NanoEngineering World Forum

Nanocomputer Systems Engineering

Michael P. Frank

University of Florida
Dept. of Computer & Information Science & Engineering
and Dept. of Electrical & Computer Engineering

Abstract

We introduce *Nanocomputer Systems Engineering* (NCSE), the integrated, interdisciplinary study of the engineering analysis and global optimization of complex, high-performance computer systems composed of nanoscale components. At the nanometer scale, near the fundamental physical limits of computing, we find that computational and physical considerations are closely intertwined, and must be optimized together in order to come close to the maximum system-level cost-efficiency that is theoretically possible. In our approach to the subject, called *Physical Computing Theory*, we explore new technology-independent, fundamental-physics-based nanocomputing models designed to remain applicable *regardless* of any future technological developments that may occur in the course of future nano-device engineering. Our model incorporates the important capabilities of *quantum computing* and *reversible computing*, which have been shown to be necessary for maximizing system-level cost-efficiency for special-purpose and general-purpose applications, respectively. Our model is argued to be the most powerful model of computing that is consistent with the known laws of modern physics. We outline how to use the model to systematically analyze and optimize key figures of merit for nanocomputer systems. Using these methods, we project that by mid-century if not earlier, nanocomputers using reversible computing will be *thousands* of times more cost-effective for most general-purpose high-performance applications than contemporaneous technology based on traditional digital logic. Such results motivate increased research on reversible computing technologies, and we summarize our current efforts in this area.

Author Biography

In Mike Frank's freshman year at Stanford, he took (and aced) the world's first class on nanotechnology, taught by nanotech pioneer K. Eric Drexler. During later M.S. and Ph.D. research at MIT, Mike designed one of the first molecular universal computers, based on DNA chemistry, and created the first universal computer chip to rely entirely on the principles of thermodynamically reversible computing. Since 1999 he has been teaching graduate subjects including computer architecture and the physical limits of computing at the University of Florida, and he is currently writing a book on Nanocomputer Systems Engineering to be published next year by CRC Press.

1. Introduction

In this paper and talk, we summarize the key foundational principles and major early results of *Nanocomputer Systems Engineering* or NCSE [1], a crucial new engineering discipline that is positioned at the intersection of computer systems engineering and nanoengineering. NCSE concerns the interdisciplinary, multi-domain study of the engineering analysis and optimization of digital systems composed of nanoscale devices, accounting for both computational and physical considerations. This integrated perspective is shown to be necessary in order to optimize overall system efficiency (by most measures) in the face of key constraints on the physics underlying computation, constraints which become evident at the nanoscale, including fundamental limits arising from basic thermodynamics and quantum mechanics. We emphasize the limits that are technology-independent, and so will still apply *regardless* of any future technological developments that may occur in the course of future nano-device engineering.

The methodology that we propose for NCSE is to use new, physically-based theoretical models of computing that encompass all of the relevant physical constraints on nanocomputing, as well as two important capabilities that have only recently been explored, known as *reversible computing* [2] and *quantum computing* [3], respectively, which together promise to improve system-level efficiency for many, even most digital applications. This new, theoretically well-founded approach to NCSE is called *physical computing theory*, and it can be shown to be strictly more relevant and accurate for real nanocomputer engineering than are the traditional models of computing that form the basis of the "classic" theories of computational complexity which have been the focus of the bulk of theoretical computer science to date.

We describe our new canonical physical model of computing, outline how to systematically analyze and optimize key figures of merit for nanocomputer systems within it, and we summarize some of the interesting results that have already been proven using these methods. One of the most interesting results is that by the middle of this century, nanocomputers that can use *reversible computing* capabilities should be literally *thousands* of times more cost-effective for most general-purpose high-performance applications than contemporaneous technology that lacks the capability of reversibility [1]. Such results motivate intensified research into reversible computing technology, and we summarize some of the current efforts being made in this direction.

2. Nanocomputer Systems Engineering

From Systems Engineering to NCSE. *Systems Engineering* in general is a very broad subject, and one that is defined in many ways by different sources (for one example, see [4]). But for our purposes in this paper, we will take systems engineering to mean the engineering study (that is, study using mathematical and scientific principles) of the design, analysis, and optimization of systems which may possibly be complex, and which may in general involve interactions between subsystems that may involve many different specific engineering sub-domains, such as electrical, mechanical, thermal, and computational domains. *Computer Systems Engineering* is, then, just this concept of systems engineering applied specifically to the design of information-processing systems, for example to design and optimize the logical organization of a computer while simultaneously accounting for things like communications delays and power and cooling requirements among its various parts. And finally, *Nanocomputer Systems Engineering*, is just this same concept of Computer Systems Engineering, but now specialized for nanocomputing systems in particular. I prefer to define a *nanocomputer* as any information-processing system in which the very smallest logical devices (those which can store and/or manipulate at least 1 bit's worth of computational state information, *e.g.*, a logic gate) are packed together with a spatial pitch that is, on a logarithmic scale, closer to 1 nanometer than to 1 micrometer, or in other words, less than about 32 nanometers [5].

2.1. Why NCSE?

Nanocomputer Systems Engineering is worth approaching as a new field, distinct from traditional (Micro-) computer systems engineering, for the simple reason that nanometer-scale devices are closely approaching various fundamental physical limits [6] to the maximum efficiency that computers can possibly have, according to a wide variety of measures. As a result, the design considerations that are significant in analyzing and optimizing these systems cannot be effectively separated into independent computational and physical considerations, as is usually done in computer engineering today, but instead the computational and physical domains turn out to heavily interact. One finds that to design the most efficient systems overall, both the logical organization and the physical characteristics of the low-level devices must be analyzed together. It has been shown [5] that one cannot converge towards the design of the most efficient nano-devices if one ignores certain important high-level architectural issues, and similarly, one cannot find the best logical architectures if one is stuck with using nano-devices that ignore certain critical characteristics. Let us now survey some of the new fundamental computational-physical interactions that come into play at the nanoscale.

2.2. Fundamental Physics of Computing

Thermodynamics of information erasure. A prime example of the fundamental computational-physical interactions, and one that I will emphasize throughout this paper, is the tight interplay between logical and thermal considerations at the nanoscale, which is due to the extremely close relationship, between digital information and physical *entropy*, which is the second most important concept (after energy) underlying the entire field of thermodynamics. As a result of this connection, which is by now very well-understood within the physics-of-computing community, it turns out that the energy efficiency of nanocomputers cannot possibly be optimized while ignoring the logical structure of the computation being performed.

Physical entropy, that originally-mysterious ratio of heat to temperature whose importance was first noticed by Rudolph Clausius in 1854, is now very well understood (as a result of various developments in modern physics and the physics of computing) to be simply a measure of that portion of the information content of a physical system that cannot be effectively compressed into a smaller amount of information by any available method [7] (for a closely related view, see [8]). Physical information may have this status of incompressibility for a variety of alternative reasons, either because its content is completely unknown, or because the content is totally disordered and random, with no discernable pattern. Even if there is some underlying pattern to the information contained in a system, if the pattern is unknown to the observer in question, or would be infeasibly difficult to discover and/or compress away, then that information is *effectively* entropy for that observer.

