9 How to Build a Quantum Computer

9.1 Fundamentals

9.1.1 Terminology

The term *quantum computer* refers to a device that processes quantum information, according to the rules outlined in Chapter 5. As one tries to build such a device, one has to make a number of decisions that depend on each other. On the physical side, one needs some hardware basis to represent the quantum information, as well as the means to perform logical operations on this information and read out the result. We review some of the existing and proposed hardware for building quantum computers in the following chapters.

Before one gets down to the details of actual implementations, there are some considerations that are relevant for all of them, independent of the specific hardware basis. The first question that we start to discuss here, is how the information flows into and through the computational device; we refer to this as the architecture of the quantum computer. The oldest and so far most successful architecture is commonly referred to as the *network model* of quantum computation [168]. This is the model that we had in mind when we discussed quantum gates in Chapter 5, and we will use it as the model for discussing existing and possible implementation. For completeness, we list some alternatives to the network model in Section 9.4 at the end of this chapter.

The present chapter is about fundamental consideration, which apply to all possible implementations. The subsequent chapters then highlight some of the many physical systems, whose capabilities are currently explored by different players in the field.

9.1.2 History

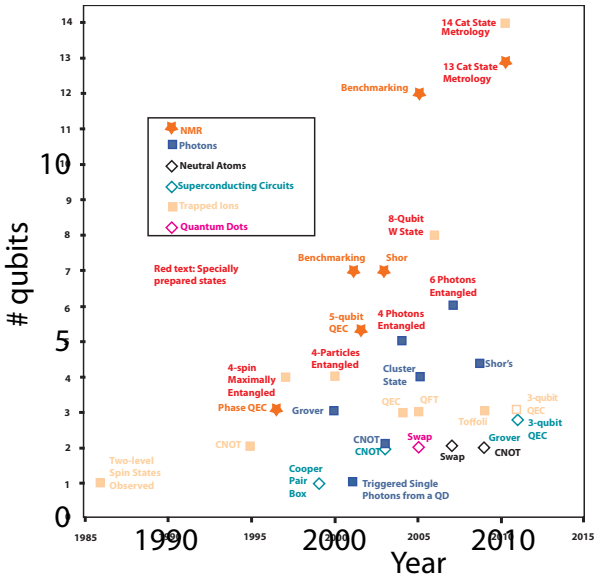

Figure 9.1: Number of controlled qubits as a function of time for different types of QIP hardware. [Adapted from Michael Mandelberg]

Over the first thirty years (from 1985) of quantum computing research, a number of different technologies have been developed that can implement quantum algorithms. As the expertise in the different fields improves, the power of these implementations increases. One measure of their capabilities is the number of qubits that can be controlled. Figure 9.1 compares graphically the evolution of different types of hardware. The individual approaches will be discussed in the subsequent chapters. Another measure is, e.g., how precisely the gates can be implemented and how many gate operations can be performed before the information is lost.

The situation changed from about 2015, when the field quantum computing moved from a

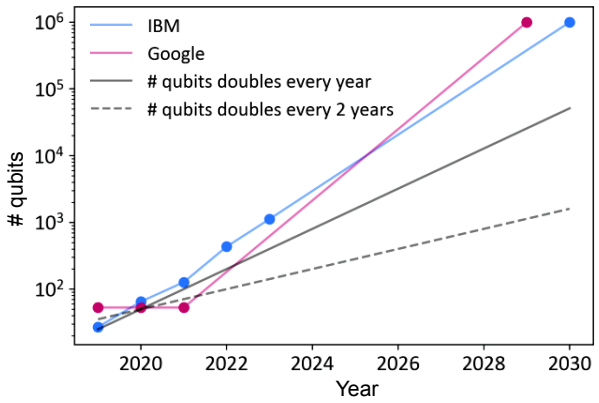

Figure 9.2: Exponential increase in the number of qubits after 2015.

purely academic endeavor to an emerging industry. The commercial actors include small startup companies as well as research departments of established technology companies. Instead of small academic research groups focusing on proofs of principle, they consist of large teams of scientists and engineers with a diverse background, with the common goal of developing a commercially viable product. This implies that their systems will have to be scaled to much larger numbers of qubits. As shown in figure 9.2, the pace accelerated significantly, potentially showing exponential scaling, similar to Moore's law.

9.1.3 The network model

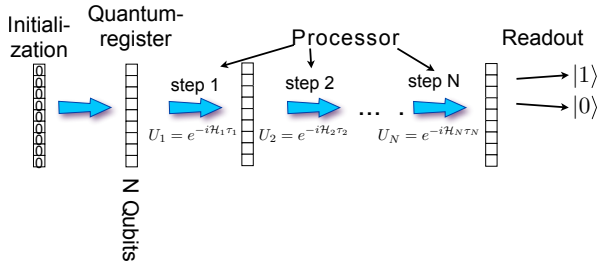

Figure 9.3: Network model of quantum computation.

We now concentrate on the usual network model [1] for constructing a quantum computer, represented schematically in figure 9.3. This is also the model that we used for the discussion of quantum algorithms. Any such implementation has to define a number of components that handle the different steps required for quantum information processing. The first and probably most obvious step is to define how the quantum information is stored. In analogy to a classical computer, where information is stored in arrays of bits called registers, quantum computers may use arrays of qubits called quantum registers. The requirements on these qubits will be discussed in more detail in Section 9.2.2. In the ideal case of a pure state, the information stored in the quantum register can then be described by the state vector Ψ.

Once the qubits are defined, the architecture must provide means of operating on this quantum register. The first step of any quantum algorithm is to initialize the quantum register, i.e., to bring the qubits into a well defined state Ψ_0, independent of its previous history. In many cases, this will be the ground state $|0\rangle$. Since such an initialization cannot be performed by unitary operations, it is necessarily a dissipative process.

The implementation must then provide a mechanism for applying computational steps to the quantum register:

$$|\psi_0\rangle \xrightarrow{U_1} |\psi_1\rangle \xrightarrow{U_2} |\psi_2\rangle \cdots \xrightarrow{U_\omega} |\psi_\omega\rangle,$$

where the unitary operations U_i implement the different computational gate operations. Each of these transformations is driven by a Hamiltonian $\mathcal{H}_i$ that acts on the system for a time τ_i:

$$U_i = e^{-i\mathcal{H}_i\tau_i}.$$

After the last processing step, the resulting state $|\Psi_\omega\rangle$ must be converted into classical information that yields the actual computational result. This typically corresponds to an ideal quantum mechanical measurement, i.e., the projection onto an eigenstate of the corresponding observable.

[1] also known as Quantum Circuit Model

9.1.4 Some existing and proposed implementations

For the first demonstrations of quantum information processing, the information was encoded in nuclear spin degrees of freedom. Processing was achieved by pulses of radio frequency radiation, applied with nuclear magnetic resonance (NMR) spectrometers. More details on this implementation will be given in chapter 10.

The next type of system that became available for storing and manipulating quantum information consists of atomic ions trapped by electromagnetic potentials [12, 11, 134]. Since trapped ions are quite well isolated from their environment, decoherence can be controlled quite well, and there is some prospect that this approach can be scaled to relatively large size [169, 170]. Similarly, it is possible to trap neutral atoms in the electromagnetic potential of standing optical waves. More details on these approaches will be given in Chapter 11.

While these two types of implementations have made the biggest progress so far, it is generally believed that systems with hundreds or thousands of qubits will be based on solid state qubits. A number of suggestions have been published so far that are based on solid-state materials. Initial demonstrations showed some potential for semiconductor materials [171], as well as electronic spins in solids. The strongest contenders from the field of solid-state systems are currently using Josephson junctions, i.e. many-body quantum states consisting of paired electrons. Some of the proposed systems are based on collective excitations in solids, including so-called Majorana fermions. Some additional details on these proposals and implementations will be discussed in Chapter 12.

Individual quantum gates and simple algorithms have also been demonstrated with photons as qubits [172, 173, 142]. Similar to liquid state NMR, this approach has no direct extension to larger numbers of qubits, unless some nonlinear elements are introduced [174, 175, 176]. Details are discussed in chapter 13.

This brief and very incomplete summary shows how diverse the approaches are, that are currently being pursued to build a quantum computer. Each of them has its specific properties that will make its operation unique in some respect. Nevertheless there are some common properties for all of them. In particular, they will all have to fulfill some stringent requirements to become useful devices [8], which we discuss in the following section.

