OPTIMIZING QUANTUM SYSTEMS–AN OVERVIEW

Chapter 4

Yongshan Ding and Frederic T. Chong
Speaker: Mariana M G Duarte

OUTLINE

- **OVERVIEW**
- 4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS
- 4.2 QUANTUM-CLASSICAL CO-PROCESSING
- 4.3 QUANTUM COMPILING
- 4.4 NISQ VS. FT MACHINES

OVERVIEW

- Key idea: **optimizing** quantum computing at a **systems** level
- This chapter describes the **layers** of a quantum computer system
- Remarkable developments: theory of quantum algorithms and the implementation of quantum hardware in the past few years
- But there are still **formidable challenges** lying ahead

OVERVIEW

- **Enormous gap**: resources **required** by the algorithms, and resources **available** today
- Need to learn to execute **large** quantum algorithms under **highly-constrained** conditions
- **Very important:** optimize for the resource consumption and success rate of a quantum program
  - Via sharing of information throughout the software-hardware stack
  - For example: this information can be the characteristics of the target application and the underlying hardware

OVERVIEW

- A **big part** of Quantum computer systems research in the NISQ (noisy intermediate-scale quantum computer) era will be focused in **vertical integration** across the systems layers (software- hardware co-design)
- A family of techniques across **many layers** will be needed
- Each and every optimization will play a vital role in enabling **practical** quantum computing
- Indeed, this is the emphasis of the book

OUTLINE

- OVERVIEW
- **4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS**
- 4.2 QUANTUM-CLASSICAL CO-PROCESSING
- 4.3 QUANTUM COMPILING
- 4.4 NISQ VS. FT MACHINES

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- Key components in QC Systems
- “Quantum computing is at a similar stage of development as classical computing in the 1950s”
- Today’s QC systems consist **three layers** in quantum computer architecture:
  - application layer
  - systems software layer
  - hardware layer

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- application layer
- systems software layer
- hardware layer

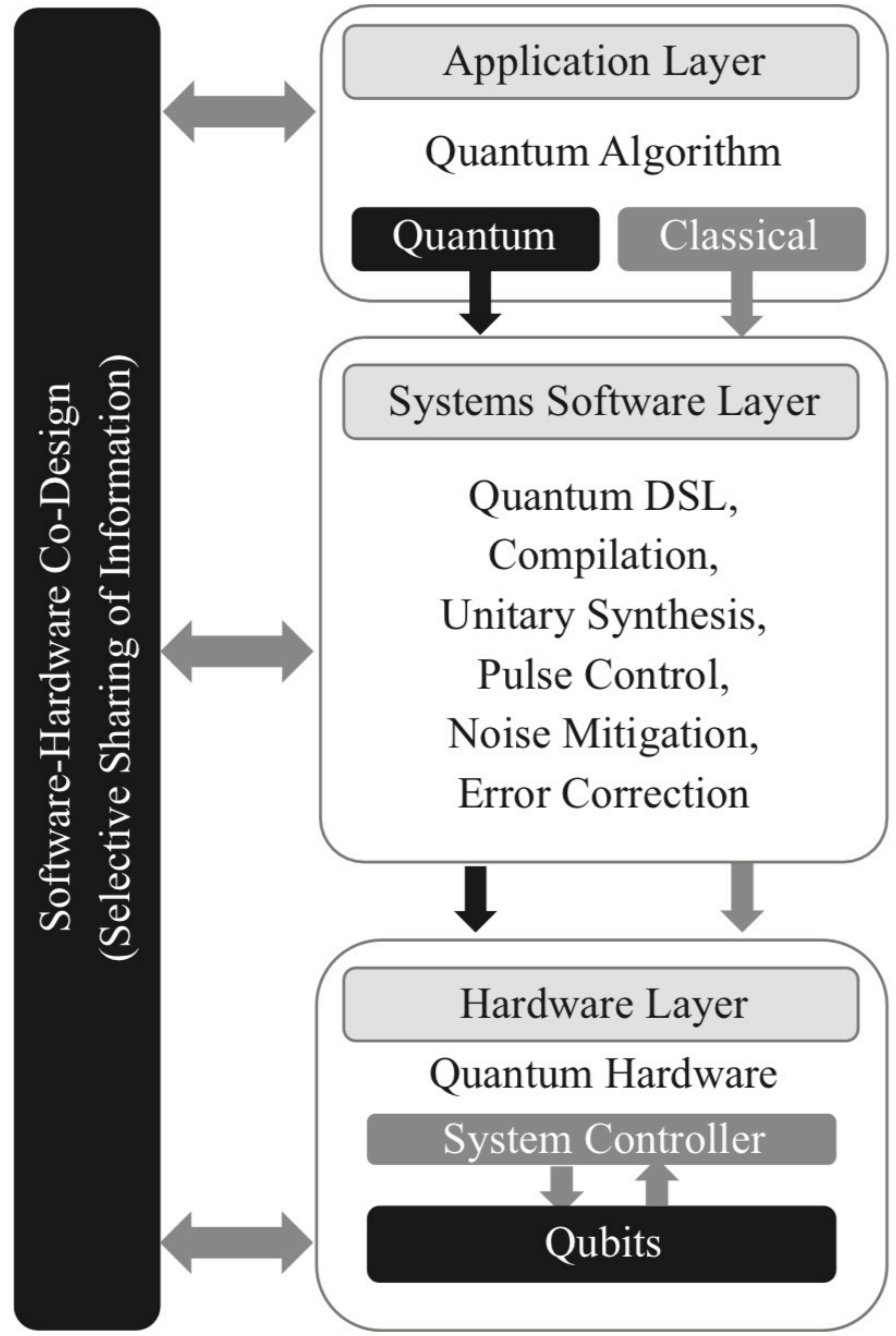


Figure 4.1: Selective sharing of information allows algorithms to use limited resource in NISQ hardware most efficiently.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- **application layer**
- systems software layer
- hardware layer

