

Chapter 1

CHALLENGES IN RELIABLE QUANTUM COMPUTING

Diana Franklin

*Department of Computer Science
California Polytechnic State University, San Luis Obispo*
dkeen@csc.calpoly.edu

Frederic T. Chong

*Department of Computer Science
University of California, Davis*
chong@cs.ucdavis.edu

Abstract This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract. This is the abstract.

Keywords: quantum, fabrication, verification, error correction, nanotechnology

Introduction

While the ideas for quantum computing have been gestating for years, it has gained momentum in the past decade due to key discoveries in potential, feasibility, and implementation. In 1994, Shor presented an algorithm that, in polynomial time, can factor very large numbers. The fastest known classical algorithm requires exponential time. With Shor's algorithm, our current cryptographic techniques would be rendered useless, if a quantum computer could be built. Building a large-scale computer became more feasible with the Threshold Theorem [Aharonov and Ben-Or, 1997], which essentially showed (in theory) an error correction technique that can correct errors as quickly as they can be introduced into the system. Finally, several groups have successfully built 5-10

quantum bit computers[Vandersypen et al., 2000, Knill et al., 1999, Sackett et al., 2000].

As promising as these developments are, we are still a long way from building a reliable, large-scale quantum computer. The physical properties of quantum matter make it very difficult to build a reliable quantum computer. There are several aspects to this - error correction, communication and fabrication.

Quantum error correction is both more challenging and more cumbersome than in classical computing. The power of quantum computation lies in the fact that each quantum bit stores much more information than a classical bit. Unfortunately, this means that a simple bit-flip is just one of many errors that can occur. This, alone, would not be a large problem except that the quantum data can not be copied exactly, nor measured precisely. In fact, a direct measurement alone changes the state of a quantum bit! This leads to a solution that, in the best case, requires seven physical bits for every logical bit. This seven to one encoding is applied hierarchically - each time it is applied, the error rate is squared (reducing it). In essence, quantum computers are trading off space for time - they take less time to compute, but error-correction requires polynomial times more space for the extra bits []. In addition, an unreliable computation must be run more times to increase the chances of getting the desired result. The original error rate is higher as the length of computation increases and the underlying technology is less reliable. The hidden advantage of error correction taking so many more resources than the computation itself is that, regardless of how future quantum algorithms behave, an architecture optimized for error-correction will be the most efficient design. We need not wait for a benchmark suite to be developed to know how to design the architecture - error-correction is the only benchmark we need.

Optimizing for error-correction means dealing with the fact that quantum circuits will be very large (since so many bits are required). In order to have these bits interact, they need to travel long distances. Thus, quantum computers need to be optimized for communication. Traditional computation uses a wire to transport a copy of a value to a new location. In quantum computing, we have neither wires nor the ability to make copies. The bit must be moved, not copied.

A highly reliable fabrication technology reduces the amount of error correction necessary, so verification is vital. Because the quantum elements are so small and fragile, incredible precision is required, much more so than with classical computing. While this could be solved through rigorous testing, the test itself is difficult because, given the same operations, many quantum operations are not expected to always

produce the same measured result because there is no way to precisely measure the complete quantum state. As much as possible, classical methods must be used to test the fabricated chip.

We begin with an introduction to quantum computation, including some proposed implementations and basic algorithms. Error correction is presented in Section 2. Section 3 focuses on implementing a quantum computer in two technologies - silicon and ion traps. Architectural difficulties in communication are presented in Section 5, and we give conclusions in Section 6.

1. Quantum Computation

Just as classical systems can be represented in boolean algebra, with no specification of the implementation, quantum systems have a mathematical representation to describe their states and operations on those states. We begin by presenting quantum bits and operators in this mathematical abstraction, followed by several possible implementations. Finally, to give an idea of the power of quantum computing, we present two famous quantum algorithms. This is just a short introduction, and more information can be found in [Nielsen and Chuang, 2000b].

Quantum Bits and Operators

The basic building block in quantum computing is a *quantum bit*, or *qubit* for short. The main difference between a bit and a qubit is that a bit is restricted to the state 0 or 1. A qubit, on the other hand, is not restricted to its analogous states $|0\rangle$ and $|1\rangle$; It is also possible to form *linear combinations* of states

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \text{ which can also be expressed } \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

α and β are complex numbers, but thinking of them as real numbers is usually sufficient for our purposes.

Initially, this seems like a huge advantage over classical computing - the ability to store complex numbers within a single bit of data. There are two basic operations that we take for granted in classical computing that are difficult in quantum computing - measurement and replication.

