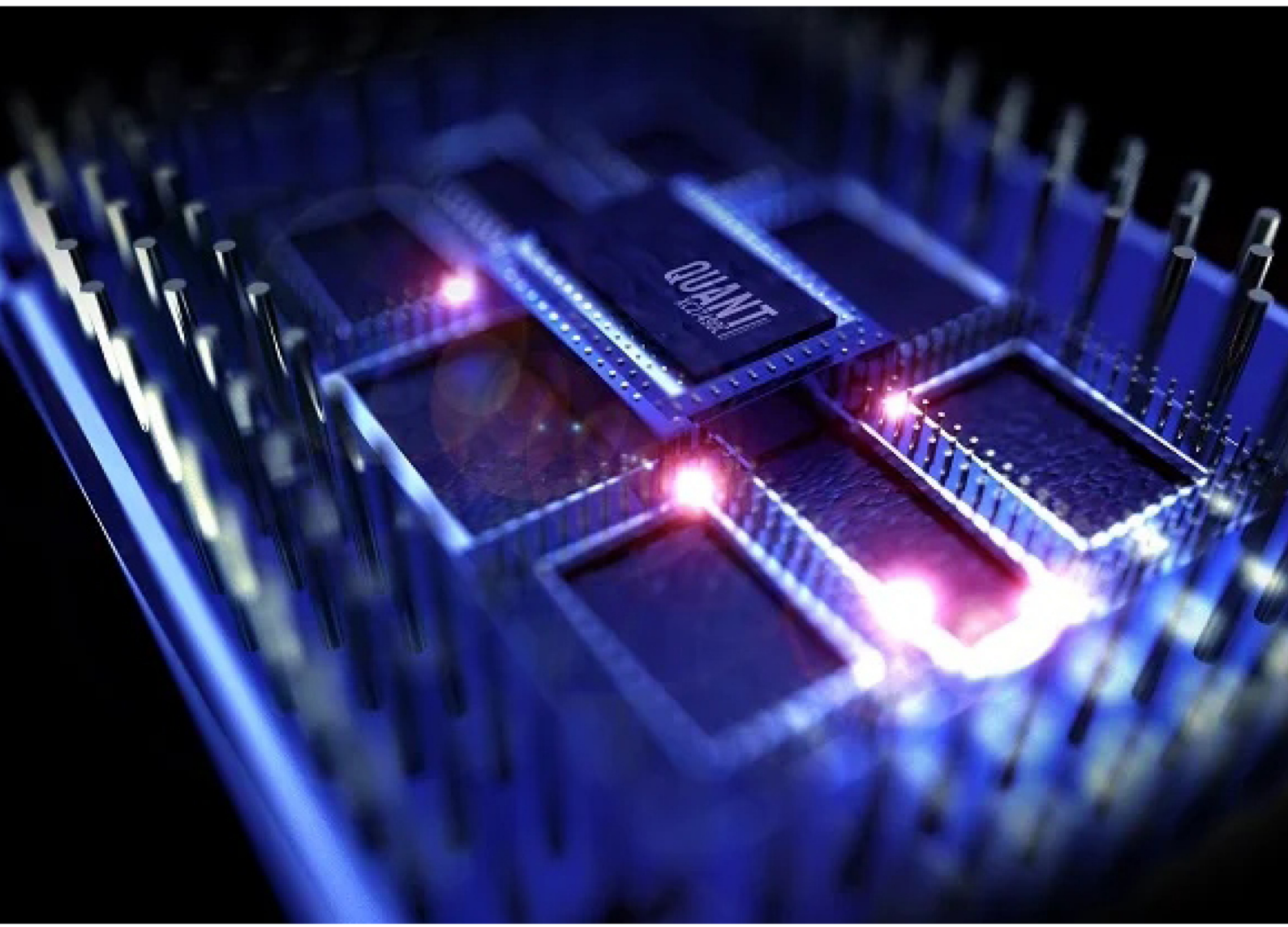
Quantum Nano Computation

Oscar Brown



QUANTUM NANO COMPUTATION

QUANTUM NANO COMPUTATION

Oscar Brown

BIBLIOTEX
Digital Library

Quantum Nano Computation
by Oscar Brown

www.bibliotex.com

Ebook ISBN: 9781984664174

Published by:
Bibliotex
Canada
Website: www.bibliotex.com

Contents

1

Introduction

The best route to create advanced quantum computers, which in theory can run more calculations in an instant than there are atoms in the universe, could be nanotechnologies, experts told UPI's Nano World.

The possibilities for making a nanotech quantum computer are many, including such exotic creations as quivering nanotubes, superconducting nanocircuits and quantum dots.

"Nanoscale devices are the best case to observe quantum mechanical phenomena in the compromise between something small enough to be quantum mechanical, but still large enough to be controllable and accessible," said physicist Franco Nori of the University of Michigan at Ann Arbor and the Frontier Research System of RIKEN near Tokyo.

Conventional computers work by symbolizing data as a series of ones and zeros - binary digits known as bits. The resulting binary code is conveyed via transistors - switches that can be flicked either on or off to represent one or zero.

Quantum computers, however, take advantage of the strange phenomenon that physicists call "superposition," where infinitesimal objects such as individual electrons or atoms can exist in two or more places at once, or spin in opposite directions at the same time.

This means computers built with superposition processors could employ quantum bits - called qubits - that exist in both on and off states simultaneously.

Quantum computers therefore can calculate every possible on-off combination at the same time, making them dramatically faster than conventional data processors when it comes to solving certain problems involving probabilities, such as code-breaking. Quantum computing research is growing rapidly at military, intelligence and university research labs worldwide, as well as at those of industrial giants such as AT&T, IBM, Hewlett-Packard, Lucent and Microsoft.

To run mind-boggling calculations, scientists will need to scale up quantum computers from the handful of qubits most now possess to hundreds.

This will be difficult, because superposition is an extremely delicate state of matter that can be disrupted by the slightest disturbance.

So far, scientists at best have managed to link up, or entangle, only a few qubits to perform simple logic operations, Nori said. The first experiments that created qubits used particles such as chloroform molecules whose components were pushed into superposition with magnetic fields and radio waves. Among the problems with such devices was they did not scale up qubits readily.

That is where nanotechnology comes in. “As a result of the existing semiconductor industry, a great deal of expertise is available for micro or nanofabri-cation,” said physicist Albert Chang at Duke University in Durham, N.C. “This knowledge base could greatly short-circuit the design and implementation of multi-qubit circuitry.” Among the most promising candidates for quantum computers are quantum dots - semiconductor crystals only nanometres, or billionth of a metre, long. Scientists can cram electric charges into quantum dots so they behave like puddles of electrons. The key to using quantum dots in quantum computing is their property known as spin. Electrons spin just as Earth spins on its poles. When two electrons occupy the same space, they must possess opposite spins - one electron spinning “up” and the other “down,” Chang said. Electrons can even be packed into quantum dots so each dot has a net spin of up or down.