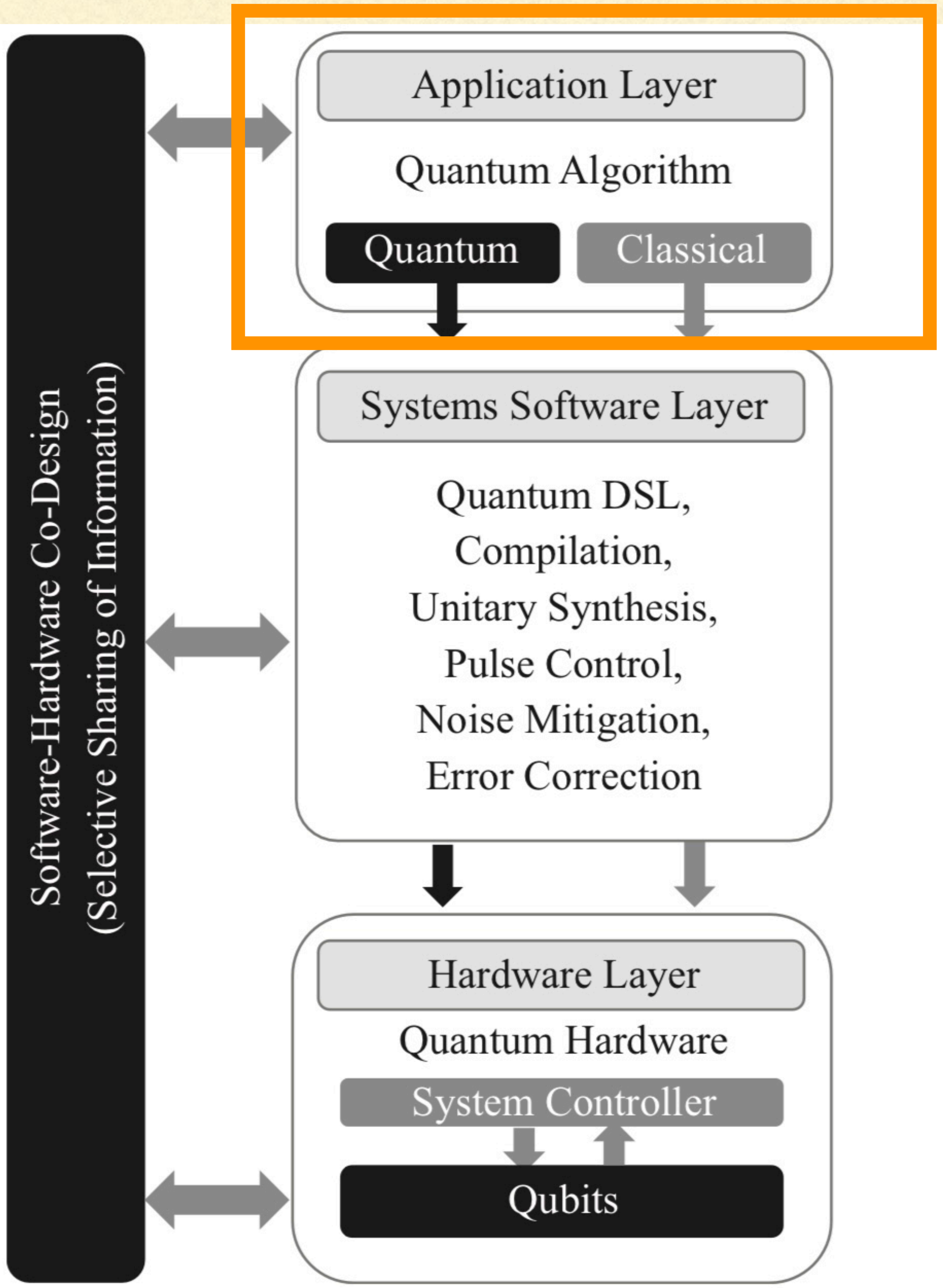


Figure 4.1: Selective sharing of information allows algorithms to use limited resource in NISQ hardware most efficiently.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- **application layer**
- **systems software layer**
- hardware layer

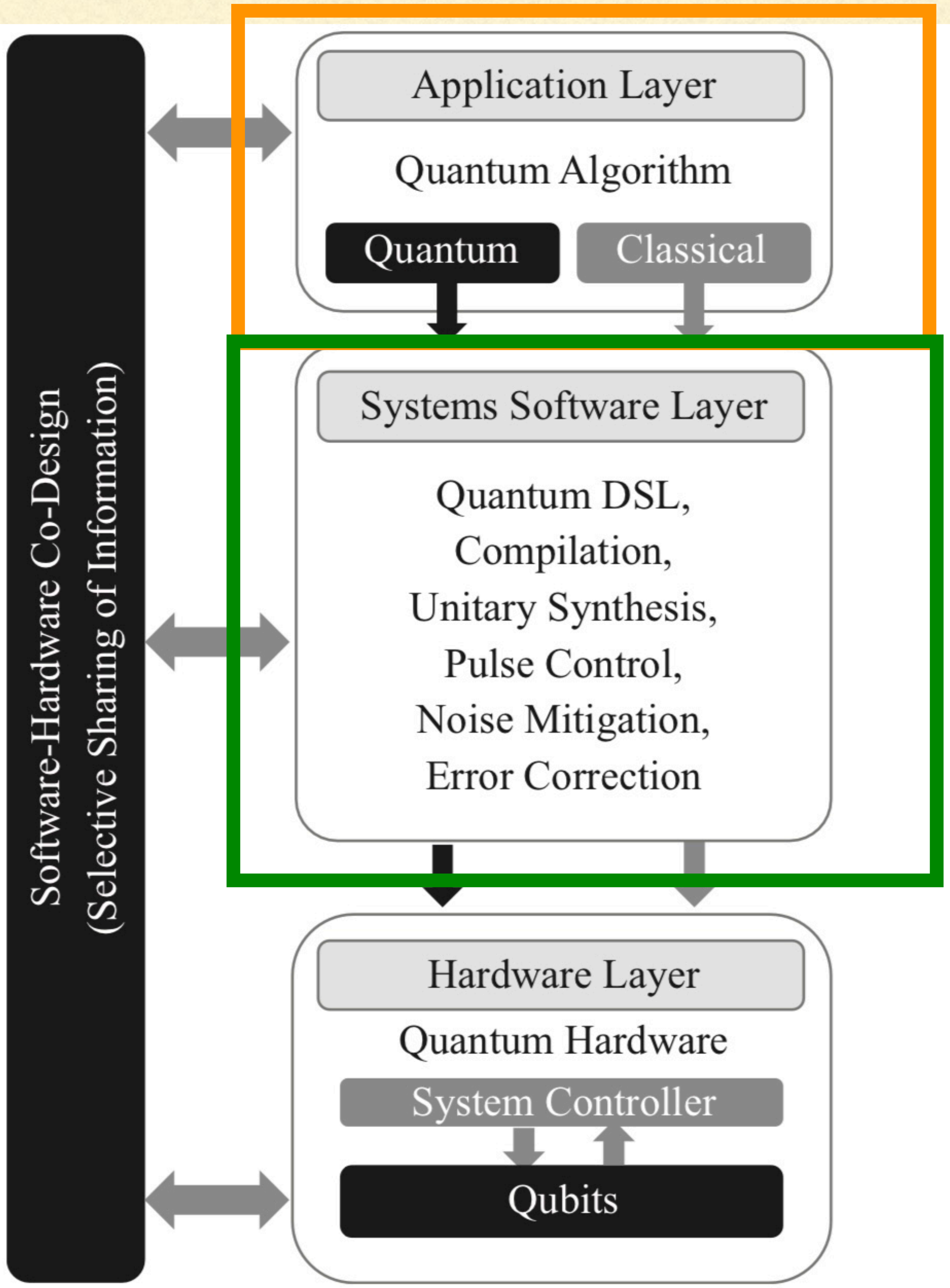


Figure 4.1: Selective sharing of information allows algorithms to use limited resource in NISQ hardware most efficiently.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- **application layer**
- **systems software layer**
- **hardware layer**