As an example, suppose that someone hands you a 10-kilobyte electronic document, but the document has been encrypted and digitally compressed. If you don't have the encryption key, you can't determine what the original document said, and since it's already compressed (by the best available method, let's assume), it's impossible for you to compress it further.

The modern understanding of this situation tells us that the 10 kilobytes or 80 kilobits of digital data contained in that document is, for you, *really in fact, physical entropy,* just as surely as the unknown information that is encoded in the random positions and velocities of the molecules of air in the room in which you are sitting is another example of physical entropy.

As a result of this absolutely *perfect* (I emphasize) fundamental identity between digital and physical entropy, together with the 2nd Law of Thermodynamics (which states that entropy cannot be destroyed), you *cannot* clear that 10KB compressed document from your computer without being forced to emit its entropy into the computer's surroundings, an amount of entropy given by 80 kb = 80,000 bits = $80{,}000 \cdot (\log_e 2)$ nats = $80{,}000\ k_B \ln 2$, where k_B is Boltzmann's constant, which is the natural-log unit of entropy or information. Here we have just converted from the base-2 unit of information (the bit) to

the base-*e* unit of information (the nat) which is more convenient for calculations in statistical mechanics.

Note however that in this modern viewpoint, the logical status of information—that is, its relationships and correlations with other available information—determines whether that information is effectively entropy or not. Suppose, for example, that I am now given the encryption key to the document. Suppose that after decompressing and decrypting the document, we notice that it turns out to contain just a printout of the first ~30,000 digits of π, that is 3.14159…. Well, digits of π can be computed using a simple and tractable algorithm, and so we can now "compress" the document down, replacing it with just a few bits which encode the statement that the original document just contained the first 30,000 digits of π. Note that this is a perfectly valid compression method, since the entire document could have been reconstructed absolutely perfectly from that short description. So, the information in that file is *not* really entropy, for anyone who has access to the decryption key, although it remains entropy from the point of view of someone who does not.

You may be thinking, "where's the physics here?" But although above we have been discussing a conventional data-processing example for clarity, I want to emphasize now that all of this is fundamentally no different from what happens in basic thermodynamic processes in physics. For example, suppose we take a quantum system, say an isolated molecule, that is prepared in a very precisely-known quantum state, whose entropy is nearly zero (experimental physicists do this routinely)—this is our analogue to the simple π file—and we allow this molecule to then interact in a predetermined way with another precisely-prepared system, say another molecule (consider it to be the "key") whose quantum state someone *else* knows perfectly, but we do not. The effect of this interaction is in general to disturb or "encrypt" the state of our molecule; due to its interaction with the unknown key system. From our perspective, our molecule generally ends up in a statistical mixture of quantum states, which has higher entropy than the original state had. (We could remove the entropy from the molecule by measuring its state, but this would just transfer that entropy into the system that records the result of the measurement, not eliminate it.)

But now, suppose that someone tells us the exact initial quantum state of the other "key" system with which our molecule interacted. We can now apply the known laws of quantum mechanics to simulate the interaction precisely (in principle), and calculate exactly what the joint quantum state of the two molecules should have become after the interaction—so now, the two molecules (considered together) still constitute a system with zero entropy. Since their state is perfectly known, we can now arrange for some additional manipulations to return them back to their starting state (thus "compressing away" the information that resulted from the collision), and then we can store other information of interest in the newly "blank" state.

At first glance, these examples might seem irrelevant. Isn't a far greater amount of entropy being generated by the experimental apparatus and by the experimenter while he is carrying out these manipulations and deductions? This is true given the scenario described, but it is unnecessary to the basic concept. We can do away with the experimental setup and the experimenter—an exactly analogous process can be carried out, in a thermodynamically reversible fashion, by, for example, individual nanoscale logical devices that are performing computational operations in a nanocomputer. This last example is what drives home and makes relevant the basic connection between computational information and physical information, because a nanodevice is so small that a significant portion of the physical information that it contains *is* the actual digital data that is being manipulated. The basic connection between computation and physics can no longer be ignored at this point.

Here is the example. Suppose that a certain nanodevice (device *A*) contains 1 bit of digital data (0 or 1) that is encoded in its physical information. A second neighboring nanodevice (device *B*) then interacts with the first device, according to some known physical interaction that we have prearranged to take place by appropriate design of the hardware. The result of this interaction is (say) to multiply the

first bit by another nearby bit in device *C* (logical AND) and store the result in device *B*. Afterwards, some other nearby nanodevice uses the result in device *B* to perform some further computation.

Now, device *B* contains 1 bit of computed data. What is the status of this particular piece of information? Is it incompressible physical entropy, or is it compressible known information? The crucial point that has emerged from modern studies is that *this choice depends entirely on the epistemological viewpoint that is hardcoded into the logical and mechanical design of the nanosystem*. Here is why. If the circuit design (courtesy of its designer) takes the stance that device B now just contains some random old temporary result that is no longer needed, some meaningless bit of garbage, then the designer probably has designed the logic mechanism to treat that old information *as if it were entropy*, ignoring its correlations with the other accessible information, and just throw it away by overwriting it with new information. But if one disposes of information using a process that would work equally well for entropy, in being so disposed of, the information actually *becomes* entropy, that is, the information cannot just magically disappear—this is guaranteed by the reversible, *unitary* nature of quantum mechanics—but instead must be emitted into the environment; it becomes unknown information in the environment, literally entropy, and contributes to the heat emission from the computer.

And in fact, *all computers today* (outside research labs) take this stance towards *every bit* of computed information—they treat it just as if it were entropy, and wantonly discard it when they need to make space for new information, and in doing so transform the old information into entropy. In fact, since this manner of disposal is built into the very mechanism of all current devices, we can say that the information effectively becomes entropy the moment that it is produced within a system whose design is hardcoded to never bother compressing that information away.

But an alternative *is possible*! This is by recognizing that we do *have the key* to unlock and compress the information. That is, the information in device *B* is really *not entropy* if we embody in the nano-circuit design the realization that this information is totally correlated with other bits that are available nearby—namely, that $B=A\cdot C$ in this example. If we design the computational mechanism to subsequently *undo* the operation that computed *B*, then we can *uncompute* the bit in device *B*, effectively "compressing it away" (this is a "compression" and not an "erasure" of the data since going forwards would generate it again), restoring the device to a standard state that can now receive a new piece of computed data. In such an alternative design, the fact that the information in *B* can be compressed away means that it is *not entropy*, and the 2nd law of thermodynamics places no restriction on what can become of it. So, we can compute and then later uncompute the data in *B* using a thermodynamically reversible mechanism, and emit essentially no entropy into the environment during the process.