Chang and colleagues created qubits from quantum dots by placing a pair of dots carrying the same net spin value near each other. They connected the dots with tiny wires and directed how much electric charge the dots could transfer among one another. By controlling the charge transfer, the team converted both dots to qubits, spinning both up and down simultaneously. “In my view, in the longer run - say on a five-to-10-year horizon, quantum dots have a good chance to emerge as one of the best systems,” Chang said. In addition to Chang’s group, other notable researchers working on quantum-dot qubits include Charles Marcus at Harvard University, Leo Kouwenhoven’s group at the Delft University of Technology in the Netherlands, Seigo Tarucha

at the University of Tokyo and Jorg Kotthaus at LMU Munich. Similar to quantum-dot computers are Kane quantum computers, named after physicist Bruce Kane who suggested the idea in 1998 when he was at the University of New South Wales in Sydney. In a Kane quantum computer, phosphorus atoms under a layer of silicon 25 nanometres or so deep behave as qubits. The device uses phosphorus because the atoms can remain in superposition for a long time.

The Kane quantum computer represents the primary quantum-computing effort in Australia. Because it also depends on silicon, the hope is techniques long refined in the semiconductor industry will help to manufacture these computers and scale them up to large qubit numbers. Still another major contender uses superconducting nanocircuits, which government and university labs worldwide are researching.

At the nanolevel, electronic circuits begin to exhibit quantum behaviour. In superconductors, electrical current flows with no resistance, which means electronic signals can travel without energy loss, helping to preserve superposition. Scientists have worked on superconducting devices for roughly 40 years, and in many ways the superconducting approach for quantum computers is very advanced compared to others, explained physicist Andrew Cleland of the University of California,Santa Barbara.

Still, the circuits are currently prone to having the superpositions break down, Chang said. A key question that remains to be answered is whether error-correction schemes can overcome this problem to make superconducting nanocircuit-based quantum computation “a practical and

useful reality," he added. A more robust quantum computer might even prove mechanical in nature, Nori said. He and colleagues recently proposed using carbon nanotubes or silicon nanorods as mechanical qubits.

"Imagine a ruler and squeeze it along its length," Nori said. A normal ruler would bend either left or right, but if shrunk to nanoscale dimensions such a ruler would take on a superposition of buckling left and right at the same time. The advantage of a mechanical qubit is it potentially could remain in superposition longer than other kinds of qubits. Moreover, mechanical qubits could be manufactured via simple carbon-nanotube growth techniques researched feverishly the world over, said Nori's collaborator, physicist Alik Kasumov of the University Paris-Sud.

"Isn't the basic idea the coolest thing?" asked Keith Schwab, senior physicist for the National Security Agency's lab at the University of Maryland in College Park. Kasumov, Nori and colleagues plan experiments on buckling nanotubes this year, and mechanical qubits could appear within three years if they can produce superposition in the nanotubes, as hoped. The science and technology of building electronic circuits and devices from single atoms and molecules is nanotechnology. This technology is also used in the field of computation. A nanocomputer is a computer whose physical dimensions are microscopic. The field of nanocomputing is part of the emerging field of nanotechnology. Electronic nanocomputers would operate in a manner similar to the way present-day microcomputers work.

The main difference is one of physical scale. More and more transistors are squeezed into silicon chips with each

passing year; witness the evolution of integrated circuits (ICs) capable of ever-increasing storage capacity and processing power. The ultimate limit to the number of transistors per unit volume is imposed by the atomic structure of matter. By 2012 today's ordinary microprocessors might be called nanodevices.

The major types of nano computers are:

- Mechanical Nanocomputers
- Chemical Nanocomputers
- Quantum Nanocomputers
- Electronic Nanocomputers

A quantum nanocomputer would work by storing data in the form of atomic quantum states or spin. Technology of this kind is already under development in the form of single-electron memory (SEM) and quantum dots. A quantum computer is any device for computation that makes direct use of distinctively quantum mechanical phenomena, such as superposition and entanglement, to perform operations on data. In a classical (or conventional) computer, information is stored as bits; in a quantum computer, it is stored as qubits (quantum bits). The basic principle of quantum computation is that the quantum properties can be used to represent and structure data, and that quantum mechanisms can be devised and built to perform operations with this data.

DESCRIPTION

The Basis of Quantum Computing

A classical computer has a memory made up of bits, where each bit holds either a one or a zero. A quantum computer

maintains a sequence of qubits. A single qubit can hold a one, a zero, or, crucially, a superposition of these, allowing for an infinite number of states. A quantum computer operates by manipulating those qubits with (possibly a suite of) quantum logic gates.

The difference between our everyday computer and a quantum computer is that the nanoworld lets a qubit be both a 0 and a 1 at the same time. This peculiar behaviour is called quantum superposition.

There is a certain probability that the qubit holds a 1 and another probability it is recording a 0. Engineers have coined the term qubit(pronounced KYEW-bit) to denote the fundamental data unit in a quantum computer. A qubit is essentially a bit (binary digit)that can take on several, or many, values simultaneously.

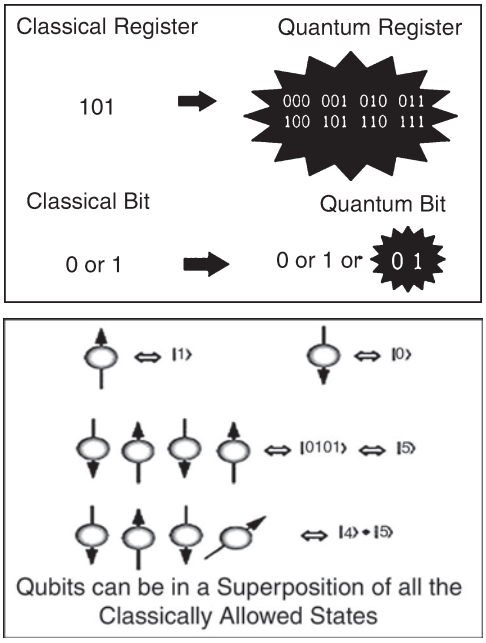
The theory behind this isas bizarre as the theory of quantum mechanics, in which individual particlesappear to exist in multiple locations. One way to think of how a qubit can exist inmultiple states is to imagine it as having two or more aspects or dimensions, each ofwhich can be high (logic 1) or low (logic 0). Thus if a qubit has two aspects, itcan have four simultaneous, independent states (00, 01, 10, and 11); if it has threeaspects, there are eight possible states, binary 000 through 111, and so on.

An example of an implementation of qubits for a quantum computer would be the use of particles with two spin states: “up” and “down” (typically written $|0\rangle$ and $|1\rangle$). But in fact any system possessing an observable quantity A which is conserved under time evolution and such that A has at least two discrete and sufficiently spaced consecutive eigenvalues,

is a suitable candidate for implementing a qubit. That's because any such system can be mapped onto an effective spin-1/2.

Bits vs. Qubits

Consider first a classical computer that operates on a 3-bit register. At any given time, the bits in the register are in a definite state, such as 101. In a quantum computer, however, the qubits can be in a superposition of all the classically allowed states.



To see how it might happen, we should compare the vision of a quantum computer with the computer that sits on your desk. In that machine, and in the biggest supercomputer in the world, information is stored very simply as strings of numbers. In fact, there are only two sorts of numbers, 0s and 1s. These are known as 'bits', short for 'binary digits'. Clever coding now lets us reduce all sorts of information, such as ordinary numbers, words, sounds, pictures and

movies to such strings of numbers, and process that information by adding, subtracting and comparing the number chains. Each bit can be stored, permanently or temporarily, in a tiny box, as a dab of electric charge in a capacitor or a tiny fragment of magnetism on a circle of magnetic film.

Something in the box means a 1; an empty box represents a 0. A quantum computer does much the same thing, but it uses nano-sized particles, such as atoms, as the storage boxes. These are called quantum bits or qubits. For example an atom spinning one way would represent a 1, spinning the other way would be a 0.