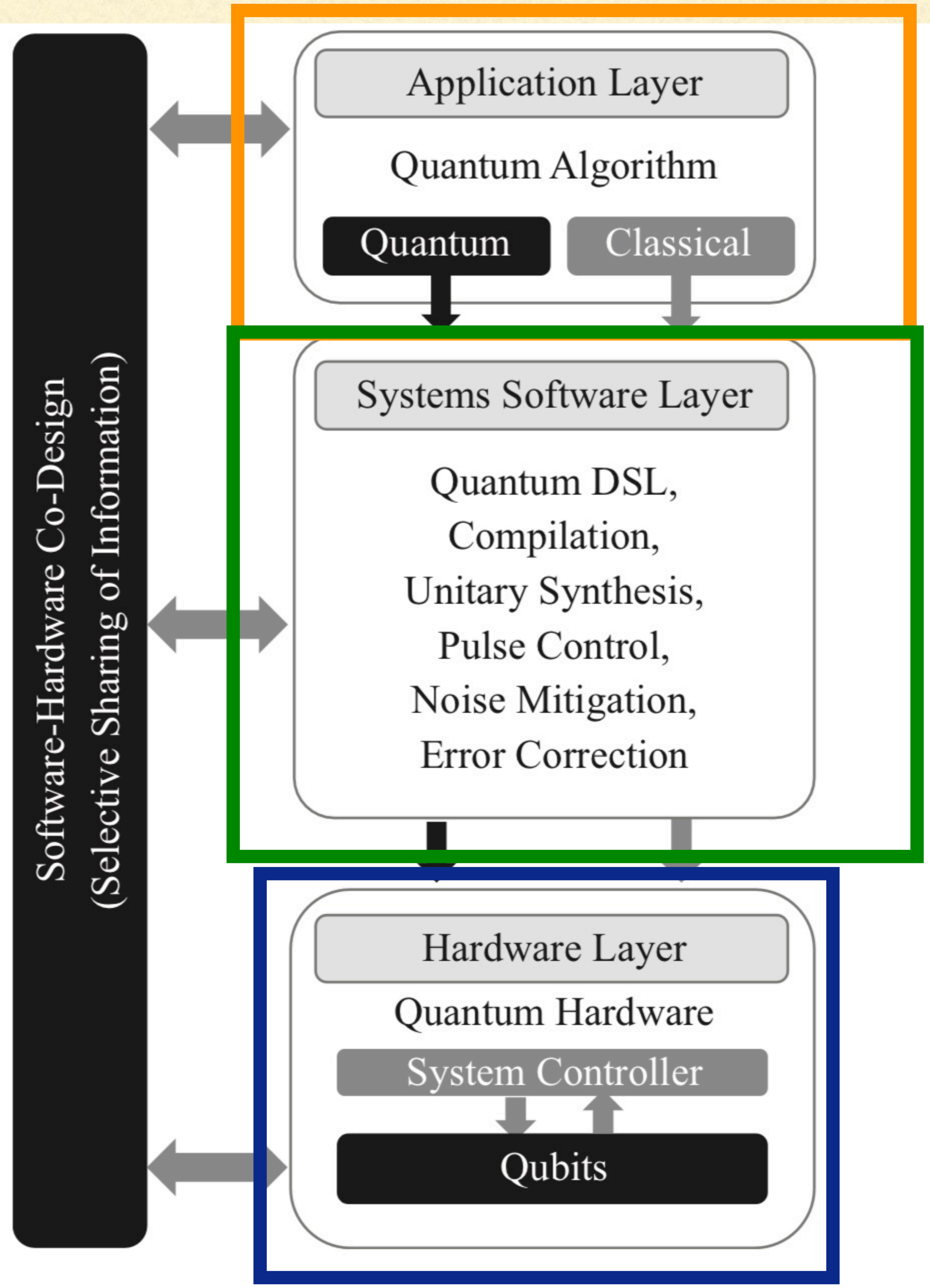


Figure 4.1: Selective sharing of information allows algorithms to use limited resource in NISQ hardware most efficiently.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- application layer
- systems software layer
- hardware layer