This scenario is not at all fantasy, but is today done routinely in the lab, in quantum computing experiments using digital data encoded in the quantum states of individual atoms [3]. The reversible version of the logical AND operation $B=A\cdot C$ is called a Toffoli gate (it was first described by Tomasso Toffoli at MIT, now at Boston University), and it has been implemented successfully within single molecules and in various other physical setups for quantum computing.

Reversible logic has been experimentally validated in superconducting circuits, in conventional electronics based on semiconducting field-effect transistors, in quantum dots, and in many other setups. So we now know for sure: Computed information need not be treated as entropy, and instead can be reversibly *uncomputed* instead. Such a reversible process need not generate significant physical entropy.

In contrast, today's computers treat as entropy (and transform into entropy) not only every bit of digital data that they process, but *also* the *much larger* amount of physical information that they use to encode that data.

The equivalence between digital information and physical information becomes especially relevant to nanocomputer design for two reasons:

(1) The efficiency of the encoding of digital information can absolutely not be reduced below the point where only one bit's worth of physical information is being used to encode each bit of digital information. Therefore the entropy generation and total heat dissipation of ordinary irreversible computation *must cease improving* once this limit is approached. At present rates of improvement, this must happen sometime within the next few decades.

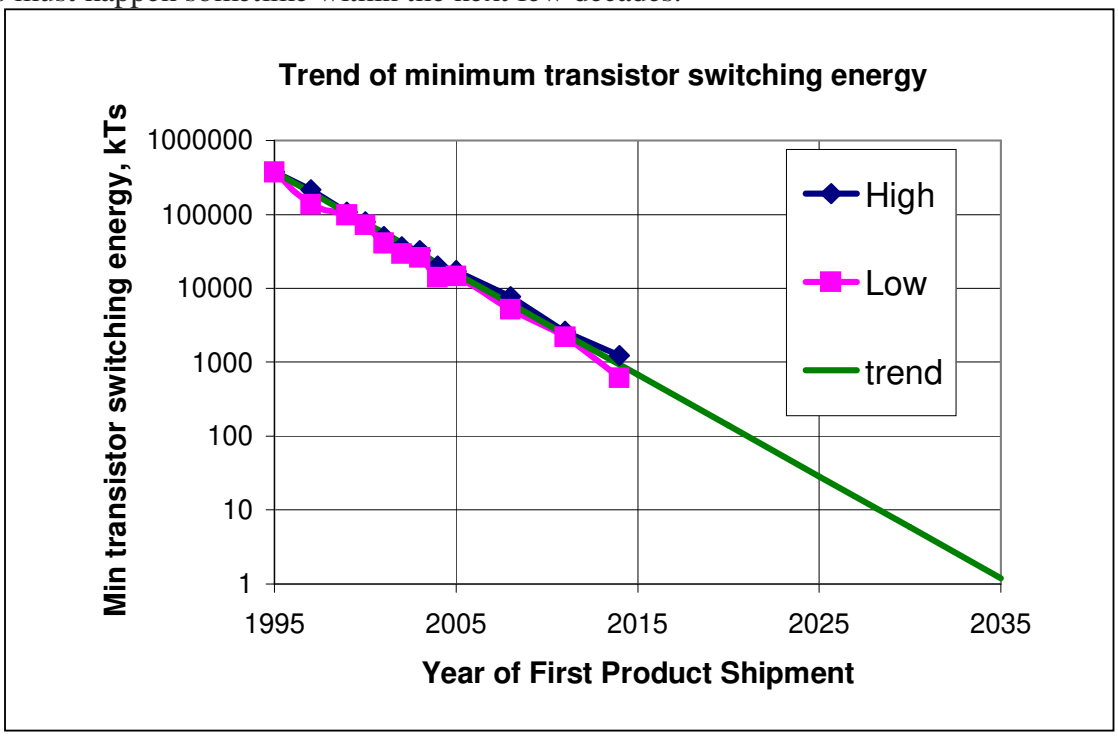

Figure 1. Trendline of minimum transistor switching energy, based on figures specified by the 1999 International Technology Roapmap for Semiconductors [9]. 0.69 *kT* (just off the right end of the chart) is the absolute minimum energy per bit stored at a given temperature, corresponding to just 1 bit of physical entropy. However, reversible computing may reduce the energy *dissipated* by reversible bit-operations below this point.

(2) Moreover, the physical *size* of bit-devices in normal atomic matter must also stop shrinking, once a significant fraction of the physical information in the material is being used to encode individual digital bits. Ordinary materials at normal temperatures and pressures have at most only a few bits per atom worth of variability that might potentially be utilized for information storage—this figure can be deduced from empirically-determined thermochemical data for these materials. Therefore, beyond this point, the only way to increase the information storage capacity for a system of given size will be to build up many layers of storage elements, in three dimensions, which we do not do much today. Keeping such a system cool, however, will require reversible computing.

Temperature and clock rate. Another recent realization in the physics of computing is the close connection between physical *temperature* and the rate of updating of information ("clock speed" in today's computers) [10,5]. A generalized temperature T for any system implies an average rate of updating of quantum information within that system of $f = 2T\mathrm{b}/h$, where h is Planck's constant, and $\mathrm{b} = k_B \ln 2$, the bit-unit of information. This formula gives the average rate of nontrivial orthogonalizing quantum operations per bit, and thus the maximum attainable rate of updating for traditional binary logic, *i.e.*, the maximum "clock speed" or operating frequency that is possible. Solving this formula for $T = hf/2\mathrm{b}$, the $f = 3$ GHz clock speeds that we enjoy today could not possibly be attained in any system at a generalized temperature lower than 0.1 K. And in warmer systems at room temperature (300 K), clock speeds no higher than ~8.7 THz are possible. Note that this is less than 3,000 times faster than today's computers, and in fact it is only slightly faster than the fastest transistors projected at the end of the

present semiconductor roadmap. After the 20 or so years that current trends would require for computer technology to achieve such a speedup, if not sooner, no further improvements in device operating frequency would subsequently be possible without the machine's internal components (its "moving parts"—be they molecules, electrons or photons) necessarily operating proportionately hotter than room temperature. As an example, an update rate of 100 THz would imply an internal temperature of about 3,500 K—hot enough to melt most ordinary materials. Therefore, any further improvements would require an increasingly high degree isolation of the computational degrees of freedom from the computer's structural material. The evolution of the computational mechanism would need to be increasingly self-contained, closed, unitary, adiabatic, and reversible.

The relevance of this fundamental connection between temperature and clock rate for nanocomputing is twofold:

(1) Once the nanoscale is reached, increasing clock speed necessarily requires increasing device temperature towards and beyond room temperature. The increased temperature proportionately increases the energy cost of bit erasure and the energetic advantages of reversible computing.

