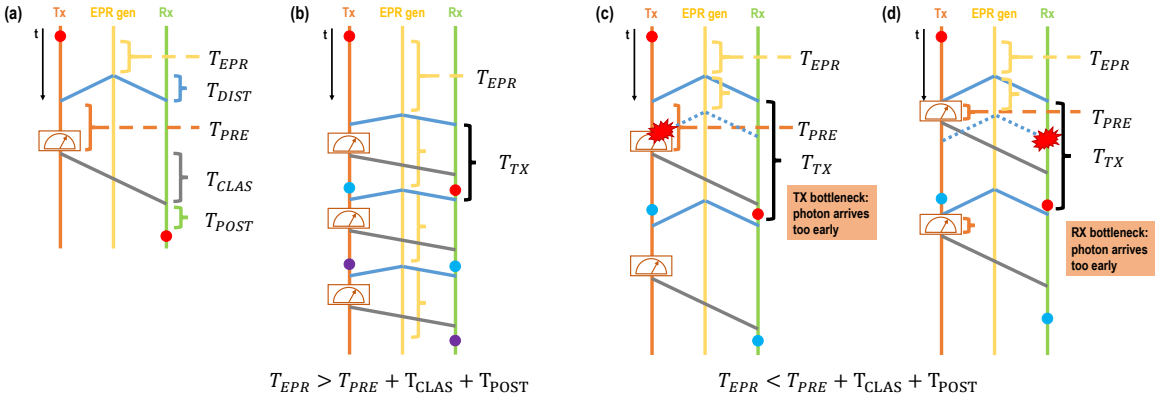


Figure 5.9: Maximum qubit rate in quantum teleportation. a) Time sequence of qubit teleportation with the corresponding phases, b) Maximum rate $R_{MAX} = 1/T_{EPR}$, when T_{EPR} takes longer than the rest of the process, c) Tx bottleneck, when $T_{PRE} > T_{EPR}$, (d) Rx bottleneck, when $T_{PRE} + T_{CLAS} + T_{POST} > T_{EPR}$. A qubit rate of $1/(T_{PRE} + T_{CLAS} + T_{POST})$ corresponds to cases where there are bottlenecks either at Tx or at Rx.

and how the quantum network layer manages this type of failure. Observe that the rest of the entire teleportation process can be said to be deterministic, so putting effort into improving this specific functionality can have a great effect on overall communications performance.

5.3.2 Qubit teleportation delay

With this probabilistic nature of the EPR pair generation process in mind, we can hence assume that it takes a non-deterministic time with mean T_{EPR} (Fig. 5.9), after which the entangled photons can be sent to the Tx and Rx nodes. We assume an ideal optical channel with no photon loss and with a fixed delay $T_{DIST} = d/c'$ related to the distance d between the EPR pair generator and the nodes, and the speed of light in the optical medium $c' = c_0/n$ with n the refractive index of the material, and c_0 the speed of light in the void. As we are assuming that the EPR pair generator is shared among all cores, T_{EPR} will depend also on its utilization by the other cores in the network (i.e. it may include waiting times due to busy EPR pair generator).

On the Tx side, the pre-processing involving the entangled qubit and the qubit to be transferred takes a fixed delay T_{PRE}, typically composed by the delay of applying a CNOT gate, a Hadamard gate, and the measurement on both qubits. The resulting classical bits are sent to the Rx node, in a process assumed error-less and taking a fixed time T_{CLAS}. At Rx, each of these bits controls whether an X or a Z gate (or both) should be applied to the received entangled qubit. Therefore, this post-processing takes on average $T_{POST} = 1/2 \cdot (T_X + T_Z)$.

Consequently, in the general case, a single qubit transfer will take T_{TX}, which corresponds to the critical path on the time diagram in Fig. 5.9,

$$T_{TX} = T_{EPR} + T_{DIST} + T_{PRE} + T_{CLAS} + T_{POST}. \tag{5.3.5}$$

5.3.3 Maximum qubit transfer rate

This communication process consists of various consecutive operations on a middle node (the EPR generator) and at both the Tx and Rx nodes. From a communications perspective, it is essential to compute the maximum qubit transfer rate (R_{MAX}) when the system is operated continuously. This will depend on the ratios among the different timing components in Eq. (5.3.5).

The EPR pair generation and distribution acts as the quantum channel. Thus, the rate at which such generation happens is the first and fundamental bottleneck, hence having a transfer rate upper-bounded by the time to generate an EPR (see Fig. 5.9b), so that

$$R_{MAX} \leq \frac{1}{T_{EPR}}. \tag{5.3.6}$$

That is, the pre- and post-processing and Rx and Tx can be hidden without affecting the actual rate. However, the equality will hold only when the pre- and post-processing do not become a bottleneck. In any of these cases, if we force the EPR generator to continuously work at $R_{EPR} = 1/T_{EPR}$, either the Tx or the Rx side will be busy when receiving the EPR photon, thus losing it. Both bottlenecks could be solved by having a EPR receiving buffer to which the arriving photon is immediately swapped, but in such a qubit-constrained environment we have chosen to avoid this assumption. For the sake of simplicity, let us study in the case $N_P = 1$ which is the threshold for these bottlenecks:

- **Pre-processing bottleneck:** Both Tx and EPR_{Tx} qubits have to be operated and measured before a new entangled pair is received, hence we will have no bottleneck iff $T_{PRE} < T_{EPR}$ (see Fig. 5.9c).
- **Post-processing bottleneck:** The Rx qubit has to wait for the classical bits and operate two single-qubit gates, hence we will have no bottleneck iff $T_{PRE} + T_{CLAS} + T_{POST} < T_{EPR}$ (see Fig. 5.9d).

Observe that the second inequality (Rx) implies the first one (if T_{EPR} is greater than $T_{PRE} + T_{CLAS} + T_{POST}$, it is also greater than T_{PRE}). That is, the post-processing at Rx has to wait for the T side to end the pre-processing, hence the Rx bottleneck prevails.
With this in mind, the rate R_{MAX} is:

$$R_{MAX} = \begin{cases} (T_{PRE} + T_{CLAS} + T_{POST})^{-1} & \text{if } T_{PRE} + T_{CLAS} + T_{POST} > T_{EPR} \\ {T_{EPR}}^{-1} & \text{otherwise} \end{cases} \tag{5.3.7}$$

5.3.4 Simulation Results

Modeling quantum teleportation process for multi-core quantum networks from a communications perspective has let us understand its performance and determined its delay and maximum rate. Departing from the theoretical analysis to perform a set of simulations will allow us to: *i)* validate the effects of delays and losses on the teleportation fidelity and error rate using experimental values, *ii)* explore the behavior of a simulated multi-core environment under stress, by executing a randomly distributed quantum circuit under varying ratios

Table 5.1: Notation, symbol definitions and values used in simulations of Fig. 5.10, taken from recent superconducting qubit technology used in [130] and [228]

Notation	**Meaning**	**Value**
N_{CORES}	Number of cores/chips	2
N_Q^{CORE}	Number of qubits per chip	1024
N_Q^{COMP}	Computation qubits in a chip	[1,1023]
N_Q^{COMM}	Communication qubits in a chip	1024 - N_Q^{COMP}
N_P	Number of inter-core ports per chip	1
T_{EPR}	Mean of EPR pair generation time	10^3 ns
T_{DIST}	EPR pair distribution time	0.01 ns
T_{PRE}	Pre-processing time	390 ns
T_{CLAS}	Classical transfer time	0.02 ns
T_{POST}	Post-processing time	30 ns
T_{TX}	Total time of quantum transfer	1420 ns
λ	Inter-core qubit rate	[10^5,10^8]
R_{MAX}	Maximum qubit transfer rate	10^6 qbps
n	Refractive index of the optical medium	1.5

of communication and computation qubits, and *iii)* dimension the network requirements for different architecture topologies.

For this task, we have used the NetSquid simulator for quantum networks [70]. Although it has been designed having in mind large-scale quantum communications and the QI, it is fully functional and adaptable to short-range multi-core quantum architectures. It has allowed us to simulate a realistic setting, with quantum memories and processors, EPR pair generators as well as classical and quantum links. The simulation includes the whole operation on qubits and each step of the teleportation process, taking into account qubit decoherence, quantum gates latencies, communication delay models, etc. under realistic assumptions.

In the first set of simulations, we have modeled two independent quantum cores connected to an EPR pair generator and interconnected also directly through a classical link (see Fig. 5.10a). Apart from the processing and networking capabilities, we have implemented a quantum First-In First-Out (FIFO) buffer for the qubit transfer, in order to explore the effects of long qubit waiting times on the fidelity of the qubit communication when the link is stressed and qubits need to wait for the EPR pair generator to be free. We have also implemented the teleportation protocol on each node, adding the buffer management. Both quantum nodes and EPR generator work with parameters summarized in Table 5.1, taken from recent superconducting qubit technology used in [130] and [228].

In order to stress the system for high communication rates, we have started our analysis with a focus on a single teleportation link among two nodes. We have introduced a constant

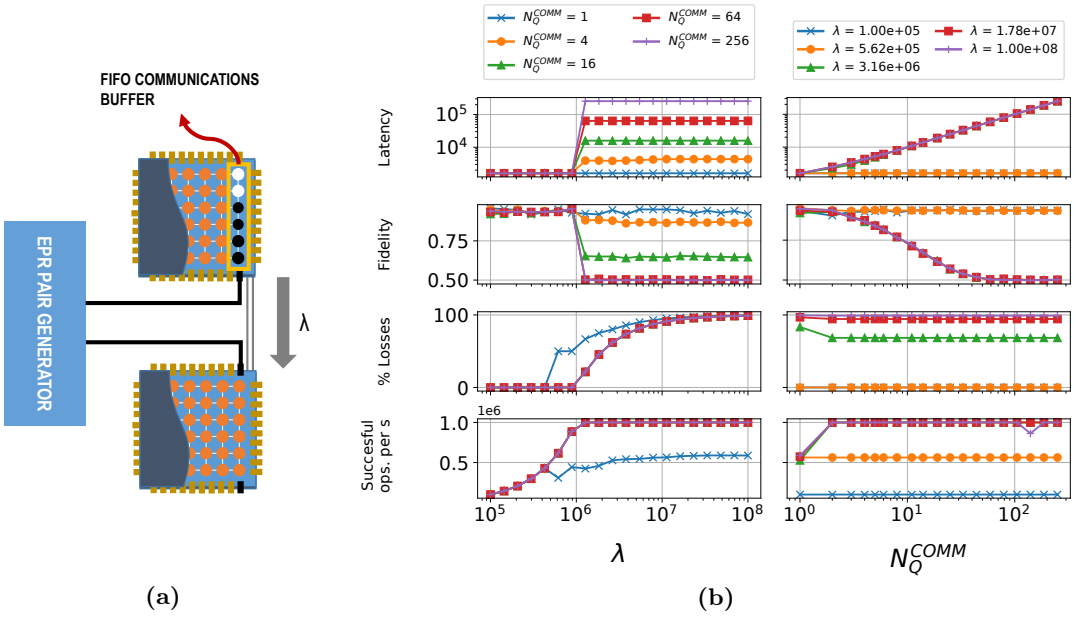


Figure 5.10: Stress test of a quantum teleportation channel at multi-core scales. The 2D input design space is swapped both in traffic (λ) and size of the quantum buffer (N_Q^{COMM})

rate (λ) qubit transmission process at the Tx node (Alice), which translates into a constant traffic arriving at the Tx waiting buffer.

The DSE over the λ input qubit rate and the buffer size at the Tx node (N_Q^{COMM}) is shown in Fig. 5.10b. From top to bottom, the successful number of teleportations per second, average communication latency, average fidelity, and the percentage of qubit losses are shown. Interestingly enough, observe how the latency and fidelity plots are mirrored, validating the direct relationship between communication latency and overall error.

See how the system saturates for $\lambda > 10^6$ (the maximum teleportation rate). As the system complies with the pre-processing and post-processing inequalities, the link capacity is bounded by the EPR generator instead of the pre- or post-processing operations. Greater input traffic only translates into a higher loss rate, without any improvement on any other performance metric. The exception is the case when there is no buffer ($N_Q^{COMM} = 1$): in that case, the absence of a buffer implies losing any arrival while a teleportation is in process, affecting the overall rate, which tops at around $5 \cdot 10^5$. Observe that having large buffers does not make the difference in terms of losses, but it does in terms of fidelity: for a saturated system, larger buffers mean longer average waiting times, and that implies lower overall fidelity, which implies worse performance. Therefore, it is sufficient to have a minimum buffer to avoid as many losses as possible without losing too much fidelity: observe that a buffer with capacity 1 leads to optimum performance for all cases.

In the analysis just presented, we have focused on the size of the buffer and the required teleportation rate. Looking from a higher perspective, we would now like to study the dimensioning and distribution of computation and communication in a full multi-core quantum architecture. In particular, the scarce resources available cause a trade-off on where to assign qubits: computation or communication. In principle, if the core has a low amount

of communication qubits ($N_Q^{COMP} >> N_Q^{COMM}$), it will be able to compute a reasonable amount of operations without needing to communicate with other cores. However, for algorithms with a high rate of inter-core operations, the constrained resources for comms will affect the performance of the overall computation. On the other hand, if the core has fewer computation qubits in order to have better communications ($N_Q^{COMM} >> N_Q^{COMP}$), we will need a greater amount of cores interconnected for computing large algorithms.

Table 5.2: Notation, symbol definitions and values used in simulations of Fig. 5.11, also with reference to values from [130] and [228]

Notation	**Meaning**	**Value**
N_{CORES}	Number of cores/chips	256
N_Q^{CORE}	Number of qubits per chip	[64,1024]
N_Q	Total number of qubits in the quantum computer	$N_{CORES} \cdot N_Q^{CORE}$
N_Q^{COMP}/N_Q^{COMM}	Computation to communication qubit ratio	[0.1,0.9]
N_P	Number of inter-core ports per chip	1
N_G	Number of single- and two-qubit gates in the circuit	n.d.*
N_G^{2Q}	Number of two-qubit gates	$0.8 \cdot N_G$
T_{EPR}	Mean of EPR pair generation time	10^3 ns
T_{DIST}	EPR pair distribution time	0.01 ns
T_{PRE}	Pre-processing time	390 ns
T_{CLAS}	Classical transfer time	0.02 ns
T_{POST}	Post-processing time	30 ns
T_{TX}	Total time of quantum transfer	1420 ns
T_{exec}	Upper-bound of the execution time of the circuit	n.d.*
λ_{core}	Inter-core qubit traffic per core	[10^5,10^8]
λ_{qubit}	Total qubit traffic per qubit	n.d.*
n	Refractive index of the optical medium	1.5

* This parameter is only used for theoretical development in Eqs. 5.3.8 and 5.3.9.

To validate and extract more conclusions around this, we have performed an analysis on the issue simulating the execution of a quantum algorithm on a multi-core quantum computer. The random algorithm is described with two ratios: the number of gates per qubit N_G/N_Q and the proportion of two-qubit gates N_G^{2Q}/N_G. Because of the randomness of the algorithm, the two-qubit gates are uniformly distributed among all qubits and over time. Then every qubit is involved in an equal amount of two-qubit gates, equally distributed along time following a rate λ_{qubit}:

$$\begin{aligned}\lambda_{qubit} &= \frac{\#\text{2-qubit gates}/\#\text{computation qubits}}{\text{execution time}} \\ &= \frac{\frac{2 \cdot N_G/N_Q \cdot N_G^{2Q}/N_G}{N_Q^{COMP} \cdot N_{CORES}}}{T_{exec}} \\ &= \frac{2 \cdot N_G^{2Q}/N_Q}{N_Q^{COMP} \cdot N_{CORES} \cdot T_{exec}}\end{aligned} \tag{5.3.8}$$

where T_{exec} is the upper bound of the time it takes to execute the whole algorithm.

Therefore, we can compute the equivalent λ_{core} (the inter-core qubit rate required from any of the cores when executing the algorithm) for each of the cores as the total traffic from all the computation qubits in the core (N_Q^{COMP}) which destination is outside the core (hence requires teleportation):

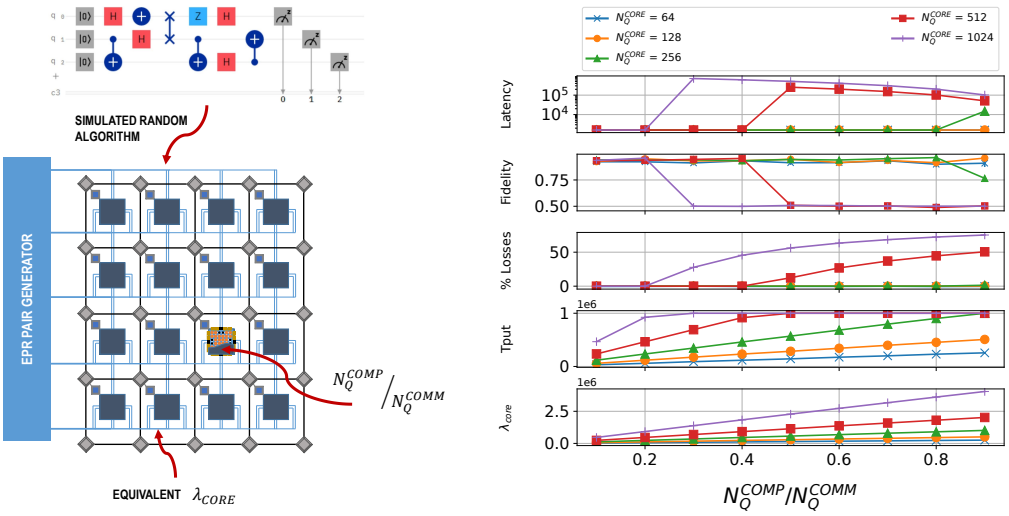
$$\begin{aligned}\lambda_{core} &= \lambda_{qubit} \cdot N_Q^{COMP} \cdot \frac{\text{qubits outside own core}}{\text{total number of qubits}} \\ &= \frac{2 \cdot N_G^{2Q}/N_Q}{N_Q^{COMP} \cdot N_{CORES} \cdot T_{exec}} \cdot N_Q^{COMP} \cdot \frac{N_Q^{COMP} \cdot (N_{CORES} - 1)}{(N_Q^{COMP} \cdot N_{CORES} - 1)} \\ &= \frac{2 \cdot N_G^{2Q}/N_Q}{N_{CORES} \cdot T_{exec}} \cdot \frac{N_Q^{COMP} \cdot (N_{CORES} - 1)}{(N_Q^{COMP} \cdot N_{CORES} - 1)}\end{aligned} \tag{5.3.9}$$

Observe that for big architectures (high N_Q^{COMP} and N_{CORES}), $\lambda_{core} \sim \lambda_{qubit} \cdot N_Q^{COMP}$.

In our simulations, we have assumed a single EPR generator connected to all cores. Having a constant λ_{core} for all cores, the optimal sharing strategy is equivalent to a Time Division Multiplexing (TDM). Therefore, each core sees the EPR generator as a source with an observed T_{EPR} equal to N_{CORES} times the actual T_{EPR}.

To explore the computation/communications ratio, we have assigned a fixed role to each of the qubits in a core: either it is used for computation or for the buffer. Therefore, changing the ratio of N_Q^{COMP}/N_Q^{COMM} effectively changes the λ_{core} (check also Eq. 5.3.9), as well as the buffering capabilities.

In Fig. 5.11, we show the corresponding exploration over the communication to computation qubit ratio, for the values of Table 5.2. This implies that the actual size of the algorithm being executed gets smaller as we increase the ratio. The bottom plot corresponds to the λ_{core} for each ratio. On the rest of the plots, from top to bottom, the successful number of teleportations per second, average communication latency, average fidelity, and the percentage of qubit losses are shown. The saturation point (check fidelity and latency subplots) in this case implies a computation to communication ratio that is producing too much traffic for the communication network and hence degrades the overall performance. Therefore, for each value of N_Q^{CORE} (total number of qubits per core), the optimum ratio would correspond to the maximum ratio for which the system does not saturate. For instance, for



(a) **(b)**

Figure 5.11: Designing the computation to communications qubit ratio. A multi-core system with 256 cores computation to communications ratio is designed by testing the performance of a random algorithm with 80% two-qubit gates.

512 qubits per core, we should have 40% of the qubits reserved for computation, while the rest would be reserved for communication buffering purposes.

5.3.5 Conclusions

Quantum communications pose completely new challenges that are fostering innovative research, especially on large-scale communications, such as QI applications. However, more attention should be paid to multi-core quantum computing, which is a key approach for quantum computing scalability and hence its ultimate success.

In this chapter, we have investigated the critical trade-offs, particularities, and optimal designs of short-range quantum communications for these architectures. In particular, we have shown how waiting times and latencies can greatly affect quantum communication quality, and have stressed such a communication system to validate its capacity bounds, which we have previously analytically modeled. Also, we have tested how a quantum algorithm (the random case) may behave in a many-core scenario. In particular, due to the scarce amount of qubits available, we have explored and obtained the optimal share of qubits among the computation and communication roles. Interesting design insights and specific parameters have been obtained for different cases.

In the next chapters, we will dive into studying the behavior of structured well-known algorithms, together with the impact of multi-core algorithm mapping and architecture topology. This research line can be followed up in the near future by exploring environments with a larger number of EPR generators and inter-core ports, as well as analyzing other communication techniques, such as teleported gates or different EPR generation and distribution architectures.

Chapter 6

Analyzing and Co-designing Multi-core Quantum Communications for Scalable Quantum Computing

It is now a good moment to face the third of the questions raised at the beginning of this thesis and summarized in Section 2.3.1: *How could we improve inter-core communications in order to have fast/efficient multi-core quantum architectures?* In the previous chapter we have been able to dive into this key design element of multi-core quantum architectures (inter-core communications), and we have even been able to determine some dimensioning of such a system, analyzing it under stress. However, we have studied it under generic assumptions of constant traffic and random algorithms, which gives us only partial information and hence limits our ability to optimize the system any further.

Indeed, blindly optimizing the communications in any system is difficult: every piece of information about the structure, behavior, frequency, expected usage, etc. can provide valuable information for the design of the network. Not only for being able to cope with the expected requirements in an efficient way but also for avoiding over-dimensioning the resources. The constrained environment we are working on is not different and needs to leverage any extra knowledge of the specific requirements of real usage for better optimizing the whole system.

In order to adequately design a communications system it is typically useful to use models from queuing theory, i.e. to study them as server systems (and their related parameters: queue length, service rate, capacity, etc.) [229]. For exploring such systems it is crucial to know as much as possible about the expected load. In a communication network, the load is called *traffic*, and can be described by means of its space and time distribution, maximum expected rate, etc. Of course, these parameters are difficult to know *a priori*, hence characterizing the traffic in a given system/environment *a posteriori* shall be a useful analysis for further design improvements and optimization of the network. We can reasonably expect that quantum networks may benefit from this same approach, though we need to take into account the specific characteristics of quantum communications, as we have reviewed in the previous chapter.

Once we have some knowledge of the traffic load in our quantum multi-core system (work presented in Sections 6.1 to 6.3.3.1), we could use that information to adequately manage the network for improving its performance already *during the execution of the quantum circuit*. In the second part of this chapter, we have explored such an idea by designing and testing the first multi-core quantum computer simulator. Through a curated implementation of the communication layers, we have been able to test how controlling communication operations might help to improve overall computing performance. We have done this through a QoS-fashioned prioritization MAC protocol specifically designed for multi-core quantum inter-core networks (see Section 6.4 and therein). By designing a MAC protocol focused on computing improvement, we are effectively entangling quantum multi-core network design with the knowledge of various layers of the multi-core quantum architecture following the constant call in this thesis for a co-design approach. In the post-NISQ era, when resources in QC will be less scarce, we might go for abstracting lower and upper layers for a more robust design, but that is a luxury that only high-performing and well-established systems can afford.

In the following, we detail our advances in such quantum traffic characterization by providing a spatiotemporal analysis of inter-core quantum networks, which allows us to assess the impact of the quantum algorithm, mapping process, and architecture on the actual execution. Our results suggest that conscious design and optimization of offline and online network management (such as the communications control layer design we present also here) in multi-core quantum architectures may help to solve current issues in the amount and distribution (over time and space) of inter-core quantum data transfers.

6.1 Characterizing the Spatio-Temporal Qubit Traffic within multi-core Quantum Computers

The first step to this co-design approach is to perform traffic characterization. To this aim, in the following, we present a technique to perform a spatio-temporal analysis of the traffic from quantum circuits running in multi-chip quantum computers. Specifically, we focus on the qubit traffic resulting from operations that involve qubits residing in different cores, and hence quantum communication across chips, while also giving importance to the amount of intra-core operations that occur in between those communications. Using specific multi-core performance metrics and a complete set of benchmarks, our analysis enables us to develop network strategies for improving multi-core quantum computer performance, such as the communications control layer design we present in Section 6.4.

6.1.1 Using traffic analysis for performance analysis

In classical multi-core computers, the design of its internal NoC has become of extreme importance due to its impact on the performance of the entire processor. Since the design of any network requires an understanding of the traffic it needs to serve, considerable efforts have been spent over the years to characterize multi-core systems and the applications that run on them.