There is no way to precisely measure the state of the quantum bit (i.e. determine α and β). Instead, when we measure the qubit, the outcome is either $|0\rangle$ or $|1\rangle$. The significance of the α and β values is that the probability of measuring a $|0\rangle$ is α^2 , and the probability of $|1\rangle$ is β^2 . Because there are only two possible outcomes, it must hold that $\alpha^2 + \beta^2 = 1$. Furthermore, the direct measurement of a qubit alters

its state. Even if the qubit has the actual state $\begin{bmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{bmatrix}$ and happens to be measured as a $|0\rangle$, from then on, it will always measure $|0\rangle$. The act of measurement changes α and β (in this case, α becomes 1, and β becomes 0).

The problem of measurement affecting the state would not be nearly as bad, surely, if we could copy the original qubit before we measured it. Then, we could measure the copy and retain the original for future computations. Unfortunately, qubits can not be copied without the destruction (through measurement) of the original bit. In classical computing, voltage can be applied and split off into an arbitrary number of wires, allowing one input bit to be outputted to several destinations. Quantum bits do not work this way. Instead, we would need to find a gate that takes as input the qubit to be copied and some known qubit, and output two identical qubits that match the first qubit. This would be analogous to having an or gate and knowing that if you input x and 0, you get x as the output. Unfortunately, for reasons we will see later, no such quantum gate can exist. These two issues - lack of measurement accuracy and the inability to copy bits - make it difficult to test fabrication and provide error correction in the classical way.

But we digress... Now that we have seen that a bit is no ordinary binary bit, but rather a vector that has the probability of expressing different results, what sort of operations occur on such a vector, and how does this turn into meaningful computation?

Because the qubit is expressed as a vector, single qubit operators, or *quantum gates*, are expressed as 2x2 matrices. In order to be a valid operator, the result must still conform to $\alpha^2 + \beta^2 = 1$. It turns out that any matrix which transforms a source vector with that property to a result vector with that same property is said to be *unitary*, that is $U^t U = I$, where:

$$U = \begin{bmatrix} a + bi & c + di \\ e + fi & g + hi \end{bmatrix} \text{ and } U^t = \begin{bmatrix} a - bi & e - fi \\ c - di & g - hi \end{bmatrix}$$

The requirement that the operator be unitary greatly restricts what can be done in quantum computing. Most classical operations are not unitary, because you can not get back the original value once the operation is performed. In fact, any operation that takes two bits and produces one is not unitary.

The ability to reverse computation has practical implications. This creates an explosion in the number of bits required to perform computation, because the information necessary to reverse all operations performed must be part of the computation. This is often in the form

Universal Quantum Operations

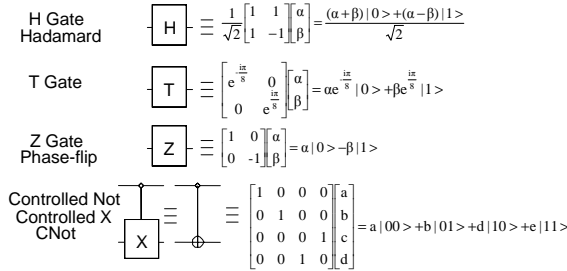


Figure 1.1. Important 1-bit quantum gates

of requiring extra bits from the very beginning. While each bit encodes much more information than in classical computing, this unitary requirement requires extra space, greatly increasing the number of bits required to execute an algorithm.

Let us play around with a few quantum gates. The first gate we present is the quantum NOT gate, which is traditionally called X. A classical not gate flips a 0 to a 1 and a 1 to a 0. The quantum equivalent flips the coefficients of $|1\rangle$ and $|0\rangle$.

$$\alpha|0\rangle + \beta|1\rangle \rightarrow \beta|0\rangle + \alpha|1\rangle.$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \text{ which has the effect: } X \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}$$

Some other important 1-bit quantum gates are shown in Figure 1.1

Things get a little more complicated when we add more qubits to the system. Each additional gate doubles the number of possible outcomes. Now, as before, the probabilities must sum to 1. The qubit state is:

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

where

$$\alpha_{00}^2 + \alpha_{01}^2 + \alpha_{10}^2 + \alpha_{11}^2 = 1$$

If we measure, say, the first bit, and it happens to be a 0, then our system has collapsed, and we have only two options left. Remember - that does not mean that α_{10} and α_{11} were zero; the act of measuring the first bit changes the system. α_{10} and α_{11} become 0 as a result of the measurement, so this may affect the other probabilities in order for the sum of the probabilities to be remain 1. The new qubit state ($|\psi'\rangle$) if the first bit is measured as 0 will be:

$$|\psi'\rangle = \frac{\alpha_{00}}{\sqrt{|\alpha_{00}|^2+|\alpha_{01}|^2}}|00\rangle + \frac{\alpha_{01}}{\sqrt{|\alpha_{00}|^2+|\alpha_{01}|^2}}|01\rangle \text{ or } \frac{\alpha_{00}|00\rangle+\alpha_{01}|01\rangle}{\sqrt{|\alpha_{00}|^2+|\alpha_{01}|^2}}$$