(2) Maintaining increasing operating temperatures will in any event require an increasing degree of isolation of these hotter-running devices from their cooler-temperature structural surroundings, just to prevent the machine's superstructure from melting. Such isolation is also a requirement for reversible operation.

(3) Once clock speeds have reached the limit of tolerable temperatures, overall throughput of nanocomputers can *only* be further improved by increasing the number of devices. Since devices will no longer be able to get smaller beyond a certain point (as noted above), to pack more devices into an enclosure of given diameter will require increasing the number of layers of devices in 3 dimensions. This too will require reversible computing, in order to avoid running up against other fundamental limits [6] on the heat flux that can be tolerated.

Information propagation speed. This is of course limited to the speed of light. As a result, the most varied class of computational tasks—those that can benefit from some degree of parallel processing, while still requiring some amount of communication between those processors—are ultimately limited by the speed-of-light signal propagation delays between processing elements. These delays can only be reduced by packing devices closer together, which is one of the benefits of the continuing device shrinkage that has taken place fairly continuously [11] over the last hundred or so years since the birth of digital computing (in the form of the first mechanical adding machines). However as we discussed above, this reduction in device size must soon halt (within the next few decades) for fundamental thermodynamic reasons. At this point, the only way to reduce propagation delays further is to compact circuits from their current two-dimensional layouts into smaller-diameter, three-dimensional structures. But this will lead to heat-removal problems if reversible computing techniques are not used.

2.3. Basic Principles of NCSE

In this section we summarize some of the key principles that we believe will be important for nanocomputer systems engineering.

Models of computing based on fundamental physics of computing. We describe such a model in the following section. Such a model is necessary because of the tight interplay between physical and computational quantities that we discussed in the previous section. All of the traditional models of computing that have been used in virtually all of the work that has taken place in theoretical computer science to date fail to take these fundamental physics of computing issues into account, and as a result, they fail to provide suitable models for the design and scaling analysis of nanocomputers [5].

Use technology-independent models for generality of results. Currently, a wide variety of radically differing device technologies have been proposed for nano-scale computing, everything ranging from carbon nanotubes to semiconductor nano-wires to electron or nuclear spins on

semiconducting islands or in organic molecules. Without knowing yet which of these proposed technologies (or some other still undiscovered technology) will turn out to be the most practical, how can we model nanocomputing? The answer is to construct generic, technology-independent models that are based on universal considerations such as information, energy, entropy, etc. Any physical technology can have its relevant performance characteristics summarized in terms of such quantities. If our model is generic in this fashion, we can be sure that it will apply to any future nano-device technology. Yet, as we will see, it is still possible to prove interesting, nontrivial results (such as various performance and cost-efficiency scaling analyses) on top of such a generic model.

Obey universal engineering principles. There are some basic engineering principles that are universal and span all domains of engineering. Of course our NCSE methodology should obey these principles. One of them is the Generalized Amdahl's Law [12]: essentially a law of diminishing returns that applies to the effect of reducing one component of any additive expression of cost (in whatever units, that is, generally, any figure of demerit) in isolation without also reducing the other components. Some corollaries of the Generalized Amdahl's Law are that the majority of design effort should always be spent on those design components that account for the majority of the cost, and that a well-optimized design will tend to have roughly comparable costs for all of its important components.

Use three-phase system design optimization. A complex system with many levels of design (from low-level bit-devices all the way up through high-level system architectures) can nevertheless be globally optimized using what I call a three-phase optimization procedure (additional description may be found in the poster [1], found at http://www.cise.ufl.edu/research/revcomp/NanoTech03-Poster-no-narration.ppt):

- **Phase 1** involves composing expressions (analytical or computational) describing the key performance characteristics (both computational and thermodynamic) of higher-level components in terms of lower-level ones.
- **Phase 2** is to extract (from the performance expressions) optimization formulas (analytically or numerically) for the design parameters at each level, given the optimization formulas to be used at the higher levels.
- **Phase 3** is to pick the values of design parameters at the lowest level that optimize system-level performance, and work our way upwards to determine the global solutions with optimized parameters at all levels.

In previous work such as [1], we have used this three-phase system design optimization procedure successfully to show how to optimize the device-level and algorithm-level parameters of reversible computers concurrently, to achieve the maximum cost-efficiency for entire general-purpose computing systems.

Include all important aspects of computer modeling. A model of computing that is sufficient for most real-world computer engineering purposes should include: a *device model*, a *technology scaling model*, an *interconnection model*, a *timing model*, an *architectural model*, a *capacity scaling model*, an *energy transfer model*, a *programming model*, an *error handling model*, a *performance model*, and a *cost model*. What each of these means is explained in more detail in the article [5], and below.

3. Physical Computing Theory

Physical computing theory is the study of formal theories of real-world computing that take into account all the fundamental issues in the physics of computing such as we summarized in §2.2. Two of the earlier models of physical computing theory that we proposed were the R3M (Reversible 3-D Mesh) and Q3M (Quantum 3-D Mesh), described in [2]. A more recent refinement of these models is the unnamed model described in [5]. In this section we present that model again, in a slightly more refined form, and give it the new name CORP (Computer with Optimal, Realistic Physics).

3.1. Background: A Computational Model of Physics

Before we present our physical model of computation, let us briefly describe a simple and increasingly conventional (*cf.* [13]) type of asymptotically precise computational model of (nongravitational) physics. This is important because it makes plain the limits of what physics could possibly do for us, computationally speaking.

For simplicity, let us assume that we have an infinite flat Euclidean spacetime (fixed reference frame), and let us model space with a three-dimensional mesh of discrete spatial locations at integer coordinates. Such a discretization of space may be only an approximation of reality; however, there are many indications that distances smaller than the Planck length ℓ_P are not physically meaningful, and many emerging theories of quantum gravity (e.g., loop quantum gravity [14]) explicitly treat spacetime as a discrete network of events at that scale, and these theories are showing increasing indications of success.

We will associate each spatial location with a small constant number of qubits (quantum bits) worth of state information. The information could specify, for example, the occupancy numbers at that location for each type of fundamental particle (*e.g.*, photons of given polarization, electrons of given spin, the various flavors and colors of quarks and gluons, *etc.*). In principle, each location interacts continually with its 6 nearest neighbors in the $+x$, $-x$, $+y$, $-y$, $+z$, and $-z$ directions. However, we can discretize time also, while approximating a continuous interaction, by, for example, performing short pairwise interactions between each location and its neighbors in each of the 6 directions sequentially. *E.g.*, there could be 6 phases per time step. In phase 1, each point with even-numbered x coordinate interacts with its neighbor in the $+x$ direction, in phase 2 the $-x$ direction, and then similarly with the other 2 dimensions and 4 directions. Each pairwise interaction can always be modeled as a simple unitary quantum interaction that is independent of the location in space, the time, and the choice of direction. The result of these interactions is in general an entangled (correlated) quantum state for the whole system. The granularity of the temporal and spatial discretization can be arranged so that that the maximum propagation speed of any disturbance is the speed of light, and apparently even proper relativistic behavior can be recovered. Thus it seems likely that with proper design of the interaction Hamiltonian and the node configuration space, most if not all of Standard Model physics can be captured fairly accurately within such a framework, indeed, related *lattice gauge* theories have already proved very successful as computational models of field theory [15].