Early works by Soteriou *et al.* [230] and Barrow *et al.* [231] analyzed a variety of multiprocessors between 16 and 32 cores running standard benchmark suites such as SPEC or

PARSEC. In the former, the temporal burstiness, spatial hotspotness, and source-destination distance were studied, whereas, in the latter, the focus was more on analyzing the memory-sharing patterns leading to such traffic characteristics.

Subsequent studies pushed the analyses to larger systems up to 64 cores and delved into particular aspects, such as the time-varying characteristics of the traffic [232], which is often periodic as analyzed in [233]. This is due to the iterative nature of most algorithms running in multiprocessors, which further suggests that traffic is predictable. Further, the work in [234, 235] focused on multicast traffic only, demonstrating that such a subset of the workload is also bursty and predictable.

These workload characterization studies had several impacts on the NoC field. In particular, they allowed to:

- study aspects such as the correlation between particular traffic characteristics and on-chip network congestion [233],
- create synthetic traffic generators better reflecting real workloads for the evaluation of NoC designs [230, 232, 235], and eventually,
- guide the design of improved topologies, routing policies, or congestion control mechanisms at the chip scale.

A pertinent question is then whether a similar approach can be used to characterize the workload of quantum processors. Before answering that question, though, it is important to see the main differences between both worlds. In classical shared-memory architectures, most of the traffic is an implicit consequence of the memory accesses produced by a multi-threaded application and, hence, very hard to infer from compiled code. On the other hand, quantum traffic in multi-core quantum communications comes from very well-known (and in principle controllable by design) sources, i.e. the quantum circuit being executed, the architecture topology we are using, etc. Moreover, communication primitives are explicit in the compiled code so that the traffic becomes not predictable, but rather known beforehand. Another difference is that due to the *no-cloning theorem*, it is hard to envisage the need for multicast communication at least resulting directly from the need to move the quantum state of qubits. Other than that, the metrics used in classical computing or the insight gained through analysis of its workloads, such as the iterative nature of communication, can still be useful in the quantum world.

6.2 A qubit traffic analysis software tool

Building a tool for analyzing traffic during the execution of quantum circuits in multi-core quantum architectures calls for firstly understanding where the traffic comes from and what are the main sources and stakeholders of traffic during the whole process.

In a generic multi-core quantum platform, qubits are constantly moving around. Indeed, whenever two distant qubits (even if they are on the same core) are to be operated by means of a two-qubit gate, they must be moved to adjacent positions. This implies that quantum circuits involve constant qubit traffic, both in and between cores (you can go back to Section 2.1 to review this idea). To showcase how this affects computation and

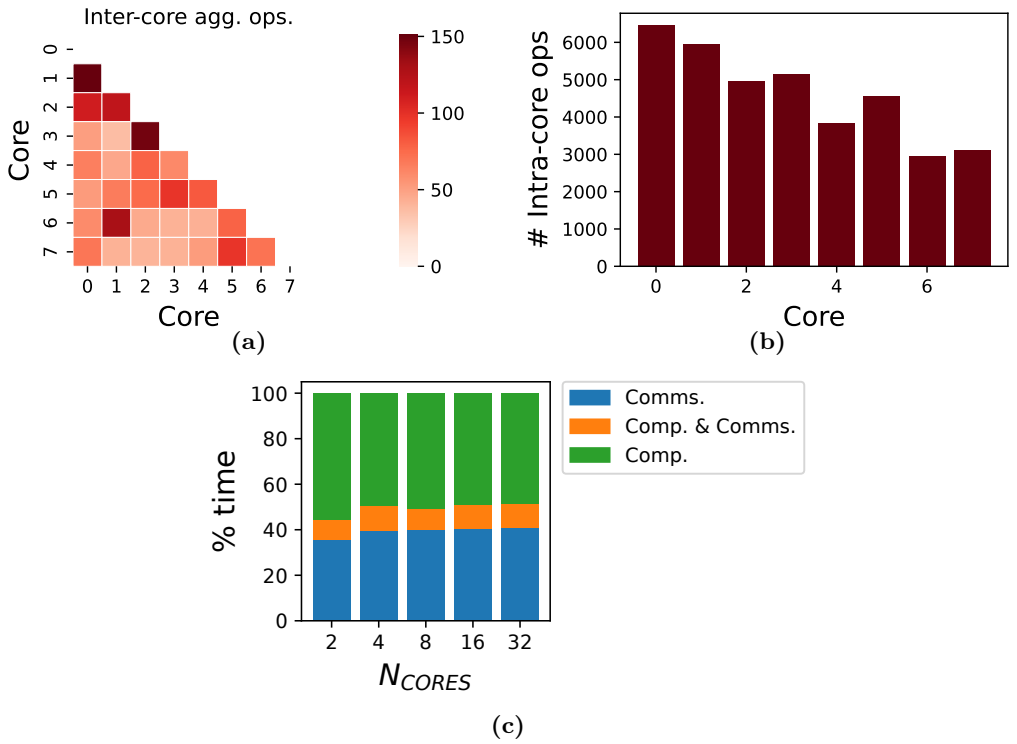


Figure 6.1: Qubit traffic in multi-core quantum architectures. a) Inter-core traffic, by pairs of communicating nodes, b) intra-core operations (qubit gates) per core, c) distribution of the execution time in computing and communicating in Cuccaro adder scaled in multi-core architectures. Observe the small amount of time in which computation and communications are allowed to be done in parallel.

communication distribution among cores, in Figs. 6.1a and 6.1b a simulated example based on a real quantum circuit is shown: specifically, the aggregated node-to-node and intra-node traffic of a sample execution of the Quantum Fourier Transform (QFT) circuit of 128 qubits on an 8-core platform with 16 qubits per core. Observe the high total count of teleportations among cores, and the existence of some hotspots, attracting most of the communication and computation (cores 0 and 1).

In addition, inter-core communication is slower than intra-core communication operations: latencies are from 5× to 100× longer [14, 51]. This, together with the generally high dependency between gates, leads to almost idle execution intervals following high-intensity ones, as well as low parallelism between computation and communication operations. In the same example as before, see in Fig. 6.1c the time distribution of computation (execution of qubit gates) and communication (teleportation operations) when scaling Cuccaro adder circuit in multi-core architectures. Most of the time, the dependencies present in the circuit make the processor idle while waiting for teleportations to end (in the example, only during about 10% of the execution are there simultaneous computation and communication operations).

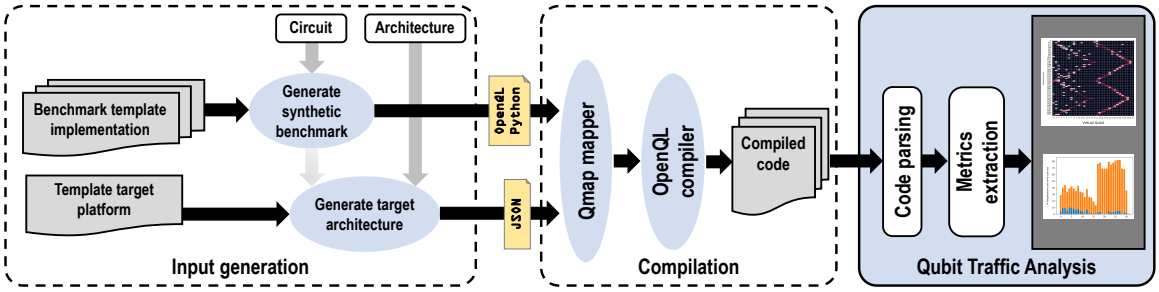


Figure 6.2: Flow diagram of the qubit traffic analysis tool

Therefore, knowing that all this communications overhead impacts the reliability of the computation, we would desire to minimize these movements and equalize the traffic. Let us quickly review the three main stakeholders involved in traffic generation and control:

- **The quantum circuit**. The number and time-wise distribution of two-qubit gates will impact on the qubit traffic during execution.
- **The processor's topology**. A scarcely connected processor leads to a higher communications overhead (in waiting times due to bottleneck links) when mapping two-qubit gates into the circuit. In particular, in a multi-core scenario, the lower the ratio of the number of qubits per core to the number of cores, the higher the need for costly inter-core qubit communications.
- **The compiler algorithm (mapper and scheduler)**. When compiling the quantum circuit into a physical platform, optimizations can be applied to allow for minimizing the traffic overhead.

Therefore, we have developed a software tool that, given a quantum circuit and a target many-core quantum platform, allows us to extract the qubit traffic by tracing all qubits along the execution and registering all the gates they participate in and the moves they are involved in.

The process, graphically explained in Fig. 6.2, consists on the following steps:

1. generation of the quantum circuit with the corresponding qubit input length,
2. compilation of the quantum circuit on the target platform, always having the same number of physical qubits as the qubits involved in the quantum circuit and the required number of cores, and
3. parsing of the resulting cQASM code in order to obtain the trace of each of the qubits. That information may be used for analysis purposes in studying traffic burstiness, hotspots, and other related metrics.

6.2.1 Extending the Qmap mapper for OpenQL

In order to evaluate the communication costs, we have mapped the previously presented benchmarks into different multi-core quantum architectures. To this purpose, we used the

OpenQL quantum programming framework [82] and the Qmap mapper [27] embedded in it. In this work, we have modified the Qmap mapper, which is meant for single-core resource-constrained quantum processors, and extended it to make it compatible with multi-core architectures following a proposal from Baker *et al.* [14]

The main modifications are the following:

1. Together with the definition of each core's description (gate latencies, qubit connectivity constraints, supported operations, etc.), the configuration file includes the topology specification of the multi-core architecture (how many cores, inter-core connectivity, and number of qubits/core) as well as the inter-core communication latency.

2. During the initial placement, a single core can be assigned multiple qubits.

3. An inter-core qubit transfer operation has been defined and is inserted during the routing process when necessary (i.e. inter-core operations).

The modularity of the OpenQL library, with the integrated QMap mapper, has allowed us to develop a many-core specific module, including the definition of the architecture, with the already named assumptions and restrictions, and the teleportation *gate*.

6.2.2 Looking at a quantum circuit in a different way

Seeing the execution from this perspective gives us some insights into the performance of both the algorithm itself and the mapper, as we may easily observe how strong dependencies that block execution are dealt with, the efficiency of the overall execution, idling periods, distribution of the job among the available cores and qubits, as well as analyzing the "life" of any qubit along the execution.

Let us analyze a single example with our tool to see how these things may be observed and analyzed. We will use a small example for better visualization: Grover's main routine for 16 qubits running in a 4-core platform (4 qubits per core). In Fig. 6.3 we present some graphical views of the execution to help us in the analysis. See in Fig. 6.3a the distribution of gate executions (in red) and teleportations (in white) versus the idling times (in black). The x-axis is the timeline of the execution (time goes from left to right) and the y-axis corresponds to the physical qubits in the system (ordered by core), i.e. the physical "placeholders" where the quantum states are stored. The amount of dependencies among operations involves a high execution inefficiency, as most of the time qubits are idle, waiting for a single operation (either a quantum gate or a teleportation) to finish. The qubits in the bottom-most core are clearly accumulating the result and do the most computation and communication. However, while looking at physical qubits is helpful for going after hot cores and qubits that are concentrating the most operations and communications, it is not as advantageous for analyzing how the qubit and gate dependencies are causing these inefficiencies.

Hence, let us now look at Fig. 6.3b, where the y-axis is now the virtual qubits (i.e. the quantum data entity): it is much easier to see the logical operation of the algorithm more clearly. Now we can observe the dependencies better and can differentiate virtual qubits that are expected to last with a coherent state for almost all the computation (e.g. qubit 9), whereas others are almost of no use (e.g. qubit 15). This shall be helpful for going

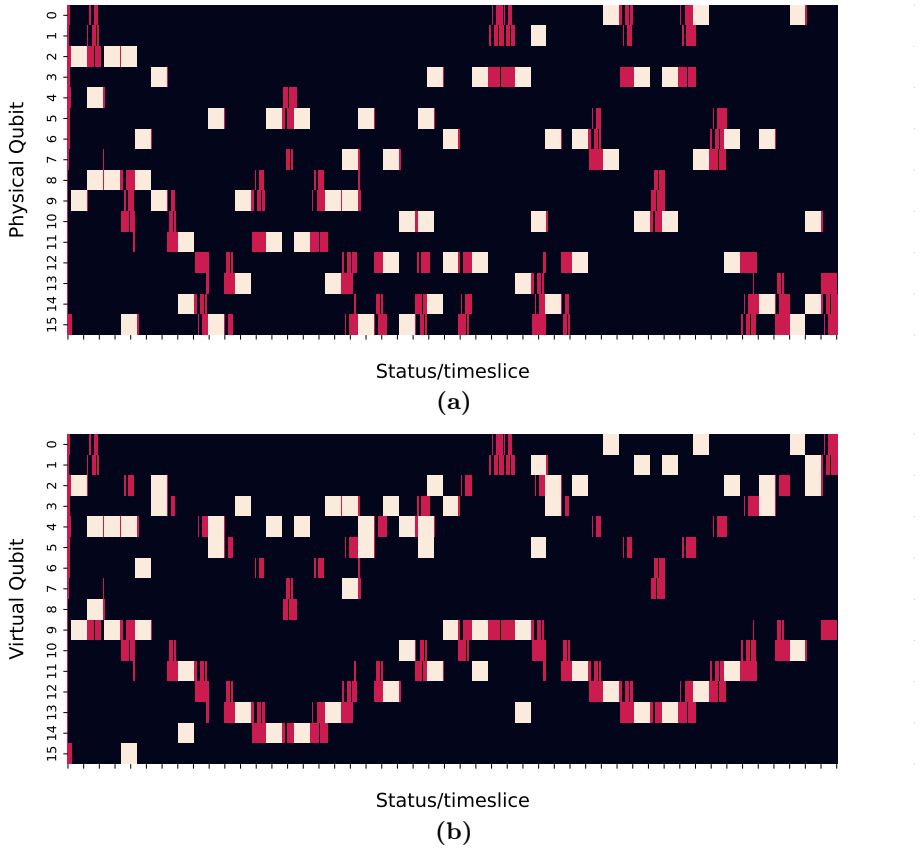


Figure 6.3: Execution trace of Grover's main routine for 16 qubits. a) on physical qubits and b) on virtual qubits. Computation, communication, and idling times are represented in red, white, and black colors respectively.

back to the original circuit and looking for disentangling core dependencies that hold the entire execution back, checking which physical topology could fit better the algorithm, or establishing an ideal target qubit coherence time that might withstand the requirements of the algorithm (or simply mapping longer-lived virtual qubits to the best behaving physical qubits in the platform).

These and other conclusions can be extracted using our tool, making it easier to give design guidelines for the algorithm, the architecture designer, and the compiler engineer.

6.3 Experimental results on quantum algorithm traffic analysis

In the previous section, we have focused on analyzing a single execution, seeing that several straightforward conclusions can be extracted. However, by aggregating these qualitative observations into numerical metrics and, in the future, using DSE techniques, we can obtain even more interesting insights.

6.3.1 Simulation set up and architectural space

In this section, we have restricted the exploration to a smaller set of examples, which can be enlarged for a wider analysis in future work.

In particular, we have used multi-core architectures with the following fixed characteristics:

- The cores are interconnected via a classical network for exchanging classical messages and measurements.
- All nodes are connected via optical channels to an EPR pair generator instrumental for qubit teleportation.
- The teleportation operation is assumed by the compiler as deterministically time-bound (set to 1000 ns, around 4 times more than a SWAP gate, in order to stress the system, as suggested in the results of our analysis in Section 7.1.5) , and always performed as a SWAP operation, i.e. the two qubits involved are swapped after the teleportation operation.
- The connectivity inside every core is full, i.e. any qubit can perform a 2-qubit gate with any other in the same core. This is done to "isolate" in the analysis the inter-core communication from the intra-core computation, which does not involve extra-SWAPs.
- The rest of the gate and qubit parameters are taken from recent superconducting qubit technology used in [130] and [228].

The exploration has been done by analyzing various algorithms (both real applications and random benchmarks, see below) and different platform configurations, varying the number of cores and number of qubits per core. In all cases, the circuits compiled in a given platform occupy all physical qubits available.

6.3.2 The selected algorithms

As benchmarks for assessment of communication overhead for multi-core architectures and their scalability, we opted for several algorithms that have the potential to show computational advantage when run on quantum in comparison to classical computers, such as QFT, Grover's search algorithm and Cuccaro Adder. These algorithms, however, have a specifically defined structure that makes them scale with the number of qubits in a steady, sometimes even linear way (e.g. Grover's), in terms of their parameters like the number of gates or two-qubit gate percentage. For that reason, we additionally used randomly generated algorithms as well as QV circuits [236], where we could have more influence on their parameters for any size of the circuit, and therefore probe our architecture in a worst-case scenario. The random algorithms we used were generated with uniformly chosen gates from a limited gate set with a uniform distribution of those gates among qubits. QV circuits are used in general for probing even single-core architectures, as they are the most complex version of a synthetic circuit with the highest two-qubit gate density (forces all qubits to be engaged in a two-qubit gate in each circuit layer).

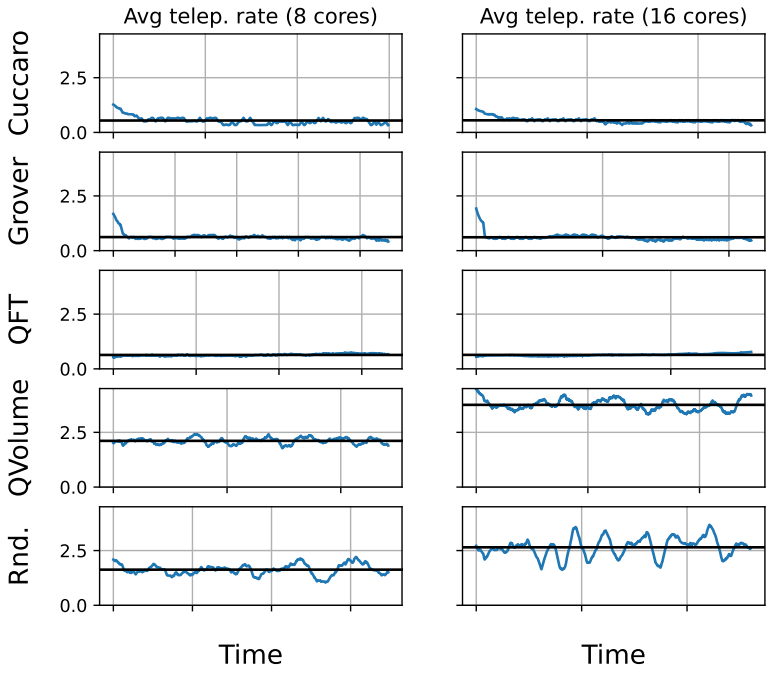


Figure 6.4: Average number of teleportations per timeslice in all benchmarks assuming 8 qubits per core and either 8 or 16 cores.

6.3.3 Space-time explorations

Following the literature on traffic analysis for multi-core scenarios (see Section 6.1.1), we have performed our exploration of the selected benchmarks in a three-phase fashion: first, studying the temporal distribution of the quantum data transfers; then, focusing on their spatial distribution, and finally, summarizing both analyses in a spatiotemporal joint exploration.

For the temporal traffic distribution, we have studied the inter-core communication trends for the different algorithms. In Fig. 6.4, the moving average of the number of teleportations per timeslice in every circuit, together with the overall mean, is plotted for all algorithms. Two different cases are studied (8 and 16 cores, both with 8 qubits per core). Observe that both Cuccaro and Grover suffer from a high inter-core data transfer burst at the start, which may easily stress the system and cause a bottleneck on loaded or poorly connected architectures. Both of them, together with QFT, have a quite low average number of concurrent teleportations (around 1), which is mostly related to the dependencies among operations in the code, forcing an almost linear, non-parallel, execution (as already observed in Fig. 6.1). Random and QV cases are good to stress the system, as they have more relaxed dependencies and allow for a higher teleportation rate. This communications requirements scale with the number of cores: this does not seem to be the case for Grover, Cuccaro, and QFT, which may facilitate scaling on large multi-core architectures.

For the spatial traffic analysis, i.e. how evenly is the overall traffic distributed among the cores, we have focused on whether the compiled circuit creates hotspots (cores attracting most communications). Hotspotness may be a natural consequence of most circuits, that

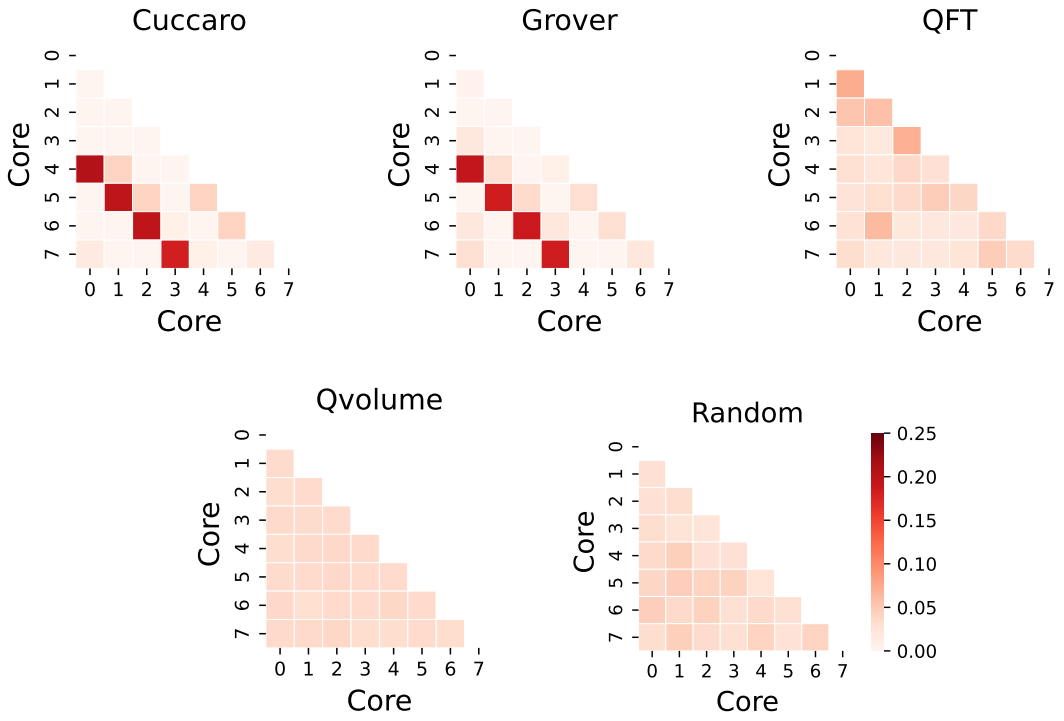


Figure 6.5: Inter-core traffic as the ratio of the number of teleportations for every pair of communicating nodes over total teleportations in all benchmarks assuming 8 cores with 16 qubits per core.

e.g. concentrate the result on a given variable, but it results in network congestion. In Fig. 6.5, the inter-core traffic is presented for all benchmarks, for the 8 cores, 16 qubits per core case. The random and QV cases are quite uniform, as expected, while that is also the case for QFT. Being a core part of some key quantum algorithms, avoiding network bottlenecks in QFT on multi-core architectures is relevant for their overall computational performance. Still, some minor hotspots (cores 0 and 1) can be detected, and probably further optimizations in the compiler could fix that. Grover and Cuccaro present a very similar behavior, which most probably has to do with the initial burst and flaws in the qubit mapping.

Finally, a joint spatio-temporal analysis is performed, using results as plotted in Fig. 6.6. A wider exploration is performed, for a core count ranging from 2 to 16 cores (8 qubits per core). We have used covariance (standard deviation σ over the mean) of both spatial and temporal traffic. For the spatial hotspotness we have used the number of teleportations per core over the whole execution, and for the temporal burstiness, we have used the number of teleportations per timeslice. Observe that there are two different regions: random and QV have low burstiness, while the rest are on the high burstiness end. Burstiness results in network inefficiencies due to unexpected bottlenecks and calls for overdimensioning the network capacity. See also that in general, the spatial hotspotness is specially high for Cuccaro, and that Grover scales quickly while QFT does it in a more controlled way.

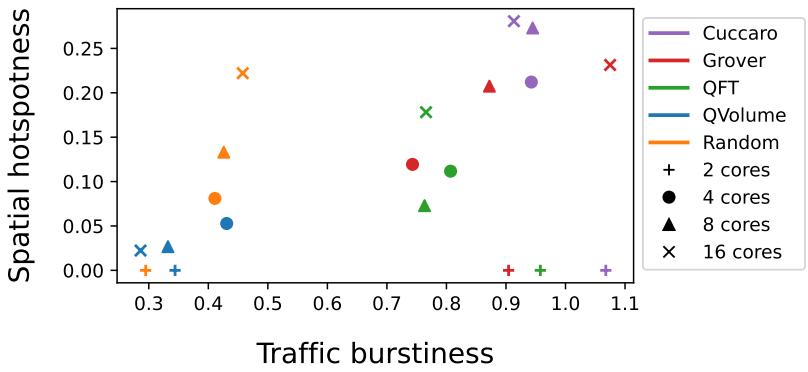


Figure 6.6: Summary of burstiness and hotspotness of all the evaluated benchmarks and core counts assuming 8 qubits per core.