Quantum gates approximate classical circuits by including a control bit that allows the gate to satisfy the unitary requirement. Consider the controlled-NOT gate - the first bit determines whether or not the NOT operation will be applied to the second bit.

$$|00\rangle \rightarrow |00\rangle; |01\rangle \rightarrow |01\rangle; |10\rangle \rightarrow |11\rangle; |11\rangle \rightarrow |10\rangle;$$

which corresponds to:

$$\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \rightarrow \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{11}|10\rangle + \alpha_{10}|11\rangle$$

Some other useful gates are shown in Figure ??.

Although there is a mapping of classical algorithms to quantum computer, this is not the true power of quantum computing. The key is in restricting the set of possible outputs and making the different bits have a relationship to each other. For example, one could built a circuit that puts 8 bits through a series of transformations such that, for each of the possibilities for the first 4 bits (x), the second 4 bits was the solution to some function f(x). Upon measurement, you would find out one (x,f(x)) pair, though you would not know which one. If you were to somehow restrict the values of the first 4 bits, you could further control which (x,f(x)) pair you measured.

Proposed Implementations

Now that you have some idea of the mathematical properties of quantum computing, we move on to possible ways to implement this system. There have been several successful attempts at building small quantum computers.

Diana - fix this section!!!!

- Ike's latest attempt was successful, right? Or was August 2000 the most recent?
- August 2000, IBM-Almaden Research Center - 5-bit quantum computer
- March 2000, Los Alamos National Laboratory
- 1998, Los

Quantum Algorithms

Designers of quantum algorithms must be very clever about how to get useful answers out of their computations. One method is to iteratively skew probability amplitudes in a qubit vector until the desired value

is near 1 and the other values are close to 0. This is used in Grover's algorithm for searching an unordered list of n elements [Grover, 1996]. The algorithm goes through \sqrt{n} iterations, at which point a qubit vector representing the keys can be measured. The desired key is found with high probability.

Another option in a quantum algorithm is to arrange the computation such that it does not matter which of many random results is measured from a qubit vector. This method is used in Shor's algorithm for prime factorization of large numbers [Shor, 1994], which is built upon the quantum Fourier transform, an exponentially fast version of the classical discrete Fourier transform. Essentially, the factorization is encoded within the period of a set of highly probable values, from which the desired result can be obtained no matter what value is measured. Since prime factorization of large numbers is the basis of nearly all modern cryptographic security systems, Shor's algorithm has received much attention.

Additional algorithms include adiabatic solution of optimization problems [Childs et al., 2002]; precise clock synchronization [Jozsa et al., 2000, Chuang, 2000]; quantum key distribution [Bennett and Brassard, 1984]; and very recently, Gauss sums [van Dam and Seroussi, 2002] and Pell's equation [Hallgren, 2002].

2. Error correction

Perhaps the single most important concept in devising a quantum architecture is quantum error correction. Unlike classical systems, where error correction can be performed by brute-force, signal-level restoration in every transistor, correction of quantum states requires a much more subtle and involved strategy. In fact, localized errors on a few qubits can have a global impact on the exponentially large state space of many qubits, because of the non-local properties of entangled quantum states. In this section, we describe the basic idea of quantum error correction, and how it can be applied recursively to provide fault-tolerant quantum computation. We summarize problems with this traditional approach, and identify opportunities for optimization and key points for architectural design improvements.

The need for error correction is made manifest by a simple calculation. If a single qubit gate fails with probability $p = 1 - e^{-\lambda}$, then in the absence of error correction, the failure probability after n gates is at worst $1 - e^{-n\lambda}$. Thus, to have a circuit fail with at most probability ϵ , the computation would have to be less than $-\ln(1 - \epsilon)/\lambda$ gates in

length, in general. Given the technological considerations of Section 1, this would be a very short computation.

The difficulty of error-correcting quantum states arises from two obstacles. First, errors in quantum computations are distinctly different from errors in classical computing. Despite the digital abstraction of qubits as two-level quantum systems, the probability amplitudes of qubit states are parameterized by continuous degrees of freedom which are not automatically protected by the abstraction. Thus, errors can be continuous in nature, and minor shifts in the superposition of a quantum bit cannot be discriminated from the desired computation. In contrast, classical bits only suffer digital errors. Also, whereas classical bits may only erroneously suffer bit flips, quantum bits also suffer phase flip errors, since the signs of their amplitudes can be negative as well as positive. Second, quantum states must be corrected without measuring the state, because that would collapse the very superpositions we desire to preserve.