Implications of the model. Since it seems that physics itself can be modeled, as described above, as a 3-D mesh of fixed-capacity quantum processors, interacting locally (also called a Q3M or quantum 3-D mesh, see [2]), it follows that no physically realizable model of computing can be more powerful asymptotically than such a model. Therefore, we base our CORP model of computing loosely on this framework.

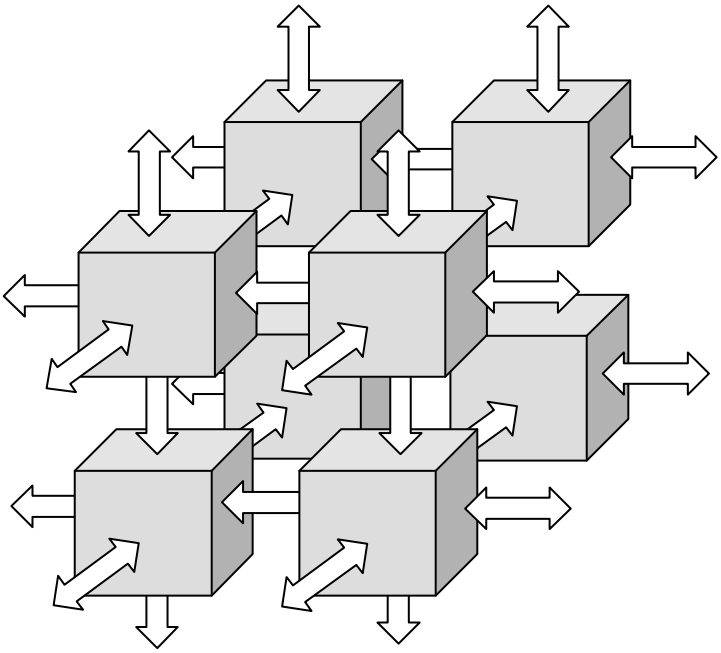

Figure 2. In lattice models of quantum physics, continuous 3-D space is broken up into discrete mesh of locations, each containing only a finite amount of quantum information, and interacting locally and reversibly with its nearest neighbors. Such models seem able to accurately simulate most if not all aspects of mainstream quantum field theory, at least in the limit as the lattice spacing approaches the Planck length.

3.1. The CORP Model

1. Device Model. A device model is a model for the lowest-level functional components of a machine design. In conventional circuits, transistors and wires are the primary devices. For us, the devices may more generally be anything from a quantum dot to a spintronic valve to a molecular electromechanical resonator; and anything from a nanotube laser, to a coolant-transport micro-channel.

In our CORP model, every computer system in any technology is conceived as an assemblage of *devices* that occupy fixed, non-overlapping connected regions of physical space. The restriction to devices held at fixed locations is not really a restriction; moving parts can still be represented as a flow of energy and information between fixed locations. And the restriction to non-overlapping regions is not strictly necessary either—with a little extra bookkeeping effort, overlapping "devices" that incorporate different degrees of freedom within the same physical space can also be accommodated, *e.g.*, intersecting free-space optical interconnects.

In the most flexible form of the CORP model, different device types may have different shapes and sizes, as convenient to represent the components of a particular technology. However, in a given technology, we can assume that there is a limited catalog of available device types, and some minimum physical diameter that the constituent devices may have.

Each device is conceptually divided into *coding* and *non-coding* subsystems; the coding subsystem is the part of the device state that is intentionally varied, to encode digital data and/or timing information. The non-coding subsystem is divided into *structural* and *thermal* subsystems; where the structural part is defined as the part that must remain unchanged in order for the device to work properly, while the state of the thermal part is allowed to vary. Furthermore, the coding subsystem is divided into the logical subsystem, which is the actual digital data of interest that is being manipulated, and the redundancy subsystem, which accounts for the remaining, redundant information that is present in the coding subsystem for purposes of noise immunity, error detection, and error correction.

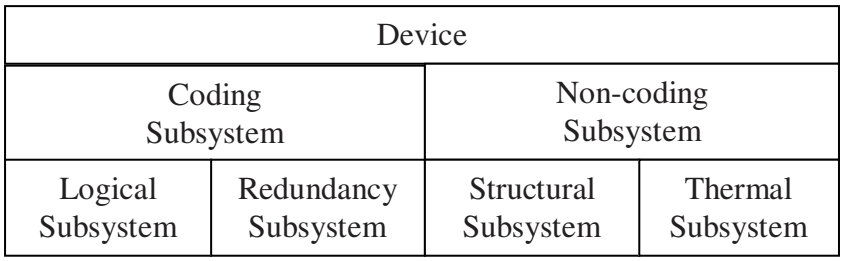

Figure 3. Conceptual hierarchy of subsystems within a CORP device.

Each subsystem is conceived as a quantum system which interacts with itself and the other subsystems in the device, and with the corresponding subsystems in adjacent devices. We assume the state space of each subsystem can be factored into qubits, or more generally into quantum systems of $\leq k$ discrete states for some small constant k. Each interaction is most generally specified by a quantum *Hamiltonian* (a matrix summarizing the quantum interaction). Each interaction can also be characterized by its associated interaction *temperature*, a generalized temperature which gives the average rate at which the quantum state information involved in the interaction will be transformed into a distinguishable state by that interaction.

In our model, all the interactions are required to be local and time-independent. The locality can be ensured by composing each subsystem out of fixed-size cubes of space that interact directly only with their nearest neighbors, just as in the Q3M model of physics described earlier. This locality constraint is important to ensure that the speed-of-light limit is not violated. And, the time-independence of the model is required to ensure that we are not imagining that we can sneak in some external control signals in a way that does not actually scale physically (*e.g.* such as if we ignored dissipation in the control system).

Modeling irreversibility. In a pure Hamiltonian formulation of the device dynamics, the quantum state of the entire system would evolve reversibly and coherently, with no accumulation of entropy. Although this is possible in principle, given a system that is prepared in a sufficiently precise initial state and is isolated to have sufficiently negligible interactions with uncontrolled external environments, in practice it is infeasible to achieve perfect reversibility, either because of imperfections in the system's structure, in our knowledge of physics, or in the isolation of the system from its environment, and so, we need to model any departures from reversibility that the device has.