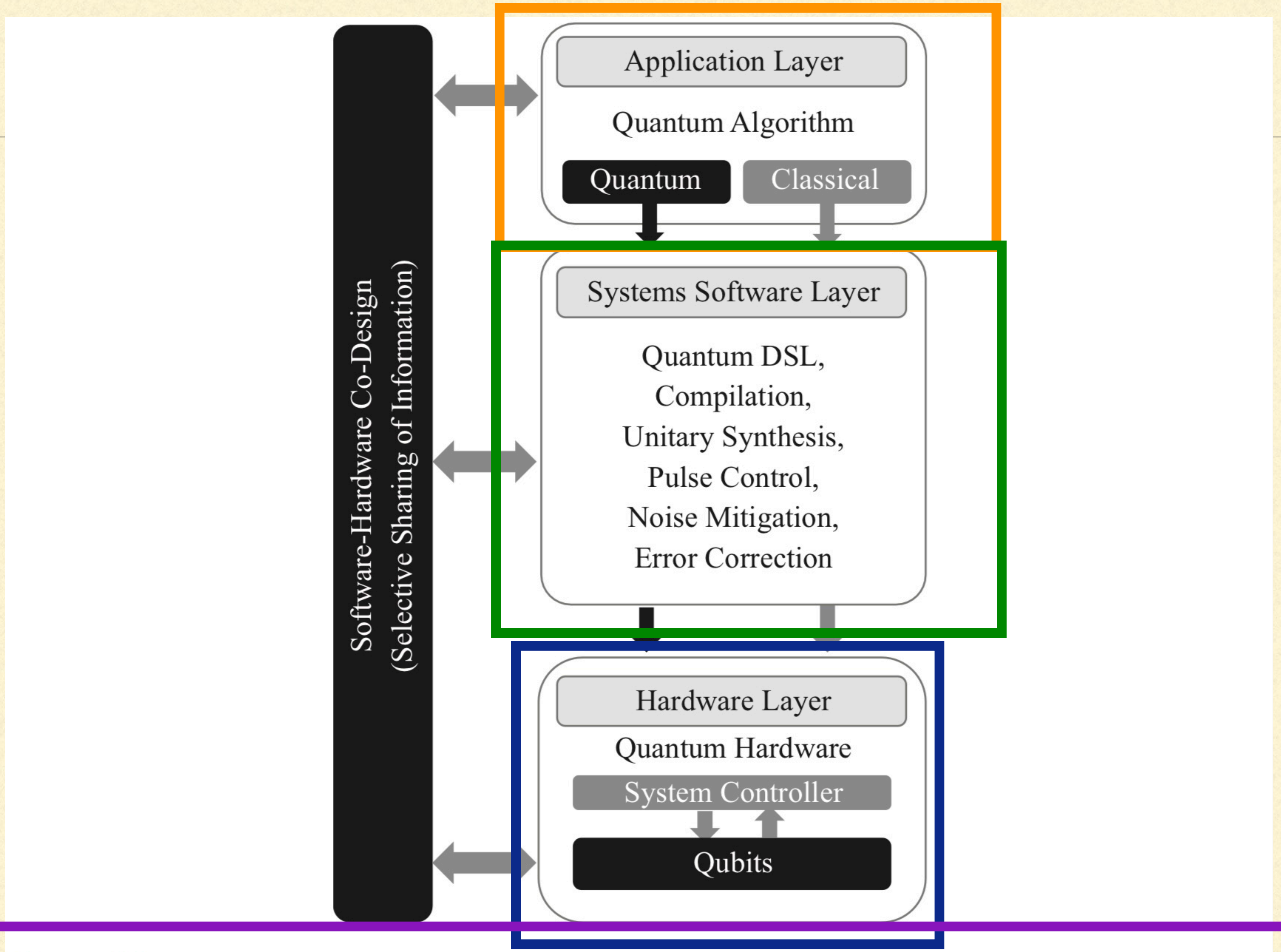


Figure 4.1: Selective sharing of information allows algorithms to use limited resource in NISQ hardware most efficiently.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- Today's **classical** computer systems manage hardware and software through **layering abstractions**
- Each layer **hides** some **implementation details** and expose a manageable set of controls for the next layer
- In contrast, the development of quantum computer systems is still at its nascent stage
  - This means that **resources** are **very scarce**
  - Researchers are motivated to break abstractions and **pay for efficiency** with **greater software complexity**

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- Even **classical computing** is backsliding a bit toward **less abstraction** as the end of Dennard scaling* puts **pressure** on architectures to become more **efficient**
- A **functional** quantum computer requires a enormous amount of attention to the isolation and control over many qubits
- The experience and lessons we learn about how to manipulate qubits in NISQ computers, will pave the way for **larger fault-tolerant** quantum devices in the future

* Dennard Scaling suggested that as transistors get smaller their power density stays constant, so that the power use stays in proportion with area.

4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS

- It is expected that, in the **NISQ era**
  - A QC toolchain must **break** the traditional **abstraction layers**
  - Use **aggressive optimizations** throughout the **full** systems stack

- The key to successful execution of quantum algorithms on NISQ devices is to **selectively share information** across layers of the stack such that programs can use the limited qubits **most efficiently**

OUTLINE

- OVERVIEW
- 4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS
- **4.2 QUANTUM-CLASSICAL CO-PROCESSING**
- 4.3 QUANTUM COMPILING
- 4.4 NISQ VS. FT MACHINES

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- An **important variation** of quantum computing systems is their use as s**pecialized hardware accelerators** within a classical computation
- This hybrid co-processing approach will **likely be** the **dominant structure** of quantum systems for the foreseeable future
- While **quantum computers** are currently **small and unreliable**, a great way to exploit their abilities is to adopt a hybrid model which leverages both quantum and classical computation

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- Almost all useful algorithms require some amount of **classical pre-processing or post-processing**
- Most promising example is in quantum chemistry, where Variational Quantum Eigensolver (VQE) algorithms perform a kind of **heuristic search** by i**terating between a quantum machine and a classical supercomputer**

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- We start from the best-known **configuration of electrons from a classical** computer and **estimate the energy** of that configuration using the **quantum machine**
- This estimate is then given back to a **classical** computer to guide its search toward a configuration with lower energy
- In this way, the **quantum machine** acts as an **accelerator** for the energy modeling part of the computation

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- Great advantages in co-processing:
  - First, it **avoids the "innovator's dilemma"** by leveraging an initial guess derived from classical technology, **rather than directly competing** with that technology

  - Second, **hybrid algorithms** break a long program into **multiple iterations of short programs**, which allows us to effectively utilize the limited number of instructions a quantum machine can reliably execute

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- Third, it allows us to pick **classically challenging problems** (ex: chemical compounds)
  - In order to determine which orbitals the electrons are in, Nature only uses n electrons to “model” **n** electrons, classical computers require combinatorially $\mathbf{k^n}$ bits, but quantum computers only need **kn** qubits

- Fourth, classically-computed ground state energy can be significantly higher than experimentally-observed values, even for small compounds
  - If our hybrid approach can get **closer to experimental values,** then the quantum machine compute something not computable classically!

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- Even as quantum machines scale, quantum algorithms are **likely to be specialized**, making the **quantum device a very domain-specific accelerator**
- Most **practical** applications will still require a combination of general classical and specialized quantum processing to be useful
- Traditional quantum algorithms can be statically compiled with a high level of optimization using known input parameters

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- With **hybrid algorithms**, some of a quantum program's input parameters **can change** each iteration
  - For example, a compiler may spend hours optimizing for quantum instructions that include **quantum rotations** for specific input angles to solve a chemistry problem, but now we find that the angles change every iteration

- This suggests that we need a **partial compilation strategy** in which programs are optimized for unchanging parameters, but then **quickly re-optimized** each iteration for parameters that change

4.2 QUANTUM-CLASSICAL CO-PROCESSING

- Hybrid algorithms also require more thought to be given to **hardware and software communication mechanisms** between quantum and classical hardware
- IBM was the first to make a physical quantum machine accessible via the cloud
- The IBM machines, however, are **heavy for hybrid computation**, as the batch queue interface is really designed for stand-alone quantum programs and the latency to couple with classical computation is long

OUTLINE

- OVERVIEW
- 4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS
- 4.2 QUANTUM-CLASSICAL CO-PROCESSING
- **4.3 QUANTUM COMPILING**
- 4.4 NISQ VS. FT MACHINES

4.3 QUANTUM COMPILING

- A quantum compiler aims to **efficiently express a high-level quantum program** using instructions that a **quantum machine recognizes and natively supports**, balancing practical architectural constraints

- A quantum algorithm is implemented in a **quantum domain-specific language** (QDSL)

- Translates the high-level program into **quantum assembly code** (QASM)
  - Accomplished with a series of **transformations and optimizations** on a **quantum intermediate representation** (QIR) of a program