Quantum error correction codes successfully address these problems by simultaneously utilizing two classical codes to redundantly protect against both bit and phase errors, while allowing measurements to determine only information about the error which occurred and nothing about the encoded data. An $[n, k]$ code encodes k qubits of data using n qubits. The encoding circuit takes the k data qubits as input, together with $n - k$ *ancilla* qubits each initialized in the state $|0\rangle$. The decoder does the reverse: it takes in an encoded n qubit state, and outputs k (possibly erroneous) qubits, together with $n - k$ qubits which specify which error occurred, with high probability. A recovery circuit then corrects the error on the data, by performing one of 2^{n-k} operations. This model assumes that qubit errors are independent and uniformly distributed, just as does classical error correction. Deviations from this model can be treated similarly to classical strategies.

The effect of quantum error correction is powerful and subtle. Without this step the “correctness” (technically, the *fidelity*) of a physical qubit decays exponentially and continuously with time. Quantum error correction transforms this exponential error model into a linear one, to leading order: a logical qubit encoded in a quantum error correcting code and undergoing periodic error measurement suffers only linear discrete amounts of error.

Not all codes are suitable for fault-tolerant computation — many different quantum codes have now been discovered, and among these the largest class, *stabilizer codes*, play a special role: computation on these encoded states can be performed *without* decoding the data, and without overly propagating errors. For this reason, we focus on the [7, 1]

Steane code, which encodes one qubit using seven physical qubits and is nearly optimal (the smallest perfect quantum code is $[5, 1]$ [?]). This code has the marvelous property that an important set of single qubit operations, and the controlled-NOT operator (used in the quantum ALU discussed in the next section) can all be performed on the encoded qubit simply by applying the operations to each individual physical qubit.

The cost of error correction is the overhead needed to compute on encoded states, and to perform periodic error correction steps (note that the correction can also be done without decoding the data[Nielsen and Chuang, 2000a]). Each such step will be called a *fault-tolerant* operation. For the Steane code, about 153 physical gates are required to construct a fault-tolerant single qubit operation.

Despite this substantial cost, the 7 qubit error correcting code dramatically improves the situation for quantum computing. The probability of a logical qubit error occurring during a single operation changes from p to $c \cdot p^2$ where c is a constant determined by the number of unique points in the error measurement, recovery and the transform being applied to the logical qubit state, where two or more failures can occur and propagate to the output of the logical qubit. For a single logical gate application, c is about 17,446. For a physical qubit transform failure rate of $p = 10^{-6}$ this means the 7 qubit Steane code improves the probable logical qubit transform failure rate to $1.6 \cdot 10^{-7}$ when an optimized error measurement procedure is used[?] and the operation is maximally parallelized. However, more significantly, error correction transforms the exponentially decaying success probability $e^{-\lambda \cdot t}$ to a linear one of $1 - t \cdot p$.

Recursive Error Correction

The most impressive and important application of quantum codes to computation is a recursive construction[Aharonov and Ben-Or, 1997] which exponentially decreases error probabilities with only polynomial effort. This is crucial because even with the 7 qubit error correction that gives us an error probability of $c \cdot p^2$ is too high for most interesting quantum applications. The construction can be understood with the following example: The Steane code transforms the physical qubit error rate p to a logical qubit error rate $c \cdot p^2$ but requires some number of physical qubit gates per logical qubit gate operation. However, suppose instead that each of those physical gates were again implemented as a logical gate on a 7 qubit code. Each of those gates would have a logical gate accuracy of $c \cdot p^2$, and hence the overall logical gate error rate would become $c \cdot (c \cdot p^2)^2$. For a technology with $p = 10^{-6}$ each upper level gate

Recursion level (k)	Storage overhead 7^k	Operation overhead 153^k	Min. time overhead 5^k	Error probability
0	1	1	1	10^{-6}
1	7	153	5	$1.6 \cdot 10^{-7}$
2	49	23,409	25	$4.3 \cdot 10^{-10}$
3	343	3,581,577	125	
4	2,401	547,981,281	625	
5	16,807	83,841,135,993	3125	

Table 1.1. Overhead of recursive error correction for a single qubit operation, $c=17,466$, $p=10^{-6}$

would have an error rate of roughly $4.3 \cdot 10^{-10}$. The key observation is that as long as $cp^2 < p$, then error probabilities decrease *exponentially* with only a *polynomial* increase in overhead. Asymptotically, this gives rise to the Threshold Theorem, according to which quantum computation can be sustained for any finite length of time so long as the underlying technology has a reliability greater than $1/c$. The relevant inequality may be expressed as:

$$\frac{(c \cdot p)^{2^k}}{c} \leq \frac{\epsilon}{p(n)} \quad (1.1)$$

This relates the overall failure rate ϵ and space-time complexity $p(n)$ of an algorithm to the individual physical qubit error rate p , logical transform complexity c , and recursive error correction level k required to satisfy the inequality. Table 1.1 summarizes the costs of recursive error correction.