Such modeling is the purpose of the thermal subsystem. All of our modeling imprecision is stuffed into the thermal subsystem, which includes all aspects of device state that are not rigidly controlled and predictable. For simplicity, we ignore any entanglements between the thermal subsystem and the rest of the system, factoring the quantum state of the entire system into separate density matrices describing mixed states for the thermal and non-thermal parts of the system separately.

This approach could conceivably backfire if correlations ejected into the thermal subsystem were not sufficiently well randomized by the interactions that subsequently took place in that subsystem, with the result that the mixture model for the thermal system was no longer applicable, and correlations between the thermal and non-thermal subsystems persisted enough to affect overall system behavior. However, we will assume that this problem can be repaired by making the thermal subsystem sufficiently complex that correlations with it are effectively irrecoverable, as in a pseudo-random number generator.

With this separation of thermal and non-thermal subsystems in place, processes such as the decoherence of the non-thermal subsystem via parasitic interactions with the thermal system, the resulting generation of entropy within the non-thermal subsystem (irreversibility of the dynamics), and the intentional isolation and ejection of this entropy into the thermal system (signal restoration / error correction / refrigeration) can all be naturally accommodated in the model. Jayes first showed in the 1950's that irreversibility and the 2nd law of thermodynamics were natural consequences of quantum interactions with unknown components [16], and Zurek extended these insights in the 80's to show how

the emergence of decoherent, classical behavior also follows naturally from pure unitary quantum mechanics in this type of model [17]. Incidentally, such insights have caused the fundamental correctness of basic quantum theory to be increasingly widely accepted in recent decades, to the point where we can now say that there is no aspect of modern physical phenomenology (aside from gravity and the actual values of the theory's free parameters) that is not in principle explained, or at least presumed to be explainable, via the fundamental mechanisms in the Standard Model.

There are two main advantages of our overall approach to device modeling proposed above:

(1) It forces us to account explicitly for the localized flows of energy, entropy and information through the machine, so that we cannot make errors in modeling associated with the ignoring of thermodynamic constraints.

(2) In characterizing device interactions abstractly, in terms of quantum interactions between qubits of unspecified detailed physical implementation, it allows the model to be useful as a technology-independent framework for exploring the interplay between device-level and systems-level issues.

2. Technology Scaling Model. A technology scaling model in general describes how device's performance characteristics change as we improve the technology in some regular way, *e.g.*, by making devices smaller, operating them at higher or lower temperatures, *etc.* In previous papers such as [1], we described some simple technology scaling models for projecting capabilities forward from current technology to the nanoscale in coming decades.

Once the nanoscale is reached, however, further scaling will no longer proceed in the same way. Beyond the atomic scale, devices will no longer be made smaller, and instead improvements can come only through modifying the device and circuit architectures, *e.g.*, to improve the isolation between the computational and thermal degrees of freedom while increasing the generalized temperature (operating frequency) of the computational modes, for greater performance. However, even this process has its limits, since as internal temperatures increase, eventually increasing pressures must be applied from outside in order to maintain a solid structure. In a conceptual extreme limit of this process, the material of the computer itself would need to be compressed to enormous densities. This would indeed mean that scaling beyond the nanoscale would in fact be achieved. There are some examples in nature of material at densities much greater than that of normal-pressure solid matter, for example, material in the cores of stars, particularly white dwarf stars and neutron stars. However, it is at best a very distant prospect to use material of such density for computing, since for instance it is unclear whether there is any means to apply the required enormous pressures other by using the gravitational pressure of several stellar masses' worth of surrounding material. Although this vision of stellar-scale engineering and highly compressed, sub-nanoscale computing appears to break no fundamental laws of physics, it seems certainly not to be a realistic prospect for the foreseeable future of technology development (say, the next hundred years or so).

Therefore, we can foresee a natural end to scaling of operating frequencies when computational temperatures reach the maximum level that can be tolerated within material at ordinary densities and pressures. For example, at room temperature (300 K), the thermal energies of individual degrees of freedom are only about 26 meV, which corresponds to a maximum frequency of quantum evolution of 12.5 THz according to the Margolus-Levitin theorem [18]. If considerably higher particle energies, *e.g.*, on the order of 1 eV (the rough order of magnitude of molecular energy barriers) can be actively yet nondissipatively sustained, corresponding to a computational temperature of ~11,600 Kelvins, then in principle a frequency of about 500 THz might be obtained. But at significantly higher energies than this, nanoscale confinement of such energetic degrees of freedom becomes difficult (*e.g.*, moving particies with much higher energies would no longer be confined by molecular-scale barriers of any material) other than perhaps by moving to the highly compressed regime dismissed earlier.

Once operating frequencies cease improving, the only thing left to improve computer performance (besides developing novel algorithms, such as quantum algorithms) is to increase the

number of devices operating in parallel. This however increases energy dissipation proportionately, so performance per unit power is not improved, unless improvements are made at the device level. As a result, the number of devices that can feasibly be packed within an enclosure of given diameter is limited far more by heat dissipation concerns than by the volume of the individual devices.

How much can performance per unit power be increased? Although performance per unit power *transmitted* between states is forever limited by the Margolus-Levitin theorem, performance per unit power *dissipated* has no known fundamental upper limit, and only the detailed design and testing of specific nanodevices will tell us how low the energy dissipation per operation can be made to be.

In fact, somewhat counter-intuitively, a detailed analysis [5] suggests that beyond a certain point, improving performance-per-power will eventually require making devices *larger* (rather than smaller) in order to reduce energy leakage. The increases in device size required are relatively modest (growing only logarithmically as leakage rates are decreased) and the resulting overheads of increased device cost, reduced operating frequency, and increased communications delay are greatly outweighed by the energy-efficiency improvements, which allow far greater numbers of devices to be packed in a given enclosure, and a far greater rate of operations to be attained at a given power level.

Finally, the last important aspect of device scaling (other than size, performance, and power) is *cost*—the manufacturing cost per device in an integrated system. However, we can hope that due to increasing automation and other economic improvements, the cost per device can continue to decrease even after the fundamental lower bounds of device size (atomic scale) are reached. This will be necessary to continue increasing the integration scale (number of devices per system) and performance beyond the next few decades. However, nanomanufacturing issues are very complex and so we have not attempted to make a detailed model of device cost scaling.

3. Interconnection Model.

An interconnection model in general describes how information is communicated between the logical devices in a system.

One advantage of the generic device model presented earlier is that interconnects can be treated as just another type of device—one whose function is simply to propagate a signal from one end to another as faithfully as possible. All present-day interconnect technologies, including simple wires, transmission lines, optical interconnects, and phonon transmission in nanomechanical systems and ion-trap quantum computers can be accurately described within our model, as can all other physically possible interconnect technologies.

4. Timing Model.

A timing model specifies how operations are synchronized in time.