9.1.5 Status and current trends

The progress in quantum computing is often measured by the number of qubits that are built into the processing unit and can be effectively controlled and read. Another important aspect it the number and size of the players entering the race to build commerically viable quantum computers. Figure 9.4 shows some of the major companies involved in the development of quantum computers at the end of 2024.

Figure 9.5 summarizes the Performance characteristics of the current generation of quantum processors. Apart from the number of qubits, the fidelity of the gate operations in one of the most important parameters.

Table 1 | Illustrative selection of quantum hardware companies, current achievements and plans

Company	Technology	Current status	Future plans (if publicly known)
IBM	Superconducting circuits	Condor processor (1,121 qubits) and the serial Heron processors (133 qubits), which can be connected using classical communication.	Delivering 200 logical error-corrected qubits, capable of executing 10 [8] gates by 2029; further upscaling to 2,000 logical qubits in the 2030s [65].
Google Quantum AI	Superconducting circuits	Claim of demonstration of quantum advantage on the Sycamore processor (54 physical qubits) in 2019 [66]; current work directed towards a logical qubit capable of executing 10 [6] gates by 2025.	Upscaling to achieve 10 [6] physical qubits as a basis for an error-corrected quantum computer within the next decade [67].
Atom Computing	Neutral atoms	Announcement of creation of a 1,180-qubit processor based on nuclear spins of optically trapped neutral atoms in 2023 [68].	—
Rigetti	Superconducting circuits	Efforts focused on improving the fidelity of gates for a 84-physical-qubit processor, followed by the development of the Lyra 336-physical-qubit processor in 2024 [69].	—
QuEra	Neutral atoms	In 2023, demonstration of a programmable quantum processor based on 48 encoded logical qubits operating with up to 280 physical qubits [61] in recon figurable neutral-atom arrays.	Upscaling the processors to more than 10^4 physical qubits and encoding more than 100 logical qubits therein [70].
Quantinuum	Trapped ions	Current H2 processors offer 32 fully connected qubits.	Achieving 10 logical error-corrected qubits by 2025 with the perspective of upscaling the technology to 1,000 logical qubits in the long term [71].
Pasqal	Neutral atoms	In 2024, development of quantum processors with about 100 qubits.	Building a processor with 10,000 physical qubits and scalable logical qubits architecture by 2026 [72].
IonQ	Trapped ions	IonQ measures the performance of its quantum processors in terms of algorithmic qubits (AQs), which are benchmarked by use-cases in optimization, quantum simulation and quantum machine learning.	Increasing the current value of 32 AQs to 1,024 AQs by 2028 [73].

The table shows future plans that could become important for future quantum-algorithm-powered simulations of nonlinear dynamics.

Figure 9.4: Major companies developing quantum computers.[146]

Parameter	Superconducting QC	Trapped-ion QC	Neutral-atom QC
Qubit lifetime	~100 µs	Several minutes	Several seconds
Single-qubit and two-qubit gate fidelity F	0.9999, 0.99	0.999999, 0.998	0.996–0.999, 0.955–0.995
Gate execution time	~10–100 ns	~10–100 µs	400 ns to 2 µs
Connectivity	4:1	40:1	10:1 to 20:1
Number of physical qubits	~1,000	~40	~1,000

Figure 9.5: Performance characteristics of the current generation of quantum processors.[146]

9.2 Requirements for quantum information processing hardware

9.2.1 DiVincenzo criteria

In principle, any system that follows the laws of quantum mechanics is a potential candidate for designing a quantum information processing system. This chapter is a first step towards differentiating between "in principle" and the real world of physical objects and implementations, where the systems are actually usable.

In practice, the number of systems that can be used for quantum information purposes is still very much finite, although expanding continuously. A good starting point for discussions about potentially useful systems are the five criteria put forward by David DiVincenzo [8]. These criteria are quite universally accepted as a kind of gold standard for verifying the suitability of a system. Briefly, the conditions are

1. Well characterized qubits, scalable system.
2. Initialization into a well defined state.
3. Long decoherence times.
4. Universal set of quantum gates.
5. Qubit-selective readout.

Mostly, these criteria are self-explanatory, but the consequences and the degree to which they are fulfilled by specific systems will be discussed below.

9.2.2 Qubits

The central part of any quantum computer is the collection of qubits that contain the quantum information being processed. Together, they form the quantum register.

While it is, in principle, possible to identify qubit states with any pair of quantum mechanical states, most of the possible choices will be impossible to implement. This is unfortunate

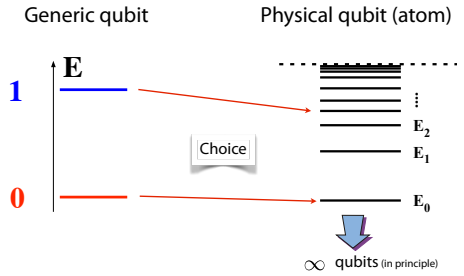

Figure 9.6: Choice of qubit in a generic quantum system.

since even a single atom has an infinite number of states and could therefore, in principle, form the basis of a very large quantum register. However, most of these states have lifetimes that are much too short for quantum computing. Therefore the lifetime of a state is an important parameter, not only for atoms.

Another important aspect is that the states must be distinguishable from each other, i.e. it must be possible to selectively target individual states or pairs of states. This excludes most eigenstates of an atom (infinitely many), since they lie in an energy range that is arbitrarily close to the ionization limit. As a result, they are not only unstable, but virtually impossible to distinguish.

To be useful for information processing, the relevant physical parameters of the individual qubits must be well known. This is necessary in order to be able to predict and control their evolution during logical operations. While this is (at least in principle) relatively straightforward in the generic case of spins $S = 1/2$, where the only possible interaction is the Zeeman coupling $\mathcal{H}_z = \gamma \vec{B} \cdot \vec{\mathbf{I}}$, it becomes a rather nontrivial task in solid state systems, where the internal Hamiltonian of the system and its coupling to the environment are not known *a priori*, but must be determined by measurements and optimized through the available degrees of freedom of the design. The relevant physical parameters include the internal Hamiltonian, the interaction of the system with external fields (electric and magnetic), the couplings between different qubits, and the relevant decoherence rates. Other sys-

tems, like spins in solids, are intermediate between these two cases: their system parameters are nominally identical, but modulated by imperfections of the lattice, like dislocations and strain.

Scalability is an important issue if quantum computers are to become more powerful than classical computers. In the simplest sense this means that one should be able to place as many qubits as one wishes in the register without affecting the operation of the device in a significant manner. More precisely, this means that while the dimension of the Hilbert space grows exponentially with the number of qubits, the computational costs (in time, energy, space etc.) are not allowed to grow exponentially, but at most polynomially. Besides just adding qubits, scalability also implies that one is able to maintain and improve the precision of addressing the qubits, the precision of the individual quantum gates, and to reduce the decoherence rate.

As discussed in section 9.1.2, the number of qubits has been rising significantly for some systems and is likely to increase further. For some problems, e.g., factorization by Shor's algorithm, even larger registers, with several thousand qubits will be required.

One of the less obvious requirements for the identification of qubits with individual quantum states is that it must be possible to create arbitrary superpositions $|\Psi\rangle = c_0|0\rangle + c_1|1\rangle$ of these states. This is usually possible unless there is a selection rule that prevents it. As an example, we consider two neighboring quantum dots, where an electron can tunnel from one dot to the other. It is then possible to identify the qubit state $|0\rangle$ with the electron being in dot 1, and qubit state $|1\rangle$ with the electron being in dot 2. However, it is not possible to identify a qubit with each quantum dot, e.g., with the assignment that the presence of an electron corresponds to $|1\rangle$, while its absence would correspond to $|0\rangle$. The superposition of these two states would then correspond to a superposition between states with different particle numbers, which is usually impossible to achieve for massive particles like electrons. This situation is different if the quantum dots are embedded in a reservoir of electrons, whose potential can be adjusted such that electrons can be exchanged between the reservoir and the quantum dot.

9.2.3 Initialization

Before the actual computation starts, the system must be put into a well defined initial state. Which state this should be is determined by the algorithm, but in many cases, it corresponds to the state

$$|\Psi_0\rangle = |0\rangle \otimes |0\rangle|0\rangle \cdots \otimes |0\rangle = |00\ldots0\rangle,$$

where all qubits are in the logical state $|0\rangle$.