COMPONENTS OF A QUANTUM COMPUTER

- Qubits Any two-level quantum system can form a qubit, and there are two ways to form a qubit using the electronic states of an ion:
  - Two ground state hyperfine levels (these are called "hyperfine qubits")
  - A ground state level and an excited level (these are called the "optical qubits")
- Initialization Ions can be prepared in a specific qubit state using a process called optical pumping.
- Measurement Typically, a laser is applied to the ion that couples only one of the qubit states. When the ion collapses into this state during the measurement process, the laser will excite it, resulting in a photon being released when the ion decays from the excited state. After decay, the ion is continually excited by

the laser and repeatedly emits photons. These photons can be collected by a photomultiplier tube (PMT) or a charge-coupled device (CCD) camera.

- Arbitrary Single Qubit Rotation One of the requirements of universal quantum computing is to coherently change the state of a single qubit. For example, this can transform a qubit starting out in 0 into any arbitrary superposition of 0 and 1 defined by the user.
- Two Qubit Entangling Gates Recent theoretical work by Garcia-Ripoll, Cirac, and Zoller have shown that there are no fundamental limitations to the speed of entangling gates, but gates in this impulsive regime (faster than 1 microsecond) have not yet been demonstrated experimentally (current gate operation time is of the order of microseconds). The fidelity of these implementations has been greater than 97%.

Scalable trap designs: Ions can be separated from the same interaction region to individual storage regions and brought back together without losing the quantum information stored in their internal states. Ions can also be made to turn corners at a "T" junction, allowing a two dimensional trap array design.

BUI LDING A QUANTUM COMPUTER

In principle we know how to build a quantum computer; we start with simple quantum logic gates and connect them up into quantum networks. A quantum logic gate, like a classical gate, is a very simple computing device that performs one elementary quantum operation, usually on two qubits, in a given time.

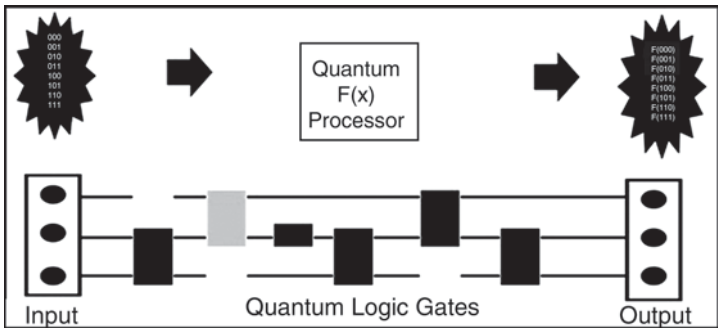
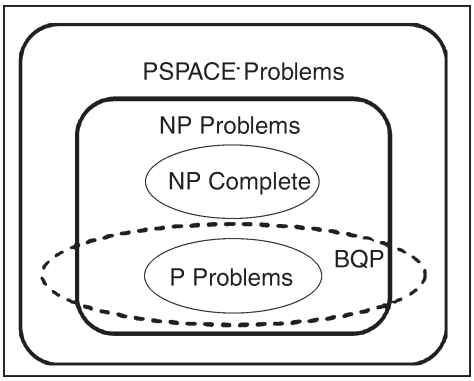


Fig. Quantum Networks

Of course, quantum logic gates differ from their classical counterparts in that they can create and perform operations on quantum superpositions. Quantum computing in computational complexity theory



This portion surveys what is currently known mathematically about the power of quantum computers. It describes the known results from computational complexity theory and the theory of computation dealing with quantum computers.

WHY WOULD WE WANT QUANTUMCOMPUTERS

The computers we have already go at blinding speed and can do pretty much whatever we want. Yet we have always been able to find ways to use each step-up in computer power as it has been presented.

We make greater demands for download speed or whiz-bang graphics or use even fancier software. Supercomputers, which usually take up a whole room, are already in demand for things like forecasting weather and climate, designing aircraft and computer chips. Yet something called a quantum computer is emerging which could give everyone that sort of grunt on their desktop.

Cracking the Code

One much-discussed need for a really slick computer is cryptography or code-cracking. This is not just spy stuff. We move money about on the Internet all the time: we are told the transactions are protected by uncrackable security codes. Many of these encryptions rely on very large numbers – like numbers with 400 digits.

To crack the code, these have to be broken down into the smaller prime numbers which create the big number when you multiply them together.

With today's computers this takes just about forever, so we can rely on such encryptions to keep our money safe. But it seems quantum computers could crack them within a few hours or even a few minutes.

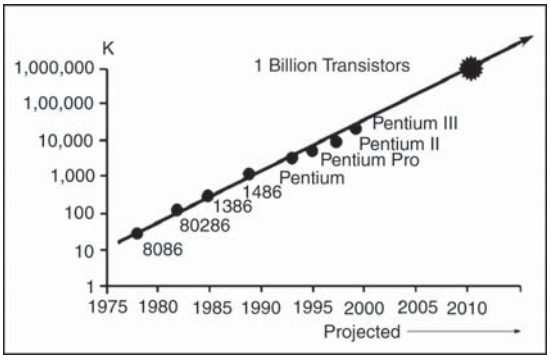
Moore's Law hits the Wall

The most famous, or perhaps the most notorious statement in computer science is Moore's law, named after a famous computer pioneer. It is not really a 'law', it is rather an observation of what has happened, and what we might expect to continue to happen, at least in the immediate future.

This 'law' says that the number of components which computer-chip makers can squeeze onto a chip for data

storage or processing doubles every 18 months or so, as design and manufacturing methods improve.

Certainly this has been going on for more than three decades. Where a few thousand transistors or capacitors or resistors would fit in 1975 – on a piece of silicon the size of your fingernail – we can now place hundreds of millions. Vastly more information is being stored and calculations now done billions of times a second, mightily increasing the power of computers.



For this to happen, the chip makers must make those components even smaller. Every 18 months or so they are cut in half in size, which is another way of expressing Moore's law.

APPLICATIONS

Quantum computers might prove especially useful in the following applications:

- Breaking ciphers
- Statistical analysis
- Factoring large numbers
- Solving problems in theoretical physics
- Solving optimization problems in many variables
- Quantum computers is a new concept in parallelism

ADVANTAGES

- Quantum computers can provide solutions extremely faster.
- The size of computers are of nano size
- The efficiency is very high
- Computations can be done more accurately
- Used in every field of technology

DISADVANTAGES

- It is difficult to build quantum computer because as the number of quantum gates in a network increases, we quickly run into some serious practical problems. The more interacting qubits are involved, the harder it tends to be to engineer the interaction that would display the quantum properties. The more components there are, the more likely it is that quantum information will spread outside the quantum computer and be lost into the environment, thus spoiling the computation. This process is called decoherence.

FUTURE OUTLOOK

At present, quantum computers and quantum information technology remains in its pioneering stage. At this very moment obstacles are being surmounted that will provide the knowledge needed to thrust quantum computers up to their rightful position as the fastest computational machines in existence. Error correction has made promising progress to date, nearing a point now where we may have the tools required to build a computer robust enough to adequately withstand the effects of decoherence.

Quantum hardware, on the other hand, remains an emerging field, but the work done thus far suggests that it will only be a matter time before we have devices large enough to test Shor's and other quantum algorithms. Thereby, quantum computers will emerge as the superior computational devices at the very least, and perhaps one day make today's modern computer obsolete. Quantum computation has its origins in highly specialized fields of theoretical physics, but its future undoubtedly lies in the profound effect it will have on the lives of all mankind.