Again, we can treat timing signals within our model as being carried by designated interconnects which are just another type of device. Self-timed asychronous circuits as well as globally clocked synchronous operation can be captured in our model. Importantly, the model forces us to pay attention to the power dissipation that is associated with timing functions. However, reversible and locally synchronous timing of systems that scale arbitrarily in 3 dimensions (up to limits set by energy leakage) is known to be possible, and this approach turns out to be required in the long run for maximally cost-efficient nanocomputing.

5. Architectural Model.

An architectural model specifies how the logical functions of the computer are organized. It extends from circuits up through complete processors. In the architectural model, we require that the architecture should provide for efficient reversible and quantum-coherent operation when needed, and should not impose any unnecessary overheads preventing the asymptotically most cost-efficient possible reversible and quantum algorithms from being programmed and executed.

For completeness, the pathways for supplying power, transmitting timing signals, and removing waste heat should also be included in the architecture, especially in a three-dimensional system design. However, insofar as these functions can be separated in the design at a high level, this is helpful for engineers specializing in different aspects of the system.

6. Capacity Scaling Model.

The capacity scaling model specifies how the capacity of the machine can be scaled up to handle larger and larger problems, and how its performance scales as a consequence. A simple type of capacity scaling model is the multiprocessor model, where we just provide increasing numbers of identical processors, each with associated local memory and storage. However, the choice of interconnection network between the processors is critical. Networks other than mesh-like networks cannot physically scale. A mesh (locally-connected) interconnection pattern is therefore critical. And, for processing elements connected 3-dimensionally, thermal concerns cannot be ignored, even at this high multiprocessor-system level. Ideally the system model includes explicitly (as devices) all of the pathways for heat removal out of the machine, to ensure that serious mistakes are not made that lead to overheating and failure of the system.

We have also previously suggested [7,19,2] a capacity scaling model for 3D reversible nanocomputing systems in which overall power-performance is maximized by scaling down the frequency of individual processors as the number of processors is increased, with frequency roughly in inverse proportion to the square root of the machine diameter, in order to leverage increasing adiabaticity of slower-speed reversible processors. This approach is valid, but is only viable as long as the devices are not already running at the lower limit of dissipation set by the energy leakage rates of the device technology.

7. Energy Transfer Model.

The energy transfer model or thermodynamic model accounts for the flow of energy and entropy through the machine. As we have already said, accounting for this is forced by our device model, and pathways for transfer of energy and waste heat can be considered as just another type of device, which can be accommodated within our model.

8. Programming Model.

The programming model specifies how the machine is to be programmed to carry out desired computations. To save space we will not delve into this here. Suffice it to say that conventional programming models (instruction set architectures, *etc.*) are more or less satisfactory for the time being, although in the long run they need to be modified to allow programmers to express and efficiently execute hand-coded reversible and quantum algorithms for specific problems of interest. See [5] for more discussion and references.

9. Error Handling Model.

This specifies how errors that accumulate in the machine state are detected and corrected or worked around. Again we will not discuss this in detail here. Various mechanisms for noise immunity and error correction (both classical and quantum) can be accommodated within our model. Importantly, errors are a form of entropy, and our model forces one to account for the resources need to expel that entropy from the machine.

10. Performance Model.

The performance model specified how the performance of a machine scales as it is applied to larger problems. Since our model accounts for all physical constraints, it automatically implies a realistic performance model, in contrast to many of the other models of computing that have been studied in

theoretical computer science. We claim that our model is a *Universally Maximally Scalable* model, which means that it is the asymptotically most efficient model of computing that is physically implementable.

11. Cost Model.
The cost model is a generalization of the performance model that takes other components of total real-world cost of a computation (besides execution time) into account. For example, there are costs associated with time-amortized manufacturing cost of the machine, and with energy dissipation. We claim that our model also provides a maximally cost-efficient scalable model of nanocomputing.

3.2. Using the Model

One difficulty with the type of model outlined above is that a full quantum-mechanical simulation of the model is in general intractable (without using a quantum computer), due to the usual difficulties in simulating quantum mechanics (namely, tracking interference effects throughout an exponentially large Hilbert space). Therefore, when simulating and analyzing the model we must in general make simplifying "classical" approximations to the full model. For example, when not executing quantum algorithms, we can model the state variables of the system classically, and describe the interactions between devices as classical reversible (or nearly reversible) operations. However we must be careful in this approximate approach not to forget to model important sources of energy dissipation, such as, for example, phonon emission into supporting structures, or electromagnetic radiation from transitioning electronic components. Ultimately, such an approximate model can only be validated with certainty through the process of actually building and experimenting with real devices.

However, despite the need to only approximate the model in simulations, we claim that the full quantum model is nevertheless the right one, theoretically speaking, to serve as the conceptual foundation for the detailed design of nanocomputers, since it does achieve our goal of taking all the relevant fundamental physical considerations into account.

4. Results to Date

The full quantum model proposed above still taking shape, and no detailed analyses are available for it yet (and may not be feasible to obtain). However, an earlier model, which can be viewed as an appropriate classical approximation to the full quantum model, has previously been described and analyzed in detail [1] via numerical optimizations using the three-phase procedure described earlier, together with a set of reasonable technology scaling assumptions covering the next 50 years or so of nanocomputer technology development.

An interesting result of this analytical work is that reversible computing technology will begin to become for cost-effective than ordinary irreversible computing in about the next 10 years, and will be on the order of 1,000 to 100,000 times as cost-effective (depending on the application) by the 2,050s.

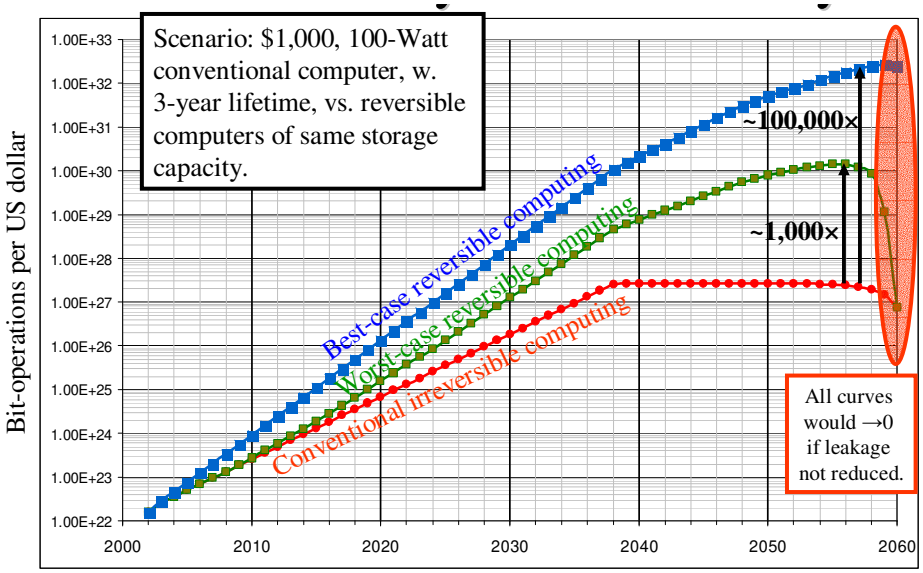

Figure 4. Cost-efficiency of reversible computing vs. irreversible computing over the next half-century. Even when taking all of the overheads of reversible computing into account, due to its greater energy efficiency, reversible computing achieves vastly greater overall cost-efficiency for general-purpose computing by the middle of the century.