Clearly, due to the high cost of such error correction a quantum computer architecture should not choose the error correction method and recursion level indiscriminately. For a given algorithm and data size the minimum recursion level should be chosen in order to reduce the overhead.

3. Quantum Computing Technologies

A variety of technologies have been used to successfully construct quantum computing prototypes [Vandersypen et al., 2000, Knill et al., 1999, Sackett et al., 2000]. For quantum machines to scale to thousands or even hundreds of thousands of quantum bits, however, more scalable technologies are necessary. We will focus on two proposals: a long-term vision of ions implanted in silicon and a nearer-term scheme of micromachined ion traps.

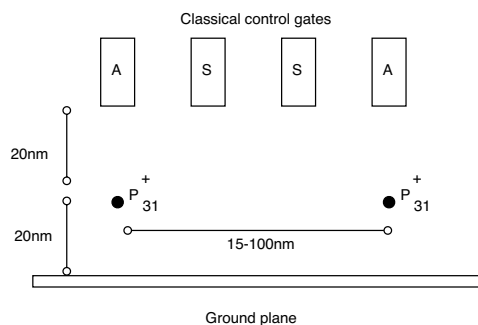


Figure 1.2. The basic quantum bit technology proposed by Kane [Skinner et al., 2002]. Qubits are embodied by the nuclear spin of a phosphorus atom coupled with an electron embedded in silicon under high magnetic field at low temperature.

Ions Implanted in Silicon

The Kane [Kane, 1998, Skinner et al., 2002] schemes of phosphorus in silicon builds upon modern semiconductor fabrication and transistor design, drawing upon understood physical properties.

Kane proposes that the nuclear spin of a phosphorus atom coupled with an electron embedded in silicon under a high magnetic field and low temperature can be used as a quantum bit, much as nuclear spins in molecules have been shown to be good quantum bits for quantum computation with nuclear magnetic resonance [Gershenfeld and Chuang, 1998]. This quantum bit is classically controlled by a local electric field. The process is illustrated in Figure 1.2. Shown are two phosphorus atoms spaced 15-100 nm apart. This inter-qubit spacing is currently a topic of debate within the physics community, with conservative estimates of 15 nm, and more aggressive estimations of 100 nm. What is being traded off is noise immunity versus difficulty of manufacturing.

Twenty nanometers above the phosphorus atoms lie three classical wires that are spaced 20 nm apart. By applying precisely timed pulses to these electrodes Kane describes how arbitrary one- and two-qubit quantum gates can be realized. Four different sets of pulse signals must be routed to each electrode to implement a universal set of quantum operations.

Micromachined Ion Traps

Nearer term, ion traps technologies provide a means to scale to perhaps thousands of quantum bits. Ion traps are one of the best understood technologies, with extensive experimental data describing their characteristics.

A typical ion trap, shown in Figure ?? contains up to half a dozen ions arranged in a linear array and trapped by a magnetic field. The ions are individually manipulated by hitting them with lasers of the appropriate frequency. This allows both quantum operations and measurement. Measurement occurs when a quantum bit is excited into either a phosphorescent state and a photodetector detects the photons emitted.

The key to scaling ion traps is the ability to move ions between traps via a series of electrodes and magnetic fields. As we shall discuss later, this motion is both slow and error prone, but allows us to build chips of micromachined ion traps, shown in Figure ??, with up to perhaps 100s or 1000s of bits [Kielpinsky et al., 2002].

Common Abstractions

At a basic level, both technologies we examine involve the spatial layout of quantum bits and control lines to implement quantum algorithms and circuits. Both will face challenges in fabrication and verification of the precisely placed bits and control signals, as well as in the communication of quantum data across each system. These are the open issues we will explore in the next two sections.

4. Fabrication and Test Challenges

Perhaps the most obvious difficulty in fabricating quantum computers is the small scale of the components and precision with which they must be placed in the system. Since reliable quantum operations are already challenging (discussed further in the next section) given a fabricated system with perfect spacing and alignment, variations are to be minimized and probably need to be detected. Furthermore, the use of quantum operations to test components should also be minimized.