6.3.3.1 Conclusions

In this exploration, we have substantiated the interest in qubit traffic analysis for efficient multi-core quantum architectures and presented a tool for carrying out this characterization. We have showcased how spatio-temporal analysis may help in quantum algorithms classification, and optimization of compilers for multi-core quantum architectures, as well as in highlighting the communications requirements for a given application and target architecture. For instance, we have been able to prove the good conditions of QFT's main routine for scalability, while Grover's search could be improved (either in compilation time or in the design of its initial phase).

6.4 In pursuit of online quantum computation improvement via communication protocols

In the first part of this chapter, we have been able to study and analyze the qubit traffic behavior of specific quantum circuits for a variety of multi-core quantum architectures. Detecting high hotspotness or burstiness provides us with a valuable source of information for improving multi-core compilers or exploring the most fitting architectures for certain algorithm structures. However, it is basically an *a posteriori* design feedback. This implies slower design improvement times, poor scalability, and adaptability to generic algorithms or architectures, as the problems detected might be very well dependent on the characteristics of the specific architecture or algorithm under study.

The goal of the rest of the chapter is to investigate whether it might be possible to implement a system that is able to detect and improve those communication inefficiencies *while the execution is ongoing* for an overall performance enhancement.

It might be difficult to precisely *equalize* temporal bursts or spatial hotspots, but since we know that any improvement of the communication latencies impacts directly quantum computational performance, the goal will be to design a communications control layer focused on enhancing transmission efficiency in a QoS fashion as a strong foundation to build on future iterations.

We have already shown how the main bottleneck in the multi-core architecture used for our previous studies (see Fig. 5.5) resides in the entanglement / EPR pair distribution. This distribution process can be seen as a channel reservation or as a shared medium access: whenever a pair of nodes want to communicate, they have to ask the EPR pair generator for an entangled pair that enables them to start a teleportation. This request implies a waiting time, that depends mainly on the efficiency of the EPR generation process and on the size of the waiting queue (i.e. the amount of other node pairs waiting for an EPR pair).

What if, instead of attending those requests in a FIFO fashion, we look at some properties and details of each of those requests in order to prioritize those operations involving qubits that might need faster communication due to, e.g. being a computational hotspot or having suffered higher amounts of decoherence due to longer uptime? In this way, we would place our QoS communications control system at the link layer, precisely at the entanglement distribution. In fact, we could properly call this a MAC protocol, in charge of distributing entanglement, i.e. letting the nodes in the network access the medium.

6.4.1 Existing quantum MAC policies

Quantum communications have only recently become practical, and hence most of the existing research is based upon lab experiments on quantum-aided classical secure communications [36, 217, 218], entanglement links on quantum chip [34, 38], or the foundations of the QI [5, 39, 41, 143, 148]. However, only a few studies exist that investigate control layers and protocols for improving quantum communications. In particular, two almost simultaneous recent papers developed a full quantum network stack with very similar approaches [42, 43]. Both also proposed several communication protocols.

In [42], the network stack they propose is divided into the physical layer (for the physical connection among nodes), connectivity layer (in charge of generating entanglement among nodes), link layer (for generating graph states of several nodes in the network), and network layer (for interconnection of several graph states). In this way, they base their architecture on multipartite entanglement for improved reliability. Apart from networking protocols for routing entanglement and some proposals for improving the reliability of the network quantum state in the presence of node failures, they introduce some auxiliary protocols for entanglement creation and distribution, but none of them include a QoS perspective, neither a mechanism for pure MAC operation. As highlighted in [43], these high-level protocols do not take into account the impact of classical control.

The work presented in [43] represents an outstanding effort for formalizing an overarching proposal attached to a practical technology, aiming at turning experimental work on entanglement generation and distribution into a well-defined service for the future development of the QI. Apart from a different approach to the full network stack, it includes two main protocols: MHP (Midpoint Heralding Protocol), for entangled EPR pairs creation at the physical layer, and QEGP (Quantum Entanglement Generation Protocol), placed at the link layer. Both protocols are interrelated in an elaborated scheme including a control layer, quantum memory management, fidelity estimation, and, very interestingly, a scheduler that may implement different scheduling strategies for prioritizing entanglement requests depending on its final application. We could consider this a first step towards a quantum MAC

protocol, though it is limited by the static definition of classes of traffic, with no *online* information extracted for a fine-grained selection of the priorities.

6.4.2 Design and implementation of a simulated fully-fledged multi-core communications network

In order to be able to develop and test our own communication protocols, we need a simulator that lets us tweak and design every aspect of the qubit transmission. For that, we have based our own multi-core quantum computer simulation tool upon the NetSquid simulator already used in Section 5.3.4 [70], by implementing on top of it all the needed components and control layers.

This work has a value of its own, as it will let us further explore these architectures while no experimental devices are available. In the following, we will detail the system arrangement and the global execution flow. In the next section, we will further enter into the details of the design choices and communications protocols we have implemented in the simulator.

6.4.2.1 The multi-core system framework

Following our architecture presented in the previous chapter (see Fig. 5.5), we assume a system composed of several identical quantum cores that are able to perform pair-wise qubit transfers among the nodes by using a shared EPR pair generation device. A layered diagram of our design for the multi-core architecture can be seen in Fig. 6.7. All nodes are connected to this EPR pair generator via direct photonic links. They are able to classically send messages to any other node via an all-to-all classical network interconnecting them.

Every quantum core has the same number of physical qubits. Two of these qubits have a specific communications role, while the rest are exclusively used to compute. We assume all-to-all connectivity within the quantum core, i.e. all qubits can inter-operate directly with any other qubit in the core. The two communication qubits are used in the teleportation qubit transfer as buffers for the EPR photons and the temporary qubits that are generated during the process.

The EPR pair generator is a network node composed of three main devices: the physical layer EPR factory, an input optical switch receiving the EPR requests, and an output optical switch for routing the generated EPR pairs. Following the suggestion in [43], this communication midpoint is also in charge of coordinating the communication control layer, as we will explain shortly.

All the elements in the system are modeled including realistic operation delays, classical and quantum noise in the optical links, qubits, and gate operations. These delay and noise models can be customized in order to represent different technologies and scenarios. At the present time, we have not yet included yield effects and variability on qubit fabrication, nor any type of qubit operation crosstalk.

6.4.2.2 Execution flow

The global execution flow of our simulator is shown in Fig. 6.8. Following the logic from NetSquid, each quantum core, as well as the EPR pair generator, is a network node having

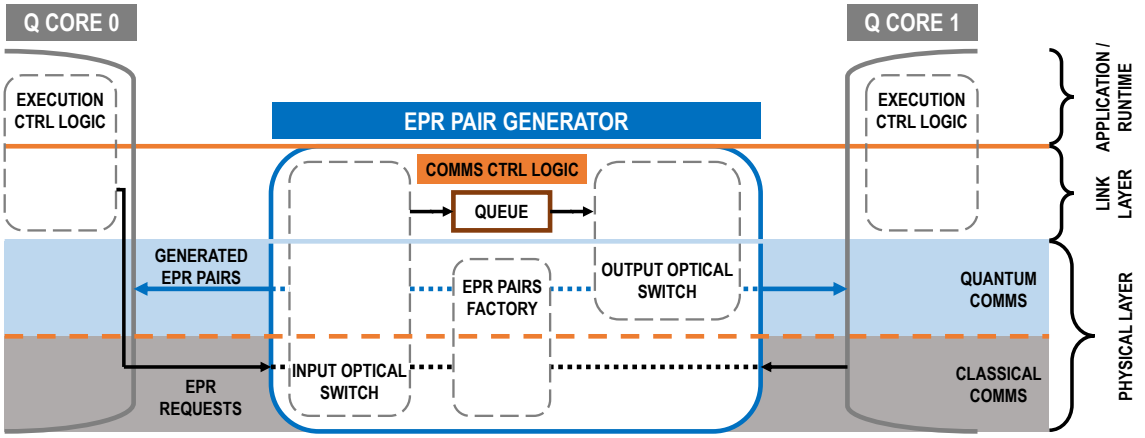


Figure 6.7: Layered diagram of the designed architecture for the multi-core computer simulator. Observe that each processor has its own control agent for the code execution. The communications control logic is placed at the EPR pair generator, where the requests queue stores the pending entanglements to be served. The physical layer (divided into classical and quantum) and the link layer form the multi-core network layer.

its own control agent. We assume that the whole multi-core architecture is single-threaded, i.e. it is fully devoted to running in isolation a single quantum circuit. Using our adapted version of OpenQL for multi-core architectures, the quantum circuit is first compiled. The output, which is a monolithic code with references to all nodes, is further parsed and divided into disjoint code chunks, each one exclusively containing the code referring to a single node while keeping the inter-core qubit transfer instructions in both affected nodes. The timing information from the compiler schedule is discarded, as within the simulator the synchronization between nodes and gate dependencies are kept dynamically via handshakes and the node control logic.

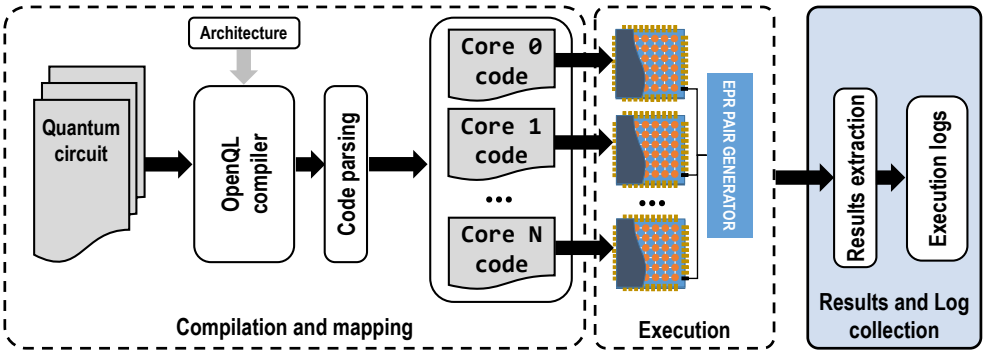


Figure 6.8: Execution flow of the multi-core computer simulator.

Each code chunk is assigned to a different node, and all of them start executing their instructions. The quantum processor logic is directly in charge of intra-core qubit initialization and other single- and two-qubit gates. Whenever teleportation is needed for qubit transfer, the protocol in 6.4.3.1 is executed. The measurements and other information are signaled to the control layer for execution logs and results.

6.4.3 A QoS-enabled communications control layer

In order to explore an *online* control mechanism for inter-core quantum communications, we need to know and monitor in detail every aspect of these qubit transmissions within our simulator. In the present section, we will describe the quantum teleportation implementation, the EoDGen (Entanglement on Demand Generation) protocol, and the implementation of a MAC protocol based upon QoS and qubit metrics.

6.4.3.1 The quantum teleportation implementation

As the main quantum communication primitive for inter-core qubit transfer in our simulator, hereafter we will explain some details of its implementation.

Contrary to the assumption taken in [172], where they consider entanglement a continuous resource, we have preferred to keep it as an on-demand system, attending to the currently achievable entangled photon generation rates. Moreover, as a way to guarantee efficient qubit memory management, we have assumed that every teleportation transfer implies a SWAP operation. That is, the two qubits involved are swapped, to avoid the need for a high amount of ancilla qubits in quantum circuits with unbalanced transmissions between nodes.

Whenever a node has to execute a qubit transfer, it starts the teleportation SWAP process (see also Fig. 6.9). Given that the qubit transfer can be characterized with the qubit pair $\langle q_0, q_1 \rangle$, where q_0 is the globally unique ID of the quantum state to be transferred from node A to node B, and q_1 is the globally unique ID of the quantum state which will be swapped with q_0, the teleportation protocol proceeds as follows:

1. **Cores synchronization handshake.** First, core A (the initiator node) sends a classical message of type `CONNECTION_REQUEST` to the core B, identifying itself as the initiator node and attaching the destination qubit ID q_1. When core B is ready for the transmission (it has finished the executions of all previous instructions up to the symmetric teleportation operation $\langle q_0, q_1 \rangle$), it responds to core A with another classical message, labeled `CONNECTION_OK`. This need for synchronization is solved in the proposal from [43] with a `min_temp` parameter for avoiding the early start of the entanglement distribution. Using a handshake is a more flexible and distributed way to achieve this. After the handshake protocol is successful, both cores are ready to proceed with the teleportation.

2. **Entanglement request.** The initiator node, once receives the `CONNECTION_OK` message, sends a `EPR_REQUEST` to the EPR generator in order to ask for an entangled photon pair. This control message has to contain, at least, the following fields:

    - `sender_ID` - the initiator node ID.

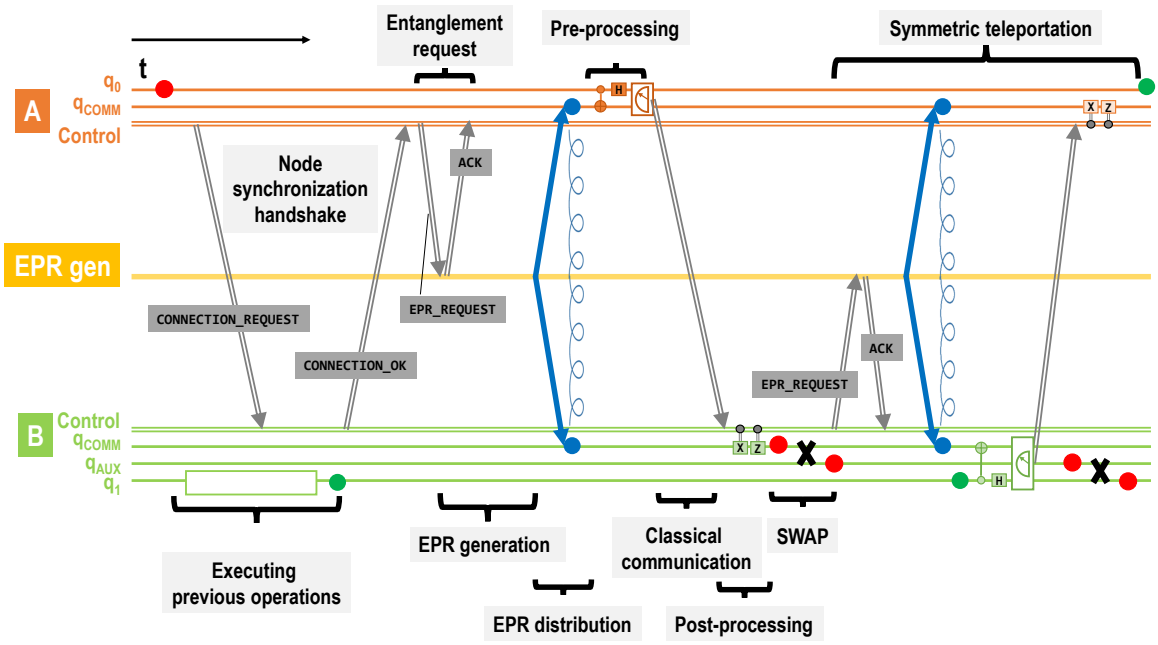


Figure 6.9: Time diagram of the implemented teleportation SWAP. The gray communication events are classical, while the blue ones are EPR photon pairs distribution (quantum). The EPR generation phase includes the time the request spends waiting in the queue, and might vary depending on the scheduling policy.

- `connect_to_ID` - the destination node ID.
- `timestamp` - time instant when the request was made.

This request is processed by the EPR generator logic, which confirms the request with an `ACK` message, and eventually (depending on the number of requests in the queue, the physical entangled photon rate, etc.) it sends the two photons of the entangled pair to cores A and B.

3. **Teleportation pre-processing.** As explained in the previous chapter, the teleportation follows by applying, at the initiator node (core A), a CNOT gate between the communication qubit q_{COMM} and q_0, and a Hadamard gate on q_0. After that, both qubits are measured, and the classical result is sent to core B.

4. **Teleportation post-processing.** As soon as the core B receives the message with the pair of classical values from core A, it conditionally applies an X gate and a Z gate on the communication qubit q_{COMM} at this side, completing thus the qubit transfer.

5. **Qubit swapping.** To balance the communication operations and facilitate memory management, the communication qubit q_{COMM} (now containing q_0) is swapped with the second ancilla communication qubit q_{AUX}, finishing thus the teleportation and preparing the symmetric operation to complete the teleportation SWAP.

6. **Symmetric teleportation.** Now it is time for core B to transfer the quantum state q_1 to core A (into the qubit previously containing q_0). The process is the same as in steps 2–4. As soon as q_1 is ready, q_{AUX} state is moved into it to finalize the process.

6.4.3.2 The Entanglement on Demand Generation protocol

Having a single EPR generator involves needing to be very careful with EPR requests management, as this node becomes the main bottleneck. Although this assumption is realistic to this day, it is a hard constraint that will need to be confronted with more flexible alternatives. However, it also poses an interesting communication problem with shared resources, time constraints, and a noisy environment. In the following, we describe the protocol we have designed for handling this issue in a flexible and efficient way.

As soon as they are received, all EPR requests are stored in the global requests queue, and the `ACK` message is sent to the initiator node. If the queue is found to be empty, the EPR generation control logic triggers the physical layer EPR factory in order to start the generation. We have chosen this behavior instead of continuously triggering the factory to facilitate cooling times and thus improve generated entanglement. It is also different from what is proposed in [43]: their midpoint heralding protocol is in charge of polling the link layer to know whether there is a need for EPR pairs, but this is mainly due to the different approach in dividing the protocol among the different layers.

Whenever an EPR pair is generated, the EPR generation control logic asks the internal scheduler which of the EPR requests in the queue has to be served first, depending on the implemented policy. This is also different from the QEGP proposal from [43], though we follow their idea of designing the communication midpoint as the master of the requests queue, centralizing also the scheduling mechanism. As soon as the request has been served to both involved nodes, the request is popped out of the queue.

If the queue remains non-empty, then the EPR generation control logic triggers the physical layer EPR factory in order to start the generation again.

6.4.3.3 Managing EPR requests queue for improving QoS: the *LifeEstimate* policy

Our modular design lets us control the qubit traffic in a flexible way, as we have a scheduler right at the key point of any inter-core communication: the EPR pair distribution.

By simply designing the scheduler policy, we are in fact enabling a MAC protocol that controls traffic prioritization. For instance, a FIFO policy (i.e. no prioritization) could be used as the base case. Nevertheless, we could instead use some parameters from every request for reordering the queue.

To do so, we can extend the `EPR_REQUEST` fields in order to collect some context information on the qubits, the nodes, or the circuit itself.

A first approach, following the work in [43], would be to add a traffic `class` field, and therefore favor any specific type of traffic when selecting the EPR request to be served, by simply assigning a higher priority (using a scalar) to the class(es) to be prioritized.

However, our approach enables us to add dynamic information to the equation: what if we add data about the number of operations performed, the number of operations still to go, the lifetime... of the affected qubits?

Reordering a queue implies lowering the expected waiting time for those elements being favored: in our scenario, time is equivalent to added decoherence, hence we could try to prioritize those requests involving qubits that have suffered more from decoherence. This can be estimated by looking at the sources of decoherence: *i)* the number of quantum operations applied to that qubit and *ii)* the time already passed since the qubit was initialized. Then, we need to track that information to keep it up to date, as well as add those items to the request message.

To do so, we have modified the system control layer to keep, for every qubit, three items of information, and have included them in the `EPR_REQUEST` fields:

- The number of operations already performed on each qubit, attached to the request as `ops_done_q0` and `ops_done_q1`.
- The number of operations still pending for each qubit, attached to the request as `ops_to_go_q0` and `ops_to_go_q1`.
- The time instant when the qubit was initialized, attached to the request as `init_timestamp_q0` and `init_timestamp_q1`.

Using this information, we have designed a scheduling policy for our QoS-centered quantum MAC protocol, which we have called "LifeEstimate". It prioritizes the pair of qubits that are expected to have the largest overall lifetimes. The estimation of their lifetimes $\widehat{\theta}_T$ is obtained as follows:

$$\widehat{\theta}_T = \sum_{q_0,q_1} \widehat{\theta}^*_T + (T_{NOW} - T_{init}) \tag{6.4.1}$$

where T_{NOW} is the current timestamp, T_{init} is the time instant when the qubit was initialized, and $\widehat{\theta}^*_T$ is the estimated time left, which is computed from the number of operations to go ($N_{OPS}{}^*$) and the estimate for the average time per operation ($\overline{T}_{OP}$):

$$\widehat{\theta}^*_T = N_{OPS}{}^* \cdot \overline{T}_{OP} \tag{6.4.2}$$

being $\overline{T}_{OP}$ the result of dividing the number of operations already performed on that qubit by its lifetime up to this moment.

It is now time to test this queue-prioritization MAC protocol proposal on our simulator to check whether it actually is able to improve the overall performance.

6.4.4 Experimental results: facing the trouble with the dependencies dead-lock

In order to test the effect of applying qubit traffic *online* control through a prioritization policy for MAC, we have thoroughly run tests on various architecture configurations on top of our multi-core quantum computer simulation.

In particular, we have chosen to use randomly generated algorithms with some modifiable parameters that could let us force the system into extreme scenarios. As we have observed in Section 6.3.3, random algorithms provide us with dense traffic while having low

hotspotness and burstiness (see Fig. 6.6). This favors our testing by focusing the communications issues in a constant high throughput among all pairs of nodes, hence in the bottleneck at the EPR pair generation.

To allow for high variability in the random algorithm within the circuit test set (i.e. avoiding the effect of the law of the large numbers), each random circuit sample is defined by four sets of random variables:

- **Init time.** Each qubit in the circuit has a random init timestamp, to allow for varying qubit lifetimes within the same circuit.
- **Idle state probability.** Each qubit in the circuit will be in an idle state (no operation) at a given circuit cycle with probability `p_idle`.
- **Two-qubit gate probability.** Each qubit in the circuit will be executing a two-qubit gate with another qubit at a given circuit cycle with probability `p_2qubitgates`. The two-qubit gate probability is the same for all qubits in the same core.
- **Inter-core gate probability.** Each qubit in the circuit will be executing a two-qubit gate with a qubit *from another core* (i.e. executing a teleportation SWAP) at a given circuit cycle with probability `p_2qubitgates * p_intercore`. The inter-core gate probability is the same for all qubits in the same core.

Following the execution flow as explained previously, we partition this random circuit (which makes use of all available qubits in the multi-core quantum processor) among all cores, and let it execute completely. For the comparison among different traffic prioritization policies of the MAC protocol, we run the exact same random circuit samples with each of the policies on the test.

In order to be able to measure the performance of the computation, we measure all qubits at the end of their respective lifetimes and collect all values. In order to be able to check the fidelity of this result, we implement another trick: every single-qubit gate and two-qubit gate does not affect the quantum state, as they are implemented as logical identity gates. However, the decoherence and gate errors are indeed applied. Therefore, we can test every qubit fidelity by comparing the measured value with the initial value. We then define the overall circuit fidelity as the arithmetic mean of the fidelities of all qubits.

After setting up these experiments, we have run a wide set of them, for architectures with up to 16 cores with 1024 qubits per core, and also some extreme cases of up to 1024 single-qubit cores. We have explored also different decoherence and gate error parameters, ranging from current experimental values (e.g. values from Sycamore quantum processor) to some orders of magnitude lower figures. The EPR pair technological parameters have also been explored, varying its final entangled photon rates. The explored variable space is summarized in Table 6.1.

Despite the fact that this exploration is wide enough, and after reviewing and repeating the simulations, checking again the system, and looking for any point of failure of the system, we have found that no improvement is made in the system performance when using a prioritization policy as a MAC protocol for the inter-core quantum communications. See, for instance, Fig. 6.10, where a representative sample of the actually explored space is shown. In particular, two different scenarios of varying EPR generation rates are shown, each of

Table 6.1: Notation, symbol definitions and values used in simulations for the present section, with reference to values from [81, 143, 227, 237]

Notation	**Meaning**	**Value (range)**
N_{CORES}	Number of cores/chips	1 - 1024
N_Q^{CORE}	Number of qubits per chip	1 - 1024
N_Q	Total number of qubits in the quantum computer	$N_{CORES} \cdot N_Q^{CORE}$
L	Length of the quantum circuit in execution cycles	100
T_{init}	Init time per qubit	$\sim U(0, L)$
`p_idle`	Idle state probability	$\sim U(0, 1)$
`p_2qubitgates`	Two-qubit gate probability	$\sim U(0, 1)$
`p_intercore`	Inter-core gate probability	$\sim U(0, 1)$
R_{EPR}	EPR pair generation rate	$10^2 - 10^8$ Hz
e_1	single-qubit gate error probability	0.0 - 0.015 [81]
e_2	two-qubit gate error probability	0.0 - 0.036 [81]
e_r	measurement/readout error probability	0.0 - 0.031 [81]
T_1	Amplitude damping for the memory noise model	$2 \cdot 10^3 - 2 \cdot 10^{18}$ [237]
T_2	Phase damping for the memory noise model	$10^3 - 10^{18}$ [237]

them on an increasing number of cores, each with 128 qubits per core. Each plot represents the change in overall fidelity when increasing the qubit coherence time (T_2), comparing FIFO and LifeEstimate policies.