QUANTUM-EFFECT NANOSCALE DEVICES

According to the laws of quantum mechanics, free carriers in a metal or semiconductor can only take on specific values of energy, as defined by the crystal structure; that is, the energy is quantized. For most practical purposes, there are so many closely spaced energy levels, it appears that the carriers have a continuum of possible energies, except for the well-defined gaps characteristic of semiconductors. When the carrier is confined to a region where one or more of the dimensions reach the range of less than 100 nm, the quantum energy levels begin to spread out and the quantum nature becomes detectable.

Illustration by Hans and Cassidy. Courtesy of Gale Group.This reduction in size can take place in one, two, or three dimensions, using the fabrication techniques discussed earlier, yielding structures known respectively as superlattices, quantum wires, and quantum dots. When electrons are introduced into a semiconductor structure, they migrate to those positions where their energy is lowest, much

like a ping-pong ball will come to rest in a dimple on a waffled surface. If the nanostructure is engineered correctly, then the electrons will settle in the nanostructure itself and not in the adjacent layers.

These carriers will then exhibit the quantum effects imposed on them by the nanostructure. The ability to engineer artificial atoms and molecules in semiconductors using nanofabrication techniques has resulted in a powerful new tool in creating novel semiconductor devices, such as quantum dots where the number of carriers trapped by the dot can be controlled by an external voltage. It appears possible that nanoscale quantum-effect devices may become widely used in complex electronic systems, such as a neural array of quantum dots spaced only a few 100 nm apart, but this will only take place after significant progress has been made in fabrication and tolerance. Scientists see a universe of potential in nanotechnology, following years or perhaps decades of research and development. Some of the applications they foresee are as follows: surgical instruments of molecular scale that are guided by computers of the same size; rocket ships for the individual made of shatterproof materials created by nanomachines; synthesis of foods by molecules and an end to famine; pollution-free manufacturing and molecular devices that clean up existing pollution without human intervention; consumer goods that assemble themselves; reforming of soil (termed terraforming) both on Earth and other planets where soil and rock may not exist; and computers capable of more computations in 1 second than all the existing semiconductor devices in the world combined.

Nanodevices may create "smart mat ter" that, when used to build a bridge or a high-rise building, knows when and how to repair itself; diamonds of perfect quality and any size may be built atom by atom to suit industrial needs or an individual's ideal; injectable molecular robots that enter the bloodstream on seek-and-destroy missions for cancer, AIDS, invading bacteria or viruses, and arterial blockages. Similarly, nanoparticles might carry vaccines and drugs directly to the source of the ailment.

IMPACTS OF STRONG (DREXLERIAN) NANOTECHNOLOGY

Though Drexler is a major contributor to the still pretty young field of nanotechnology, he has by no means invented the infinitesimal realm of nano. Prior to the advent of his diamondoid SISD generic nanoassembler design, diverse ideas about both top-down downscaling of solid state devices and using autoorganizing and autoassembling intrinsic properties of solvated biopolymers in a bottom-up approach were already fairly widespread. The crux of Drexler's design outline is high matter throughput/processing capability which at the same time features extremely precise, atomic, level of control. Should the design prove itself implementable, the implications are literally unimaginable. Especially, greatly facilitated creation of noticeably superrealtime, significantly above human intelligence level information processing systems are thought to cause a positive design autofeedback loop, almost instantly precipitating an extremely rapid (days to months scale) intellectual runaway, the so-called Singularity. The Singularity Theory, or, better, conjecture, is further fueled by the well-known fact that many current

development trends can be fitted to exponential or even hyperbolic functions with negligeable deviations. Extrapolation of these trends into the near future (25-40 years) thus forecasts obviously impossible, since absolutely ridiculous values. Either, the predictions are entirely wrong, and we will experience a saturation at a high level, the neo-"Golden Age", or, more likely, a catastrophic breakdown, not unlike the autoinhibition of bacterial growth in a finite medium, orelse we will enter the brave new era of trans- or posthumanism, which significance can only be compared with the primeval life nucleation event on Earth.

While this may sound like pompous garbage at best, or, obscure, pseudoreligious faith at worst, as future development trajectories are thought to be fundamentally uncomputable, this view can be veriefied easily enough, as most of us will live to see it. The only fact which so far seems fairly certain, is that "the world will turn strange, soon". In fact, the continuously increasing rate of change, the first forequake of Singularity, has produced a kind of world in which native neolithical societies, though rapidly vanishing, can transiently coexist with maturing aerospace, infoprocessing and biotechnology industries. As in analytical chromatography, the wandering spots on the substate plate are being increasingly spread over a larger area, some of them almost keeping up with the solvent (shockwave) front. I dunno, "strange" may already seem a quite apt epithet. Relax, you ain't seen nothing yet.

ALTERNATIVES

Anyway, should one strive "merely" for building a drastically better computer, without assuming imminent availability of

all-purpose Drexlerian nanoassemblers, which bootstrap or even implementability window might well be too narrow for practicability, one is left with the weak, "wet" autoassembly biopolymer nanotechnology. Should we succeed in its implementation, the resulting hardware will at the very least be extremely instrumental in the subsequent bootstrap of Drexler-flavoured systems. Why biopolymers? Can't we simply utilize the numerous structures and powerful methods organic chemistry can offer us?

There are two basic reasons. Contrary to the widespread scientific legend, biopolymers are by no means inefficient or flabby. Proteins are all-purpose linear polymers, nevertheless their bandwidth reaches from thixotrophic, highly hydrated gels and phospholipid-protein conglomerates to the pretty dry, noticeably better-than-kevlar-calibre cobwebs and other tough fibrous structural proteins. Thus, circuitry matrix needs not to be flabby nor heavily hydrated. Proteins have been subjected to heavy optimization during GYrs of evolution, their precipitated 20 amino acid minimum alphabet surely sufficiently all-purpose.

The best reason, of course, is our already extremely large and exponentially expanding fount of knowledge about protein structure, their folding kinetics and the rapidly growing skill of manipulating (by recombinant DNA technology) the so far the only working instance of nanotechnology we know of: In contrast to that, the level of complexity achievable by organic chemistry is very limited. Catalytically active, organic switchable enzyme analogons bear great promise, but are terribly difficult to design and to optimize.

Alas, awareness of their potential feasibility and necessity of a drastically different approach in their synthesis management has not yet become widespread in the mainstream chemical community. Further progresses in organic synthesis are only possible by further migration towards mechanosynthesis. Mechanically highly constrained, aligned systems, e.g. organic photochemistry in crystal lattice as opposed to photochemistry in (inert) solvate, show drastical deviations in their products as many configuration space trajectories are excluded by constraints imposed by the embedding crystal lattice matrix. Biochemistry already utilizes mechanosynthesis at a noticeable degree, at least at the degree possible in a nonstiff, solvated system.

It should be noted, that natural biological systems rarely exploit the maximum possible stiffness of enzymes. Even preliminary results of site-directed mutagenesis optimization of antibodies has shown that natural proteins can be made more stable, sometimes even drastically so by the mutation of one to few amino acids. Another key factor is disulfide bridge crosslink.

Nature cannot tolerate deposition of heavily crosslinked albeit signficiantly stiffer systems with the cell, as their creation is a highly irreversible process, which might also generate significant amounts of damaging radicals. Would this path be taken, it would lead to wastage of material, and thus of metabolic energy necessary for the biosynthesis of the monomer precursors, synthesis and subsequent secretion. Moreover, stiff, nondegradable systems are fundamentally incompatible with the homeostasis maintanance control paradigm of the biological cell.