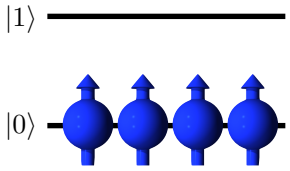

Figure 9.7: Initialization of the system into the ground state.

If this is the ground state of the system, it is in principle possible to let relaxation perform the initialization by cooling until the system reaches the ground state. This is only possible, if the thermal energy k_BT is small compared to the energy difference $\hbar\omega_0$ between the two qubit states.

As a specific example, we consider electron spins in a magnetic field of $B_0 = 2\,\mathrm{T}$, the magnetic energy is

$$\hbar\Omega_L \approx 10^{-34} 2\pi\, 56 \cdot 10^9\,\mathrm{J} \approx 3.5 \cdot 10^{-23}\,\mathrm{J}.$$

This can be compared to the thermal energy. At a temperature of $T = 0.1\,\mathrm{K}$, the thermal energy is

$$k_BT \approx 0.1 \cdot 1.4 \cdot 10^{-23}\,\mathrm{J} \approx 1.4 \cdot 10^{-24}\,\mathrm{J}.$$

This is significantly less than the Zeeman energy. Accordingly, under these conditions, the equilibrium spin polarization approaches unity,

$$\frac{n_\uparrow - n_\downarrow}{n_\uparrow + n_\downarrow} \approx 1.$$

For nuclear spins, the Larmor frequency is ≈ 1000 times smaller and the resulting spin polarisation is

$$\frac{n_\uparrow - n_\downarrow}{n_\uparrow + n_\downarrow} \approx 5 \cdot 10^{-3},$$

i.e. small compared to unity: a nuclear spin system at room temperature is usually in a highly mixed state.

For a nuclear spin system with a Larmor frequency $\omega_0 = 500$ MHz, this would imply that the temperature has to be significantly lower than $T = \frac{\hbar\omega_0}{k_B} = 3$ mK. Such temperatures can be reached today, and it is then possible, in principle, to cool also nuclear spin systems into their ground states. However, at these low temperatures, the time that the system requires to reach thermal equilibrium (i.e. to condense into the ground state) becomes often very long, up to many years, which makes this approach almost useless for practical applications.

9.2.4 Initialization time

Like the gate operations, initialization always requires a finite time to complete. If thermal relaxation is used, this time can be quite long, particularly in nuclear spin systems. The reason is the weak interaction between system and environment, which is a desired effect for the computation, but undesired for initialization.

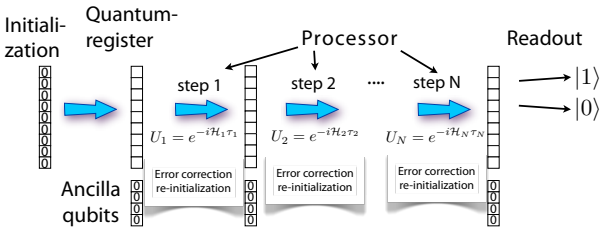

Figure 9.8: Repetitive initialization of ancilla qubits for error correction.

A slow initialization process is not critical for the computation process itself: it occurs before the actual computation and does not affect the time it takes to execute the algorithm. However, it will become a significant issue for any quantum computer that is more powerful than a classical computer: such a system will have to rely on an error correction scheme. All error correction schemes known to date require an input in the form of freshly initialized qubits. These error correction qubits must be initialized at a rate that is large compared to the dephasing rate.

If we write T_2 for the time during which the fidelity of the quantum state decays to $1/e$, the threshold condition requires that error correction cycles are no longer than

$$\tau_{EC} \ll [10^{-4}T_2 \ldots 10^{-2}T_2],$$

where the numerical prefactor depends on the error correction scheme used. Since the re-initialization is part of the error correction cycle, it must therefore be significantly faster than τ_{EC}:

$$\tau_{init} < \tau_{EC} \ll 10^{-3}T_2.$$

This is not fulfilled by thermal relaxation, since it's time-constant T_1 is at best of the same order of magnitude as the dephasing rate,

$$T_1 \geq T_2.$$

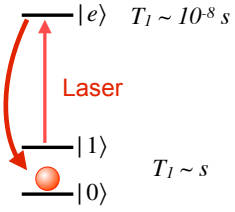

Figure 9.9: Initialization of atomic qubit by a laser pulse.

The requirement can be met, however, in many atomic or optical systems, such as ion traps, where the initialization procedures use optical excitation (see Fig 9.9), which may be performed over a time of the order of nanoseconds, while the coherence time of ground-state qubits can be of the order of seconds. In other systems,

particularly in solid state systems, future implementations will probably rely on switching on some strong coupling to a "cold" system, which brings the qubit to its ground state, and can be switched off during the actual computation. Switching it off is essential, since such a strong interaction would invariably give rise to a fast decoherence process.

9.2.5 Decoherence time

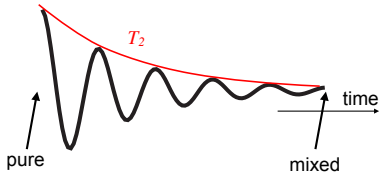

Figure 9.10: Dephasing of a superposition state due to decoherence.

The information in the quantum register is subject to decay through the interaction with external degrees of freedom. The computation must therefore be completed before this decay has significantly degraded the information. For most physical systems being considered for quantum information processing, estimates for the decoherence times vary by many orders of magnitude. This is partly due to the difficulty of performing such measurements; in addition, the decoherence that one can attain in a specific device is usually many orders of magnitude shorter than for an ideal isolated system and varies with many parameters of the fabrication process that can only partially be controlled. This is particularly true for solid state systems where the qubits are either defects embedded into a macroscopic environment consisting of thousands of atoms, or they themselves consist of mesoscopic structures with thousands or millions of particles. Accordingly this field has invested heavily into improving the properties of their qubits and has seen enormous progress in increasing the lifetime of their qubits by many orders of magnitude by improving all aspects of design and fabrication.

The effect of decoherence can partly be eliminated by quantum error correction or error prevention schemes like dynamical decoupling, as discussed in Section 7.4. However, error correction also increases the duration of the computation and introduces additional errors. Theoretical analysis shows [95, 96] that computations can proceed for an arbitrary duration provided that quantum error correction is used and the error probability for individual (uncorrected) gate operations is below some threshold, which can be of the order of $10^{-2} \ldots 10^{-4}$. The relevant figure of merit for the viability of a particular implementation will therefore eventually be whether it can reach this threshold where reliable quantum computing can proceed for arbitrary duration.

When estimating the prospects for achieving this threshold, one has to take into account that the relevant dephasing time is not that of the individual qubits, but that of the total information stored in the quantum register. While details for the decoherence in such highly entangled quantum systems are not known, it is generally expected (and verified for some specific systems) that decoherence processes are much faster for the total quantum register than for the individual qubits. Details are discussed in section 7.2.8.

9.2.6 Quantum gates

If one wishes to build a "universal" quantum computer, i.e., one that can process arbitrary algorithms, one needs a universal set of quantum gates, as discussed in section 5.3. The unitary operations that act as gates on the qubits must be implemented by Hamiltonians that act on the system for a specified time.

Generating the single-qubit Hamiltonians is in general relatively straightforward: typically they correspond to external fields acting on the qubits for a specified duration. As we have shown in section 5.1, single-qubit gate operations correspond to rotations. It is thus best to visualize them as being driven by a magnetic field acting on a spin 1/2. In the example shown in Figure 9.11, the field is applied at 45° between the x and z axis

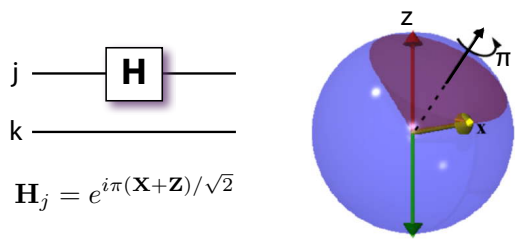

Figure 9.11: Hadamard gate on qubit j as a representative example of a single qubit gate.

of qubit j to generate a Hadamard gate:

$$\mathbf{H}_j = e^{i\pi(\mathbf{X}_j+\mathbf{Z}_j)/\sqrt{2}}.$$

A nontrivial requirement is, in many systems, that these gates must be applied selectively, i.e., it must be possible to apply a logic gate to qubit j in such a way that no other qubit is affected by it:

$$U_i = \mathbf{1}_1 \otimes \mathbf{1}_2 \otimes \cdots \otimes \mathbf{H}_j \otimes \cdots \otimes \mathbf{1}_N.$$

This addressing problem is not significantly different from classical computers, where the gate operations are typically applied by changing local potentials using voltages in conducting wires. The same principle can be applied to many solid-state qubits, like semiconductors and superconductors, which are controlled by nanostructured electronic circuits.