Manufacturing

For the Kane technology, the first hurdle is the placement of the phosphorus atoms themselves. The leading work in this area has involved precise ion implantation through masks, and manipulation of single atoms on the surface of silicon [Kane et al., 1999]. For applications where substantial monetary investment is not an issue, slowly placing a few hundred thousand phosphorus atoms with a probe device [Globus et al., 1998] may be possible. For bulk manufacturing the advancement of DNA or other chemical self-assembly techniques [Adleman, 2000] may need to be developed. Note, while new technologies may be developed to enable precise placement, the key for our work is only the spacing (60 nm) of the phosphorus atoms themselves, and the number of control

lines (3) per qubit. The relative scale of quantum interaction and the classical control of these interactions is what will lead our analysis to the fundamental constraints on quantum computing architectures.

A second challenge is the scale of classical control. Each control line into the quantum datapath is roughly 10 nm in width. While such wires are difficult to fabricate, we expect that either electron beam lithography [Anderson et al., 1991], or phase-shifted masks [Sanie et al., 2001] will make such scales possible.

A remaining challenge is the temperature of the device. In order for the quantum bits to remain stable for a reasonable period of time the device must be cooled to less than one degree Kelvin. The cooling itself is straightforward, but the effect of the cooling on the classical logic is a problem. Two issues arise: first conventional transistors stop working as the electrons become trapped near their dopant atoms, which fail to ionize. Second, the 10 nm classical control lines begin to exhibit quantum-mechanical behavior such as conductance quantization and interference from ballistic transport [Ferry and Goodnick, 1997].

Fortunately, many researchers are already working on low-temperature transistors. For instance, single-electron transistors (SET's) [Likhareve, 1999] are the focus of intense research due to their high density and low power properties. SET's, however, have been problematic for conventional computing because they are sensitive to noise and operate best at low temperatures. For quantum computing, this predilection for low temperatures is exactly what is needed! Tucker and Shen describe this complementary relationship and propose several fabrication methods in [Tucker and Shen, 2000].

Testing

Once fabricated, qubits and control will be difficult to test. Tolerances are tight and it may be necessary to avoid using qubits in the system that are spaced incorrectly or have control signals that are misaligned.

Probably the most effective test for the spacing of control signals is to inspect, using an SEM or other device, the pattern of small 10 nm vias above each ion before they are covered by subsequent layers of metal. Connectivity from wide control wires to the vias will have to be verified via a quantum test program.

The spacing and alignment of the ions that implement the qubits is also problematic. Defects could be caught via quantum test programs, but the test would have difficulty distinguishing between ion spacing errors, misalignment between control vias and ions, and control via spacing

errors. Efficient test patterns will be needed to test individual qubits and the two-qubit operations between neighboring qubits.

5. Architectural Challenges

Basic Geometric Constraints

On the other hand, the quantum-mechanical behavior of the control lines presents a subtle challenge that has been mostly ignored to-date. At low temperatures, and in narrow wires, the quantum nature of electrons begins to dominate over normal classical behavior. For example, in 100 nm wide polysilicon wires at 100 millikelvin, electrons propagate ballistically like waves, through only one conductance channel, which has an impedance given by the quantum of resistance, $h/e^2 \approx 25 \text{ k}\Omega$. Impedance mismatches to these and similar metallic wires make it impossible to properly drive the AC current necessary to perform qubit operations.

Avoiding such limitations mandates a geometric design constraint: narrow wires must be short and locally driven by nearby wide wires. Using 100 nm as a rule of thumb¹ for a minimum metallic wire width sufficient to avoid undesired quantum behavior at these low temperatures, we obtain a control gate structure such as that depicted in Figure ???. Here, wide wires terminate in 10 nm vias that act as local gates above individual phosphorus atoms.

Producing a line of qubits that overcomes all of the above challenges is possible. We illustrate a design in Figure ???. Note how access lines quickly taper into upper layers of metal and into control areas of a classical scale. These control areas can then be routed to access transistors that can gate on and off the frequencies (in the 10's to 100's of MHz) required to apply specific quantum gates.

Quantum Communication

We examine the problem of scaling a quantum system by focusing on perhaps the primary task of quantum computer – error correction. Recall that error correction is applied recursively to achieve enough fault tolerance to allow sustainable quantum computation. The basic quantum error correction circuit can be implemented with a double row of quantum bits, one for ancilla and one for the actual quantum data. Recursively applying more levels error correction results in a natural H-tree structure, as shown in Figure ??.