Enzymatic activity is controlled both by the total number of enzymes within the cell as their individual modification, e.g. by phosphorylation or interaction with an allosteric cofactor. This is impossible, or, at least very hard to do on a crosslinked enzyme. Further difficulty in crosslinking by disulfide bridges is, apart from the fact that there is an upper limit the the fraction of crosslinks per volume, as (a potentially large) percentile of the sequence string is necessary for the encoding of a robust folding trajectory as well as proper alignment of the complementary cystine residues.

Even tiny deviation from these constraints will result in nonviable proteins. Nevertheless, the design of such systems appears possible. The biopolymers as implementation tools impose a large number of constraints, which rule out lots of possible design targets. For one, though proteins are great for autoassembly and support matrix, they are virtually useless for computation.

Hence, one is forced to use the (yet almost entirely hypothetical) molecular circuitry, which works by utilizing electronically excited states of small organical and metallorganical molecules, polyenes, columnar complexes, dyes, etc. Lasers or a potential gradient can be utilized to pump a quantum system into a higher state, using several, switchable relaxation pathways for computation. The achievable switching rates are noticeably above those of diamond rod logic, power dissipation should also be smaller (?), perhaps significantly so. The upper limit of switching rate is obviously given by photodissotiation of the molecular circuitry. However, protein matrix constituting the circuitry cavity, if properly engineered, can both influence excited state

lifetime and contribute to drastically enhanced robustness by providing well defined relaxation pathways for dangerous states.

The need for molecular circuitry, which has to be generated by the very limited means of organical chemistry and biochemistry, obviously imposes an upper limit for atomic subsystem complexity. Instead of seeing this as a minus, one might as well see it as a challenge turning to the most minimalistic computer design known: cellular automata machines, abbreviated as CAMs. Apart from their intrinsic simplicity, CA show an excellent congruency of with nature's ways of doing computation: namely, by means of a very large number of asynchronous, purely locally coupled simple subsystems.

Contrarily to common belief, most real-world problems are massively parallel. Their apparent sequentiality is merely an artefact of limited introspection; a severely biased perception of the physical reality. Furthermore, CA can greatly profit from results of complexity science, utilizing new knowledge gained in emergent behaviour studies, making adaptive, robust self-healing systems more than just a dream. Once the decision for a protein-matrix, molecular-circuitry 3d molecular automaton machine (molCAM) has been made, many things start making remarkable sense. Many proteins and large protein complexes, e.g. viruses, can be crystallized in macroscopic, optically clear crystals, which inherently shield damaging shortwave radiation. While many of the crystals are very weak mechanically (you can poke with your finger through them), some being thixotropic gels, one should not forget the fact that biopolymer crystal precipitation within

the cells is greatly inhibited by the artefacting generating process, the Darwinian evolution (target has 'evolution' written in huge letters all over it). In fact, it is a miracle most proteins require only very little coaxing to crystallize at all.

An engineered or de novo designed protein crystal can well form a clear, tough matrix with the circuitry embedded. Again, it might be instrumental to remember that spider silk protein significantly surpasses kevlar in its mechanical properties. Crystal growth is an emergent, self-organizing parallel process without a central organizing authority. Since the elementary cells of molCAM are so large and form intricate complementary-surface docking sites, the number of defects can be made virtually nil. Due to extensive crosslinking, both within the elementary cell and at the intercell level (complementary surface interaction play also a major part), the crystal can be made mechanically tough, resisting translocation defects.

Due to stiffness and defect considerations, the connectivity must be low. Should we settle for the cubic primitive lattice (which might not be the optimum but is pretty convenient to reason about), only the six nearest neighbours can be directly contacted. Rule complexity, especially in a lookup implementation, can be greatly reduced by utilizing a rotation invariant rule. Again, due to low achievable complexity, the cell's state space must be limited to few-bit values.

The protein matrix and the scaffold (auxiliary proteins that are needed transiently and are not incorporated into the final matrix) play the following two roles. First, they envelop organically or biochemically synthesized circuitry subblocks, creating complexes with differing properties of both the

uncomplexed protein and the piece of circuitry. This may utilized for efficient purification (e.g. gel electrophoresis, etc.), as affinity chromatography, the utilization of extremely high specifity _and_ high binding constant of immobilized antibodies raised against certain haptenes (small antigens, usually presented on a spacer) for an excellent purification. Second, they envelop, align and join pieces of the circuitry machinery in a given sequence, until in an hierarchic assembly few-step process the individual CA cell has been constructed. Since the assembly is SIMD in one pot, albeit with interim purification, the absolute number of elementary cells to be later assembled into a CAM macrocrystal is large.

The elementary cell is roughly 0.1-1.0 um sized, which may seem fairly large even by today's standards of photolithographically attainable structures. It should not be underestimated however, as the cell's complexity is noticeably higher (it is a tiny few-bit computer) than the according semiconductor structure, the cells are aligned in a true 3-lattice instead of 2-lattice of photolithography, the concentration of circuitry per volume is limited due to inherent structural overhead neccessary both for the encoding of a robust folding pathway as well as a well-defined target fold, the need of optical transparency for optical power and I/O as well as maximum tolerable power/volume dissipation.

The synchroniziation problem is not entirely trivial. The application of the local composite discrete Hamiltonian upon the neighbourhood (base of the light cone) must be perfectly parallel, allowing no runtime differences to accumulate during subsequent iterations, even at utilized ns and

noteceably sub-ns cycle times. As infinitely precise timers within each cell are obviously impossible, a kind of external clocking must applied. Two basic flavours are possible: clocking by laser pulses (which may be also utilized for power source) and an active clock by propagating wave phenomena.

The former utilizes the high speed of light which significantly surpasses signal travel and switching time in a CAM, the latter achieves the synchronization by subsequently traveling wavefronts which are originated by a few cells, which may be random. Both assume that at a single iteration time, the infinitesimal cell differences are yet invisible. The autopropagating wave state is assumed to be immediately forwarded to the neighbours, while initiating a computation cycle after a short delay. After the computation cycle the cell is temporarily insusceptible to subsequent waves, suffering the nonexcitable 'refractory time' before restoring the original state. The Gentle Reader may grant me that the above rough hardware outline (there is more to it than mentioned here) seems to be both implementable and operable, but what about the thorny issues of interfacing and programming?

BOTTOM-UP TECHNOLOGY

The tour showed the sizes, forces, and general nature of objects in the molecular world. Building on this, we can get a better picture of where developments seem to be leading, a better picture of molecular manufacturing itself.

To show the sizes, forces, and general nature of things in molecular manufacturing, we first invite the reader (and the reader's inquisitive alter ego) to take a second and final tour

before returning to the world of present-day research. As before, the pre-1990 history is accurate, and the science isn't fiction.

THE SILICON VALLEY FAIRE

The tour of the molecular world showed some products of molecular manufacturing, but didn't show how they were made. The technologies you remember from the old days have mostly been replaced—but how did this happen? The Silicon Valley Faire is advertised as "An authentic theme park capturing life, work, and play in the early Breakthrough years." Since "work" must include manufacturing, it seems worth a visit. A broad dome caps the park —"To fully capture the authentic sights, sounds, and smells of the era," the tourguide politely says. Inside, the clothes and hairstyles, the newspaper headlines, the bumper-to-bumper traffic, all look much as they did before your long nap. A light haze obscures the buildings on the far side of the dome, your eyes burn slightly, and the air smells truly authentic.