A key assumption made in that work was that device leakage could be made small and kept small as devices approached their fundamental minimum size. However, doing this will require clever engineering at the device level. One of the goals of my own work is to convince the community of nano-device engineers and physicists to focus increased attention on the reversible computing paradigm when they design new devices; to design devices with reversible operation in mind, and to minimize as much as possible the energy leakage parameters which turn out to be ultimately the key limiting factor on system performance.

5. Future Work & Conclusion

Most work of course still remains to be done in the nascent new field of NanoComputer Systems Engineering. The sort of generic, technology-independent quantum model that was outlined above and was used (in approximation) to generate our results to date needs to be fleshed out in detail, for the case of analyzing the system-level cost efficiency of various nano-device designs that have been proposed and demonstrated in the lab.

But, even more importantly, the results that we have obtained already show that it will be critical for the success of future nanocomputing to pay more attention to reversible mechanisms as a basis for computing, and to design and engineer nanodevices with reversible operation firmly in mind. For example, an increasing focus must be placed on minimizing energy losses by improving isolation and reversibility of devices, since energy dissipation will ultimately be the true limiting factor on nanocomputer power-efficiency and cost-efficiency. In the meantime, device engineers should continue to reduce device size and increase device frequency towards the fundamental quantum limits—but without ever forgetting that the smallest possible device may not turn out to actually be the best one, at the system level, when energy dissipation issues are considered.

Some other important issues to be addressed in forthcoming reversible computing research, which I and my students are currently pursuing at UF, include:

(1) Design of appropriate hardware description languages for designing and synthesizing mostly-reversible logic circuits [20].
(2) Design of high-quality resonators (probably based on NEMS technology) for driving the timing signals of reversible electronic circuits.

(3) Design of hand-optimized reversible algorithms for specific applications of widespread interest (*e.g.*, digital signal and image processing).

At present, there is still a fair bit of time (a couple more decades, at least) before the fundamental thermodynamic limits to the energy efficiency of ordinary irreversible computing are really reached. But, we emphasize that those limits can *only* ever be worked around through the use of reversible computing. If we hope to ensure that the progress of nanocomputer technology will continue advancing throughout the lives of our children and grandchildren, bringing along with it ever-increasing benefits to society, then we must pay increasing attention to this subject in future NCSE research.

References

1 Michael P. Frank, "Nanocomputer Systems Engineering," in *Nanotech 2003: Technical Proceedings of the 2003 Nanotechnology Conference and Trade Show*, held Feb. 23-27, 2003, San Francisco, CA, vol. 2, pp. 182-185, http://www.cise.ufl.edu/research/revcomp/theory/NanoTech2003/.

2 Michael P. Frank, *Reversibility for Efficient Computing*, Ph.D. dissertation, Massachusetts Institute of Technology, Dept. of Electrical Eng. & Computer Science, June 1999.

3 Michael A. Nielsen and Isaac L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2000.

4 Benjamin S. Blanchard and Walter J. Fabrycky, *Systems Engineering and Analysis*, 3rd ed., Prentice-Hall, 1998.

5 Michael P. Frank, "Nanocomputers—Theoretical Models," review chapter to appear in the *Encyclopedia of Nanoscience and Nanotechnology*, Hari Singh Nalwa, ed., American Scientific Publishers, 2003, http://www.cise.ufl.edu/research/revcomp/Nanocomputers.doc.

6 Michael P. Frank, "Physical Limits of Computing," *Computing in Science and Engineering*, **4**(3):16-25, May/June 2002.

7 Michael P. Frank and Thomas F. Knight, Jr., "Ultimate Theoretical Models of Nanocomputers," *Nanotechnology* **9**(3):162-176, 1998, also presented at the 5th Foresight Conf. on Molecular Nanotechnology, Palo Alto, CA, Nov. 1997, http://www.cise.ufl.edu/~mpf/Nano97/abstract.html.

8 Wojciech Hubert Zurek, "Algorithmic Randomness, Physical Entropy, Measurements, and the Demon of Choice," in Anthony J. G. Hey, ed., Feynman and Computation: Exploring the Limits of Computers, Perseus Books, 1999.

9 Semiconductor Industry Association, "International Technology Roadmap for Semiconductors: 1999 Edition," http://public.itrs.net/files/1999_SIA_Roadmap/Home.htm. For the most recent edition, see http://public.itrs.net.

10 Seth Lloyd, "Computational Capacity of the Universe," *Physical Review Letters*, **88**(23):237901, 10 June 2002.

11 Ray Kurzweil, *The Age of Spiritual Machines*, Penguin Books, 2000.

12 Michael P. Frank, lecture notes for *Principles of Computer Architecture*, UF course CDA5155, Fall 2001-Summer 2003.

13 Seth Lloyd, "Universal Quantum Simulators," *Science* **273**:1073-1078, 23 Aug. 1996.

14 Lee Smolin, *Three Roads to Quantum Gravity*, Basic Books, 2002. For a more recent and more technical introduction, also see Smolin, "Quantum gravity with a positive cosmological constant," http://arxiv.org/abs/hep-th/0209079.

15 Michio Kaku, *Quantum Field Theory: A Modern Introduction*, Oxford University Press, 1993.

16 E. T. Jayes, "Information Theory and Statistical Mechanics," *Physical Review* **106**(4), 15 May 1957, pp. 620-630, http://bayes.wustl.edu/etj/articles/theory.1.pdf.

17 Wojciech H. Zurek, "Decoherence, einselection, and the quantum origins of the classical," July 2002, http://arxiv.org/abs/quant-ph/0105127.

18 N. Margolus and L. Levitin, "The maximum speed of dynamical evolution," *Physica D*, **120**:188-195, 1998. http://arxiv.org/abs/quant-ph/9710043.

19 Michael P. Frank, Thomas F. Knight, Jr., and Norman H. Margolus, "Reversibility in Optimally Scalable Computer Architectures," in Calude, Casti, Dineen, eds., *Unconventional Models of Computation*, Springer, 1998, pp. 183-200, http://www.cise.ufl.edu/~mpf/rc/scaling_paper/scaling.html.

20 Michael P. Frank, "Common Mistakes in Adiabatic Logic Design and How to Avoid Them," submitted to *Methodologies in Low-Power Design*, Las Vegas, NV, June 2003, http://www.cise.ufl.edu/research/revcomp/Frank-MPLD-03.doc.