4.3 QUANTUM COMPILING

- A number of **architectural constraints** must be satisfied:
  - **Instruction set**:
    - There are **some** quantum instructions that are supported
    - In most cases, this instruction set is **“Clifford+T” gates**, that consists of
      - CNOT (controlled-NOT) gate
      - X (NOT) gate
      - H (Hadamard) gate
      - T gate
    - Common set for most gate-based NISQ machines, and large-scale FT machines
    - Some **NISQ compilers** choose to **target directly** the physical analog pulses for improved hardware control

4.3 QUANTUM COMPILING

- **Qubit communication**
  - A quantum algorithm is hardly interesting if it can be implemented with only single-qubit gates, as **two-qubit** gates (or **multi-qubit gates**) provides the entangling power between qubits
  - **Two-qubit** gates are implemented by **qubit-qubit interaction/communication**
    - Qubit communication has different meanings in the NISQ vs. the FT contexts

4.3 QUANTUM COMPILING

- In a **NISQ** machine, not all qubits can **directly interact** with each other, two qubits **interact by moving closer** to one another via a chain of swap gates until they are directly connected hence allowed to interact
    - The time to complete a swap chain is proportional to the length of the chain

- In **FT** machines, qubit interactions are accomplished through **fault-tolerant operations** depending on the error correcting codes

4.3 QUANTUM COMPILING

- With today's technology, building large-scale quantum machines with all-to-all qubit **connectivity** is shown to be **extremely challenging**
- The latest effort from IonQ offers a machine with **eleven fully connected** qubits using trapped-ion technology
- Superconducting machines, for instance by IBM and Rigetti, typically have **much lower connectivity**

4.3 QUANTUM COMPILING

- **Hardware noise**
  - **Minimize errors** caused by hardware noise
  - Typically include memory errors (caused by decoherence of qubits) and gate errors (caused by imprecise control of gates)
  - In general, the **longer** the program runs, the **higher** the chance that the qubits experience **decoherence**
  - The **more gates** are applied, the **lower the chance** that the program succeeds at the end

4.3 QUANTUM COMPILING

- In today's technology, a **two-qubit gate** proves be challenging, hence it is one of the **dominant sources of error**

- A **compiler** normally aims to express a quantum program in
  - fewer qubits
  - fewer number of gates
  - shorter circuit depth

- More advanced **noise-aware compilers** have also been proposed
  - In **NISQ** machines, some qubits are more **robust** then others, so picking the longer-lived qubits to perform important computation can **improve** the overall success rate

4.3 QUANTUM COMPILING

- **Available parallel control**
  - Depending on the technology that implements the qubits, a compiler can be **constrained by the available parallelism**
  - The parallelism limitation is usually the consequence of **hardware control** mechanism, or error mitigation protocols
  - Some error mitigation protocols dictate that no **parallel gates** are allowed when they are **physically located** close to each other, reducing crosstalk errors between them

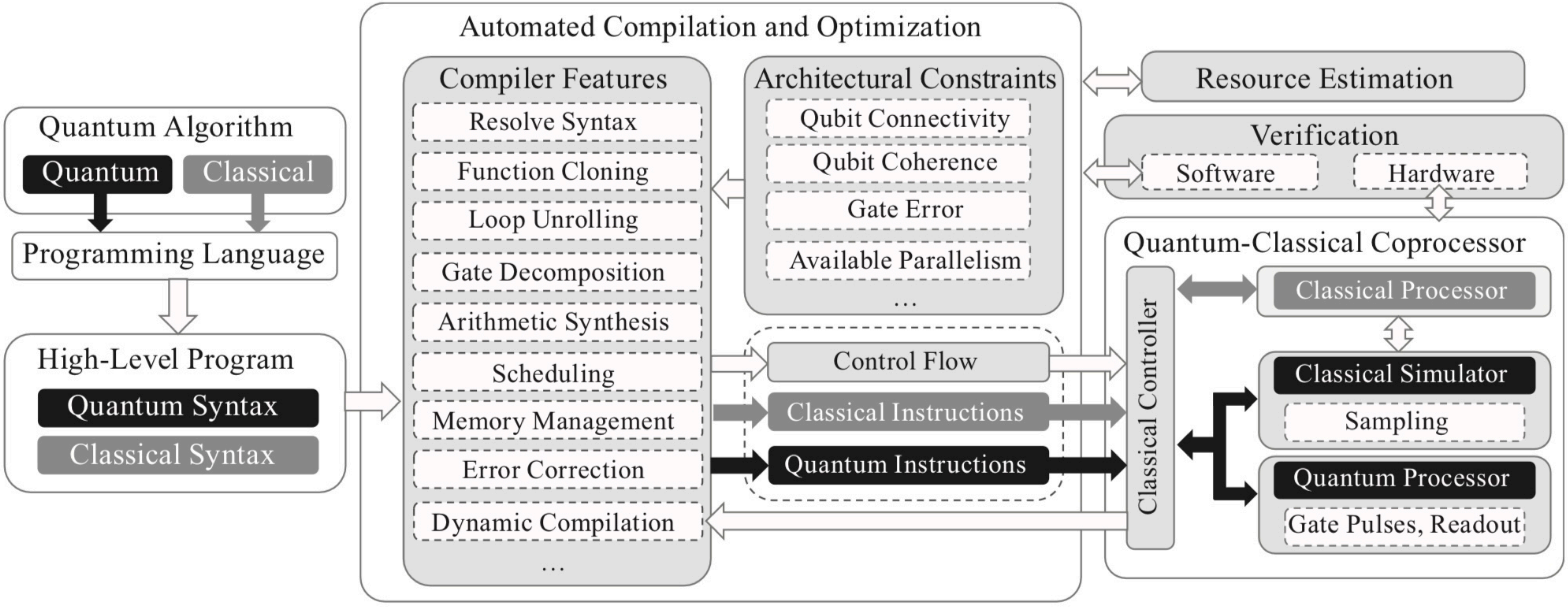


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.