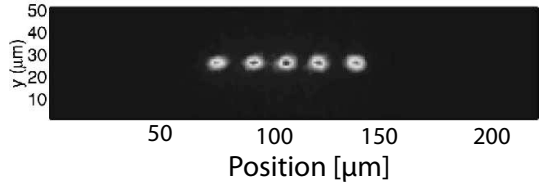

Figure 9.12: Fluorescence of trapped $^{40}Ca^{+}$ ions. Addressing is possible by focusing a laser on the selected ion.

In the case of ion trap quantum computers, it is possible to apply laser pulses that are so tightly focused that the interaction with all but one ion can be neglected. This means that the ions can be addressed individually if they are separated by a distance that is large compared to the optical wavelength.

9.2.7 Frequency-domain addressing

The situation is different for spin-based quantum computers. In liquid state NMR, e.g., the wavelength of the applied radio frequency field is of the order of 1 meter; all qubits therefore experience roughly the same coupling to the radio-frequency (RF) field.

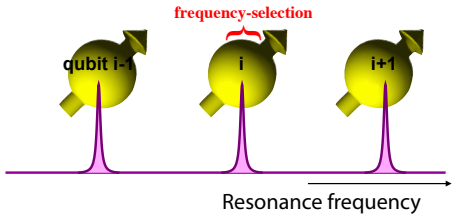

Figure 9.13: Addressing in frequency-space.

Nevertheless it is possible to address individual qubits independently of each other, since the excitation is a resonant process: only spins whose magnetic resonance transitions are close to the frequency of the RF field interact strongly with the field. The selection process occurs in this case in frequency space.

In solid state systems, the selective addressing of individual qubits will typically be achieved by nanometer-sized electrodes that must reach close to each qubit. While the technology of building these circuits is maturing rapidly, the effect that these structures and the applied fields have on the decoherence of the qubits will have to be analyzed in more detail.

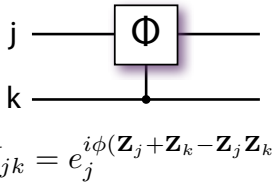

Figure 9.14: Controlled phase gate as a representative example of a two-qubit gate.

In many systems, the two-qubit operations are more difficult to implement, since they also require, apart from external fields, interactions between qubits. In the example of Figure 9.14, the controlled phase gate

$$U_{jk} = e^{i\phi(\mathbf{Z}_j + \mathbf{Z}_k - \mathbf{Z}_j\mathbf{Z}_k)}$$

acting on qubits j and k includes external fields $\mathbf{Z}_{j,k}$ along the z-axis of qubits j and k, in addition to a bilinear coupling $\mathbf{Z}_j\mathbf{Z}_k$ between these qubits. Such interactions exist in all systems proposed for quantum computing. A typical case is the exchange interaction

$$\mathcal{H}_E = J\vec{I}^j \cdot \vec{I}^k,$$

which is found, e.g., in the case of electrons in semiconductors. However, 2-qubit gates require not static interactions, but the interactions should be off for most of the time. Only when a two-qubit gate is to be applied to the qubit-pair j, k, the interaction between qubit j and k must be switched on for a well defined duration. In some systems, this procedure cannot be implemented directly: in liquid state NMR, e.g., the couplings are determined by the structure of the molecule, which remains constant during an experiment. A possible alternative is then to use static interactions and eliminate the unwanted ones by a procedure called refocusing. This procedure is applied routinely in NMR quantum computers and will be discussed in Chapter 10. The concept has also been generalized to other systems [177].

9.2.8 Imperfections

Every experimentally realizable gate includes imperfections, i.e., deviations from the ideal behavior. They can be quantified, e.g., by the gate fidelity defined in section 7.3.2:

$$F(U_e, U_t) = \frac{|Tr\{U_e^\dagger U_t\}|}{\sqrt{Tr\{U_e^\dagger U_e\}}\sqrt{Tr\{U_t^\dagger U_t\}}}.$$

Here, U_e is the experimentally generated gate operation while U_t represents the target operation.

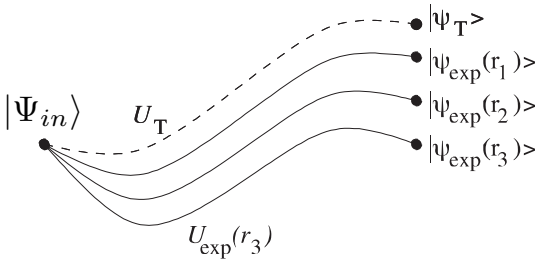

Figure 9.15: Different evolutions resulting from finite precision of gate potentials.

For single-qubit gates, whose ideal form $\mathbf{U}(\theta, \phi, \beta)$ may be parametrized with three angles, deviations may correspond to errors in these angles. In systems, where the qubits are only part of a larger Hilbert space, leakage may be a problem: the real operation may take part of the state out of the qubit space. As an example, consider a harmonic oscillator, where the states $|n = 0\rangle$ and $|n = 1\rangle$ have been chosen to represent a qubit [178, 134]. Since the energy level separations between all states are identical, there is always a tendency to excite higher lying vibrational states. In addition, addressing is usually not perfect: Any excitation of a single qubit j will always excite neighboring qubits to some degree.

The effect of most errors is a degrading of the information in the quantum register and is therefore similar to an additional source of decoherence. Consequently, these errors can also be eliminated by error correction schemes, provided they are small enough.

9.3 Converting quantum to classical information

At the end of the computation process, the result of the computation must be retrieved from the final state of the quantum register: The result of the quantum computation is not the final quantum state, but rather classical information that may consist of a sequence of (classical) bits. Converting the quantum state into classical bits is achieved by the readout process. What exactly

has to be read out is determined by the quantum algorithm being considered.

While the analysis of the result is, in principle, similar to the corresponding procedure in a classical computer, where one reads the logical state of the individual classical bits, it involves here measurements on a quantum mechanical system. The quantum mechanical measurement process (see chapter 4.5) is a highly nontrivial topic, and quantum computers touch some of its central issues. We therefore discuss some of these issues in this separate section.

9.3.1 Principle and strategies

After the last logical operation of a quantum algorithm, the quantum register is left in its final state

$$|\psi_{\text{fin}}\rangle = c_0|0,0,0...0\rangle + c_1|0,0,0...1\rangle + c_2...,$$

which contains the solution of the problem being investigated. The sum runs over all 2^N basis states, where N is the number of qubits. According to this formal analysis, the result of the computation is contained in the 2^N coefficients c_i that determine the final state. It is thus possible to extract the result by determining these coefficients, e.g. by state tomography (→ 7.3.3). However, determining 2^N coefficients clearly scales exponentially with the number of qubits and is therefore not an efficient process. Furthermore, in almost all situations, we are not interested in all these values. The *useful* final result should have a numerical or Boolean logical value, such as *true* or *false* or 42[2]. We therefore discuss here how to convert the final state of the unitary transformation into the desired classical information.