With the naked eye, these bar plots show no difference between both, but even when looking at the difference (LifeEstimate - FIFO) on a greater scale (green bars and green axis), that intuition is confirmed: with our current parameters and on the whole design space explored, no improvement can be said to exist when applying prioritization to teleportation operations involving qubits with longer expected lifetimes. It is to be said that we have explored variations of the LifeEstimate policy, but the same results were extracted.

In order to figure out what the matter was, we analyzed in detail the simulation traces, coming out with a compelling finding: if we look at the comparison between the total execution time and the total number of operations (quantum gates) when operating a circuit under FIFO policy and LifeEstimate policy, we observe that even our proposed policy is effectively reordering many times the queue, the average amount of time spent in the quantum operations stays the same, and consequently the average qubit lifetime remains unaffected. That is, the qubit that has been favored by the MAC protocol has to spend that saved waiting time later on. We can observe that from the global circuit perspective in Fig. 6.11: the average time per operation (including teleportation transfers) remains the same when applying LifeEstimate. The same conclusion can be extracted when looking at the total circuit execution time.

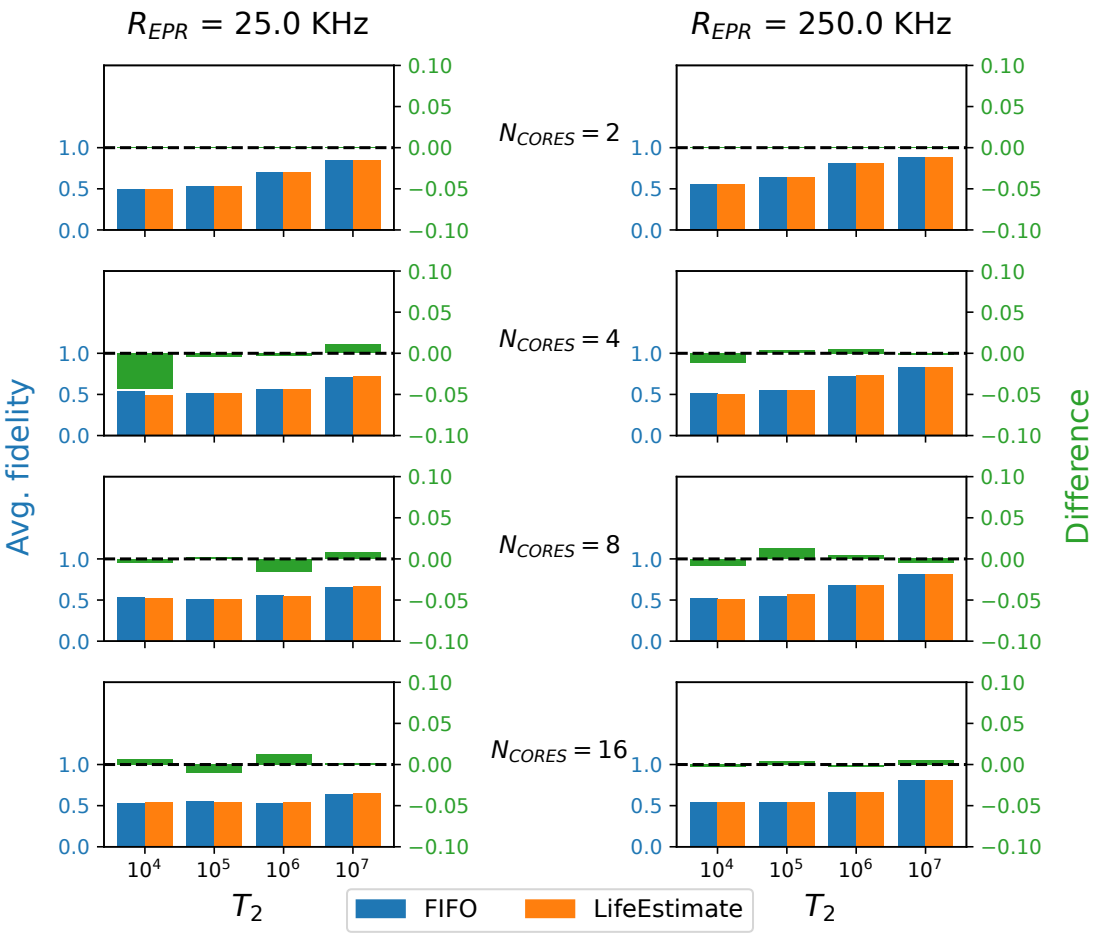
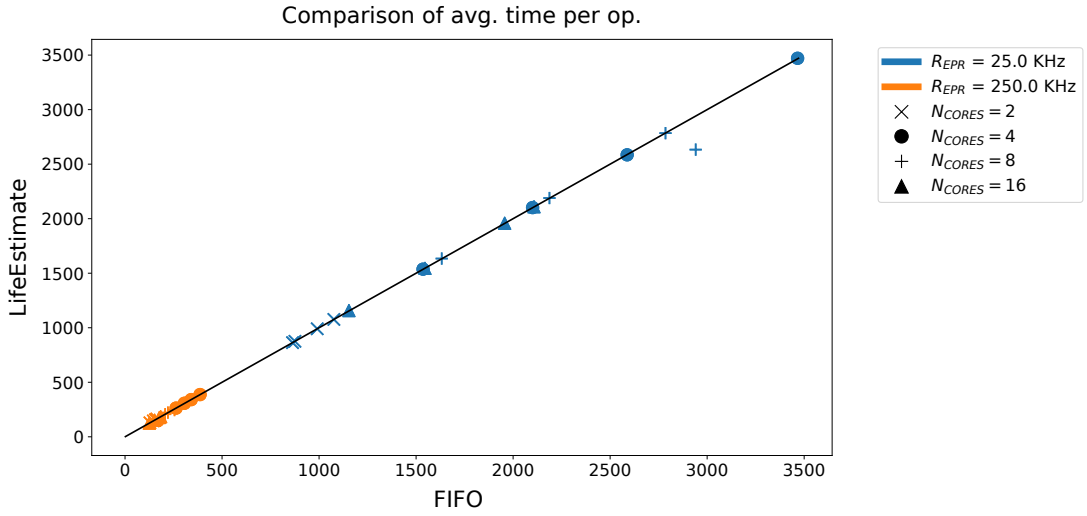


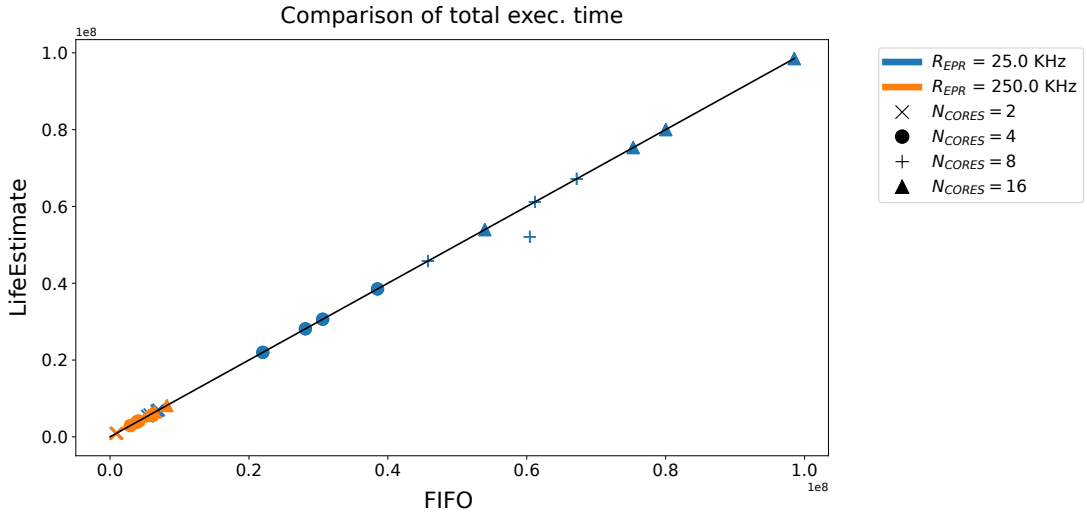
Figure 6.10: Comparing LifeEstimate and FIFO policies for queue prioritization on the proposed MAC protocol. All the scenarios show no visible trends nor differences in overall fidelity between both policies. The green bars and axis represent the difference (overall fidelity using LifeEstimate minus overall fidelity using FIFO) at a larger scale.

Where is this persistent blocking coming from? The qubit waiting times do not come only from pending EPR requests: qubits are constantly waiting for other qubits. Remember that the quantum circuit, like any orderly set of instructions, usually has some dependencies among instructions that prevent some of them from being executed before others have already been successfully done. Our random quantum circuits, and more specifically its dependencies, behave as a tight crew from which no component can be separated. Even if at a certain moment we could advance the execution of an operation (in our case, a teleportation), that will only imply swapping it with idle time that the involved qubits had to spend anyway some cycles later. Eventually, all qubits will wait for the last operation to finish: no qubit left behind.

In fact, applying priorities in a queue only improves the overall performance in several cases:



(a)



(b)

Figure 6.11: Performance comparison between LifeEstimate and FIFO policies in various scenarios. The black diagonal line in both plots is the identity line. a) Average time per quantum operation (either single- or two-qubit gate), b) Total execution time.

- When working with clearly differentiated classes of packets/clients, we can indeed locally provide better service to a certain class (at a cost for the lower priority classes), as in Internet traffic QoS.
- when working with varying (and long) serving times (as in operating systems task scheduling), where we interfere with the service to avoid starvation by implementing some sort of fair share policy (time division, Round Robin, etc.)
- we have deadlines to meet, and therefore we can establish priorities based on them (also used in task scheduling).

However, we are facing a scenario where we cannot establish different client classes (all teleportations are equally important), the expected service times are the same for all of them (the service is always a teleportation SWAP), and it is not evident to implement deadlines, as we are in fact working always in an ASAP scenario (except for the init operations, which we prefer to delay as much as possible).

Therefore, the proposed approach is not able to deal with dependencies. But all this apparently useless experiment opens up the door for an enticing research line: the easiest way to "break" dependencies in any computer is to allow for multi-programming. In the first part of this chapter, we have looked at some circuits with their qubits idling almost half of the time: we believe that "filling" those unused cycles with other quantum circuits running in parallel will power up the scalability of quantum computers. In the case of multi-core quantum processors, a MAC protocol at the entanglement distribution link layer as the one we have presented will enhance the efficiency and overall performance of such a multi-threaded system.

Chapter 7

Scaling of Multi-Core Quantum Architectures: A Simulation-based Structured Analysis

We have already presented in Chapter 4 DSE, a powerful tool for exploring the design space of multi-core quantum computers approach and its potential as a scalability-enabler for QC. However, the analysis carried out in that chapter is limited to a behavioral analytical model without realistic qubit error models nor any detail on specific quantum gates or quantum communications technology performance. Nevertheless, it has served us to learn how such exploration technique might help us in testing QC scalability as well as providing design parameters and guidelines more fitting to each architecture and computing scenario.

In Chapters 5 and 6, we have been able to dive into each of the layers of the QC stack, getting into the details of the inter-core communications, with enticing findings and conclusions on its weaknesses and strengths. With this knowledge in the backpack, in this chapter we aim at answering this thesis' *big questions*, but now with the aid of fully-detailed explorations based on our developed multi-core compiler and QC simulator. These explorations, which take into account actual quantum circuits, runtime issues, and the advantages and disadvantages of a variety of architecture configurations, as well as quantum noise models, help us to provide experimentalists with design guidelines enabling them to build scalable multi-core QC.

We have structured this chapter as follows. First, we have presented a detailed DSE-based study on determining, for any given multi-core configuration, the threshold value for the inter-core quantum communications latency, which, as we have observed in previous chapters, is a fundamental piece for the success of multi-core quantum architectures. Without the need for a costly simulation, our analysis is carried out by comparing information from compiled code for single- and multi-core quantum architectures. This study provides a focused metric for benchmarking this development, which is still in its initial stage, helping to optimize the technology push in the most efficient direction.

On the other hand, thanks to our development of a simulator specific to multi-core quantum architectures (go back to Section 6.4.2 to revisit the details), in the second part of this chapter we replicate the scalability analysis performed in the last part of Chapter

4. This time, the analysis is run by means of a fully-fledged simulator, covering the full QC stack, from the qubit to the application layer. This includes accurate quantum noise models, actual quantum gates, quantum circuit execution, etc. In this way, we close the *loop* of the thesis by highlighting the promising scalability design regions for multi-core quantum architectures.

7.1 A communications-aware code-based approach

Developing efficient quantum interconnects for low-error high-rate qubit transfers is key to enabling scalability of the promising multi-core quantum architectures. This has been sufficiently remarked on by the conclusions of the two previous chapters. Avoiding bottlenecks in quantum inter-core communications implies studying the trade-off between their implicit overhead and the gain in the size of the algorithms that can be executed on them. Profiling this design junction will help not only to determine the viability of multi-core computing but also to characterize the *decision threshold* where the communications cost of distributing quantum computation among several cores pays off.

Studies on this trade-off are available [14, 66–69], although the different approaches are centered in different specific goals: purely analytical study [66, 69] focused in compiler optimizations [14] or not comparing the many-core exploration with the baseline single-core performance [67, 68]. In particular, none of them allows us to answer any of these key questions: how fast should inter-core communications be in order to allow many-core architectures to supersede traditional single-core quantum processors? For a given interconnect technology, which is the *minimum* architectural configuration (number of cores, number of qubits per core...) that we need in order to pay off the communication costs?

This is the aim of the present section. We attempt to do so by performing a DSE of the architectural and technological variables that configure multi-core quantum computers specifically affecting this trade-off. We have used the OpenQL [82] compiler and QMap mapper [27] for several random and real application benchmarks to compute their actual computing performance for different multi-core configurations, comparing it with the traditional single-core results.

Let us now dive into some key items for the analysis: architectural assumptions and performance metrics, chosen benchmarks, compilation framework, and problem formulation. See in Fig. 7.1 a flow diagram of the design exploration process.

7.1.1 Modeling Assumptions

Multi-core quantum topology: We assume the quantum computer to be composed of one or several interconnected cores, which could be compared to any current NISQ quantum computer; that is, a computing core consisting of some tens or hundreds of qubits. In order to keep the focus on the effect of inter-core communication, we have assumed all-to-all intra-core connectivity, i.e. every physical qubit can interact with any other as long as they are placed in the same core. As in [51] and [31], full connectivity is also assumed among the cores, meaning that qubits placed in different cores are at "one hop" distance, as there is an entangled pairs generator which is shared among all cores.

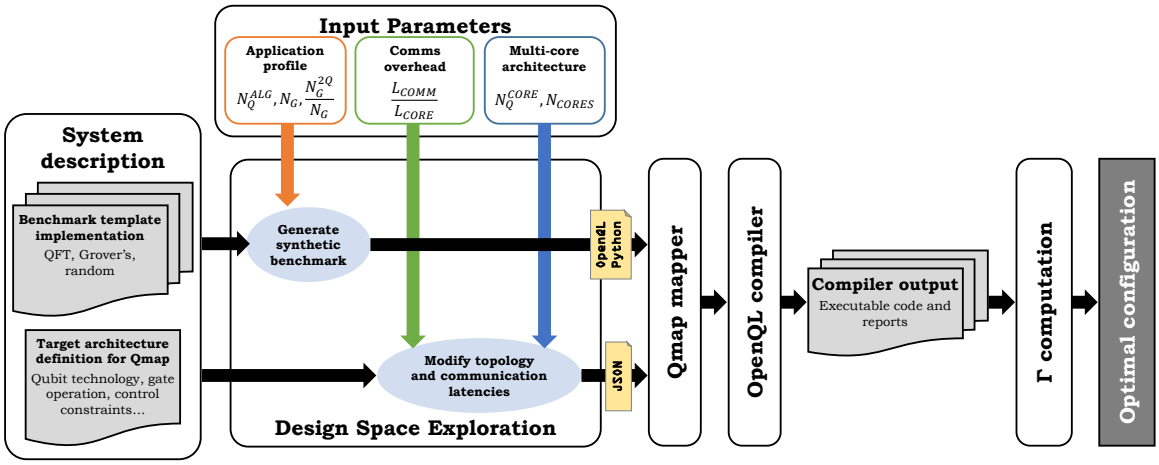


Figure 7.1: Flow diagram of the evaluation framework used for the DSE

Inter-core communication: As in previous chapters, we have chosen quantum teleportation as the inter-core communication primitive. It is a firm candidate for this role and has been already demonstrated with several technologies [142, 146, 148, 238]. Quantum teleportation is based on the properties of entangled pairs of qubits, that are shared among two entities and enable them to communicate a quantum state, through a few simple operations and some classical communication, without the need to send the physical qubit, hence increasing the success rate. You will find more details on quantum teleportation and entanglement distribution in Section 5.3.

7.1.2 Communication overhead sources in multi-core quantum architectures

The optimization task on quantum mappers (already explained in Section 3.4), which aims at translating the hardware-agnostic quantum circuits to executable programs, has to deal mainly with qubit-to-qubit connectivity constraints [29] through its main three steps (i.e. initial placement, qubit routing and scheduling of quantum operations).

These mapping techniques can be applied and extended when scaling the architecture to multiple cores as the connectivity among cores is also limited. In this case, virtual qubits can be mapped to a physical qubit in any core. Ideally, qubits with a high degree of interaction should be placed in the same core. When two qubits that are in different cores need to interact, they will have to be moved to the same core. Therefore, multi-core architectures require not only intra-core qubit movement operations (e.g. SWAP gates) but also inter-core communication operations such as quantum teleportation or qubit shuttling.

Depending on the interconnect topology of the multi-core architecture, the restrictions on inter-core communications will vary. As in the single-core case, this limitation will result in an increase in quantum operations. In addition, inter-core communication is slower and more error-prone than intra-core communication operations: latencies are from 5× to 100× longer, and the error rates are on average 10× to 100× worse for quantum teleportation than for two-qubit gates [14, 51, 57]. All of these overhead sources need to be analyzed as they will substantially affect the algorithm execution and reliability of the final results.

7.1.3 Selected Benchmarks

With the aim of evaluating the communications overhead in multi-core quantum architectures and its potential for solving the scalability issue, we have used the QFT and Grover's search algorithm as benchmarks [14, 16]. These benchmarks, however, have a specific structure and scale in a predefined steady manner, which limits our control over their parameters. For instance, the number of gates for Grover's algorithm scales in a linear manner with the number of qubits, whereas its percentage of two-qubit gates (after decomposition) is around 30%.

In order to overcome this limitation, additionally we decided to use synthetically generated benchmarks in the form of random circuits. These can provide the necessary freedom to explore algorithm parameters like the number of qubits, gates, and the percentage of two-qubit gates mentioned previously. The importance of using the family of random circuit benchmarks was pointed out by some previous works as well [14, 207–209, 239–241], which also introduced various ways of generating them. They differ in type of randomness, shape (width vs. depth), gate density per layer, etc. In our case, we opted for random circuits that are generated by uniformly selecting gates from a predefined set and uniformly selecting qubits (one or two qubits for single- or two-qubit gates, respectively) to apply those gates on. Beforehand, we had to decide on the percentage of two-qubit gates, as this parameter is the one that defines the amount of qubit interactions. We chose the two extreme values of 20% and 80% to showcase the impact of communication.

For the compilation process, as in the previous chapter, we have used OpenQL [82] and a modified version of the Qmap mapper [27] embedded in it, extended to make it compatible with multi-core architectures following a proposal from Baker *et al.* [14].

7.1.4 Problem Formulation

We are considering multi-core architectures as an approach that may help to unleash the QC potential, which is right now limited by the number of qubits of current processors. In the present study, we are focusing on the trade-off between inter-core communications overhead and the implicit gain in computational power (number of integrated qubits) in this approach. For that, we need to formally define the metrics of interest for the DSE as the first step for stating the DSE problem. You can revisit Section 4.3 for a detailed explanation of how to correctly define a DSE analysis.

In this case, the problem requiring the exploration (i.e. inter-core communications cost VS computational capabilities trade-off in multi-core quantum computers) facilitates the selection of the adequate metrics:

- **Quantum circuit size (computational capabilities):** as we are working with actual quantum circuits compiled with multi-core architectures, instead of just comparing the number of qubits integrated into the single-core and multi-core architectures, we can directly make use of the size of the algorithm being compiled, which is related not only to the number of qubits it requires but also to the number of quantum operations involved.
- **Execution latency overhead (inter-core communications cost):** in order to do a technology-agnostic analysis with higher validity, we have modeled the communications

performance through the definition of the upper-bound delay of the inter-core qubit transfer. This affects directly the instructions scheduling process of the compiler, causing the corresponding overall execution overhead that we want to examine.

Of course, considering only one of these two metrics would not be fair for the comparison: in terms of latency, the single-core processor will always outperform the many-cores approach, and, conversely, considering only available computing space will favor the increase in number of cores due to the current limitation in integrating many qubits in a single core.

This metric selection helps us to focus the analysis on determining the *decision threshold* between monolithic single-core and multi-core quantum architectures, for which we do not need to consider other elements that might produce similar effects in both approaches (single- and multi-core). That is, no decoherence and other quantum noise models are considered: the latency overhead is the main hurdle of inter-core communications when compared to intra-core operations and deserves an isolated study. This does not imply that they should not be included in larger analyses. In fact, we have added the effects of quantum decoherence and gate errors from accurate models in the scalability analysis we present in the second part of this chapter.

In any case, it is key to note that in quantum communications latency and error rate are tightly coupled: the more time is consumed in the data transfer, the higher will be the decoherence effect on the resulting error (assuming no error correction or entanglement distillation techniques). Also, correcting errors and dealing with them incurs higher execution latency. Accordingly, the present study, although focused on latency overhead, gives guidelines that are applicable as well for real-world error-prone systems.

Our analysis, with reference to the selected metrics, depends on three different aspects of the problem: the architecture configuration, the application-specific parameters, and the technology specifications. According to our modeling assumptions, we can narrow down these dependencies to five variables, grouped into these three aspects:

- **Architecture**. Number of qubits per core (N_Q^{CORE}) and number of cores (N_{CORES}).
- **Application**. The total number of qubits (N_Q^{ALG})[1] and gates (N_G) required by the algorithm and 2-qubit gates fraction (N_G^{2Q}/N_G), as these are the ones that might need for inter-core teleportations when the involved qubits are in different cores.
- **Communications**. The ratio between the inter-core communication latency and the cost of an intra-chip communication operation (L_{COMM}/L_{CORE}).

Now we have collected all the elements to start the definition of the FoM, which, to avoid collisions with the one defined in Chapter 4, will be called Γ'. Based on the variables under study, the definition of both metrics follows: *i)* Quantum circuit size (computational capabilities) and *ii)* Execution latency overhead (inter-core communications cost):

Quantum circuit size: the product of its number of qubits and its number of gates.

$$J_{QSIZE} = N_G \cdot N_Q^{ALG} \tag{7.1.1}$$

[1]Observe that N_Q^{ALG} might be different from the available total number of qubits N_Q

Execution latency overhead: the ratio between the execution latency [2] of a given circuit on a multi-core architecture ($L_{multi-core}$) and the execution latency when that same circuit is run on an equivalent (same number of total qubits) idealistic single-core counterpart ($L_{single-core}$). This execution latency accounts for the total number of processor cycles the code takes, after performing the platform-specific compilation (i.e., mapping and instructions scheduling).

$$J_{COMMS} = \frac{L_{multi-core}}{L_{single-core}} \tag{7.1.2}$$

Taking into account the simplicity of the current exploration in terms of performance metrics, we can simply use as the multiplicative aggregation function f a ratio among the two metrics, without the need for normalizing any of them: in this case, the trade-off between two metrics facilitates the observation of trends without the need of further mathematical formulation. Hence, the proposed FoM for this exploration has been defined as:

$$\Gamma' = \frac{J_{QSIZE}}{J_{COMMS}} = \boldsymbol{f}\left(N_Q^{CORE}, N_{CORES}, N_Q^{ALG}, N_G, N_G^{2Q}, \frac{L_{COMM}}{L_{CORE}}\right) \tag{7.1.3}$$

This exploration let us perform further optimization, as we are looking in the design space for the points where multi-core architectures perform better than single-core processors. That is to say, to find the maximum performance Γ' for any given architecture configuration (number of qubits per core N_Q^{CORE} and the number of cores N_{CORES}), for every application (i.e. number of gates N_G, 2-qubit gates fraction N_G^{2Q}/N_G and number of qubits required by the algorithm N_Q^{ALG}) and for fixed communication overhead (L_{COMM}/L_{CORE}). This can be summarized in the formulation below:

$$\begin{aligned} \min \quad & \frac{L_{COMM}}{L_{CORE}} \\ \text{s.t.} \quad & \Gamma'\left(N_Q^{CORE}, N_{CORES}\right) \geq \Gamma'\left(N_Q^{CORE}, 1\right) \\ & N_Q^{ALG}, N_G = \text{const} \end{aligned} \tag{7.1.4}$$

In our exploration, intra- and inter-core topology, qubit implementation, and interconnect technology are assumed to be fixed, while the application-specific parameters vary on every benchmark.