A crucial trait of this recursive structure is that communication distances increase as we approach the root of the tree for any substantial

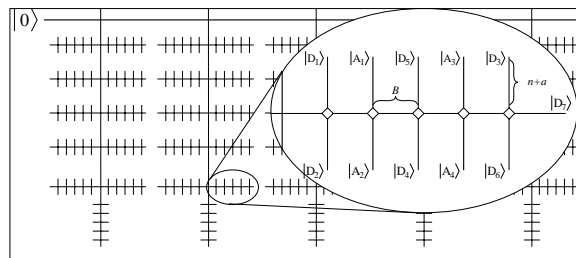


Figure 1.3. Schematic layout of the H-tree structure of a concatenated code. The branches in the inset represent the logical two-rail qubits, with the bold lines representing ancillae.

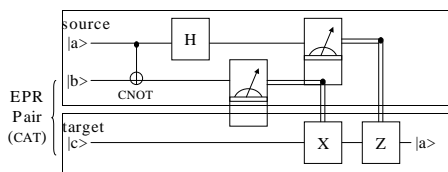


Figure 1.4. Quantum Teleportation of state $|a\rangle$ over distance. First, *entangled* qubits $|b\rangle$ and $|c\rangle$ are exchanged. Then, $|a\rangle$ is combined with $|b\rangle$ after which measurements produce two *classical* bits of information (double lines). After transport, these bits are used to manipulate $|c\rangle$ to regenerate state $|a\rangle$ at destination.

level of recursion. In fact, the naive approach of swapping quantum data from bit to bit becomes untenable. There are too many swaps to accomplish without error correction, yet we are trying to construct the basic error correction circuit! In fact, it is possible to apply intermediate error correction, but this would substantially increase the overhead of an already costly process. Instead, we examine another method of achieving quantum communication over long distance – quantum teleportation.

Quantum teleportation

Quantum teleportation is the re-creation of a quantum state at a distance. Contrary to its science fiction counterpart, quantum telepor-

tation is not instantaneous transmission of information. Rather, it uses an entangled *EPR pair*, $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ [Bell, 1964].

Figure 1.4 gives an overview of the teleportation process. We start by generating an EPR pair. We separate the pair, keeping one qubit, $|b\rangle$, at the source and transporting the other, $|c\rangle$, to the destination. When we want to send a qubit, $|a\rangle$, we first interact $|a\rangle$ with $|b\rangle$ using a CNOT gate. We then measure $|a\rangle$ and $|b\rangle$ in the computational basis, and send the two one-bit classical results to the destination, and use those results to re-create the correct phase and amplitude in $|c\rangle$ such that it takes on the original state of $|a\rangle$. The re-creation of phase and amplitude is done with X and Z gates, whose application is contingent on the outcome of the measurements of $|a\rangle$ and $|b\rangle$. Intuitively, since $|c\rangle$ has a special relationship with $|b\rangle$, interacting $|a\rangle$ with $|b\rangle$ makes $|c\rangle$ resemble $|a\rangle$, modulo a phase and/or amplitude error. The two measurements allow us to correct these errors and re-create $|a\rangle$ at the destination. Note that the original state of $|a\rangle$ is destroyed when we take our two measurements. This is consistent with the “no-cloning” theorem, which states that a quantum state cannot be copied.

Why bother with teleportation when we end up transporting $|c\rangle$ anyway? Why not just transport $|a\rangle$ directly? First, we can pre-communicate EPR pairs with extensive pipelining without stalling computations. Second, it is easier to transport EPR pairs than real data. Since $|b\rangle$ and $|c\rangle$ have known properties, we can employ a specialized procedure known as *purification* to turn a collection of pairs partially damaged from transport into a smaller collection of asymptotically perfect pairs. Third, transmitting the two classical bits resulting from the measurements is more reliable than transmitting quantum data.