POCKET LIBRARIES

The Nanofabricators, Inc., plant offers the main display of early nanotechnology. As you near the building, the tourguide mentions that this is indeed the original manufacturing plant, given landmark status over twenty years ago, then made the centrepiece of the Silicon Valley Faire ten years later, when... With a few taps, you reset the pocket tourguide to speak up less often.

As people file into the Nanofabricator plant, there's a moment of hushed quiet, a sense of walking into history. Nanofabricators: home of the SuperChip, the first mass-

market product of nanotechnology. It was the huge memory capacity of SuperChips that made possible the first Pocket Library. This section of the plant now houses a series of displays, including working replicas of early products. Picking up a Pocket Library, you find that it's not only the size of a wallet, but about the same weight.

Yet it has enough memory to record every volume in the Library of Congress–something like a million times the capacity of a personal computer from 1990. It opens with a flip, the two-panel screen lights up, and a world of written knowledge is at your fingertips. Impressive. "Wow, can you believe these things?" says another tourist as he fingers a Pocket Library.

"Hardly any video, no 3-D–just words, sound, and flat pictures. And the cost! I wouldn't've bought `em for my kids at that price!" Your tourguide quietly states the price: about what you remember for a top-of-the-line TV set from 1990. This isn't the cheap manufacturing promised by mature nanotechnology, but it seems like a pretty good price for a library. Hmm... how did they work out the copyrights and royalties? There's a lot more to this product than just the technology...

NANOFABRICATION

The next room displays more technology. Here in the workroom where SuperChips were first made, early nanotech manufacturing is spread out on display. The whole setup is surprisingly quiet and ordinary. Back in the 1980s and 1990s, chip plants had carefully controlled clean rooms with gowns and masks on workers and visitors, special workstations, and carefully crafted air flows to keep dust

away from products. This room has none of that. It's even a little grubby. In the middle of a big square table are a half-dozen steel tanks, about the size and shape of old-fashioned milk cans. Each can has a different label identifying its contents: Memory blocks, data-transmission blocks, interface blocks. These are the parts needed for building up the chip. Clear plastic tubes, carrying clear and tea-coloured liquids, emerge from the mouths of the milk cans and drape across the table. The tubes end in fist-sized boxes mounted above shallow dishes sitting in a ring around the cans.

As the different liquids drip into each dish, a beater like a kitchen mixer swirls the liquid. In each dish, nanomachines are building SuperChips. A Nanofab "engineer," dressed in period clothing complete with name badge, is setting up a dish to begin building a new chip. "This," he says, holding up a blank with a pair of tweezers, "is a silicon chip like the ones made with pre-breakthrough technology. Companies here in this valley made chips like these by melting silicon, freezing it into lumps, sawing the lumps into slices, polishing the slices, and then going through a long series of chemical and photographic steps.

When they were done, they had a pattern of lines and blobs of different materials on the surface. Even the smallest of these blobs contained billions of atoms, and it took several blobs working together to store a single bit of information. A chip this size, the size of your fingernail, could store only a fraction of a billion bits. Here at Nanofab, we used bare silicon chips as a base for building up nanomemory.

The picture on the wall here shows the surface of a blank chip: no transistors, no memory circuits, just fine wires to

connect up with the nanomemory we built on top. The nanomemory, even in the early days, stored thousands of billions of bits. And we made them like this, but a thousand at a time–" He places the chip in the dish, presses a button, and the dish begins to fill with liquid.

"A few years latter," he adds, "we got rid of the silicon chips entirely"–he props up a sign saying this chip build began at: 2:15 p.m., estimated completion time: 1:00 a.m.-" and we sped up the construction process by a factor of a thousand." The chips in the dishes all look pretty much the same except for colour. The new chip looks like dull metal. The only difference you can see in the older chips, further along in the process, is a smooth rectangular patch covered by a film of darker material.

An animated flowchart on the wall shows how layer upon layer of nanomemory building blocks are grabbed from solution and laid down on the surface to make that film. The tourguide explains that the energy for this process, like the energy for molecular machines within cells, comes from dissolved chemicals–from oxygen and fuel molecules.

The total amount of energy needed here is trivial, because the amount of product is trivial: at the end of the process, the total thickness of nanomemory structure–the memory store for a Pocket Library–amounts to one-tenth the thickness of a sheet of paper, spread over an area smaller than a postage stamp.

MOLECULAR ASSEMBLY

The animated flowchart showed nanomemory building blocks as big things containing about a hundred thousand atoms apiece (it takes a moment to remember that this is

still submicroscopic). The build process in the dishes stacked these blocks to make the memory film on the SuperChip, but how were the blocks themselves built? The hard part in this molecular-manufacturing business has got to be at the bottom of the whole process, at the stage where molecules are put together to make large, complex parts. The Silicon Valley Faire offers simulations of this molecular assembly process, and at no extra charge.

From the tourguide, you learn that modern assembly processes are complex; that earlier processes–like those used by Nanofabricators, Inc. –used clever-but-obscure engineering tricks; and that the simplest, earliest concepts were never built. Why not begin at the beginning? A short walk takes you to the Museum of Antique Concepts, the first wing of the Museum of Molecular Manufacturing.

A peek inside the first hall shows several people strolling around wearing loosely fitting jumpsuits with attached goggles and gloves, staring at nothing and playing mime with invisible objects. Oh well, why not join the fools' parade? Stepping through the doorway while wearing the suit is entirely different. The goggles show a normal world outside the door and a molecular world inside.

Now you, too, can see and feel the exhibit that fills the hall. It's much like the earlier simulated molecular world: it shares the standard settings for size, strength, and speed. Again, atoms seem 40 million times larger, about the size of your fingertips. This simulation is a bit less thorough than the last was–you can feel simulated objects, but only with your gloved hands. Again, everything seems to be made of quivering masses of fused marbles, each an atom.

"Welcome," says the tourguide, "to a 1990 concept for a molecular - manufacturing plant. These exploratory engineering designs were never intended for actual use, yet they demonstrate the basics of molecular manufacturing: making parts, testing them, and assembling them." Machinery fills the hall.

Overall, the sight is reminiscent of an automated factory of the 1980s or 1990s. It seems clear enough what must be going on: Big machines stand beside a conveyor belt loaded with half-finished-looking blocks of some material (this setup looks much like Figure); the machines must do some sort of work on the blocks. Judging by the conveyor belt, the blocks eventually move from one arm to the next until they turn a corner and enter the next hall.

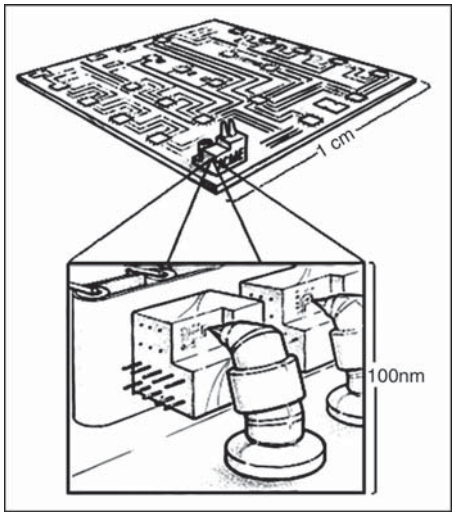


Fig. Assembler With Factory On Chip

A factory–large enough to make over 10 million nanocomputersper day would fit on the edge one of today's integrated circuits. Inset shows an assembler arm together with workpiece on a conveyor belt. Since nothing is real, the exhibit can't be damaged, so you walk up to a machine and

give it a poke. It seems as solid as the wall of the nanocomputer in the previous tour. Suddenly, you notice something odd: no bombarding air molecules and no droplets of water–in fact, no loose molecules anywhere. Every atom seems to be part of a mechanical system, quivering thermal vibration, but otherwise perfectly controlled. Everything here is like the nanocomputer or like the tough little gear; none of it resembles the loosely coiled protein or the roiling mass of the living cell.