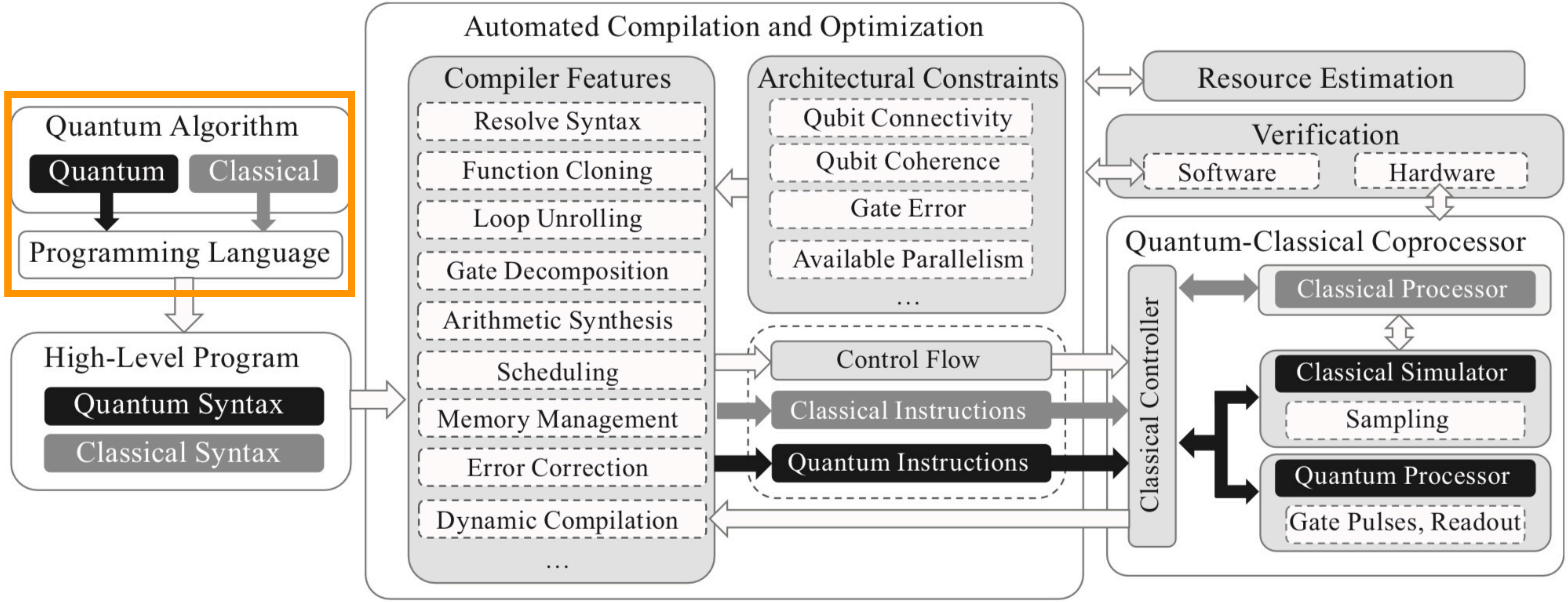


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.

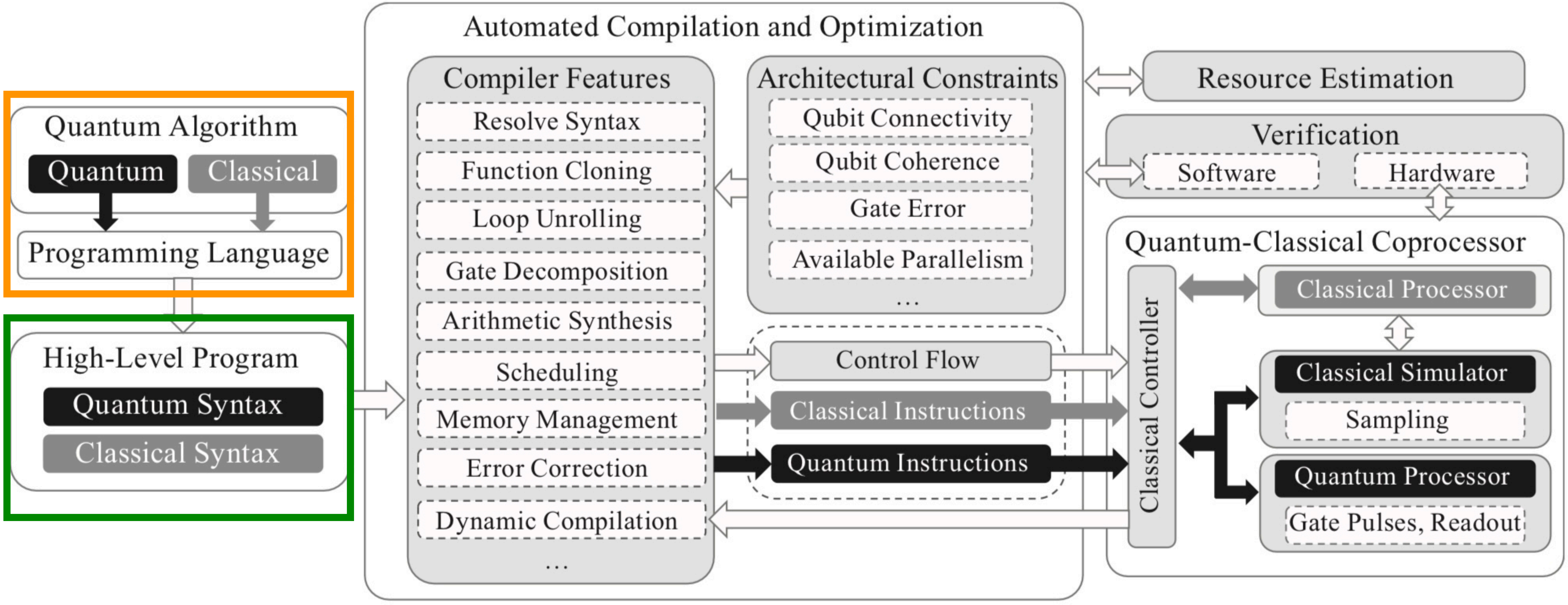


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.