Like the initialization process, the readout is a non-unitary operation that cannot be reversed. The wavefunction of the quantum register collapses during readout. Many algorithms rely on measuring the populations of the individual qubit states $|0\rangle$ and $|1\rangle$. In this case, the relevant observables are the longitudinal components $\mathbf{Z}$ of the pseudo-spin operators or the projectors $\mathbf{P}_{|0\rangle}$ or $\mathbf{P}_{|1\rangle}$. Other algorithms, like the Deutsch–Jozsa scheme, require readout of the transverse component $\mathbf{X}$, and some quantum computer architectures, like the one-way quantum computer (see section 9.4.5), require the readout of arbitrary components of the pseudo-spin. The relevant observable is then

$$O = \alpha\mathbf{X} + \beta\mathbf{Y} + \gamma\mathbf{Z}.$$

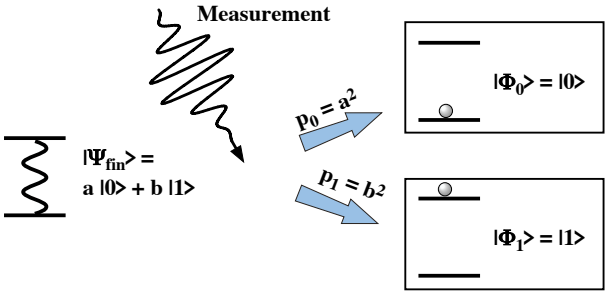

Figure 9.16: State reduction during the measurement process.

According to the quantum mechanical projection postulate discussed in Chapter 4, an ideal quantum mechanical measurement collapses the state $|\psi\rangle$ into an eigenstate $|\phi_i\rangle$ of the observable being measured and returns the eigenvalue λ_i of the corresponding state with probability $|c_i|^2$, where c_i is the expansion coefficient of the state $|\psi\rangle = \sum c_i|\phi_i\rangle$. Assuming that such an ideal measurement is possible, reading out the result of a quantum computation is relatively straightforward - provided the measurements are ideal and we only need the absolute value of the coefficients.

Since the purpose of this chapter is to discuss issues arising in real experimentally realizable systems, we have to consider the different behavior encountered in real measurements. In many realistic systems, measurement attempts return

[2] Result given by the computer named Deep Thought when asked to calculate the Answer to the Ultimate Question of Life, the Universe, and Everything in the Hitchhiker's Guide to the Galaxy.

no result. Typical cases are attempts to measure the state of a qubit by scattering a photon from it. If the photon is not scattered, this is not critical, as one may simply repeat the attempt. If the photon is scattered but not detected, this is more critical. In this case, an interaction of the qubit with an external system (the photon) has changed the state of the qubit, and a repetition of the measurement may produce a different result.

Several strategies are possible to circumvent this problem: one can try to use a QND (=quantum non-demolition) measurement [179, 180]. Such a measurement arranges for the unavoidable influence that the measurement must have on the qubit to be such that it does not affect later measurements of the same variable. Not all variables can be measured this way, but in most cases it is possible to arrange the system in such a way that QND measurements can be used at least in principle. A good example is a free particle: Its momentum is a QND variable, i.e. it can be measured (in principle) repeatedly without compromising the measurement accuracy of the later measurements. The position of a free particle, however, is not a QND variable: if we measure the position with any finite precision, we necessarily increase the momentum uncertainty, as required by the Heisenberg uncertainty relation. Under the subsequent evolution driven by the free particle Hamiltonian

$$\mathcal{H}_f = \frac{1}{2m}\vec{p}^2,$$

the momentum spread is translated into a spread of the position and therefore reduces the accuracy of later measurements.

9.3.2 Repeated measurements through ancilla qubits

Another possibility is to read out not the qubit itself, but a copy of it. This has several advantages: it allows one to

1. Repeat the measurement as often as needed while minimizing the perturbation of the output state.
2. Optimize the properties of these ancilla qubits for readout, independently of the properties of the computational qubits.

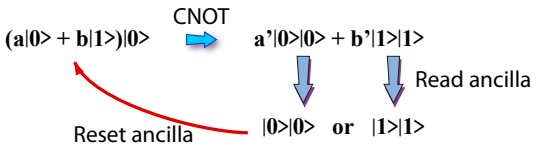

Figure 9.17: Basis of repetitive readout procedures.

As discussed in Chapter 4, copying quantum information is possible within limitations. The copy process will not provide an exact copy of the quantum state (no cloning theorem!), but it can copy exactly the probabilities of obtaining certain measurement results. As long as the copying process is exact, one can therefore repeatedly measure copies of the qubit. If the measurement is not successful, or to check the validity of the measurement result, one can then make an additional copy and read that out. Such a procedure could be repeated many times to achieve very reliable readout even with very unreliable single measurements.

If the qubit that we want to read is in a superposition state

$$|q\rangle = a|0\rangle + b|1\rangle = \begin{pmatrix} a \\ b \end{pmatrix},$$

and the measurement qubit is initially in state $|0\rangle$, the combined system is

$$(a|0\rangle + b|1\rangle) \otimes |0\rangle = a|00\rangle + b|10\rangle.$$

The copy (CNOT) operation changes the state of the two qubits into the correlated state

$$\begin{aligned} (a|0\rangle + b|1\rangle) \otimes |0\rangle \quad &\rightarrow \quad (a|0\rangle \otimes |0\rangle + b|1\rangle \otimes |1\rangle) \\ &= \quad a|00\rangle + b|11\rangle. \end{aligned}$$

If a measurement of the measurement qubit yields a result (i.e., finds it in state $|0\rangle$ or $|1\rangle$),

it collapses the wavefunction of both qubits simultaneously. If the measurement fails, i.e. it does not collapse the state and does not provide a result, one has the option of discarding the measurement qubit and re-initialize it to state $|0\rangle$. The combined system is thus returned to a product state. The register qubit is returned to its original state if the measurement attempt did not collapse the state or to a computational basis state if the entangled state has collapsed without providing a result. The measurement process can then be repeated until a result is obtained.

9.3.3 Example: Deutsch–Jozsa algorithm

As an example readout process consider a function evaluation, such as in the Deutsch–Jozsa problem (see Section 8.2). Here the processing can be written as

$$|\psi_0\rangle = \sum_x |x, 0\rangle \Rightarrow |\Psi_{fin}\rangle = \sum_x |x, f(x)\rangle,$$

where the superposition of all possible input states is transformed into a superposition of all possible input states and function values. As discussed in Chapter 8, the goal of the Deutsch–Jozsa algorithm is to learn, with a single function call, wether a function is constant or balanced. For the simple case of a single qubit (plus auxiliary qubit), we found that if the two function values are the same, $f(0) = f(1)$, then the final state of the quantum register is

$$|\psi_{eq}\rangle = (|0\rangle + |1\rangle) \otimes (|f(0)\rangle - |\bar{f}(0)\rangle),$$

but if they are different, $f(0) \neq f(1) = \bar{f}(0)$,

$$|\psi_{ne}\rangle = (|0\rangle - |1\rangle) \otimes (|f(0)\rangle - |\bar{f}(0)\rangle).$$

In this trivial example, the type of measurement that must be performed is obvious. In both cases, the input register is in an eigenstate of $\mathbf{X}$. Its eigenvalue is $+1$ if the two possible function values are different (i.e., the function is balanced) or -1 if the two values are the same (i.e., the function is constant). The result can be determined from the single successful measurement of the variable $\mathbf{X}$ of qubit 1.

An alternative to measuring $\mathbf{X}$ is to apply another transformation to the first qubit, such as a Hadamard gate or a $\pi/2$ rotation around y. The states $(|0\rangle \pm |1\rangle)$ are then converted to $|0\rangle$ and $|1\rangle$ and can be measured by using the more conventional observable $\mathbf{Z}$. The same principle can be applied to other states, which require measurements in a direction defined by the spherical angles θ,φ.

9.3.4 Complete state information

The example shows that (for a single qubit) a single measurement is sufficient to determine the result (constant or balanced). This power does not come for free: while one gains this ability, one loses the possibility to find out what these values are, i.e., whether the (constant) results are $(0, 0)$ or $(1, 1)$ or (for the case of a balanced function) $f(0) = 0$ and $f(1) = 1$ or $f(0) = 1$ and $f(1) = 0$. Answering such a question requires measuring a different observable, which does not commute with $\mathbf{X}$ and is therefore not compatible with this measurement.

The complete information that is contained in the final state consists of the 2^N coefficients c_i that define the superposition. How this can be done was discussed in section 7.3.3. However, to determine all 2^N coefficients requires at least 2^N measurements, i.e., an effort that increases exponentially with the number of qubits. Obviously this is not possible without losing the scaling advantage, a major motivation for the implementation of quantum computing.

Furthermore, it can be difficult to make measurements that are state-selective, i.e., distinguish state $|i\rangle$ from the other $2^N - 1$ states. Instead one is usually content with measurements on single qubits, which are often referred to as local measurements. As discussed above, this will not allow for a complete determination of the state.