The conveyor belt seems motionless. At regular intervals along the belt are blocks of material under construction: workpieces. The nearest block is about a hundred marble-bumps wide, so it must contain something like 100 x 100 x 100 atoms, a full million. This block looks strangely familiar, with its rods, crank, and the rest. It's a nanocomputer–or rather, a blocklike part of a nanocomputer still under construction. Standing alongside the pieces of nanocomputer on the conveyor belt, dominating the hall, is a row of huge mechanisms. Their trunks rise from the floor, as thick as old oaks. Even though they bend over, they rear overhead. "Each machine," your tourguide says, "is the arm of a general-purpose molecular assembler.

One assembler arm is bent over with its tip pressed to a block on the conveyor belt. Walking closer, you see molecular assembly in action. The arm ends in a fist-sized knob with a few protruding marbles, like knuckles. Right now, two quivering marbles–atoms–are pressed into a small hollow in the block. As you watch, the two spheres shift, snapping into place in the block with a quick twitch of motion: a chemical reaction. The assembler arm just stands there,

nearly motionless. The fist has lost two knuckles, and the block of nanocomputer is two atoms larger.

The tourguide holds forth: "This general-purpose assembler concept resembles, in essence, the factory robots of the 1980s. It is a computer-controlled mechanical arm that moves molecular tools according to a series of instructions. Each tool is like a single-shot stapler or rivet gun. It has a handle for the assembler to grab and comes loaded with a little bit of matter–a few atoms–which it attaches to the workpiece by a chemical reaction." This is like the rejoining of the protein chain in the earlier tour.

MOLECULAR PRECISION

The atoms seemed to jump into place easily enough; can they jump out of place just as easily? By now the assembler arm has crept back from the surface, leaving a small gap, so you can reach in and poke at the newly added atoms. Poking and prying do no good: When you push as hard as you can (with your simulated fingers as strong as steel), the atoms don't budge by a visible amount. Strong molecular bonds hold them in place. Your pocket tourguide–which has been applying the power of a thousand 1990s supercomputers to the task of deciding when to speak up–remarks, "Molecular bonds hold things together. In strong, stable materials atoms are either bonded, or they aren't, with no possibilities in between. Assemblers work by making and breaking bonds, so each step either succeeds perfectly or fails completely. In pre-breakthrough manufacturing, parts were always made and put together with small inaccuracies. These could add up to wreck product quality. At the molecular scale, these problems vanish. Since each step is perfectly precise, little

errors can't add up. The process either works, or it doesn't." But what about those definite, complete failures? Fired by scientific curiosity, you walk to the next assembler, grab the tip, and shake it. Almost nothing happens. When you shove as hard as you can, the tip moves by about one-tenth of an atomic diametre, then springs back. "Thermal vibrations can cause mistakes by causing parts to come together and form bonds in the wrong place," the tourguide remarks. "Thermal vibrations make floppy objects bend further than stiff ones, and so these assembler arms were designed to be thick and stubby to make them very stiff.

Error rates can be kept to one in a trillion, and so small products can be perfectly regular and perfectly identical. Large products can be almost perfect, having just a few atoms out of place." This should mean high reliability. Oddly, most of the things you've been seeing outside have looked pretty ordinary–not slick, shiny, and perfect, but rough and homey. They must have been manufactured that way, or made by hand. Slick, shiny things must not impress anyone anymore.

MOLECULAR ROBOTICS

By now, the assembler arm has moved by several atom-widths. Through the translucent sides of the arm you can see that the arm is full of mechanisms: twirling shafts, gears, and large, slowly turning rings that drive the rotation and extension of joints along the trunk.

The whole system is a huge, articulated robot arm. The arm is big because the smallest parts are the size of marbles, and the machinery inside that makes it move and bend has many, many parts. Inside, another mechanism is at work: The arm now ends in a hole, and you can see the old, spent

molecular tool being retracted through a tube down the middle. Patience, patience. Within a few minutes, a new tool is on its way back up the tube. Eventually, it reaches the end. Shafts twirl, gears turn, and clamps lock the tool in position. Other shafts twirl, and the arm slowly leans up against the workpiece again at a new site. Finally, with a twitch of motion, more atoms jump across, and the block is again just a little bit bigger. The cycle begins again. This huge arm seems amazingly slow, but the standard simulation settings have shifted speeds by a factor of over 400 million. A few minutes of simulation time correspond to less than a millionth of a second of real time, so this stiff, sluggish arm is completing about a million operations per second.

Peering down at the very base of the assembler arm, you can get a glimpse of yet more assembler-arm machinery underneath the floor: Electric motors spin, and a nanocomputer chugs away, rods pumping furiously. All these rods and gears move quickly, sliding and turning many times for every cycle of the ponderous arm. This seems inefficient; the mechanical vibrations must generate a lot of heat, so the electric motors must draw a lot of power. Having a computer control each arm is a lot more awkward now than it was in pre-breakthrough years. Back then, a robot arm was big and expensive and a computer was a cheap chip; now the computer is bigger than the arm. There must be a better way–but then, this is the Museum of Antique Concepts.

BUILDING-BLOCKS INTO BUILDINGS

Where do the blocks go, once the assemblers have finished with them? Following the conveyor belt past a dozen arms, you stroll to the end of the hall, turn the corner, and find

yourself on a balcony overlooking a vaster hall beyond. Here, just off the conveyor belt, a block sits in a complex fixture. Its parts are moving, and an enormous arm looms over it like a construction crane. After a moment, the tourguide speaks up and confirms your suspicion: "After manufacturing, each block is tested. Large arms pick up properly made blocks. In this hall, the larger arms assemble almost a thousand blocks of various kinds to make a complete nanocomputer. The grand hall has its own conveyor belt, bearing a series of partially completed nanocomputers. Arrayed along this grand belt is a row of grand arms, able to swing to and fro, to reach down to lesser conveyor belts, pluck million-atom blocks from testing stations, and plug them into the grand workpieces, the nanocomputers under construction. The belt runs the length of the hall, and at the end, finished nanocomputers turn a corner–to a yet-grander hall beyond? After gazing at the final-assembly hall for several minutes, you notice that nothing seems to have moved. Mere patience won't do: at the rate the smaller arms moved in the hall behind you, each block must take months to complete, and the grand block-handling arms are taking full advantage of the leisure this provides. Building a computer, start to finish, might take a terribly long time. Perhaps as long as the blink of an eye.

Molecular assemblers build blocks that go to block assemblers. The block assemblers build computers, which go to system assemblers, which build systems, which–at least one path from molecules to large products seems clear enough. If a car were assembled by normal-sized robots from a thousand pieces, each piece having been assembled by

smaller robots from a thousand smaller pieces, and so on, down and down, then only ten levels of assembly process would separate cars from molecules. Perhaps, around a few more corners and down a few more ever-larger halls, you would see a post-breakthrough car in the making, with unrecognizable engine parts and comfortable seating being snapped together in a century-long process in a hall so vast that the Pacific Ocean would be a puddle in the corner.. Just ten steps in size; eight, starting with blocks as big as the ones made in the hall behind you. The molecular world seems closer, viewed this way.