7.1.5 Latency ratio threshold experimental results

We have performed an in-depth exploration of the design space, sweeping all the input variables in wide ranges. Standard benchmarks (QFT, Grover's) have their own profile in terms of number of gates (N_G) and 2-qubit gates fraction (N_G^{2Q}/N_G), both of which depend on the size of the input (number of qubits required, N_Q^{ALG}), which we have varied from 10 to 3000. In the random case, we have fixed the number of gates, while sweeping the number of qubits in the same range, as a way to keep constant the number of independent variables modifying the output performance. Aiming at the future high-scaling multi-core computers,

[2] Note that this latency is different from the qubit communication latencies L_{COMM} and L_{CORE}.

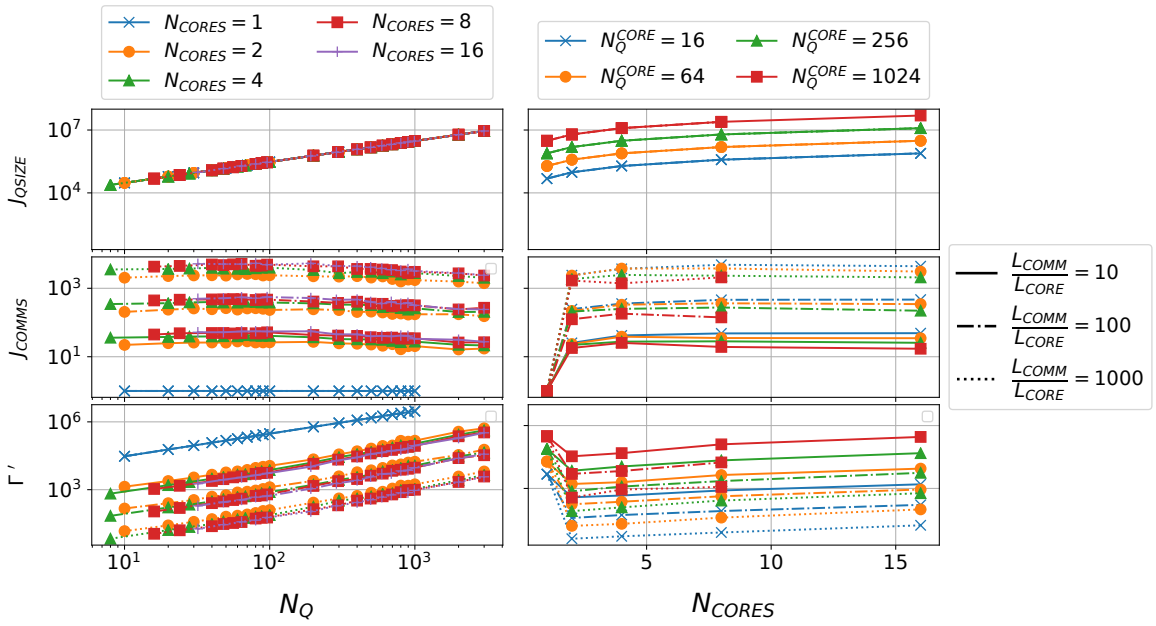


Figure 7.2: Full-blown variable exploration of multi-core quantum architectures for the random benchmark (80% of gates are two-qubit operations). From top to bottom, the size of the algorithm (qubits × gates), communications overhead, and FoM Γ' are plotted against the number of qubits of the algorithm, and the number of cores of the architecture, varying the number of qubits per core (and the total number of physical qubits accordingly). In this benchmark, the number of gates has been fixed, while the number of qubits of the algorithm increases along the size of the architecture.

we have explored ranging from modest sizes (2 cores and 10 qubits per core) to 16 cores and 1024 qubits per core. Finally, we have characterized costly inter-core communications with latencies ratio (L_{COMM}/L_{CORE}) that starts from 10 and goes up to 1000, in order to stress the analysis in the most crucial point. A summary of these input parameters can be found in Table 7.1.

Table 7.1: Design Space Exploration variables and parameters (notation and values for each benchmark)

Notation	QFT	Grover's	Random 20 %	Random 80 %
N_G	$\mathcal{O}(N_Q^{ALG})^2$	$\mathcal{O}(N_Q^{ALG})$	3000	3000
N_G^{2Q}/N_G	$\sim 50\%$	$\sim 30\%$	20%	80%
N_Q^{ALG}	$[10, 3000]$			
N_{CORES}	$[1, 16]$			
N_Q^{CORE}	$\{16, 64, 256, 1024\}$			
L_{COMM}/L_{CORE}	$\{10, 100, 1000\}$			

In Fig. 7.2, the full-blown variable space exploration for a single benchmark (the random benchmark with 80% of 2-qubit gates) is shown. The design space is projected onto the two architecture-related dimensions, i.e. total number of qubits and number of cores available in the computer. From top to bottom, the FoM Γ' components are shown (the algorithm size is plotted in the first row, right below the communications overhead can be found, and the aggregated final metric is on the last row). In this way, we can see the effects of both parameters on the performance.

On the left side, we can see the different clusters of lines in the communications overhead plot (in the middle), for the three different ratios of L_{COMM}/L_{CORE}: the performance scales linearly with the inter-core latency. This does not benefit the multi-core approach, as the single-core processor performs always better. Very importantly, observe that the single-core curve is discontinued at 10^3 qubits, as that is the forecasted approximate upper limit (number of qubits in a single core) for current qubit integration technologies [2]. This allows multi-core architecture performance to grow past that limit, achieving single-core levels of performance by increasing the number of qubits and number of cores, for the same qubit technology. Moreover, we expect that errors (not modeled in this analysis) will explode with the number of qubits per core (i.e. most of these errors come from the crosstalk and other interferences, thus aggravated in the single-core case, as we are integrating higher numbers of qubits in the same core). For a fixed number of cores, the communications overhead decreases slowly (as more qubits are fitted inside the same core, fewer inter-core movements are needed), and thus the performance increases. Note also that the communications overhead is higher than the L_{COMM}/L_{CORE}, as the overhead accounts as well for delays in instructions scheduling that are a byproduct of longer qubit transportation waiting times.

In this case, it is interesting to observe in both plots the concave trend of the communications overhead, which after a maximum point between 10 and 100 qubits, starts a monotonic descent. This is due to having less percentage of inter-core communication as we increase the number of qubits per core along with the size of the algorithm.

The bottom-right plot contains the most valuable information in this first wide exploration: we can see from left to right the cost of going from single-core to multi-core (see the steep descent of the performance in the interval $[1, 2]$, and most importantly, how the multi-core architecture, as the number of cores is increased (i.e. parallelism), recovers performance until almost reaching single-core performance values with 16 cores working together. That is, adding cores allows increasing the number of available physical qubits for computation, breaking the single-core qubit integration limit, and overcoming the latency overhead from inter-core communications. This sheds light on the potential of multi-core architectures.

The same full-blown exploration for QFT, Grover's, and the low-communication random benchmark are also shown in Fig. 7.3, and compared to the already seen high-communication random benchmark (tagged as 'random 80%'). The lowest performance is that of the random algorithms, as the two-qubit gates are distributed uniformly, without any structure, among all qubits: even the best efforts coming from the mapper in the low-communication benchmark are not capable of decreasing inter-core qubit interaction overhead. QFT shows a steeper increase in performance as the architecture grows: even having a moderate amount of qubit communication (see Table 7.1), its structure allows the mapper to distribute it properly. Note that its minimum performance is the worst, but it also shows the higher maximum, surpassing the single-core performance with as few as 4 cores. In a nutshell,

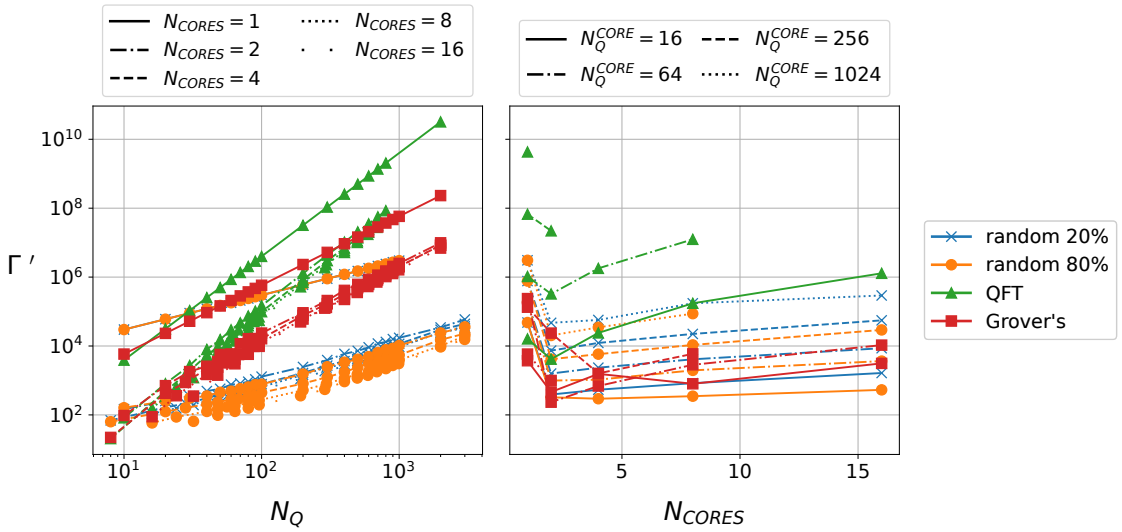


Figure 7.3: Several benchmarks' scalability on different multi-cores architectures: two different random benchmarks (varying the fraction of two qubit-gates (20%–80%), QFT and Grover's search

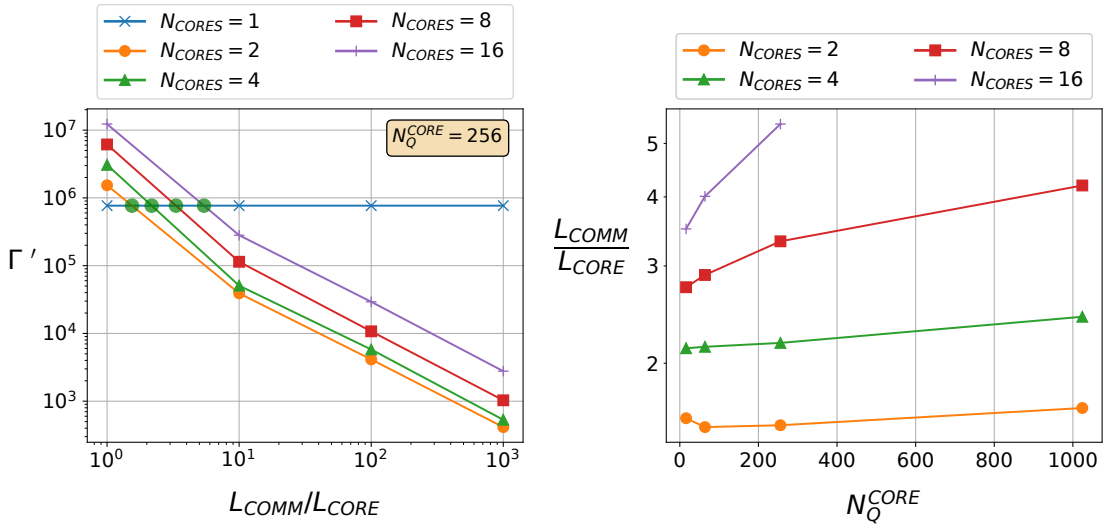


Figure 7.4: DSE ultimate goal: optimal inter-core latency for every architecture (here the case for random (80%) is shown). The evaluation points lacking in the plot correspond to compilations that took an excessive amount of time to complete

multi-core architectures benefit from highly structured pieces of quantum code, which are frequently used in many well-known algorithms.

Having explored the design space in this way, we have been able to see, for different architectures, applications, and inter-core communication performance, the *point of balance* where multi-core architectures supersede single-core performance. Therefore, let us now explore this the other way around, looking for the *decision threshold* where the inter-core communication adoption in multi-core architectures starts to pay off and exceed single-core

performance. This *decision threshold* will depend on the application and the architecture. See in Fig. 7.4, left plot, the graphical representation of the *decision threshold* finding for a specific application and several architectures, as the crossing point of the performance single-core line (which is flat, as it does not depend on the inter-core communication latency) with the performance curves of different architectures. In Fig. 7.4, right plot, the *decision thresholds* curves for a wide range of architectures are shown. Observe that, for instance, a quantum computer with 4 cores and 256 total qubits will outperform the corresponding single core's performance if $L_{COMM} < 2.1 \times L_{CORE}$ (take into account that the benchmark shown is communication intensive). Note also that, even though the use of more cores implies higher inter-core qubit communication, the more cores we use, the lower the communication performance (i.e. the higher the maximum L_{COMM}/L_{CORE} ratio) is required.

7.1.6 Discussion

The presented exploration of the multi-core quantum architectures space has let us find the threshold where these distributed architectures outperform single-core traditional quantum processors, focusing on a highly specific technology parameter such as the inter-core communication latency (more precisely, the ratio between the intra-core latency and the inter-core latency). Together with these results, which indicate upper bounds for inter-core communication technology performance, we have shown that multi-core architectures, in addition to breaking the qubit integration limits of single-core quantum computers, exploit parallelism for widely used well-known structured quantum algorithms.

Adding to this work some other important constraints, such as qubit control and operation, connectivity-constrained intra-core topology as well as intra- and inter-core operation error rates will let us refine the obtained results, which pave the way to exposing compelling design guidelines on these architectures using standard quantum applications.

7.2 The first fully-fledged simulation-based multi-core quantum computers scalability analysis

After such promising results, in this second part of the chapter we aim at culminating the thesis by reproducing the DSE exploration for a scalability analysis on multi-core quantum architectures already presented in Chapter 4, now backed by a fully-fledged simulation including accurate quantum noise models and actual execution of quantum circuits. In fact, every layer in the double full-stack is present in the simulation. Showcasing this tool lets us foresee the advantages of utilizing advanced design tools even in the early stages of the development of QC.

In the following, we describe in detail the scope of the simulated model, define the adequate performance metric for this analysis, and present some results.

7.2.1 Traversing the double full-stack through simulation

As we already did with the DSE study in Chapter 4, it is convenient in order to set the stage to review the design space to be explored. Specifically, we are going to go over the

entire double full-stack for multi-core QC, layer by layer, as an orderly way to check how our simulator models the variables and parameters as defined in Section 4.3.1.

We have used the same simulator presented in Section 6.4.2: a fully-fledged model based on the NetSquid package, which we have developed to implement a functional simulation of a detailed multi-core quantum computer, from the qubit level (with its corresponding decoherence model) to the quantum and classical networking among the different cores. This simulator is capable of executing a given quantum circuit, aided by a modified version of OpenQL for the compilation and mapping process.

Observe that this time, as opposed to the analysis in Chapter 4, with the simulated model we cover not only the physical computing and communication layers (qubit, core and network), but also the runtime/compiler and application ones.

7.2.1.1 Qubit layer

Although NetSquid is a simulation package focused on large-scale quantum networking, it includes accurate models for operating with qubits, as well as modeling quantum decoherence on them and adding quantum noise to qubit gates. Every aspect at the single-qubit control level can be customized, which has allowed us to cover in detail this layer's most important variables and parameters for our exploration of the design space:

- **Coherence time.** The coherence time τ_c (i.e. an upper bound for the time we can operate and read out the state of the qubit) has been modeled by means of the T_1 and T_2 constants (also called amplitude and phase damping, respectively). In this *memory noise model* typically T_2 is approximately two times the value of T_1, reducing it into a one-dimension variable. For having a reference from recent experimental values, we have studied in detail refs. [237], [130], and [242] as in the previous chapter (see Table 7.2 for more details on the explored range).

- **Quality factor - single qubit gates.** Control operations quality, conveyed by the quality factor QF, is limited by the already reviewed coherence time, and the gate latency L_G (i.e. the time spent in performing a qubit gate), as $QF = \tau_c / L_G$. In our simulator, using state-of-the-art superconducting qubit values as a reference [130], we have set the single-qubit gate latency to 30 ns, while the measurement operation latency is set to 300 ns.

- **Single-qubit gate fidelity.** The single-qubit gate fidelity (F_G) is affected not only by the decoherence noise but also by the cumulative errors from manipulating qubits when applying gates and other control operations. In our model, we have included a random dephasing noise model for the gate operations (following error model analysis from [215]). For this, we have followed experimental values from a state-of-the-art superconducting quantum processor such as the one presented in [81], taking both its single-qubit gate and measurement error probabilities as the probabilities of dephasing after each of these operations are applied.

7.2.1.2 Core layer

In the simulation, the core layer is simulated with quantum processing nodes containing a set of qubits and a set of defined qubit gates. This let us also implement an accurate model for the execution of quantum instructions in each core, with their corresponding quantum noise models and possible topology limitations. However, it is not feasible to model crosstalk effects yet (i.e. qubit perturbations due to neighboring operations). Considering this would help in observing how putting together too many qubits in the same chip makes things worse. Moreover, we have not considered either fabrication variability or yield effects.

- **Two-qubit gate fidelity.** Similarly to what has been described for the single-qubit gate noise model, a dephasing effect is implemented following values from [81] when applying two-qubit gates.
- **Quality factor - two-qubit gates.** In our simulation, we have included a CNOT gate as the single available two-qubit gate, which can also be used for two-qubit swapping. This gate's latency has been set, according to state-of-the-art experimental values [130] to 60 ns.
- **Number of qubits per core.** Through our simulated model, we have been able to explore up to 1024-qubit cores, although the total number of qubits that can be simulated is of course limited to the memory and processing capabilities of the processor running the simulation, as well as by the simulation time constraints.
- **Number of communication qubits per core.** Out of the N_Q^{CORE} qubits, as we already mentioned in the experiments from the second part of Chapter 6, two of them are reserved for inter-core communications. They are used in the teleportation qubit transfer as buffers for the EPR photons and the temporary qubits that are generated during the process.
- **Intra-core topology.** In order to focus on the multi-core architecture and facilitate the compiler's task to enable the simulation of larger architectures, we have assumed all-to-all connectivity within each core. That is, every pair of qubits within a core can directly interoperate by means of a CNOT gate, in any direction.
- **Quantum intra-core communication latency.** As we are considering an all-to-all intra-core topology, the time it takes to communicate qubits within the chip (i.e. to move them around in order to be able to apply a two-qubit gate between two given qubits) is 0: there is no need to move qubits around inside the core.
- **Qubit intra-core transfer rate.** Following the previous reasoning, due to the assumptions made, this rate is infinite in our model.

7.2.1.3 Network layer

As explained in Section 6.4.2.1, we have focused on a multi-core network centered on a shared EPR pair generator, which controls the entanglement distribution for the qubit transfers among any pair of nodes by using quantum teleportation.

- **Number of cores.** Depending on the number of qubits per core, we have been able to simulate up to 128 cores, though the main exploration has been carried out on smaller sets of 8, 16, or 32 cores.

- **Total number of qubits.** Up to architectures implementing 16000 qubits have been simulated. Although this might seem a big number for the state-of-the-art experimental devices, we need in fact larger sets of qubits for fully-fledged QC. However, within the evident limitations of a classical simulation of a quantum computer, this number of qubits allows us to extract some practical conclusions for the design of future multi-core quantum computers.

- **Inter-core topology.** The optically-enabled star topology among all nodes and the EPR pair generator (at the center of the star) allows for an effective all-to-all inter-core connectivity once the entanglement is distributed, as the classical multi-core network connects also directly all pairs of cores.

- **Quantum inter-core communication latency.** The inter-core qubit transfer primitive is quantum teleportation, which latency has been modeled in Section 5.3.2. In our simulator, following the SWAP teleportation protocol described in 6.9, we have modeled realistic EPR distribution, classical communication, and pre- and post-processing delays.

- **Qubit inter-core transfer rate.** Following our study in Section 5.3.3, this inter-core qubit commmunication rate is typically limited by the EPR pair generation frequency. Nonetheless, in the present exploration, we have not applied any EPR generation output error, in order to simplify this first fully-fledged analysis: EPR generation delays provide already a sufficient source of qubit decoherence. Therefore, we have explored different scenarios with varying R_{EPR} in order to see its effects on the overall performance.

7.2.1.4 Runtime/compiler layer

The present analysis is not limited to the physical modeling of both the computing and communication physical implementations of the multi-core quantum computer. On the contrary, the developed extension to NetSquid allows us to fully run a quantum circuit on top of the architecture.

In order to be able to do so, we need a compiler for quantum circuits which may allow us to distribute the circuit among the different cores, which we have implemented as an extension to OpenQL as explained in Section 6.2.1.

On the other hand, the compiled code has to comply with the input requirements for quantum processors in NetSquid, not capable of executing directly a QASM input. Therefore, we have developed a tool for parsing the compiled output from our modified version of OpenQL in order to generate the different sets of code for each core. Then, the execution logic implemented on top of each core is in charge of interpreting each instruction and executing the corresponding physical instruction. You can revisit Fig. 6.8 to see a diagram of this execution flow.

7.2.1.5 Application layer

This layer corresponds to the uppermost quantum circuit logic. As it is hardware-agnostic, there are no specific items related to the novelty of multi-core quantum computers to deal with here. In any case, our simulator, being able to execute any quantum circuit input, enables the possibility of exploring some cross-layer optimizations of the execution from the application layer (e.g. applying QoS class-like policies as suggested at the end of the previous chapter).

In order to have a uniform benchmark for this first fully-fledged scalability analysis, we have chosen the same random benchmark used in the second part of Chapter 6 for the possibility it offers to control some parameters that let us force the system into extreme scenarios.

7.2.2 A simulation-based model for a DSE scalability analysis of multi-core quantum architectures

Although the detailed study presented in Section 4.3.2 has given us a wide perspective of the potential of a technology-agnostic multi-core quantum computer performance exploration, it is basically a preliminary approach, limited to an analytical FoM, not related to any previously existing well-known quantum metric or simulations based upon experiment-supported models. Now, with the aid of a fully-fledged multi-core architecture simulator, we can develop a simulation-based perfected model to supersede it.

Let us first review the metrics chosen in Chapter 4 for our behavioral model as a reference, in order to identify how to define them in the current scenario: *i)* Qubit quality, *ii)* aggregated fidelity in a quantum core, *iii)* computational power, *iv)* performance degradation due to qubit single-core integration limit and *v)* inter-core communications performance.

Observe that, now that we have a simulated model, most of these behavioral metrics become variables influencing the overall performance. In particular, qubit layer quality metrics (coherence time, single-qubit gate latencies and error rates), as well as aggregated fidelity in intra-core operations (due mainly to two-qubit gate latencies and error rates), degradation due to qubit integration limits (inter-qubit crosstalk and controllability issues) and inter-core communications performance can be aggregated into the overall fidelity of the circuit, as they might affect in different ways to this overarching performance metric.

On the other hand, computational power/capabilities is still of the utmost importance for a scalability analysis, setting a trade-off with the limitations of the multi-core approach as we have seen in the previous section. There is no information in knowing that an unknown circuit has been executed resulting in a high-fidelity result: this will be more valuable the larger the circuit may be (in terms of number of qubits used and number of operations performed). Therefore, we have considered the following two metrics:

Computational effort: the total number of gates executed in the quantum circuit.

$$J_{QSIZE} = N_G \tag{7.2.1}$$

Overall fidelity: the average fidelity $F(\cdot)$ of all qubits (q_i) used in the circuit. This performance metric encompasses the effects from all the error models at the qubit and core

Table 7.2: Notation, symbol definitions and values used in simulation-based scalability analysis, with reference to values from [81, 143, 227, 237]

Notation	Meaning	Value (range)
N_{CORES}	Number of cores/chips	$\{1, 2, 4, 8, 16\}$
N_Q^{CORE}	Number of qubits per chip	16 - 1024
N_Q	Total number of qubits in the quantum computer	$N_{CORES} \cdot N_Q^{CORE}$
N_G	Total number of gates	$\propto N_Q$
L	Length of the quantum circuit in execution cycles	$[100, 250]$
T_{init}	Init time per qubit	$\sim U(0, L)$
`p_idle`	Idle state probability	$\sim U(0, 1)$
`p_2qubitgates`	Two-qubit gate probability	$\sim U(0, 1)$
`p_intercore`	Inter-core gate probability	$\sim U(0, 1)$
R_{EPR}	EPR pair generation rate	$10^6, 10^7, 10^8$ Hz
e_1	single-qubit gate error probability	0.015 [81]
e_2	two-qubit gate error probability	0.036 [81]
e_r	measurement/readout error probability	0.031 [81]
T_1	Amplitude damping for the memory noise model	$2 \cdot T_2$ [237]
T_2	Phase damping for the memory noise model	$10^6, 10^8, 10^{10}, 10^{12}$ ns [237]

level, as well as the inefficiencies at the inter-core networking level.

$$J_F = \frac{1}{N_Q} \cdot \sum_{\forall q_i} F(q_i) \tag{7.2.2}$$

As both metrics complement each other to produce an overall performance metric and their effect on the overall trends can be easily told, we can use the multiplicative approach used in Eqs. 4.3.12 and 7.1.3, defining the new FoM Γ'' as the product of the two defined metrics, after normalizing J_{QSIZE} to the $[0, 1]$ interval (note that J_F is already bounded within that same interval), with their corresponding weights γ_i [189]:

$$\Gamma'' = \left(J_F\right)^{\gamma_F} \cdot \left(J_{QSIZE}\right)^{\gamma_{QSIZE}} \tag{7.2.3}$$

7.2.3 Scalability analysis

In order to replicate the scalability analysis, now based upon a fully-fledged simulation model, we have performed a wide exploration of the design space, varying the multi-core size and networking parameters, as well as the fundamental qubit quality models. In particular, we have pivoted the analysis on the following parameters: number of cores, number of qubits per core, decoherence, and gate error rates, and the EPR pair generation rate. The explored variable space is summarized in Table 7.2.