Consider, e.g., the two states

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

and

$$|\psi_2\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle).$$

If the two qubits described by this state are measured independently, one will obtain $|0\rangle$ in 50% of all cases and $|1\rangle$ in the other 50% for each of the qubits. Looking only at individual results, the two states would then appear to be indistinguishable.

It is nevertheless possible to distinguish between them by taking correlations into account. Correlations are measured by two-qubit (or, in the general case, multi-qubit) observables. For the present case, the operator $\mathbf{Z}_1\mathbf{Z}_2$ is suitable: it returns

$$\begin{aligned}\langle\Psi_1|\mathbf{Z}_1\mathbf{Z}_2|\Psi_1\rangle &= \frac{1}{2}(+1+1) = 1\\ \langle\Psi_2|\mathbf{Z}_1\mathbf{Z}_2|\Psi_2\rangle &= \frac{1}{4}(+1-1-1+1) = 0.\end{aligned}$$

In the first case, measurements on the individual spins always yield the same result; in the second case, they are completely uncorrelated.

If not only the amplitudes (probabilities) of the different states are required, but also the (relative) phases, additional measurements must be performed that are not compatible with the measurements that determine the amplitudes. An important example is given by the two states

$$|\psi_\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle),$$

which distinguish the two outcomes of the DJ-algorithm. Clearly, they are orthogonal states and must be distinguished to determine the result of the DJ-algorithm. The absolute values of the two coefficients are the same, but they differ by the relative phase. They cannot be distinguished by measuring the operator Z, but they can be distinguished by measuring the operator $\mathbf{X}$: The states $|\psi_\pm\rangle$ are eigenstates of $\mathbf{X}$, with eigenvalues ± 1.

9.3.5 Quantum state tomography

If the state of the system is not pure (which is never truly the case), the system has to be described by a density operator and the number of coefficients that have to be determined for a complete characterization of the quantum state rises from 2^N to 2^{2N} (or $2^{2N}-1$, if we take into account that the trace of the density operator is 1). Determining all these numbers requires multiple incompatible measurements. The corresponding procedure is known as 'quantum state tomography' (see also chapter→7.3.3), in analogy to tomographic procedures in imaging, such as magnetic resonance tomography (MRT) of X-ray tomography (CT). With the appropriate combination of measurements and computational procedures, it is possible to reconstruct the full density operator.

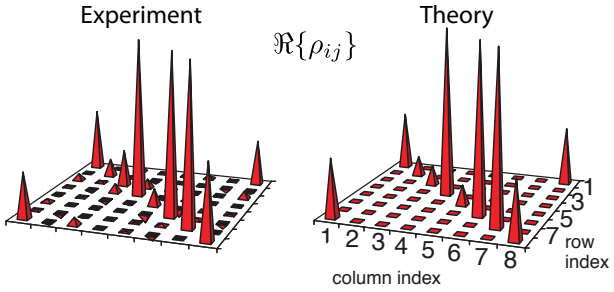

Figure 9.18: Tomographic representation of the real part of a density operator.Kampermann *et al.* [181]

Fig. (9.18) shows, as an example, the real part of a density matrix measured experimentally by a series of tomographic experiments [181]. The state measured here is

$$\begin{aligned}\rho_{BE} = \frac{1}{N}\Big[&2|\mathrm{GHZ}\rangle\langle\mathrm{GHZ}| + a_1|001\rangle\langle 001|\\ &+a_2|010\rangle\langle 010|\\ &+\frac{1}{a_3}|011\rangle\langle 011| + a_3|100\rangle\langle 100|\\ &+\frac{1}{a_2}|101\rangle\langle 101| + \frac{1}{a_1}|110\rangle\langle 110|\Big],\end{aligned}$$

which is a "bound entangled state". A definition of the GHZ-state is given, e.g., in section 4.6.6.

The only method for verifying that the state is really created is to perform full quantum state tomography. Clearly, this is not an efficient way and does not scale well for large systems.

In some cases, a full tomography of a quantum state is not necessary, but it is sufficient to perform partial state tomography, i.e. to determine a subset of the density operator elements. An important example is the so-called 'population tomography', where one determines the diagonal elements of the density operator.

9.4 Alternatives to the network model

So far, we have always analyzed quantum computing in terms of the so-called network model. This is certainly the most frequently used computational paradigm and it has the closest correspondence to classical computational models. However, there are some other models that have some advantages - either in terms of offering additional insight into the basic requirements for computing or in terms of realization of large-scale quantum computing. Here, we give a short introduction into some of these concepts.

9.4.1 Local addressability

One requirement of the network model of quantum computation is local addressing, i.e., the ability to perform logical operations on arbitrary individual qubits. This requirement is relatively easy to satisfy for the present demonstration models with only a few qubits. It is, however, a major problem for increasing the number of qubits. In liquid state NMR, e.g., the number of resonance lines increases exponentially with the number of coupled spins, making individual addressing virtually impossible for systems with 10 and more qubits.

In solid-state systems, the individual qubits can be addressed by suitably designed control lines. However, current systems are already facing technical challenges in the number of such control lines: for each qubit, some 10 control- and readout lines are typically required.

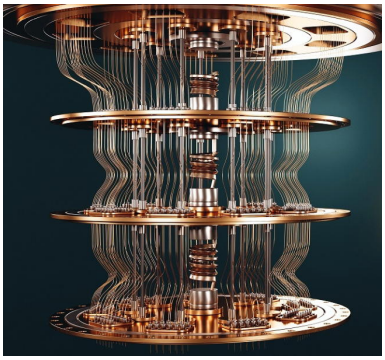

Figure 9.19: Wiring in a solid-state quantum processor.

As figure 9.19 shows, this puts strain in current quantum processors operating a few dozen qubits at low temperature. Extending this to thousands or millions of qubits will be highly challenging. Apart from the difficulty of constructing such devices with sufficiently high precision, the large number of control gates may introduce too many channels for decoherence. These difficulties motivate a search for alternative architectures that do not require local addressing of individual qubits.

9.4.2 Cellular automata

Cellular automata (CAs) offer a possible architecture that does not require local control. They are known from classical information processing. The term "cellular" refers to the fact that in this model, the information is contained in cells, which change their state on the basis of some simple rules and the state of some neighboring cells. In the simplest case, the system is a one-dimensional register, in which each cell can take the values 0 or 1. This is called a binary CA. A processing step corresponds to each cell changing its state on the basis of the state of itself and its immediate neighbors. This state change is initiated for all cells simultaneously by a single trigger signal.

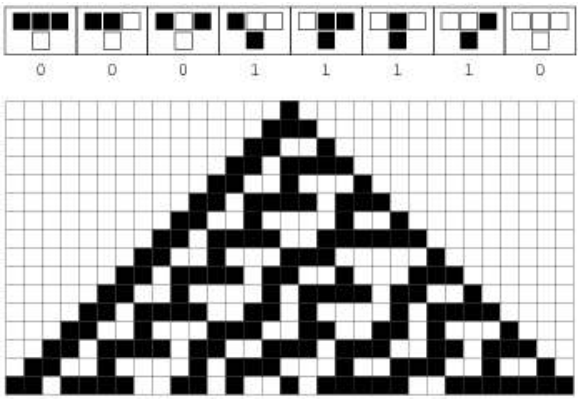

Figure 9.20: An example of a simple classical cellular automaton.

The top trace of Fig. 9.20 shows a simple example: here, the new state of each cell depends on the present state of itself and its two immediate neighbors. For a binary CA, these three cells can have $2^3 = 8$ different values on input. The CA rule assigns a new state to each of these eight initial states. This means that there are $2^8 = 256$ possible rules. They can be numbered, as shown in Fig. 9.20, by ordering the eight input states sequentially and labelling the corresponding output state by its binary value. The corresponding bit sequence defines a number in the range [0..255]. In the example in Fig. 9.20, the bit sequence 00011110 defines it as "rule 30". The lower half of Fig. 9.20 shows 16 generations of the register, starting from an initial state where all except one cell contain the value 0.

Figure 9.21: Picture of a Textile Cone Snail showing a CA pattern [182].

Different schemes, including different dimensions, different geometries and different number of states are possible. Many natural systems like the one shown in figure 9.21 operate as cellular automata, changing their state in response to local input from neighboring cells.

Figure 9.22: Snapshot from Convay's game of life.