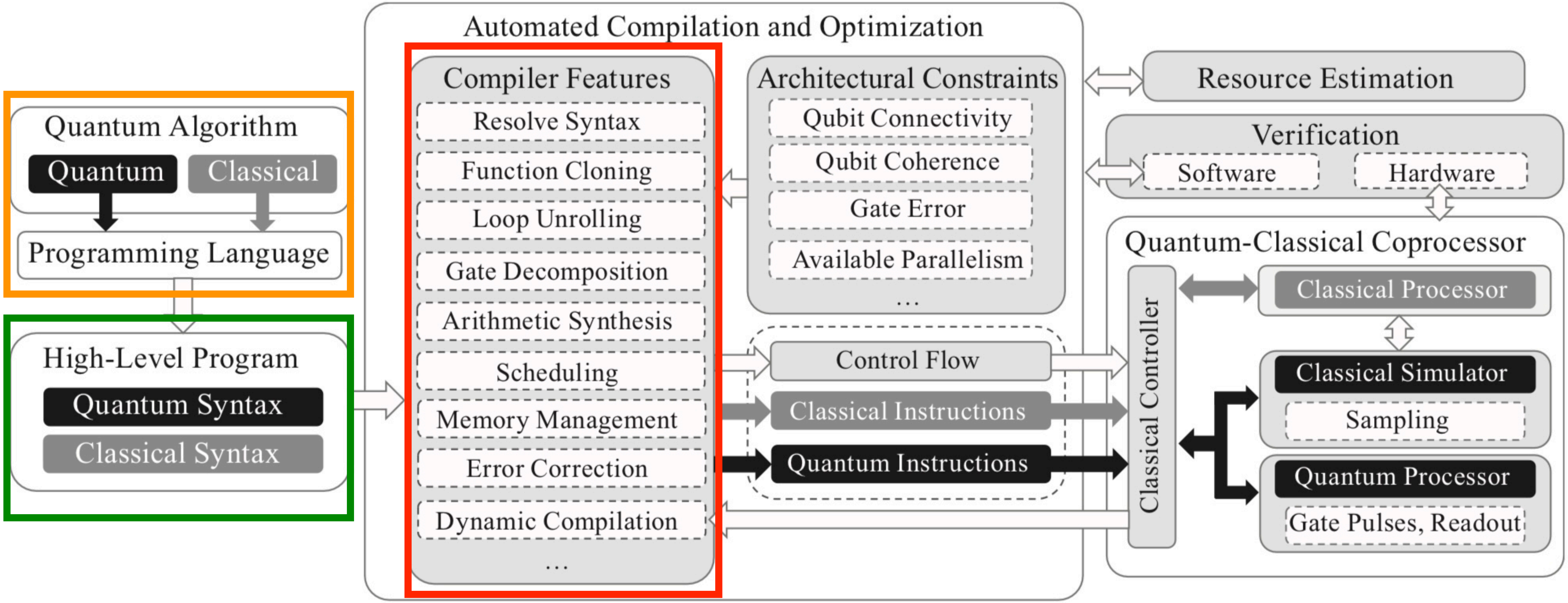


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.

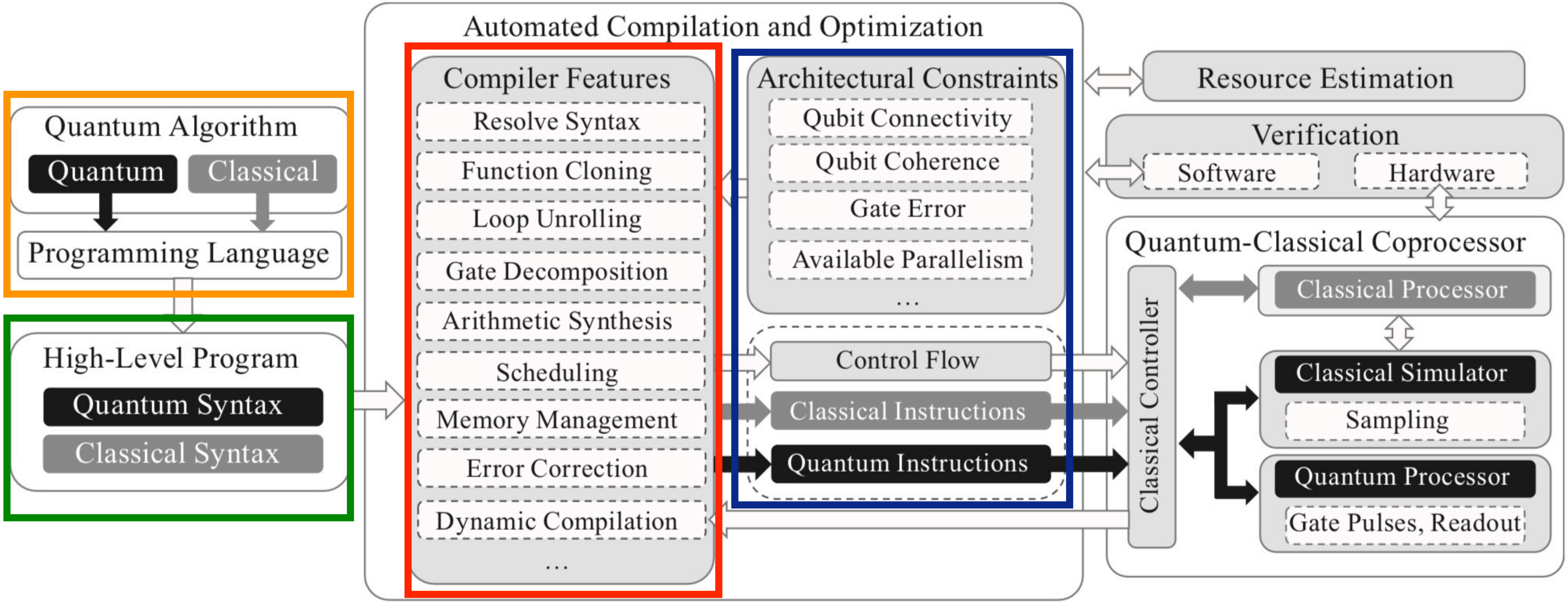


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.