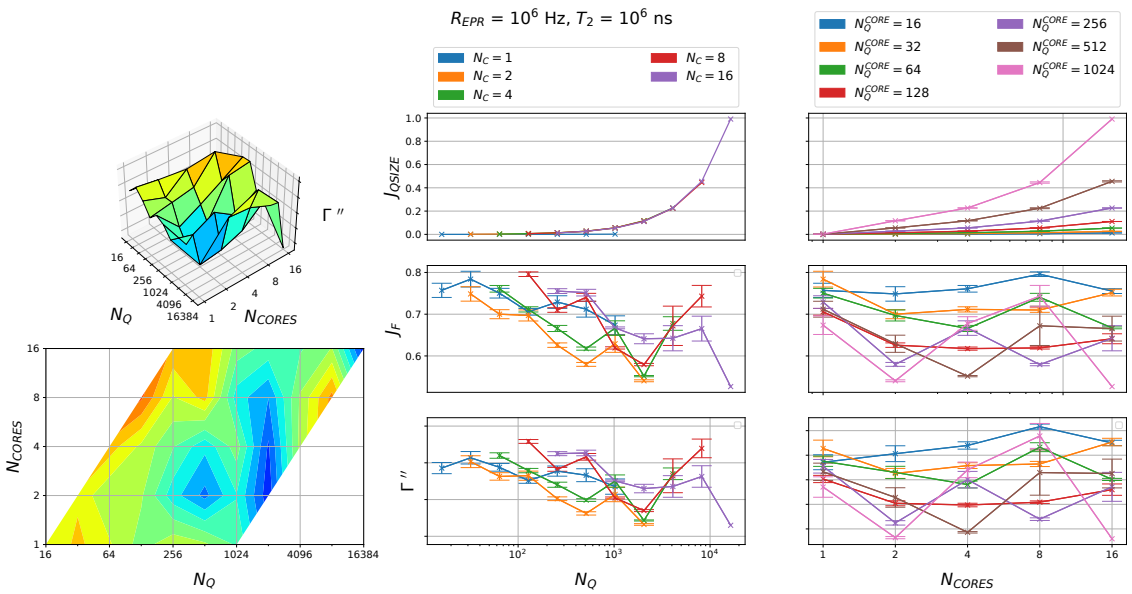


Figure 7.5: Design Space Exploration for $R_{EPR} = 10^6$ Hz and $T_2 = 10^6$ ns

We show three samples of such exploration in Figs. 7.5, 7.6 and 7.7, corresponding to three different technological points, going from almost state-of-the-art to an improvement of several orders of magnitude. In these three figures, the 3D plotting of Γ'' is shown versus N_Q and N_{CORES} on the left side, while the disaggregation of the FoM on both axes is shown on the right side. In this way, we can see clearly how each metric affects to the overall result. Remember that Γ'' is not to be interpreted as a numerical result, but just as an aggregation of metrics of interest: even though most of the time single-core cases may not have high values for Γ'', that is due to the lower computational capability (J_{QSIZE}) and not because of having low fidelity results.

Fig. 7.5 let us see that with these low-performing technology parameters, there is still no chance for multi-core to result in any gain when compared to single-core. Observe how the maxima are basically around the lowest N_Q^{CORE} with degradation of performance when pushing up the number of cores.

Going up in the technology (see Fig. 7.6) implies a good step up (observe that the color map in the 3D and heatmap plots is common to all three figures in order to facilitate the analysis). In this case, the performance improves uniformly providing a greater region of efficient multi-core architectures: see for instance cases with 8 cores and 1024 qubits, 4 cores with 256-1024 qubits, or 2 cores and 256-512 qubits. Naturally, larger core sizes imply lower fidelities but observe that the optimum values in terms of Γ'' for N_Q^{CORE} for a specific number of cores is different from 16 qubits (the most conservative value).

Finally, stretching technology up to 1000 seconds of decoherence time and almost 1 GHz of entanglement generation rate (see Fig. 7.7), we have the higher performing platforms already in the 1024 qubits per core side, with a maximum around 2 and 4 cores. However, observe that even though the towering technology gap from the previous plot, the performance is somewhat saturated in the same range of values (here the absence of a qubit

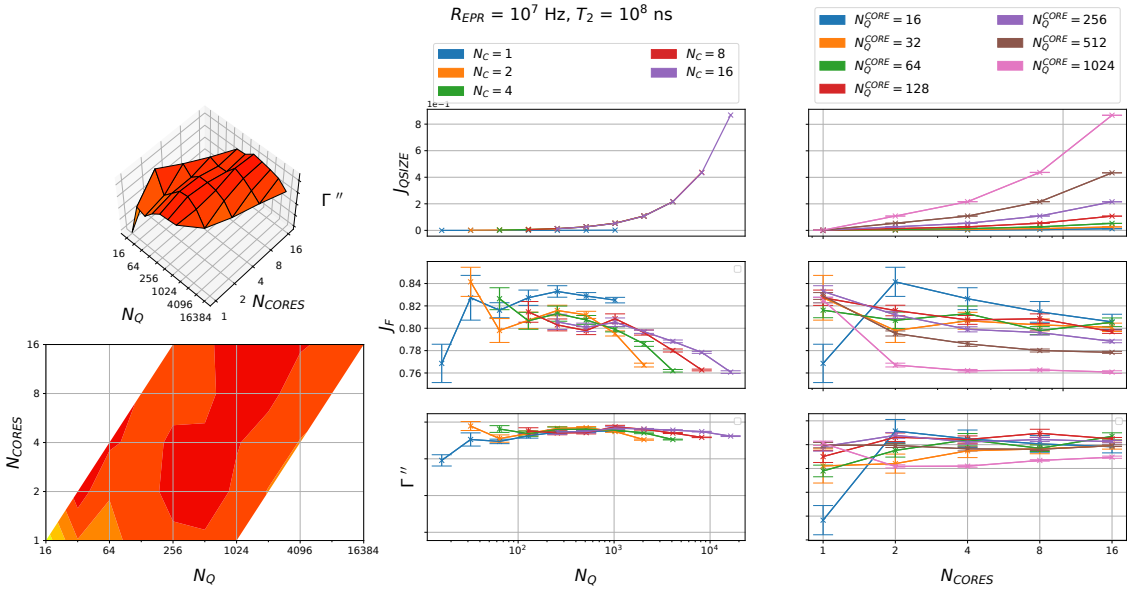


Figure 7.6: Design Space Exploration for $R_{EPR} = 10^7$ Hz and $T_2 = 10^8$ ns

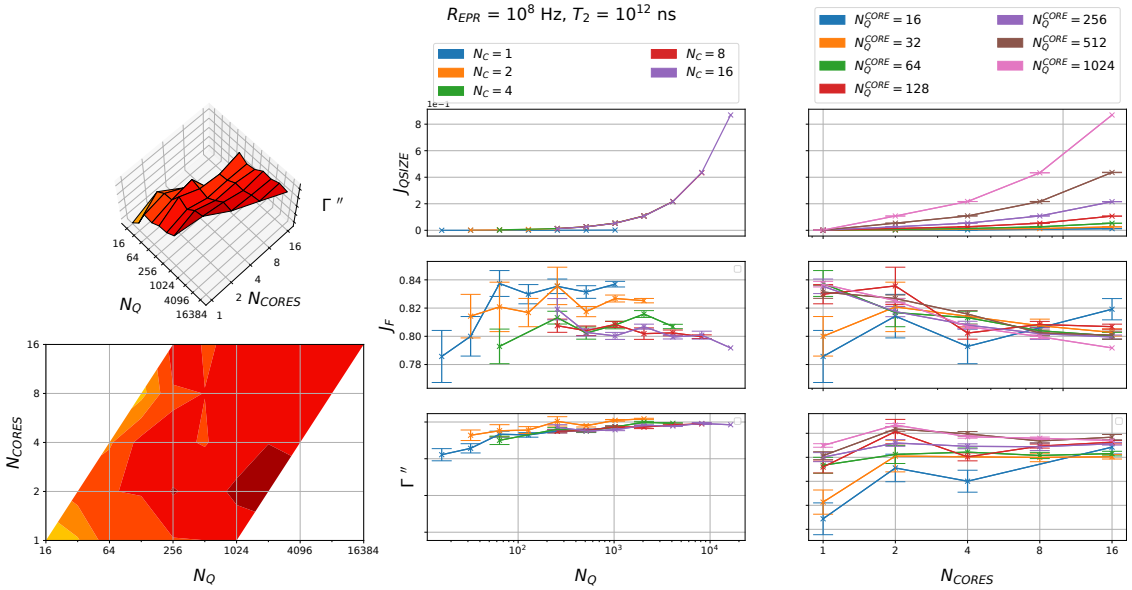


Figure 7.7: Design Space Exploration for $R_{EPR} = 10^8$ Hz and $T_2 = 10^{12}$ ns

crosstalk model in the simulation is also helping to not having a decrease in performance for larger core sizes). Importantly, in this case, the speed-up when adding cores to the system actually pays off, having an increasing trend in that direction.

An interesting complementary analysis is that of a technology gap analysis. As for every technology parameter set we have found a varying optimal region, it is of high interest to study how those "high-performing architectures" regions evolve with the improvement of

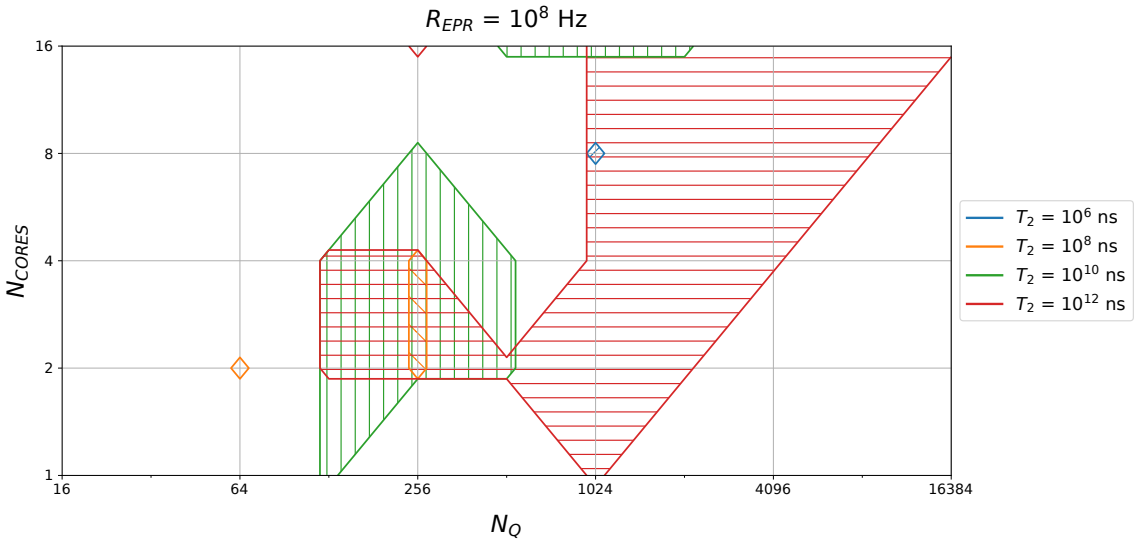


Figure 7.8: Multi-Core Quantum Technology Gap Analysis for $R_{EPR} = 10^8$ Hz

technology. To do so, we have done a specific study, which is showcased in Fig. 7.8. In this plot, which is done for a static value of R_{EPR}, simulations for increasing values of T_2 are shown, by only drawing the "architecture area" where that technology performs over the 75% quartile of all simulations (in terms of Γ'').

Observe how for increasing values of T_2 the optimal architecture range increases. Very importantly, the right-most part of the plot is almost completely filled for $T_2 = 10^{12}$ ns, i.e. multi-core-enabled quantum computer scalability will be effective for such technology parameters.

7.2.4 Discussion

The importance of these analyses is not related to the exact trends or figures obtained, but to the tool they provide to us for running the exploration for any given technology and extracting critical conclusions which we have been looking for throughout this whole work. In particular, think of e.g. IBM and their scalability roadmap [1]: how might they know where to invest more efforts, time, and money, whether in the inter-core communications technology, or in the qubit decoherence time, or else in the two-qubit gate fidelities, etc? Such an analysis as the one we have just presented greatly facilitates this task which is crucial for a rapid and efficient design of the future of QC.

Chapter 8

Conclusions

At the beginning of this thesis, (see section 2.3.1) we raise three fundamental questions that could focus the research of this thesis. After having gone through the development and results of this work, it is now the perfect moment to review them and check how much we have been able to answer them:

1. *Will the multi-core approach unlock the current monolithic single-core quantum computers' scalability bottlenecks?* In the different analyses we have performed (see especially the most complete results in Chapter 7), we have explored the multi-core quantum architecture space observing that current technology is not yet ready for these architectures. However, we have been able to provide some figures (and most importantly, reusable tools) in order to find the thresholds and regions where these distributed architectures outperform single-core traditional quantum processors. When exploring technology improvements, we have been able to prove that these architectures will effectively enable QC scalability. We have also shown that multi-core architectures, in addition to breaking the qubit integration limits of single-core quantum computers, exploit parallelism for widely used well-known structured quantum algorithms. Of course, improving simulation models and amplifying exploration to other qubit technologies, parameters, or architectures will provide us with more valuable information on this matter.

2. *For a given technology used to implement (or a specific application to be executed on) a multi-core quantum architecture, which is the optimal architectural configuration?* In the different explorations carried out along this work (Chapters 4-7), we have always explored different architectural configurations in terms of the number of cores and the number of qubits, though future research will for sure include other types of inter-core communication, intra-core topologies, and other elements. In any case, for our explorations, we have been able to detect architectures and algorithms that pair better than others, depending on the circuit structure and multi-core topology: in particular, tightly inter-qubit dependent algorithms impose a hard challenge for having efficient executions on multi-core platforms.

3. *How could we improve inter-core communications in order to have fast/efficient multi-core quantum architectures?* Most specifically in Chapters 5 and 6, we have inves-

tigated critical trade-offs, overhead, particularities and optimal designs in quantum inter-core communication for these architectures. With a thorough analytical study of the quantum teleportation circuit, we have shown how waiting times and latencies can greatly affect quantum communication quality, and have tested how a quantum algorithm may behave in a many-core scenario. In particular, due to the scarce amount of qubits available, we have explored and obtained the optimal share of qubits among the computation and communication roles. For the chosen inter-core teleportation technology and configuration (single EPR pair generator shared among all nodes for quantum teleportation) it is evident to quickly cause inefficiency due to bottlenecks, that could be improved with faster entanglement generation rates or parallelizing several EPR pair generators. Other inter-core communication technologies should be explored, as this is a critical point of failure in these architectures.

Most importantly, on top of these specific answers, we have contributed in several ways to the foundations of multi-core quantum computing design and research:

- **Double full-stack layered architecture for multi-core quantum computers**: as a key part of our proposal, entangling quantum communications and computing within the design of a multi-core platform is part of the cross-layer approach we need in order to enable QC scalability. This vision is substantiated by the great dependency between communication latencies, execution times and cumulative qubit decoherence in the current NISQ era.

- **DSE-based system-wide optimization proposal**: using such a structured design technique might facilitate once-for-all design guidelines unifying the still separated quantum computer's parts into a consolidated solution with optimal technologies and parameters for every situation. All the presented explorations are in fact more valuable for the code implemented to perform them than the results themselves, as the code can be reused and improved with actual parameters and data coming from experimentalists, helping them to get their investment and research focus right on the optimal spots detected in the DSE explorations.

- **Analytical and simulated models of multi-core quantum computing performance**: In Chapters 4 and 7 we have presented both behavioral and simulated models for testing multi-core quantum computer performance, based on FoM functions that allow us to focus on specific metrics with high flexibility. Such models can be easily improved in the future and allow for clean analysis of QC performance.

- **Fully-fledged multi-core quantum computing simulator**: in order to develop the fully-fledged explorations in Chapter 7, as well as to study different communication protocols in the second part of Chapter 6, we have developed (based upon the NetSquid library) a full reusable multi-core quantum computer simulator capable of running quantum circuits and including the modeling of qubit decoherence, gate errors, actual communication latencies and inefficiencies, etc.

- **Qubit traffic analysis tool**: In order to visualize and extract performance metrics specific to the inter-core qubit traffic for different algorithms and architectures, we

have developed a reusable tool for exploring and analyzing multi-core communication networks. This might help, among other things, to develop better multi-core quantum compilers and to compare topologies and circuit efficiency.

- **First approach to a MAC layer protocol for improving QC performance**: the entanglement between quantum communications and computation suggests that improving communications performance directly impacts the overall multi-core quantum computer's efficiency. Although in our first approach, we have had to face the hurdle of qubit gates dependency, we have been able to study different approaches for controlling inter-core communications from an interesting QoS/MAC design view.

8.1 Future work

The present thesis has opened several promising research lines that have interesting branches and follow-ups to be investigated yet. In particular, the multi-core quantum computer simulator, as a key tool for design exploration and research on such approach to QC architectures, is to be improved by adding constraints to intra-core qubit topology, a model of qubit crosstalk and inter-core communication technologies and architectures other than the one we have already implemented for this thesis.

The qubit traffic and communications overhead analysis, as well as the full DSE scalability and technology gap exploration, will benefit from including more well-known structured algorithms, as well as new multi-core mapping proposals that might show higher efficiency (e.g. [243]).

We plan to use the qubit traffic analysis and multi-core quantum simulator tools to do further explorations with larger sets of benchmarks and a range of target architectures, as well as complement this analysis with fully-fledged simulations that may shed light on online quantum network management for error mitigation. Also, it is worth exploring which structural parameters in quantum circuits are the reason behind most inefficiencies found (data transfer bursts, hotspots, code dependencies...). This in-depth analysis might help us improve our inter- and intra-core communication strategy and later on give us the guidelines for multi-core device design that is more compatible with specific types of algorithms.

Promising outcomes are to come from the exploration of the inter-core communications control protocol from the point of view of the MAC layer, where we will try to overcome the complexity of inter-qubit dependencies-related blockages. We believe such a protocol can be a key part of developing the dynamic scheduling of quantum circuit execution from a communications perspective.

Appendix A

Derived Publications

The contributions of this thesis have been adapted, submitted and/or published both in journals and conferences. The publications related to the work of this thesis are as follows:

- S. Rodrigo, S. Abadal, C. G. Almudever, and E. Alarcón, "Efficient exploration of design guidelines for scalable multi-core quantum computers on a fully-fledged simulator," *in preparation*
- M. Bandic, L. Prielinger, J. Nüßlein, A. Ovide, S. Rodrigo, S. Abadal, H. van Someren, G. Vardoyan, E. Alarcón, C. G. Almudever *et al*, "Mapping quantum circuits to modular architectures with QUBO," *arXiv preprint arXiv:2305.06687*, 2023
- A. Ovide, S. Rodrigo, M. Bandic, H. Van Someren, S. Feld, S. Abadal, E. Alarcón, and C. G. Almudever, "Mapping quantum algorithms to multi-core quantum computing architectures," *arXiv preprint arXiv:2303.16125*, 2023.
- S. Rodrigo, D. Spanò, M. Bandic, S. Abadal, H. Van Someren, A. Ovide, S. Feld, C. G. Almudever, and E. Alarcón, "Characterizing the spatio-temporal qubit traffic of a quantum intranet aiming at modular quantum computer architectures," in Proceedings of the 9th ACM International Conference on Nanoscale Computing and Communication, 2022.
- S. Rodrigo, S. Abadal, C. G. Almudever, and E. Alarcón, "Modelling short-range quantum teleportation for scalable multi-core quantum computing architectures," in Proceedings of the 8th ACM International Conference on Nanoscale Computing and Communication, 2021.
- S. Rodrigo, S. Abadal, E. Alarcón, M. Bandic, H. van Someren, and C. G. Almudever, "On double full-stack communications-enabled architectures for multi-core quantum computers," in IEEE micro, 41(5), 48-56, 2021.
- S. Rodrigo, M. Bandic, S. Abadal, H. van Someren, E. Alarcón, and C. G. Almudever, "Scaling of multi-core quantum architectures: a communications-aware structured gap analysis," in Proceedings of the 18th ACM International Conference on Computing Frontiers, 2021.

- S. Rodrigo, S. Abadal, E. Alarcón, and C. G. Almudever, "Will quantum computers scale without inter-chip comms? A structured design exploration to the monolithic vs distributed architectures quest," in 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS), November 2020.
- S. Rodrigo, S. Abadal, E. Alarcón, and C. G. Almudever, "Exploring a double full-stack communications-enabled architecture for multi-core quantum computers," *arXiv preprint arXiv:2009.08186*, 2020.

Bibliography

[1] J. Gambetta, "Expanding the IBM Quantum roadmap to anticipate the future of quantum-centric supercomputing (IBM Research Blog)." [Online]. Available: https://research.ibm.com/blog/ibm-quantum-roadmap-2025

[2] N. A. of Sciences, *Quantum computing: progress and prospects*. National Academies Press, 2019.

[3] S. Resch and U. R. Karpuzcu, "Quantum computing: an overview across the system stack," *arXiv preprint arXiv: 1905.07240*, 2019.

[4] M. Martonosi and M. Roetteler, "Next steps in quantum computing: Computer science's role," *arXiv preprint arXiv:1903.10541*, 2019.

[5] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, no. 6412, 2018.

[6] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, "Quantum computers," *Nature*, vol. 464, no. 7285, pp. 45–53, 2010.

[7] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[8] R. Van Meter and C. Horsman, "A blueprint for building a quantum computer," *Communications of the ACM*, vol. 56, no. 10, pp. 84–93, 2013.

[9] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[10] J. Gambetta, "Quantum-centric supercomputing: The next wave of computing." [Online]. Available: https://research.ibm.com/blog/next-wave-quantum-centric-supercomputing

[11] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins *et al.*, "Quantum computational advantage with a programmable photonic processor," *Nature*, vol. 606, no. 7912, pp. 75–81, 2022.

[12] J. Chow, O. Dial, and J. Gambetta, "IBM Quantum breaks the 100-qubit processor barrier." [Online]. Available: https://research.ibm.com/blog/127-qubit-quantum-processor-eagle

[13] H.-S. Zhong, Y.-H. Deng, J. Qin, H. Wang, M.-C. Chen, L.-C. Peng, Y.-H. Luo, D. Wu, S.-Q. Gong, H. Su *et al.*, "Phase-programmable gaussian boson sampling using stimulated squeezed light," *Physical review letters*, vol. 127, no. 18, p. 180502, 2021.

[14] J. M. Baker, C. Duckering, A. Hoover, and F. T. Chong, "Time-sliced quantum circuit partitioning for modular architectures," in *Proceedings of the 17th ACM International Conference on Computing Frontiers*, 2020, pp. 98–107.

[15] A. S. Cacciapuoti, M. Caleffi, F. Tafuri, F. S. Cataliotti, S. Gherardini, and G. Bianchi, "Quantum internet: Networking challenges in distributed quantum computing," *IEEE Network*, vol. 34, no. 1, pp. 137–143, 2019.

[16] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.

[17] A. B. Finnila, M. A. Gomez, C. Sebenik, C. Stenson, and J. D. Doll, "Quantum annealing: A new method for minimizing multidimensional functions," *Chemical physics letters*, vol. 219, no. 5-6, pp. 343–348, 1994.

[18] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, "Measurement-based quantum computation," *Nature Physics*, vol. 5, no. 1, pp. 19–26, 2009.

[19] S. E. Venegas-Andraca, "Quantum walks: a comprehensive review," *Quantum Information Processing*, vol. 11, no. 5, pp. 1015–1106, 2012.

[20] S. Resch and U. R. Karpuzcu, "Benchmarking quantum computers and the impact of quantum noise," *ACM Computing Surveys (CSUR)*, vol. 54, no. 7, pp. 1–35, 2021.

[21] X. Wang, Y. Zheng, and S. Yin, "Spin relaxation and decoherence of two-level systems," *Physical Review B*, vol. 72, no. 12, p. 121303, 2005.