A popular variant of this scheme is "Conway's game of life", a CA that was invented by the British mathematician John Horton Conway in 1970. As shown in figure 9.22, it generates interesting patterns and may be considered a model for the evolution of life in general.

9.4.3 Quantum cellular automata

Quantum cellular automata (QCAs) are the quantum mechanical version of CA's. Lloyd proposed them as an alternative quantum computer

architecture that does not need to address every qubit individually in 1993, [183] i.e. before the first experimental realizations of quantum computers. In a QCA, each cell is characterized by its quantum state and changes its value (i.e. it evolves) in response to a global (i.e. identical for all cells) control operation and the state of itself and some neighboring states. As a result, only a few control qubits are needed.

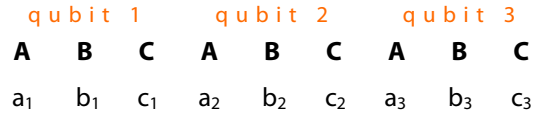

Figure 9.23: Architecture of an ABC-type QCA.

In Lloyd's QCA architecture, the quantum information is stored in a chain of qubits that consists of repeated units ABC of only three distinguishable *physical qubits* (see Fig. 9.23). Each group of three physical qubits stores one logical qubit.

Logical operations acting on these qubits can be broken down into operations that act on all A, B or C physical qubits whose neighbors are in the appropriate state. In addition, the units at the end of the chain can be distinguished, since they only have a single neighbor. For the following discussion, we use the notation ω_{01}^{A} to indicate a π-pulse that inverts all A spins provided that the spin to the left (a C-type spin) is in the state 0 and the spin to its right (a B-type spin) is in the state 1. The sequence

$$\omega_{01}^{A} - \omega_{11}^{A} - \omega_{10}^{B} - \omega_{11}^{B} - \omega_{01}^{A} - \omega_{11}^{A}$$

of global operations can then be analyzed by considering pairs of pulses: The first and the last pair act on all A-type spins, irrespective of the state of the left-hand neighbor, bot conditioned on the right-hand neighbor (B) being in the state 1. Similarly, the middle pair acts on the B-spins irrespective of the C-spin on its right, but conditioned on the A-spin on its left being in the state 1. Together, this sequence corresponds to the following operations:

	$\omega_{01}^{A}, \omega_{11}^{A}$	$\omega_{10}^{B}, \omega_{11}^{B}$	$\omega_{01}^{A}, \omega_{11}^{A}$
$\|a_i b_i\rangle = 00$	00	00	00
$\|a_i b_i\rangle = 01$	11	10	10
$\|a_i b_i\rangle = 10$	10	11	01
$\|a_i b_i\rangle = 11$	01	01	11

Here, the first column represents the input state of spins AB, the second the state after the first pair of operations etc. If we compare the input and output state, the overall operation corresponds to a SWAP operation between A and B. Sequences of SWAP operations can be used, e.g., to move logical qubits through the quantum register. This applies to all pairs AB, except those at the end of the string. They can be addressed separately, e.g. to load information into the register.

It was shown that this architecture is universal, i.e., it can efficiently run all algorithms that are efficient on a network quantum computer with an overhead, compared to the network model, that is polynomial in the number of qubits. This motivates its consideration as an alternative to the network model.

9.4.4 QCA with 2 types of cells

A modification of this scheme was proposed by Simon Benjamin [184, 185, 186]. His scheme uses only two distinguishable units α, β. In addition, some fixed units are used that are not affected by the control pulses. The relevant operations are then implemented by combining the appropriate two-dimensional arrangement of the physical qubits with global control operations.

Fig. 9.24 shows, as an example, how information is transferred along a chain of physical qubits. Again, each logical qubit x_i is represented by three physical qubits, two of type α, the third of type β. In the initial state represented by the left-hand part of Fig. 9.24, the information is stored in type α qubits. The required update operations are represented as $\alpha_v^{t\to u}$, which indicates that all cells of type α are taken from state t to state u, provided they experience a 'net field'

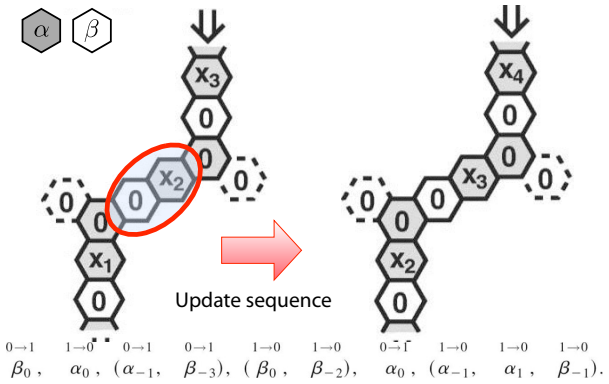

Figure 9.24: Shift of quantum information in Benjamin's QCA architecture. Dashed cells have fixed values.

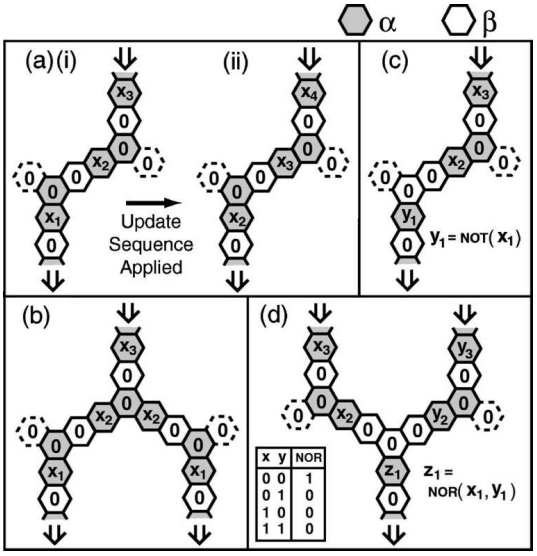

Figure 9.25: Cell arrangements for a) shift operation, b) COPY operation, c) NOT operation and d) NOR operation.

of v. Here, the term 'net field' is defined as the number of neighbors in the state 1 minus the number of neighbors in the state 0. As shown in figure 9.24, each cell can have up to three neighbors, so v can have values from -3 to +3. In this architecture, the type or the location of the neighboring cells is therefore not important.

If we consider the pair of cells marked in Fig. 9.24, the first two pulses of the update sequence change their state as follows:

	$\beta_0^{0\to1}$	$\alpha_0^{1\to0}$
00	10	10
01	01	00

Apparently, the cell containing the qubit value x_2 has been inverted and moved by one place down the string,

$$\Psi = |0, x_2\rangle \;\rightarrow\; \Psi' = |\mathrm{NOT}(x_2), 0\rangle.$$

Together, the combined effect of the update pulses and the fixed cells is a transport of the whole register by one logical qubit corresponding to three cells.

The same update sequence can be used with different cell arrangements to implement different operations. Figure 9.25 shows some examples.

Although the overhead, in terms of physical qubits and in terms of gate operations, is significantly larger with this scheme, it almost eliminates the problem of local addressing. The model may be well suited for an implementation based suitable physical qubits. As an example, it was proposed to be implemented with endohedral fullerenes as qubits [187]. Nitrogen and phosphorous endohedral atoms have long decoherence times and are well distinguishable. They could thus serve as type α and β cells.

9.4.5 One-way quantum computer

An even more radical deviation from the network computational model was suggested by Raussendorf and Briegel [188]. Their approach, which is referred to either as *one-way quantum computer* or *cluster-state quantum computer* or *measurement-based quantum computer,* replaces most unitary transformations by single-qubit measurements. These measurements must be performed in a specific sequence and in directions determined by the algorithm. Before these measurements can be performed, the system has to be brought into a highly entangled state (the "cluster state"). This approach therefore shifts the interactions between qubits from the process-

ing stage to the preparation stage and explicitly uses entanglement as a computational resource.

The cluster state can be generated by initializing all qubits into the state

$$\Psi_0 = \frac{1}{2^{N/2}} \left(|0\rangle + |1\rangle\right)^{\otimes N}$$

und subsequently letting it evolve under an Ising-type Hamiltonian

$$\mathcal{H}_I = J(t) \sum_{jk} S_z^{(j)} S_z^{(k)} \tag{9.1}$$

for a time τ such that $\int_0^\tau J(t)dt = \pi$. After this evolution, the first set of measurements prepares the state for the actual algorithm.