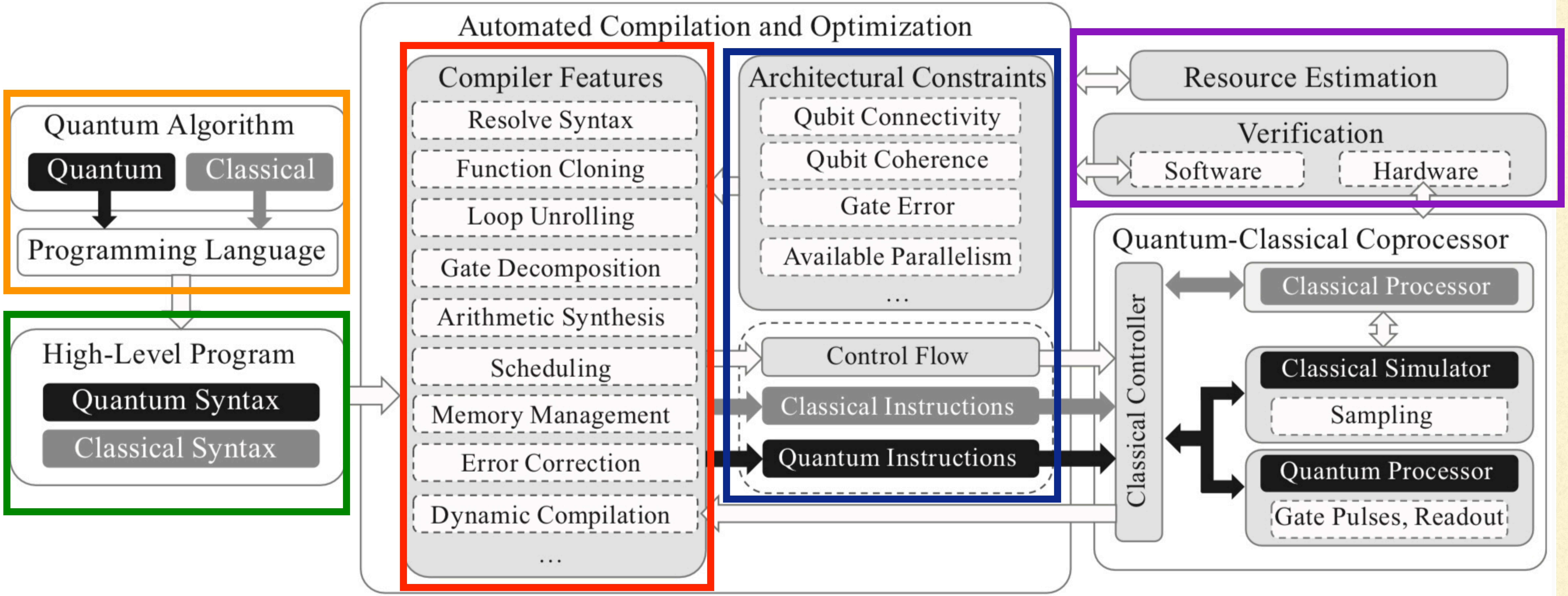
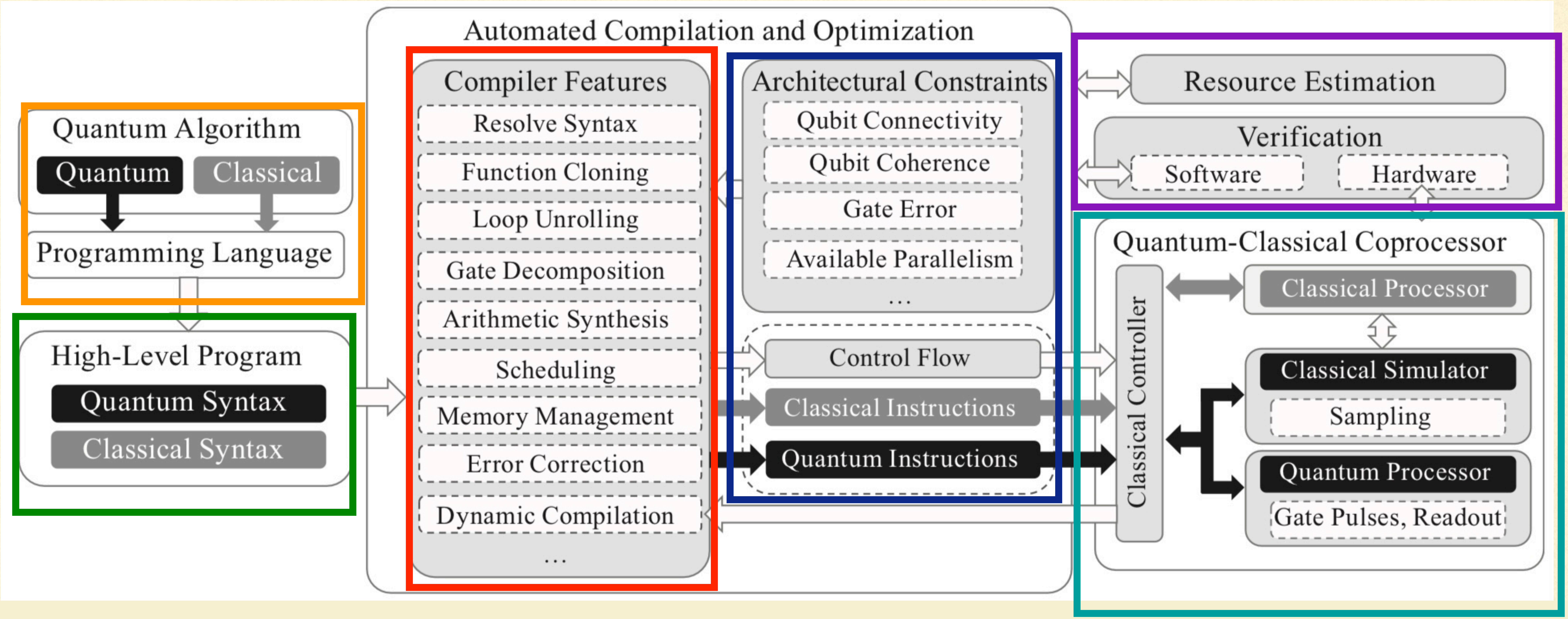


Figure 4.2: A detailed quantum compilation flow outlining the transformations and optimizations involved in a generic compiler.



- At its core, the quantum compiler passes a high-level quantum program through a series of optimizations, for the target hardware, balancing different architectural constraints

OUTLINE

- OVERVIEW
- 4.1 STRUCTURE OF QUANTUM COMPUTER SYSTEMS
- 4.2 QUANTUM-CLASSICAL CO-PROCESSING
- 4.3 QUANTUM COMPILING
- **4.4 NISQ VS. FT MACHINES**

4.4 NISQ VS. FT MACHINES

- Quantum compiling in the context of NISQ and FT era can be drastically different
- Notably, quantum compiling in the NISQ era tends to be **more dynamic**
- For **NISQ** applications, with hybrid/interleaved classical and quantum processing
  - Quantum circuits are parameterized with the parameters optimized by a classical algorithm
- Traditional model of **compiling static** quantum programs once would not work well in the NISQ context

4.4 NISQ VS. FT MACHINES

- Another difference is in the **topology of the architecture** and the **model** for resolving two-qubit interactions
- As a result, **communication costs** will differ
- In the context of a NISQ machine, the most frequently used approach to resolve a long-distance two-qubit gate is to **move one qubit closer** to the other through a chain of swaps

4.4 NISQ VS. FT MACHINES

- In a **F T** machine, we can resolve long-distance interactions between logical qubits through a process called braiding (i.e., movement and transformation of qubits)
- Braiding has **very different cost models** than swapping
  - Braids can extend to arbitrary length and shape in **constant time**, given that they never cross other braids
  - Latency (i.e., time cost) of a swap chain is proportional to the length of the chain

4.4 NISQ VS. FT MACHINES

- A third difference is the choice of **instruction set**:
  - Quantum circuit synthesis has been largely done with Clifford+T gate set, due to its algebraic structures

  - Although that is a **reasonable choice** for **FT** machines (as Clifford gates are straight- forward to implement fault-tolerantly for stabilizer error correction codes)

  - It is not the ideal choice for **NISQ** machines
    - For example, NISQ machines suffer on two-qubit gates such as CNOT gates

4.4 NISQ VS. FT MACHINES

- It remains an **open problem** in discovering optimal device or **application-adapted** synthesis algorithms

- Last but not least, **quantum compiling** in the presence of **noise** has been **under-studied**

- Are among the challenges in quantum computer systems:
  - Integrating noise-awareness in circuit synthesis
  - gate scheduling
  - qubit mapping
  - pulse synthesis
  - compiler validation

OPTIMIZING QUANTUM SYSTEMS–AN OVERVIEW

Chapter 4

Yongshan Ding and Frederic T. Chong
Speaker: Mariana M G Duarte

Questions?