6. Conclusions

References

- [Adleman, 2000] Adleman, Leonard (2000). Toward a mathematical theory of self-assembly. USC Tech Report.
- [Aharonov and Ben-Or, 1997] Aharonov, D. and Ben-Or, M. (1997). Fault tolerant computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 176–188.
- [Anderson et al., 1991] Anderson, E.H., V.Boegli, Schattenburg, M.L., Kern, D.P., and Smith, H.I. (1991). Metrology of electron beam lithography systems using holographically produced reference samples. *J. Vac. Sci. Technol.*, B-9.
- [Bell, 1964] Bell, John S. (1964). On the Einstein-Podolsky-Rosen paradox. *Physics*, 1:195–200. Reprinted in J. S. Bell, *Speakable and Un-speakable in Quantum Mechanics*, Cambridge University Press, Cambridge, 1987.
- [Bennett and Brassard, 1984] Bennett, Charles H. and Brassard, Gilles (1984). Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179.
- [Childs et al., 2002] Childs, A. M., Farhi, E., and Preskill, J. (2002). Robustness of adiabatic quantum computation. *Phys. Rev. A*, (65).
- [Chuang, 2000] Chuang, Isaac L. (2000). Quantum algorithm for clock synchronization. *Phys. Rev. Lett.*, 85:2006.
- [Ferry and Goodnick, 1997] Ferry, David K. and Goodnick, Stephen M. (1997). *Transport in Nanostructures*. Cambridge Studies in Semiconductor Physics & Microelectronic Engineering, 6. Cambridge University Press, Cambridge.
- [Gershenfeld and Chuang, 1998] Gershenfeld, N. and Chuang, I.L. (1998). Quantum computing with molecules. *Scientific American*.
- [Globus et al., 1998] Globus, Al, Bailey, David, Han, Jie, Jaffe, Richard, Levit, Creon, Merkle, Ralph, and Srivastava, Deepak (1998). Nasa

- applications of molecular nanotechnology. *Journal of the British Interplanetary Society*, 51.
- [Grover, 1996] Grover, L. (1996). In *Proc. 28th Annual ACM Symposium on the Theory of Computation*, pages 212–219, New York. ACM Press.
- [Hallgren, 2002] Hallgren, Sean (2002). *Quantum Information Processing '02 Workshop*.
- [Jozsa et al., 2000] Jozsa, R, Abrams, DS, Dowling, JP, and Williams, CP (2000). Quantum atomic clock synchronization based on shared prior entanglement. *Phys. Rev. Lett.*, pages 2010–2013.
- [Kane et al., 1999] Kane, B. E., McAlpine, N. S., Dzurak, A. S., Clark, R. G., Milburn, G. J., Sun, He Bi, and Wiseman, Howard (1999). Single spin measurement using single electron transistors to probe two electron systems. *arXive e-print cond-mat/9903371*. Submitted to *Phys. Rev. B*.
- [Kane, 1998] Kane, Bruce (1998). A silicon-based nuclear spin quantum computer. *Nature*, 393:133–137.
- [Kielpinsky et al., 2002] Kielpinsky, D., Monroe, C., and Wineland, D.J. (2002). Architecture for a large-scale ion trap quantum computer. *Nature*, 417:709.
- [Knill et al., 1999] Knill, E., Laflamme, R., Martinez, R., and Tseng, C.-H. (1999). A cat-state benchmark on a seven bit quantum computer. *arXive e-print quant-ph/9908051*.
- [Likhareve, 1999] Likhareve, Konstantin K. (1999). Single-electron devices and their applications. *Proceedings of the IEEE*, 87.
- [Nielsen and Chuang, 2000a] Nielsen, M. A. and Chuang, I. L. (2000a). *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, UK.
- [Nielsen and Chuang, 2000b] Nielsen, M.A. and Chuang, I.L. (2000b). *Quantum computation and quantum information*. Cambridge University Press, Cambridge, England.
- [Sackett et al., 2000] Sackett, C.A., Kielpinsky, D., King, B.E., Langer, C., Meyer, V., Myatt, C.J., Rowe, M., Turchette, Q.A., Itano, W.M., Wineland, D.J., and Monroe, C. (2000). Experimental entanglement of four particles. *Nature*, 404:256–258.
- [Sanie et al., 2001] Sanie, Michael, Cote, Michel, Hurat, Philippe, and Malhotra, Vinod (2001). Practical application of full-feature alternating phase-shifting technology for a phase-aware standard-cell design flow.

- [Shor, 1994] Shor, P. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *Proc. 35th Annual Symposium on Foundations of Computer Science*, page 124, Los Alamitos, CA. IEEE Press.
- [Skinner et al., 2002] Skinner, A. et al. (2002). Hydrogenic spin quantum computing in silicon: a digital approach. *quant-ph/0206159*.
- [Tucker and Shen, 2000] Tucker, J. R. and Shen, T.-C. (2000). Can single-electron integrated circuits and quantum computers be fabricated in silicon? *International Journal of Circuit Theory and Applications*, 28:553–562.
- [van Dam and Seroussi, 2002] van Dam, W. and Seroussi, G. (2002). Efficient quantum algorithms for estimating gauss sums. *quant-ph*, page 0207131.
- [Vandersypen et al., 2000] Vandersypen, Lieven M.K., Steffen, Matthias, Breyta, Gregory, Yannoni, Costantino S., Cleve, Richard, and Chuang, Isaac L. (2000). Experimental realization of order-finding with a quantum computer. *Phys. Rev. Lett.*, December 15.