[22] C. G. Almudever, L. Lao, X. Fu, N. Khammassi, I. Ashraf, D. Iorga, S. Varsamopoulos, C. Eichler, A. Wallraff, L. Geck *et al.*, "The engineering challenges in quantum computing," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 836–845.

[23] P. Murali, N. M. Linke, M. Martonosi, A. J. Abhari, N. H. Nguyen, and C. H. Alderete, "Full-stack, real-system quantum computer studies: architectural comparisons and design insights," in *Proceedings of the 46th International Symposium on Computer Architecture*, 2019, pp. 527–540.

[24] R. Jozsa, "Fidelity for mixed quantum states," *Journal of modern optics*, vol. 41, no. 12, pp. 2315–2323, 1994.

[25] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A*, vol. 86, no. 3, p. 032324, 2012.

[26] A. Y. Kitaev, "Quantum codes on a lattice with boundary," *arXiv preprint quant-ph/9811052*, 1998.

[27] L. Lao, H. van Someren, I. Ashraf, and C. G. Almudever, "Timing and resource-aware mapping of quantum circuits to superconducting processors," *arXiv preprint arXiv: 1908.04226*, 2019.

[28] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Experimental comparison of two quantum computing architectures," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3305–3310, 2017.

[29] C. G. Almudever, L. Lao, R. Wille, and G. G. Guerreschi, "Realizing quantum algorithms on real quantum computing devices," in *Proceedings of the 23rd Conference on Design, Automation and Test in Europe*, ser. DATE '20. EDA Consortium, 2020, p. 864–872.

[30] P. Murali, N. M. Linke, M. Martonosi, A. J. Abhari, N. H. Nguyen, and C. H. Alderete, "Architecting noisy intermediate-scale quantum computers: A real-system study," *IEEE Micro*, vol. 40, no. 3, pp. 73–80, 2020.

[31] K. R. Brown, J. Kim, and C. Monroe, "Co-designing a scalable quantum computer with trapped atomic ions," *npj Quantum Information*, vol. 2, no. 1, pp. 1–10, 2016.

[32] V. Kaushal, B. Lekitsch, A. Stahl, J. Hilder, D. Pijn, C. Schmiegelow, A. Bermudez, M. Müller, F. Schmidt-Kaler, and U. Poschinger, "Shuttling-based trapped-ion quantum information processing," *AVS Quantum Science*, vol. 2, no. 1, p. 014101, 2020.

[33] A. Einstein, B. Podolsky, and N. Rosen, "Can quantum-mechanical description of physical reality be considered complete?" *Physical review*, vol. 47, no. 10, p. 777, 1935.

[34] D. Llewellyn, Y. Ding, I. I. Faruque, S. Paesani, D. Bacco, R. Santagati, Y.-J. Qian, Y. Li, Y.-F. Xiao, M. Huber *et al.*, "Chip-to-chip quantum teleportation and multi-photon entanglement in silicon," *Nature Physics*, vol. 16, no. 2, pp. 148–153, 2020.

[35] S. Pirandola, J. Eisert, C. Weedbrook, A. Furusawa, and S. L. Braunstein, "Advances in quantum teleportation," *Nature photonics*, vol. 9, no. 10, pp. 641–652, 2015.

[36] F. Xu, X. Ma, Q. Zhang, H.-K. Lo, and J.-W. Pan, "Secure quantum key distribution with realistic devices," *Reviews of Modern Physics*, vol. 92, no. 2, p. 025002, 2020.

[37] M. Peev, C. Pacher, R. Alléaume, C. Barreiro, J. Bouda, W. Boxleitner, T. Debuisschert, E. Diamanti, M. Dianati, J. Dynes *et al.*, "The secoqc quantum key distribution network in vienna," *New Journal of Physics*, vol. 11, no. 7, p. 075001, 2009.

[38] J. Yin, Y. Cao, Y.-H. Li, S.-K. Liao, L. Zhang, J.-G. Ren, W.-Q. Cai, W.-Y. Liu, B. Li, H. Dai *et al.*, "Satellite-based entanglement distribution over 1200 kilometers," *Science*, vol. 356, no. 6343, pp. 1140–1144, 2017.

[39] H. J. Kimble, "The quantum internet," *Nature*, vol. 453, no. 7198, pp. 1023–1030, 2008.

[40] M. Caleffi, A. S. Cacciapuoti, and G. Bianchi, "Quantum internet: From communication to distributed computing!" in *Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication*, 2018, pp. 1–4.

[41] W. Dür, H.-J. Briegel, J. I. Cirac, and P. Zoller, "Quantum repeaters based on entanglement purification," *Physical Review A*, vol. 59, no. 1, p. 169, 1999.

[42] A. Pirker and W. Dür, "A quantum network stack and protocols for reliable entanglement-based networks," *New Journal of Physics*, vol. 21, no. 3, p. 033003, 2019.

[43] A. Dahlberg, M. Skrzypczyk, T. Coopmans, L. Wubben, F. Rozpedek, M. Pompili, A. Stolk, P. Pawełczak, R. Knegjens, J. de Oliveira Filho *et al.*, "A link layer protocol for quantum networks," in *Proceedings of the ACM Special Interest Group on Data Communication*, 2019, pp. 159–173.

[44] H. Bombin, I. H. Kim, D. Litinski, N. Nickerson, M. Pant, F. Pastawski, S. Roberts, and T. Rudolph, "Interleaving: Modular architectures for fault-tolerant photonic quantum computing," *arXiv preprint arXiv:2103.08612*, 2021.

[45] D. Zajac, T. Hazard, X. Mi, E. Nielsen, and J. R. Petta, "Scalable gate architecture for a one-dimensional array of semiconductor spin qubits," *Physical Review Applied*, vol. 6, no. 5, p. 054013, 2016.

[46] "Unveiling IonQ Forte: The First Software-Configurable Quantum Computer." [Online]. Available: https://ionq.com/posts/may-17-2022-ionq-forte

[47] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied Physics Reviews*, vol. 6, no. 2, p. 021314, 2019.

[48] A. Eddins, M. Motta, T. P. Gujarati, S. Bravyi, A. Mezzacapo, C. Hadfield, and S. Sheldon, "Doubling the size of quantum simulators by entanglement forging," *PRX Quantum*, vol. 3, p. 010309, Jan 2022. [Online]. Available: https://link.aps.org/doi/10.1103/PRXQuantum.3.010309

[49] D. Sutter and C. Piveteau, "At what cost can we simulate large quantum circuits on small quantum computers?" [Online]. Available: https://research.ibm.com/blog/circuit-knitting-with-classical-communication

[50] J. Gambetta and M. Steffen, "Charting the course to 100,000 qubits." [Online]. Available: https://research.ibm.com/blog/100k-qubit-supercomputer

[51] C. Monroe, R. Raussendorf, A. Ruthven, K. Brown, P. Maunz, L.-M. Duan, and J. Kim, "Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects," *Physical Review A*, vol. 89, no. 2, p. 022317, 2014.

[52] L. Vandersypen, H. Bluhm, J. Clarke, A. Dzurak, R. Ishihara, A. Morello, D. Reilly, L. Schreiber, and M. Veldhorst, "Interfacing spin qubits in quantum dots and donors — hot, dense, and coherent," *npj Quantum Information*, vol. 3, no. 1, pp. 1–10, 2017.

[53] N. H. Nickerson, Y. Li, and S. C. Benjamin, "Topological quantum computing with a very noisy network and local error rates approaching one percent," *Nature communications*, vol. 4, no. 1, pp. 1–5, 2013.

[54] L. Jiang, J. M. Taylor, A. S. Sørensen, and M. D. Lukin, "Distributed quantum computation based on small quantum registers," *Physical Review A*, vol. 76, no. 6, p. 062323, 2007.

[55] S. Sargaran and N. Mohammadzadeh, "Saqip: A scalable architecture for quantum information processors," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 16, no. 2, pp. 1–21, 2019.

[56] N. Isailovic, Y. Patel, M. Whitney, and J. Kubiatowicz, "Interconnection networks for scalable quantum computers," in *33rd International Symposium on Computer Architecture (ISCA'06)*. IEEE, 2006, pp. 366–377.

[57] B. Kannan, D. Campbell, F. Vasconcelos, R. Winik, D. Kim, M. Kjaergaard, P. Krantz, A. Melville, B. M. Niedzielski, J. Yoder *et al.*, "Generating spatially entangled itinerant photons with waveguide quantum electrodynamics," *arXiv preprint arXiv: 2003.07300*, 2020.

[58] X. Fu, M. A. Rol, C. C. Bultink, J. van Someren, N. Khammassi, I. Ashraf, R. Vermeulen, J. De Sterke, W. Vlothuizen, R. Schouten *et al.*, "An experimental microarchitecture for a superconducting quantum processor," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, 2017, pp. 813–825.

[59] R. Van Meter and S. J. Devitt, "The path to scalable distributed quantum computing," *Computer*, vol. 49, no. 9, pp. 31–42, 2016.

[60] N. C. Jones, R. Van Meter, A. G. Fowler, P. L. McMahon, J. Kim, T. D. Ladd, and Y. Yamamoto, "Layered architecture for quantum computing," *Physical Review X*, vol. 2, no. 3, p. 031007, 2012.

[61] D. Gottesman, "An introduction to quantum error correction and fault-tolerant quantum computation," in *Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics*, vol. 68, 2010, pp. 13–58.

[62] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," *Nature*, vol. 402, no. 6760, pp. 390–393, 1999.

[63] L. Benini and G. De Micheli, "Networks on chips: A new soc paradigm," *computer*, vol. 35, no. 1, pp. 70–78, 2002.

[64] C. Araguz, D. Llaveria, E. Lancheros, E. Bou-Balust, A. Camps, E. Alarcón, I. Lluch, H. Matevosyan, A. Golkar, S. Tonetti *et al.*, "Optimized model-based design space exploration of distributed multi-orbit multi-platform earth observation spacecraft architectures," in *2018 IEEE Aerospace Conference*. IEEE, 2018, pp. 1–16.

[65] R. J. Blume-Kohout and K. Young, "Metrics and benchmarks for quantum processors: State of play," Sandia National Lab (SNL-NM), Albuquerque, NM (United States), Tech. Rep., 2019.

[66] M. Oskin, F. T. Chong, I. L. Chuang, and J. Kubiatowicz, "Building quantum wires: the long and the short of it," in *30th Annual International Symposium on Computer Architecture, 2003. Proceedings.* IEEE, 2003, pp. 374–385.

[67] J. Heckey, S. Patil, A. JavadiAbhari, A. Holmes, D. Kudrow, K. R. Brown, D. Franklin, F. T. Chong, and M. Martonosi, "Compiler management of communication and parallelism for quantum computation," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2015, pp. 445–456.

[68] R. Risque and A. Jog, "Characterization of quantum workloads on SIMD architectures," in *2016 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 2016, pp. 1–9.

[69] D. D. Thaker, T. S. Metodi, and F. T. Chong, "A realizable distributed ion-trap quantum computer," in *International Conference on High-Performance Computing*. Springer, 2006, pp. 111–122.

[70] T. Coopmans *et al.*, "Netsquid, a discrete-event simulation platform for quantum networks," *arXiv preprint arXiv:2010.12535*, 2020.

[71] R. P. Feynman *et al.*, "Simulating physics with computers," *Int. j. Theor. phys*, vol. 21, no. 6/7, 1982.

[72] D. Deutsch, "Quantum theory, the church–turing principle and the universal quantum computer," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 400, no. 1818, pp. 97–117, 1985.

[73] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.

[74] E. Gerjuoy, "Shor's factoring algorithm and modern cryptography. an illustration of the capabilities inherent in quantum computers," *American journal of physics*, vol. 73, no. 6, pp. 521–540, 2005.

[75] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Physical review A*, vol. 52, no. 4, p. R2493, 1995.

[76] A. M. Steane, "Error correcting codes in quantum theory," *Physical Review Letters*, vol. 77, no. 5, p. 793, 1996.

[77] I. L. Chuang, N. Gershenfeld, and M. Kubinec, "Experimental implementation of fast quantum searching," *Physical review letters*, vol. 80, no. 15, p. 3408, 1998.

[78] J. A. Jones and M. Mosca, "Implementation of a quantum algorithm on a nuclear magnetic resonance quantum computer," *The Journal of chemical physics*, vol. 109, no. 5, pp. 1648–1653, 1998.

[79] J. Preskill, "Quantum computing and the entanglement frontier," *arXiv preprint arXiv: 1203.5813*, 2012.

[80] ——, "John Preskill Explains 'Quantum Supremacy' (Quanta Magazine)." [Online]. Available: https://www.quantamagazine.org/john-preskill-explains-quantum-supremacy-20191002/

[81] F. Arute *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[82] N. Khammassi, I. Ashraf, J. van Someren, R. Nane, A. Krol, M. A. Rol, L. Lao, K. Bertels, and C. G. Almudever, "OpenQL: a portable quantum programming framework for quantum accelerators," *arXiv preprint arXiv: 2005.13283*, 2020.

[83] A. Paler, A. Zulehner, and R. Wille, "Nisq circuit compilers: search space structure and heuristics," *arXiv preprint arXiv: 1806.07241*, 2018.

[84] F. T. Chong, D. Franklin, and M. Martonosi, "Programming languages and compiler design for realistic quantum hardware," *Nature*, vol. 549, no. 7671, pp. 180–187, 2017.

[85] J. J. Wallman and J. Emerson, "Noise tailoring for scalable quantum computation via randomized compiling," *Physical Review A*, vol. 94, no. 5, p. 052325, 2016.

[86] A. Javadi-Abhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, "Scaffcc: Scalable compilation and analysis of quantum programs," *Parallel Computing*, vol. 45, pp. 2–17, 2015.

[87] G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for nisq-era quantum devices," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 1001–1014.

[88] S. S. Tannu and M. K. Qureshi, "Not all qubits are created equal: a case for variability-aware policies for nisq-era quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 987–999.

[89] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah, "On the qubit routing problem," *arXiv preprint arXiv: 1902.08091*, 2019.

[90] A. Zulehner, A. Paler, and R. Wille, "An efficient methodology for mapping quantum circuits to the ibm qx architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 7, pp. 1226–1236, 2018.

[91] L. Lao, B. van Wee, I. Ashraf, J. van Someren, N. Khammassi, K. Bertels, and C. G. Almudever, "Mapping of lattice surgery-based quantum circuits on surface code architectures," *Quantum Science and Technology*, vol. 4, no. 1, p. 015005, 2018.

[92] M. Y. Siraichi, V. F. d. Santos, S. Collange, and F. M. Q. Pereira, "Qubit allocation," in *Proceedings of the 2018 International Symposium on Code Generation and Optimization*, 2018, pp. 113–125.

[93] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, "Mixed-state entanglement and quantum error correction," *Physical Review A*, vol. 54, no. 5, p. 3824, 1996.

[94] D. G. Cory, M. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. Somaroo, "Experimental quantum error correction," *Physical Review Letters*, vol. 81, no. 10, p. 2152, 1998.

[95] N. Isailovic, M. Whitney, Y. Patel, J. Kubiatowicz, D. Copsey, F. T. Chong, I. L. Chuang, and M. Oskin, "Datapath and control for quantum wires," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 1, no. 1, pp. 34–61, 2004.

[96] R. Li, L. Petit, D. P. Franke, J. P. Dehollain, J. Helsen, M. Steudtner, N. K. Thomas, Z. R. Yoscovits, K. J. Singh, S. Wehner *et al.*, "A crossbar network for silicon quantum dot qubits," *Science advances*, vol. 4, no. 7, p. eaar3960, 2018.

[97] B. Patra, R. M. Incandela, J. P. Van Dijk, H. A. Homulle, L. Song, M. Shahmohammadi, R. B. Staszewski, A. Vladimirescu, M. Babaie, F. Sebastiano *et al.*, "Cryo-cmos circuits and systems for quantum computing applications," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 1, pp. 309–321, 2017.

[98] J. Hornibrook, J. Colless, I. C. Lamb, S. Pauka, H. Lu, A. Gossard, J. Watson, G. Gardner, S. Fallahi, M. Manfra *et al.*, "Cryogenic control architecture for large-scale quantum computing," *Physical Review Applied*, vol. 3, no. 2, p. 024010, 2015.

[99] H. Homulle, S. Visser, B. Patra, G. Ferrari, E. Prati, F. Sebastiano, and E. Charbon, "A reconfigurable cryogenic platform for the classical control of quantum processors," *Review of Scientific Instruments*, vol. 88, no. 4, p. 045103, 2017.

[100] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Physical review letters*, vol. 74, no. 20, p. 4091, 1995.

[101] D. Loss and D. P. DiVincenzo, "Quantum computation with quantum dots," *Physical Review A*, vol. 57, no. 1, p. 120, 1998.

[102] B. E. Kane, "A silicon-based nuclear spin quantum computer," *nature*, vol. 393, no. 6681, pp. 133–137, 1998.

[103] Y. Nakamura, Y. A. Pashkin, and J. S. Tsai, "Coherent control of macroscopic quantum states in a single-cooper-pair box," *nature*, vol. 398, no. 6730, pp. 786–788, 1999.

[104] S. Prawer and A. D. Greentree, "Diamond for quantum computing," *Science*, vol. 320, no. 5883, pp. 1601–1602, 2008.

[105] J. P. Gaebler, T. R. Tan, Y. Lin, Y. Wan, R. Bowler, A. C. Keith, S. Glancy, K. Coakley, E. Knill, D. Leibfried *et al.*, "High-fidelity universal gate set for be 9+ ion qubits," *Physical review letters*, vol. 117, no. 6, p. 060505, 2016.

[106] C. Ballance, T. Harty, N. Linke, M. Sepiol, and D. Lucas, "High-fidelity quantum logic gates using trapped-ion hyperfine qubits," *Physical review letters*, vol. 117, no. 6, p. 060504, 2016.

[107] R. Srinivas, S. Burd, H. Knaack, R. Sutherland, A. Kwiatkowski, S. Glancy, E. Knill, D. Wineland, D. Leibfried, A. C. Wilson *et al.*, "High-fidelity laser-free universal control of trapped ion qubits," *Nature*, vol. 597, no. 7875, pp. 209–213, 2021.

[108] Y. Sung, L. Ding, J. Braumüller, A. Vepsäläinen, B. Kannan, M. Kjaergaard, A. Greene, G. O. Samach, C. McNally, D. Kim *et al.*, "Realization of high-fidelity cz and z z-free iswap gates with a tunable coupler," *Physical Review X*, vol. 11, no. 2, p. 021058, 2021.

[109] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell *et al.*, "Superconducting quantum circuits at the surface code threshold for fault tolerance," *Nature*, vol. 508, no. 7497, pp. 500–503, 2014.

[110] V. Negîrneac, H. Ali, N. Muthusubramanian, F. Battistel, R. Sagastizabal, M. Moreira, J. Marques, W. Vlothuizen, M. Beekman, C. Zachariadis *et al.*, "High-fidelity controlled-z gate with maximal intermediate leakage operating at the speed limit in a superconducting quantum processor," *Physical Review Letters*, vol. 126, no. 22, p. 220502, 2021.

[111] X. Xue, M. Russ, N. Samkharadze, B. Undseth, A. Sammak, G. Scappucci, and L. M. Vandersypen, "Computing with spin qubits at the surface code error threshold," *arXiv preprint arXiv:2107.00628*, 2021.

[112] S. Pezzagna and J. Meijer, "Quantum computer based on color centers in diamond," *Applied Physics Reviews*, vol. 8, no. 1, p. 011308, 2021.

[113] M. T. Mądzik, S. Asaad, A. Youssry, B. Joecker, K. M. Rudinger, E. Nielsen, K. C. Young, T. J. Proctor, A. D. Baczewski, A. Laucht *et al.*, "Precision tomography of a three-qubit electron-nuclear quantum processor in silicon," *arXiv preprint arXiv:2106.03082*, 2021.

[114] B. Náfrádi, M. Choucair, K.-P. Dinse, and L. Forró, "Room temperature manipulation of long lifetime spins in metallic-like carbon nanospheres," *Nature communications*, vol. 7, no. 1, pp. 1–8, 2016.

[115] L. Contamin, T. Cubaynes, W. Legrand, M. Marganska, M. Desjardins, M. Dartiailh, Z. Leghtas, A. Thiaville, S. Rohart, A. Cottet *et al.*, "Staggered spin-orbit interaction in a nanoscale device," *arXiv preprint arXiv:2104.12181*, 2021.

[116] N. Beysengulov, C. Mikolas, J. Kitzman, J. Lane, D. Edmunds, D. Rees, E. Henriksen, S. Lyon, and J. Pollanen, "Helium surface fluctuations investigated with superconducting coplanar waveguide resonator," *Journal of Low Temperature Physics*, pp. 1–10, 2022.

[117] N. W. Hendrickx, W. I. Lawrie, M. Russ, F. van Riggelen, S. L. de Snoo, R. N. Schouten, A. Sammak, G. Scappucci, and M. Veldhorst, "A four-qubit germanium quantum processor," *Nature*, vol. 591, no. 7851, pp. 580–585, 2021.

[118] E. Knill, R. Laflamme, and G. J. Milburn, "A scheme for efficient quantum computation with linear optics," *nature*, vol. 409, no. 6816, pp. 46–52, 2001.

[119] J. McKeever, A. Boca, A. Boozer, R. Miller, J. Buck, A. Kuzmich, and H. Kimble, "Deterministic generation of single photons from one atom trapped in a cavity," *Science*, vol. 303, no. 5666, pp. 1992–1994, 2004.

[120] T. Zhong, J. M. Kindem, E. Miyazono, and A. Faraon, "Nanophotonic coherent light–matter interfaces based on rare-earth-doped crystals," *Nature communications*, vol. 6, no. 1, pp. 1–6, 2015.

[121] A. Politi, M. J. Cryan, J. G. Rarity, S. Yu, and J. L. O'brien, "Silica-on-silicon waveguide quantum circuits," *Science*, vol. 320, no. 5876, pp. 646–649, 2008.

[122] S. Bartolucci, P. Birchall, H. Bombin, H. Cable, C. Dawson, M. Gimeno-Segovia, E. Johnston, K. Kieling, N. Nickerson, M. Pant *et al.*, "Fusion-based quantum computation," *arXiv preprint arXiv:2101.09310*, 2021.

[123] J. E. Bourassa, R. N. Alexander, M. Vasmer, A. Patil, I. Tzitrin, T. Matsuura, D. Su, B. Q. Baragiola, S. Guha, G. Dauphinais *et al.*, "Blueprint for a scalable photonic fault-tolerant quantum computer," *Quantum*, vol. 5, p. 392, 2021.

[124] C. R. Clark, H. N. Tinkey, B. C. Sawyer, A. M. Meier, K. A. Burkhardt, C. M. Seck, C. M. Shappert, N. D. Guise, C. E. Volin, S. D. Fallek *et al.*, "High-fidelity bell-state preparation with ca+ 40 optical qubits," *Physical Review Letters*, vol. 127, no. 13, p. 130505, 2021.

[125] J. P. Home, D. Hanneke, J. D. Jost, J. M. Amini, D. Leibfried, and D. J. Wineland, "Complete methods set for scalable ion trap quantum information processing," *Science*, vol. 325, no. 5945, pp. 1227–1230, 2009.

[126] I. Pogorelov, T. Feldker, C. D. Marciniak, L. Postler, G. Jacob, O. Krieglsteiner, V. Podlesnic, M. Meth, V. Negnevitsky, M. Stadler *et al.*, "Compact ion-trap quantum computing demonstrator," *PRX Quantum*, vol. 2, no. 2, p. 020343, 2021.

[127] D. Hucul, P. Haas, H. J. Rutbeck-Goldman, Z. S. Smith, B. Tabakov, J. A. Williams, C. F. Woodford, K.-A. B. Soderberg, J. E. Christensen, E. R. Hudson *et al.*, "133ba+: High fidelity goldilocks qubits (conference presentation)," in *Quantum Information Science, Sensing, and Computation XII*, vol. 11391. SPIE, 2020, p. 1139107.

[128] D. Ristè, C. Bultink, K. W. Lehnert, and L. DiCarlo, "Feedback control of a solid-state qubit using high-fidelity projective measurement," *Physical review letters*, vol. 109, no. 24, p. 240502, 2012.

[129] R. Versluis, S. Poletto, N. Khammassi, B. Tarasinski, N. Haider, D. J. Michalak, A. Bruno, K. Bertels, and L. DiCarlo, "Scalable quantum circuit and control for a superconducting surface code," *Physical Review Applied*, vol. 8, no. 3, p. 034021, 2017.

[130] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting qubits: Current state of play," *Annual Review of Condensed Matter Physics*, vol. 11, pp. 369–395, 2020.

[131] I. Hansen, A. E. Seedhouse, K. W. Chan, F. Hudson, K. M. Itoh, A. Laucht, A. Saraiva, C. H. Yang, and A. S. Dzurak, "Implementation of the smart protocol for global qubit control in silicon," *arXiv preprint arXiv:2108.00836*, 2021.