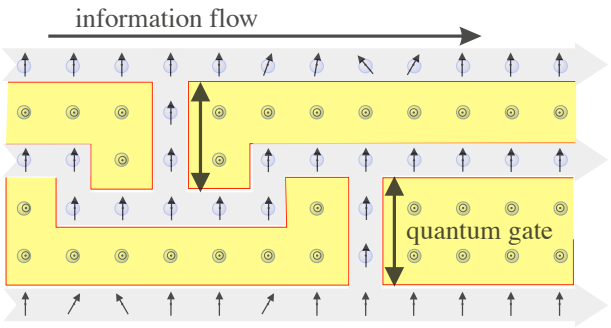

Figure 9.26: Two-dimensional cluster state. The qubits in the yellow segments are measured along the z-axis and thereby effectively removed from the cluster state. Vertical arrows indicate measurements in x-direction, tilted arrows in the xy-plane. (Adapted from [188])

As shown in Fig. 9.26, some qubits (marked in yellow) are measured along the z-axis, i.e. in the computational basis. The collapse of these qubits eliminates their entanglement with the rest of the cluster. The resulting state consists then of a tensor product of all measured qubits with an entangled state of the unmeasured qubits.

9.4.6 Example : CNOT

To process quantum information with this network, the particles are measured in a certain order and in a certain basis. Quantum information is thereby propagated horizontally through the cluster by measuring the qubits on the wire while qubits on vertical connections are used to realize two-bit quantum gates. The basis in which a certain qubit is measured depends in general on the results of the preceding measurements.

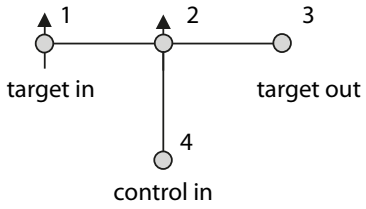

Figure 9.27: CNOT gate in the cluster-state computer.

Fig. 9.27 shows how a CNOT gate is implemented in a measurement-based quantum computer. The input state is encoded into the qubits marked 'target-in' and 'control-in', while qubits 2 and 3 are initialized into the states

$$|+\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle + |1\rangle\right).$$

The tensor product of these four qubit states is then subjected to the entanglement operation described above, i.e. the evolution under the Hamiltonian (9.1). Then, measurements are performed on qubits 1 and 2 along the x-axis, as marked by the arrows. This puts the state of qubit 3 into $|(i_1 + i_4) \bmod 2\rangle$, where $|i_k\rangle$ indicates that qubit k is in state $i = \{0, 1\}$. For the four possible values of the two input qubits, we obtain

00	0
01	1
10	1
11	0

.

This corresponds to a CNOT operation.

The proposed device appears to be at least as powerful as a network quantum computer and for certain tasks it is more powerful [188, 189]. For a possible implementation, it was suggested

to represent the qubits by atoms stored in an optical lattice [190] formed by the electric field of a standing light wave (→ section 11.6). Alternatively, photons can be prepared in cluster states [191], using, e.g. active feed-forward techniques [192].

9.4.7 Adiabatic computation

Another alternative computational model is the adiabatic quantum computer, which was originally proposed by Farhi et al. [157]. As in the case of the network model, the adiabatic model represents the result of a computational task by a quantum state Ψ_{res}. The main difference is that one does not worry about gate operations to generate this state from a chosen initial state. Instead, one designs a Hamiltonian such that its ground state corresponds to the target state Ψ_{res}. The only remaining task is then to force the system into its ground state. This can in principle be achieved by cooling, but in many systems, it is difficult or impossible to reach the ground state in finite time.

The solution for this problem is the adiabatic theorem of quantum mechanics. It states that

> A quantum mechanical system remains in its instantaneous eigenstate if the Hamiltonian changes sufficiently slowly and there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum.

To use this theorem, one first designs a system Hamiltonian for which the ground state can be prepared efficiently. A possible example is the initial Hamiltonian

$$\mathcal{H}_0 = -\Omega_0 \sum_i S_z^{(i)}.$$

Clearly, the ground state of this Hamiltonian is

$$\Psi_0 = |\uparrow\uparrow\uparrow \ldots \uparrow\rangle,$$

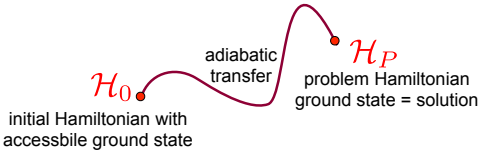

Figure 9.28: Schematic representation of adiabatic quantum computing.

which is often used as the initial state of an algorithm in the network model and relatively easy to prepare.

As shown in figure 9.28, the system Hamiltonian is then slowly transformed into the 'problem Hamiltonian' $\mathcal{H}_P$:

$$\mathcal{H}(t) = (1 - \frac{t}{T})\mathcal{H}_0 + \frac{t}{T}\mathcal{H}_P.$$

According to the adiabatic theorem, the system remains in the ground state of the instantaneous Hamiltonian [193] throughout this process and therefore encodes the solution when the system Hamiltonian has become the problem Hamiltonian.

The main challenges of this scheme are the design of the problem Hamiltonian and the adiabatic transfer process, making sure that the conditions for adiabaticity remain satisfied throughout.

9.4.8 Adiabatic factoring

As an example, we consider an adiabatic factoring algorithm [194]. It allows one to determine the prime factors of a number N by determining the ground state of the Hamiltonian

$$\mathcal{H}_P = \sum_{x,y} (N - xy)^2 |x, y\rangle\langle x, y|.$$

Here, $|x, y\rangle = |x\rangle \otimes |y\rangle$ represents a quantum register that encodes possible trial factors in two partial registers. The Hamiltonian is diagonal in this basis and the energy of each state is given by the square of the difference between the product xy of the trial factors and the number N to be factorized. If the product of the two trial factors is N, i.e. if they are the true factors, the energy

becomes zero, which is the lowest possible value. The ground state thus encodes the solution of the factoring problem.

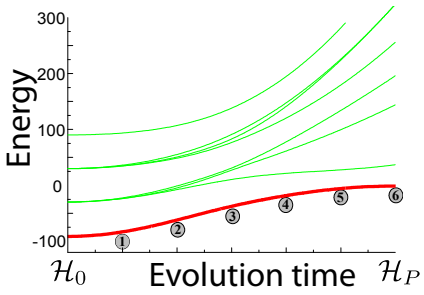

Figure 9.29: Energy levels during the adiabatic factoring algorithm.

Fig. 9.29 shows, as an example, the energy levels of this Hamiltonian as it changes slowly from $\mathcal{H}_0$ to $\mathcal{H}_P$. In this case, the initial Hamiltonian was chosen as

$$\mathcal{H}_0 = -\Omega_0 \sum_i S_x^{(i)}$$

and the number to be factorized was 21. Since this number is odd, the possible factors must also be odd. In a binary representation, the lowest-order qubit is therefore always 1 and it is not necessary to encode its value in the quantum register. It is then sufficient to use 1 qubit for the smaller of the two possible factors and 2 qubits for the larger one.

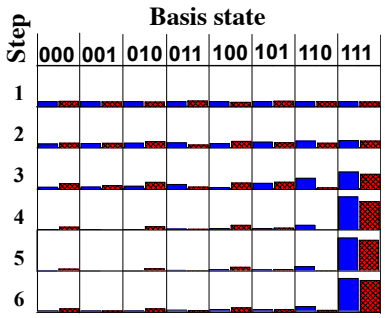

Figure 9.30: Measured probability distribution of the states during the adiabatic factoring algorithm. Blue bars show the theoretical prediction, red ones the experimental results.

Fig. 9.30 shows, how the ground state of this Hamiltonian changes during the adiabatic evolution. Six time points of the evolution are marked by the numbers 1-6 in Fig. 9.29. Initially, the system is in an equal superposition of all computational basis states. As the algorithm progresses, all the probability amplitudes decrease, except for that of the $|111\rangle$-state, which increases. The first two qubits (plus the not encoded lsb) thus represent the factor 7, while the third qubit, together with the implicit lsb, encode the second factor 3.

Adiabatic quantum computing is also the basis of the commercial quantum computer of the Canadian company d-wave, which uses superconducting microchips for processing the quantum information.