[132] A. Zwerver, T. Krähenmann, T. Watson, L. Lampert, H. C. George, R. Pillarisetty, S. Bojarski, P. Amin, S. Amitonov, J. Boter *et al.*, "Qubits made by advanced semiconductor manufacturing," *Nature Electronics*, vol. 5, no. 3, pp. 184–190, 2022.

[133] R. Hanson and D. D. Awschalom, "Coherent manipulation of single spins in semiconductors," *Nature*, vol. 453, no. 7198, pp. 1043–1049, 2008.

[134] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu *et al.*, "Quantum computational advantage using photons," *Science*, vol. 370, no. 6523, pp. 1460–1463, 2020.

[135] S. Pellerano, S. Subramanian, J.-S. Park, B. Patra, T. Mladenov, X. Xue, L. M. Vandersypen, M. Babaie, E. Charbon, and F. Sebastiano, "Cryogenic cmos for qubit control and readout," in *2022 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2022, pp. 01–08.

[136] O. Mukhanov, A. Kirichenko, C. Howington, J. Walter, M. Hutchings, I. Vernik, D. Yohannes, K. Dodge, A. Ballard, B. Plourde *et al.*, "Scalable quantum computing infrastructure based on superconducting electronics," in *2019 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2019, pp. 31–2.

[137] L. Egan, D. M. Debroy, C. Noel, A. Risinger, D. Zhu, D. Biswas, M. Newman, M. Li, K. R. Brown, M. Cetina *et al.*, "Fault-tolerant control of an error-corrected qubit," *Nature*, vol. 598, no. 7880, pp. 281–286, 2021.

[138] L. Postler, S. Heuβen, I. Pogorelov, M. Rispler, T. Feldker, M. Meth, C. D. Marciniak, R. Stricker, M. Ringbauer, R. Blatt *et al.*, "Demonstration of fault-tolerant universal quantum gate operations," *Nature*, vol. 605, no. 7911, pp. 675–680, 2022.

[139] C. Ryan-Anderson, J. Bohnet, K. Lee, D. Gresh, A. Hankin, J. Gaebler, D. Francois, A. Chernoguzov, D. Lucchetti, N. Brown *et al.*, "Realization of real-time fault-tolerant quantum error correction," *Physical Review X*, vol. 11, no. 4, p. 041058, 2021.

[140] Z. Chen, K. J. Satzinger, J. Atalaya, A. N. Korotkov, A. Dunsworth, D. Sank, C. Quintana, M. McEwen, R. Barends, P. V. Klimov *et al.*, "Exponential suppression of bit or phase flip errors with repetitive error correction," *arXiv preprint arXiv:2102.06132*, 2021.

[141] M. Akhtar, F. Bonus, F. Lebrun-Gallagher, N. Johnson, M. Siegele-Brown, S. Hong, S. Hile, S. Kulmiya, S. Weidt, and W. Hensinger, "A high-fidelity quantum matter-link between ion-trap microchip modules," *arXiv preprint arXiv:2203.14062*, 2022.

[142] P. Kurpiers, P. Magnard, T. Walter, B. Royer, M. Pechal, J. Heinsoo, Y. Salathé, A. Akin, S. Storz, J.-C. Besse *et al.*, "Deterministic quantum state transfer and remote entanglement using microwave photons," *Nature*, vol. 558, no. 7709, pp. 264–267, 2018.

[143] L. Stephenson, D. Nadlinger, B. Nichol, S. An, P. Drmota, T. Ballance, K. Thirumalai, J. Goodwin, D. Lucas, and C. Ballance, "High-rate, high-fidelity entanglement of qubits across an elementary quantum network," *Physical review letters*, vol. 124, no. 11, p. 110501, 2020.

[144] S. Ritter, C. Nölleke, C. Hahn, A. Reiserer, A. Neuzner, M. Uphoff, M. Mücke, E. Figueroa, J. Bochmann, and G. Rempe, "An elementary quantum network of single atoms in optical cavities," *Nature*, vol. 484, no. 7393, pp. 195–200, 2012.

[145] J. Hofmann, M. Krug, N. Ortegel, L. Gérard, M. Weber, W. Rosenfeld, and H. Weinfurter, "Heralded entanglement between widely separated atoms," *Science*, vol. 337, no. 6090, pp. 72–75, 2012.

[146] H. Bernien, B. Hensen, W. Pfaff, G. Koolstra, M. S. Blok, L. Robledo, T. H. Taminiau, M. Markham, D. J. Twitchen, L. Childress *et al.*, "Heralded entanglement between solid-state qubits separated by three metres," *Nature*, vol. 497, no. 7447, pp. 86–90, 2013.

[147] A. Delteil, Z. Sun, W.-b. Gao, E. Togan, S. Faelt, and A. Imamoğlu, "Generation of heralded entanglement between distant hole spins," *Nature Physics*, vol. 12, no. 3, pp. 218–223, 2016.

[148] J. I. Cirac, P. Zoller, H. J. Kimble, and H. Mabuchi, "Quantum state transfer and entanglement distribution among distant nodes in a quantum network," *Physical Review Letters*, vol. 78, no. 16, p. 3221, 1997.

[149] S. Daiss, S. Langenfeld, S. Welte, E. Distante, P. Thomas, L. Hartung, O. Morin, and G. Rempe, "A quantum-logic gate between distant quantum-network modules," *Science*, vol. 371, no. 6529, pp. 614–617, 2021.

[150] D. D. Awschalom, R. Hanson, J. Wrachtrup, and B. B. Zhou, "Quantum technologies with optically interfaced solid-state spins," *Nature Photonics*, vol. 12, no. 9, pp. 516–527, 2018.

[151] D. Awschalom, K. K. Berggren, H. Bernien, S. Bhave, L. D. Carr, P. Davids, S. E. Economou, D. Englund, A. Faraon, M. Fejer *et al.*, "Development of quantum interconnects (quics) for next-generation information technologies," *PRX Quantum*, vol. 2, no. 1, p. 017002, 2021.

[152] M. Wallquist, K. Hammerer, P. Rabl, M. Lukin, and P. Zoller, "Hybrid quantum devices and quantum engineering," *Physica Scripta*, vol. 2009, no. T137, p. 014001, 2009.

[153] G. Waldherr, Y. Wang, S. Zaiser, M. Jamali, T. Schulte-Herbrüggen, H. Abe, T. Ohshima, J. Isoya, J. Du, P. Neumann *et al.*, "Quantum error correction in a solid-state hybrid spin register," *Nature*, vol. 506, no. 7487, pp. 204–207, 2014.

[154] Z.-L. Xiang, S. Ashhab, J. You, and F. Nori, "Hybrid quantum circuits: Superconducting circuits interacting with other quantum systems," *Reviews of Modern Physics*, vol. 85, no. 2, p. 623, 2013.

[155] S. Baier, M. Pompili, S. L. Hermans, H. K. Beukers, P. C. Humphreys, R. N. Schouten, R. F. Vermeulen, M. J. Tiggelman, L. dos Santos Martins, B. Dirkse *et al.*, "Realization of a multi-node quantum network of remote solid-state qubits," in *Quantum Information and Measurement*. Optica Publishing Group, 2021, pp. M2A–2.

[156] D. Kielpinski, C. Monroe, and D. J. Wineland, "Architecture for a large-scale ion-trap quantum computer," *Nature*, vol. 417, no. 6890, pp. 709–711, 2002.

[157] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. Allman, C. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer *et al.*, "Demonstration of the trapped-ion quantum ccd computer architecture," *Nature*, vol. 592, no. 7853, pp. 209–213, 2021.

[158] T. S. Metodi, D. D. Thaker, A. W. Cross, F. T. Chong, and I. L. Chuang, "A quantum logic array microarchitecture: Scalable quantum data movement and computation," in *38th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'05)*. IEEE, 2005, pp. 12–pp.

[159] M. G. Whitney, N. Isailovic, Y. Patel, and J. Kubiatowicz, "A fault tolerant, area efficient architecture for shor's factoring algorithm," in *Proceedings of the 36th annual international symposium on Computer architecture*, 2009, pp. 383–394.

[160] M. J. Dousti, A. Shafaei, and M. Pedram, "Squash: a scalable quantum mapper considering ancilla sharing," in *Proceedings of the 24th edition of the great lakes symposium on VLSI*, 2014, pp. 117–122.

[161] M. Ahsan, R. V. Meter, and J. Kim, "Designing a million-qubit quantum computer using a resource performance simulator," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 4, pp. 1–25, 2015.

[162] P. Chapman, ""scaling ionq's quantum computers: The roadmap," *Website: https://ionq. com/posts/december-09-2020-scaling-quantum-computer-roadmap*, 2020.

[163] P. Magnard, S. Storz, P. Kurpiers, J. Schär, F. Marxer, J. Lütolf, T. Walter, J.-C. Besse, M. Gabureac, K. Reuer *et al.*, "Microwave quantum link between superconducting circuits housed in spatially separated cryogenic systems," *Physical Review Letters*, vol. 125, no. 26, p. 260502, 2020.

[164] A. Gold, J. Paquette, A. Stockklauser, M. J. Reagor, M. S. Alam, A. Bestwick, N. Didier, A. Nersisyan, F. Oruc, A. Razavi *et al.*, "Entanglement across separate silicon dies in a modular superconducting qubit device," *npj Quantum Information*, vol. 7, no. 1, pp. 1–10, 2021.

[165] N. LaRacuente, K. N. Smith, P. Imany, K. L. Silverman, and F. T. Chong, "Short-range microwave networks to scale superconducting quantum computation," *arXiv preprint arXiv:2201.08825*, 2022.

[166] V. Kliuchnikov, A. Bocharov, M. Roetteler, and J. Yard, "A framework for approximating qubit unitaries," *arXiv preprint arXiv:1510.03888*, 2015.

[167] N. J. Ross and P. Selinger, "Optimal ancilla-free clifford+ t approximation of z-rotations." *Quantum Inf. Comput.*, vol. 16, no. 11&12, pp. 901–953, 2016.

[168] A. Bocharov, M. Roetteler, and K. M. Svore, "Efficient synthesis of probabilistic quantum circuits with fallback," *Physical Review A*, vol. 91, no. 5, p. 052317, 2015.

[169] G. Watkins, H. Nguyen, H.-K. Lau, A. Paler, S. Pearce, R. Raussendorf, V. Seshadri, and K. Watkins, "Lattice surgery quantum error correction compiler," *Bulletin of the American Physical Society*, 2022.

[170] X. Xu, S. C. Benjamin, and X. Yuan, "Variational circuit compiler for quantum error correction," *Physical Review Applied*, vol. 15, no. 3, p. 034068, 2021.

[171] M. J. Dousti and M. Pedram, "Minimizing the latency of quantum circuits during mapping to the ion-trap circuit fabric," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2012, pp. 840–843.

[172] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, "Optimized compiler for distributed quantum computing," *ACM Transactions on Quantum Computing*, vol. 4, no. 2, feb 2023. [Online]. Available: https://doi.org/10.1145/3579367

[173] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, "Quipper: a scalable quantum programming language," in *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation*, 2013, pp. 333–342.

[174] D. Wecker and K. M. Svore, "Liqui|>: A software design architecture and domain-specific language for quantum computing," *arXiv preprint arXiv:1402.4467*, 2014.

[175] D. S. Steiger, T. Häner, and M. Troyer, "Projectq: an open source software framework for quantum computing," *Quantum*, vol. 2, p. 49, 2018.

[176] K. M. Svore, A. V. Aho, A. W. Cross, I. Chuang, and I. L. Markov, "A layered software architecture for quantum computing design tools," *Computer*, vol. 39, no. 1, pp. 74–83, 2006.

[177] N. Khammassi, G. G. Guerreschi, I. Ashraf, J. W. Hogaboam, C. G. Almudever, and K. Bertels, "cqasm v1. 0: Towards a common quantum assembly language," *arXiv preprint arXiv:1805.09607*, 2018.

[178] X. Fu, L. Riesebos, M. Rol, J. Van Straten, J. Van Someren, N. Khammassi, I. Ashraf, R. Vermeulen, V. Newsum, K. Loh *et al.*, "eqasm: An executable quantum instruction set architecture," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019, pp. 224–237.

[179] A. W. Cross, L. S. Bishop, J. A. Smolin, and J. M. Gambetta, "Open quantum assembly language," *arXiv preprint arXiv:1707.03429*, 2017.

[180] A. Cross, A. Javadi-Abhari, T. Alexander, N. de Beaudrap, L. S. Bishop, S. Heidel, C. A. Ryan, P. Sivarajah, J. Smolin, J. M. Gambetta *et al.*, "Openqasm 3: A broader and deeper quantum assembly language," *ACM Transactions on Quantum Computing*, 2021.

[181] I. Buluta, S. Ashhab, and F. Nori, "Natural and artificial atoms for quantum computation," *Reports on Progress in Physics*, vol. 74, no. 10, p. 104401, 2011.

[182] P. Gerbert and F. Rueß, "The next decade in quantum computing and how to play," *Boston Consulting Group, November*, 2018.

[183] S. Harvey, "Quantum dots/spin qubits," *arXiv preprint arXiv:2204.04261*, 2022.

[184] J. Wang, F. Sciarrino, A. Laing, and M. G. Thompson, "Integrated photonic quantum technologies," *Nature Photonics*, vol. 14, no. 5, pp. 273–284, 2020.

[185] J. Gambetta, "IBM's roadmap for scaling quantum technology (IBM Research Blog)." [Online]. Available: https://research.ibm.com/blog/ibm-quantum-roadmap

[186] M. Maronese, L. Moro, L. Rocutto, and E. Prati, "Quantum compiling," in *Quantum Computing Environments*. Springer, 2022, pp. 39–74.

[187] E. Kang, E. Jackson, and W. Schulte, "An approach for effective design space exploration," in *Monterey Workshop*. Springer, 2010, pp. 33–54.

[188] M. Gries, "Methods for evaluating and covering the design space during early design development," *Integration*, vol. 38, no. 2, pp. 131–183, 2004.

[189] C. Araguz López, "In pursuit of autonomous distributed satellite systems," Ph.D. dissertation, Universitat Politècnica de Catalunya, 2019.

[190] G. W. Torrance, M. H. Boyle, and S. P. Horwood, "Application of multi-attribute utility theory to measure social preferences for health states," *Operations research*, vol. 30, no. 6, pp. 1043–1069, 1982.

[191] F. T. Chong, "Technical perspective: Applying design-space exploration to quantum architectures," *Communications of the ACM*, vol. 65, no. 3, pp. 100–100, 2022.

[192] M. Soeken, M. Roetteler, N. Wiebe, and G. De Micheli, "Design automation and design space exploration for quantum computers," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. Ieee, 2017, pp. 470–475.

[193] M. Alam, A. Ash-Saki, and S. Ghosh, "Design-space exploration of quantum approximate optimization algorithm under noise," in *2020 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2020, pp. 1–4.

[194] P. Niemann, A. A. de Almeida, G. Dueck, and R. Drechsler, "Design space exploration in the mapping of reversible circuits to ibm quantum computers," in *2020 23rd Euromicro Conference on Digital System Design (DSD)*. IEEE, 2020, pp. 401–407.

[195] H. Zarein, "Design space exploration for mapping in quantum computers," Master's thesis, Universitat Politècnica de Catalunya, 2020.

[196] M. Bandic, H. Zarein, E. Alarcon, and C. G. Almudever, "On structured design space exploration for mapping of quantum algorithms," in *2020 XXXV conference on design of circuits and integrated systems (DCIS)*. IEEE, 2020, pp. 1–6.

[197] S. T. Flammia and Y.-K. Liu, "Direct fidelity estimation from few pauli measurements," *Physical review letters*, vol. 106, no. 23, p. 230501, 2011.

[198] E. Nielsen, J. K. Gamble, K. Rudinger, T. Scholten, K. Young, and R. Blume-Kohout, "Gate set tomography," *Quantum*, vol. 5, p. 557, 2021.

[199] G. M. D'Ariano, M. G. Paris, and M. F. Sacchi, "Quantum tomography," *Advances in Imaging and Electron Physics*, vol. 128, pp. 206–309, 2003.

[200] M. O. Scully and M. Zubairy, "Quantum optical implementation of grover's algorithm," *Proceedings of the National Academy of Sciences*, vol. 98, no. 17, pp. 9490–9493, 2001.

[201] A. G. Fowler, S. J. Devitt, and L. C. Hollenberg, "Implementation of shor's algorithm on a linear nearest neighbour qubit array," *arXiv preprint quant-ph/0402196*, 2004.

[202] J. Emerson, R. Alicki, and K. Życzkowski, "Scalable noise estimation with random unitary operators," *Journal of Optics B: Quantum and Semiclassical Optics*, vol. 7, no. 10, p. S347, 2005.

[203] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, "Randomized benchmarking of quantum gates," *Physical Review A*, vol. 77, no. 1, p. 012307, 2008.

[204] L. S. Bishop, S. Bravyi, A. Cross, J. M. Gambetta, and J. Smolin, "Quantum volume," *Quantum Volume. Technical Report*, 2017.

[205] A. J. McCaskey, Z. P. Parks, J. Jakowski, S. V. Moore, T. D. Morris, T. S. Humble, and R. C. Pooser, "Quantum chemistry as a benchmark for near-term quantum computers," *npj Quantum Information*, vol. 5, no. 1, pp. 1–8, 2019.

[206] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt, "Characterizing large-scale quantum computers via cycle benchmarking," *Nature communications*, vol. 10, no. 1, pp. 1–7, 2019.

[207] D. Mills, S. Sivarajah, T. L. Scholten, and R. Duncan, "Application-motivated, holistic benchmarking of a full quantum computing stack," *arXiv preprint arXiv: 2006.01273*, 2020.

[208] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," *Physical Review A*, vol. 100, no. 3, p. 032328, 2019.

[209] R. Blume-Kohout and K. C. Young, "A volumetric framework for quantum computer benchmarks," *arXiv preprint arXiv: 1904.05546*, 2019.

[210] A. Wack, H. Paik, A. Javadi-Abhari, P. Jurcevic, I. Faro, J. M. Gambetta, and B. R. Johnson, "Quality, speed, and scale: three key attributes to measure the performance of near-term quantum computers," *arXiv preprint arXiv:2110.14108*, 2021.

[211] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, no. 5886, pp. 802–803, 1982.

[212] D. Dieks, "Communication by epr devices," *Physics Letters A*, vol. 92, no. 6, pp. 271–272, 1982.

[213] A. Ross, M. G. O'Neill, D. Hastings, and D. Rhodes, "Aligning perspectives and methods for value-driven design," in *AIAA Space 2010 Conference & Exposition*, 2010, p. 8797.

[214] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[215] A. S. Cacciapuoti, M. Caleffi, R. Van Meter, and L. Hanzo, "When entanglement meets classical communications: Quantum teleportation for the quantum internet," *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3808–3833, 2020.

[216] D. Chandra, M. Caleffi, and A. S. Cacciapuoti, "The entanglement-assisted communication capacity over quantum trajectories," *IEEE Transactions on Wireless Communications*, 2021.

[217] A. Ambainis, "A new protocol and lower bounds for quantum coin flipping," *Journal of Computer and System Sciences*, vol. 68, no. 2, pp. 398–416, 2004.

[218] J. F. Fitzsimons, "Private quantum computation: an introduction to blind quantum computing and related protocols," *npj Quantum Information*, vol. 3, no. 1, pp. 1–11, 2017.

[219] D. Sánchez, G. Michelogiannakis, and C. Kozyrakis, "An Analysis of On-Chip Interconnection Networks for Large-Scale Chip Multiprocessors," *ACM Transactions on Architecture and Code Optimization*, vol. 7, no. 1, p. Article 4, 2010.

[220] M. Schoeberl, F. Brandner, J. Sparsø, and E. Kasapaki, "A statically scheduled time-division-multiplexed network-on-chip for real-time systems," in *2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*. IEEE, 2012, pp. 152–160.

[221] S. Pal, D. Petrisko, M. Tomei, P. Gupta, S. S. Iyer, and R. Kumar, "Architecting waferscale processors-a gpu case study," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019, pp. 250–263.

[222] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, "Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels," *Physical review letters*, vol. 70, no. 13, p. 1895, 1993.

[223] D. Bouwmeester, J.-W. Pan, K. Mattle, M. Eibl, H. Weinfurter, and A. Zeilinger, "Experimental quantum teleportation," *Nature*, vol. 390, no. 6660, pp. 575–579, 1997.

[224] J.-G. Ren, P. Xu, H.-L. Yong, L. Zhang, S.-K. Liao, J. Yin, W.-Y. Liu, W.-Q. Cai, M. Yang, L. Li *et al.*, "Ground-to-satellite quantum teleportation," *Nature*, vol. 549, no. 7670, pp. 70–73, 2017.

[225] R. Valivarthi, S. I. Davis, C. Peña, S. Xie, N. Lauk, L. Narváez, J. P. Allmaras, A. D. Beyer, Y. Gim, M. Hussein *et al.*, "Teleportation systems toward a quantum internet," *PRX Quantum*, vol. 1, no. 2, p. 020317, 2020.

[226] W. Kozlowski, A. Dahlberg, and S. Wehner, "Designing a quantum network protocol," in *Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies*, 2020, pp. 1–16.

[227] D. Hucul, I. V. Inlek, G. Vittorini, C. Crocker, S. Debnath, S. M. Clark, and C. Monroe, "Modular entanglement of atomic qubits using photons and phonons," *Nature Physics*, vol. 11, no. 1, pp. 37–42, 2015.

[228] C. Dickel, J. Wesdorp, N. Langford, S. Peiter, R. Sagastizabal, A. Bruno, B. Criger, F. Motzoi, and L. DiCarlo, "Chip-to-chip entanglement of transmon qubits using engineered measurement fields," *Physical Review B*, vol. 97, no. 6, p. 064508, 2018.

[229] G. Giambene, *Queuing theory and telecommunications*. Springer, 2014, vol. 585.

[230] V. Soteriou, H. Wang, and L. Peh, "A statistical traffic model for on-chip interconnection networks," in *14th IEEE International Symposium on Modeling, Analysis, and Simulation*. IEEE, 2006, pp. 104–116.

[231] N. Barrow-Williams, C. Fensch, and S. Moore, "A communication characterisation of splash-2 and parsec," in *2009 IEEE international symposium on workload characterization (IISWC)*. IEEE, 2009, pp. 86–97.

[232] P. Bogdan and R. Marculescu, "Non-stationary traffic analysis and its implications on multicore platform design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 4, pp. 508–519, 2011.

[233] P. Gratz and S. W. Keckler, "Realistic workload characterization and analysis for networks-on-chip design," in *The 4th workshop on chip multiprocessor memory systems and interconnects (CMP-MSI)*. Citeseer, 2010, pp. 1–10.

[234] S. Abadal, A. Mestres, R. Martinez, E. Alarcon, A. Cabellos-Aparicio, and R. Martinez, "Multicast on-chip traffic analysis targeting manycore noc design," in *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 2015, pp. 370–378.

[235] S. Abadal, R. Martínez, J. Solé-Pareta, E. Alarcón, and A. Cabellos-Aparicio, "Characterization and modeling of multicast communication in cache-coherent manycore processors," *Computers & Electrical Engineering*, vol. 51, pp. 168–183, 2016.

[236] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," *arXiv:1811.12926*, 2018.

[237] J. Bylander, S. Gustavsson, F. Yan, F. Yoshihara, K. Harrabi, G. Fitch, D. G. Cory, Y. Nakamura, J.-S. Tsai, and W. D. Oliver, "Dynamical decoupling and noise spectroscopy with a superconducting flux qubit," *arXiv preprint arXiv:1101.4707*, 2011.

[238] S. Welte, B. Hacker, S. Daiss, S. Ritter, and G. Rempe, "Photon-mediated quantum gate between two neutral atoms in an optical cavity," *Physical Review X*, vol. 8, no. 1, p. 011018, 2018.

[239] S. Herbert and A. Sengupta, "Using reinforcement learning to find efficient qubit routing policies for deployment in near-term quantum computers," *arXiv preprint arXiv: 1812.11619*, 2018.

[240] M. G. Pozzi, S. J. Herbert, A. Sengupta, and R. D. Mullins, "Using reinforcement learning to perform qubit routing in quantum compilers," *arXiv preprint arXiv: 2007.15957*, 2020.

[241] S. Sivarajah, S. Dilkes, A. Cowtan, W. Simmons, A. Edgington, and R. Duncan, "t|ket>: A retargetable compiler for nisq devices," *Quantum Science and Technology*, 2020.

[242] Y. Lu, A. Bengtsson, J. J. Burnett, E. Wiegand, B. Suri, P. Krantz, A. F. Roudsari, A. F. Kockum, S. Gasparinetti, G. Johansson *et al.*, "Characterizing decoherence rates of a superconducting qubit by direct microwave scattering," *npj Quantum Information*, vol. 7, no. 1, p. 35, 2021.

[243] M. Bandic, L. Prielinger, J. Nüßlein, A. Ovide, S. Rodrigo, S. Abadal, H. van Someren, G. Vardoyan, E. Alarcon, C. G. Almudever *et al.*, "Mapping quantum circuits to modular architectures with qubo," *arXiv preprint arXiv:2305.06687*, 2023.