MOLECULAR PROCESSING

Stepping back into that hall, you wonder how the process begins. In every cycle of their sluggish motion, each molecular assembler gets a fresh tool through a tube from somewhere beneath the floor, and that somewhere is where the story of molecular precision begins. And so you ask, "Where do the tools come from?", and the tourguide replies, "You might want to take the elevator to your left."

Stepping out of the elevator and into the basement, you see a wide hall full of small conveyor belts and pulleys; a large pipe runs down the middle. A plaque on the wall says, "Mechanochemical processing concept, circa 1990." As usual, all the motions seem rather slow, but in this hall everything that seems designed to move is visibly in motion. The general flow seems to be away from the pipe, through several steps, and then up through the ceiling towards the hall of assemblers above. After walking over to the pipe, you can see that it is nearly transparent.

Inside is a seething chaos of small molecules: the wall of the pipe is the boundary between loose molecules and controlled ones, but the loose molecules are well confined. In this simulation, your fingertips are like small molecules. No matter how hard you push, there's no way to drive your finger through the wall of the pipe.

Every few paces along the pipe a fitting juts out, a housing with a mechanically driven rotating thing, exposed to the liquid inside the pipe, but also exposed to a belt running over one of the pulleys, embedded in the housing. It's hard to see exactly what is happening. The tourguide speaks up, saying, "Pockets on the rotor capture single molecules from the liquid in the pipe.

Each rotor pocket has a size and shape that fits just one of the several different kinds of molecule in the liquid, so the process is rather selective. Captured molecules are then pushed into the pockets on the belt that's wrapped over the pulley there, then–" "Enough," you say. Fine, it singles out molecules and sticks them into this maze of machinery. Presumably, the machines can sort the molecules to make sure the right kinds go to the right places.

The belts loop back and forth carrying big, knobby masses of molecules. Many of the pulleys–rollers?–press two belts together inside a housing with auxiliary rollers. While you are looking at one of these, the tourguide says, "Each knob on a belt is a mechanochemical-processing device. When two knobs on different belts are pressed together in the right way, they are designed to transfer molecular fragments from one to another by means of a mechanically forced chemical reaction.

In this way, small molecules are broken down, recombined, and finally joined to molecular tools of the sort used in the assemblers in the hall above. In this device here, the rollers create a pressure equal to the pressure found halfway to the centre of the Earth, speeding a reaction that–" "Fine, fine," you say. Chemists in the old days managed to make amazingly complex molecules just by mixing different chemicals together in solution in the right order under the right conditions. Here, molecules can certainly be brought together in the right order, and the conditions are much better controlled.

It stands to reason that this carefully designed maze of pulleys and belts can do a better job of molecule processing than a test tube full of disorganized liquid ever could. From a liquid, through a sorter, into a mill, and out as tools: this seems to be the story of molecule processing. All the belts are loops, so the machinery just goes around and around, carrying and transforming molecular parts.

BEYOND ANTIQUES

This system of belts seems terribly simple and efficient, compared to the ponderous arms driven by frantic computers in the hall above. Why stop with making simple tools? You must have muttered this, because the tourguide speaks up again and says, "The Special-Assembler Exhibit shows another early molecular-manufacturing concept that uses the principles of this molecule-processing system to build large, complex objects.

If a system is building only a single product, there is no need to have computers and flexible arms move parts around. It is far more efficient to build a machine in which everything

just moves on belts at a constant speed, adding small parts to larger ones and then bringing the larger ones together as you saw at the end of the hall above." This does seem like a more sensible way to churn out a lot of identical products, but it sounds like just more of the same. Gears like fused marbles, belts like coarse beadwork, drive shafts, pulleys, machines and more machines.

In a few places, marbles snap into new patterns to prepare a tool or make a product. Roll, roll, chug, chug, pop, snap, then roll and chug some more. As you leave the simulation hall, you ask, "Is there anything important I've missed in this molecular manufacturing tour?"

The tourguide launches into a list: "Yes–the inner workings of assembler arms, with drive shafts, worm gears, and harmonic drives; the use of Diels-Alder reactions, interfacial free-radial chain reactions, and dative-bond formation to join blocks together in the larger-scale stages of assembly; different kinds of mechanochemical processing for preparing reactive molecular tools; the use of staged-cascade methods in providing feed-molecules of the right kinds with near-perfect reliability; the differences between efficient and inefficient steps in molecular processing; the use of redundancy to ensure reliability in large systems despite sporadic damage; modern methods of building large objects from smaller blocks; modern electronic nanocomputers; modern methods for–"

"Enough!" you say, and the tourguide falls silent as you pitch it into a recycling bin.A course in molecular manufacturing isn't what you're looking for right now; the general idea seems clear enough. It's time to take another

look at the world on a more normal scale. Houses, roads, buildings, even the landscape looked different out there beyond the Faire dome–less crowded, paved, and plowed than you remember. But why?

The history books (well, they're more than just books) say that molecular manufacturing made a big difference; perhaps now the changes will make more sense. Yes, it's time to leave. As you toss your goggled, gloved jumpsuit into another bin, a striking dark-haired woman is taking a fresh one from a rack. She wears a jacket emblazoned with the name "Desert Rose NanoManufacturing."

"How'd you like it?" she asks with a smile.

"Pretty amazing," you say.

"Yes," she agrees. "I saw this sim back when I was taking my first molecular-manufacturing class. I swore I'd never design anything so clunky! This whole setup really brings back the memories–I can't wait to see if it's as crude as I remember." She steps into the simulation hall and closes the door.

CRUDE TECHNOLOGY

As the Silicon Valley Faire scenario shows, molecular manufacturing will work much like ordinary manufacturing, but with devices built so small that a single loose molecule of pollutant would be like a brick heaved into a machine tool. John Walker of Autodesk, a leading company in computer-aided design, observes that nanotechnology and today's crude methods are very different: "Technology has never had this kind of precise control; all of our technologies today arebulk technologies. We take a big chunk of stuff and

hack away at it until we're left with the object we want, or we assemble parts from components without regard to structure at the molecular level."

Molecular manufacturing will orchestrate atoms into products of symphonic complexity, but modern manufacturing mostly makes loud noises. These figurative noises are sometimes all too literal: A crack in a metal forging grows under stress, a wing fails, and a passenger jet crashes from the sky. A chemical reaction goes out of control, heat and pressure build, and a poisonous blast shakes the countryside. A lifesaving product cannot be made, a heart fails, and a hospital's heart-monitoring machine signals the end with a high-pitched wail.

Today, we make many things from metal, by machining. From the perspective of our standard, simulated molecular world, a typical metal part is a piece of terrain many days' journey across. The metal itself is weak compared to the bonds of the protein chain or other tough nanomechanisms: solid steel is no stronger than your simulated fingers, and the atoms on its surface can be pushed around with your bare hands. Standing on a piece of metal being machined in a lathe, you would see a cutting blade crawl past a few times per year, like a majestic plough the size of a mountain range. Each pass would rip up a strip of the metal landscape, leaving a rugged valley broad enough to hold a town.

This is machining from a nanotechnological perspective: a process that hacks crude shapes from intrinsically weak materials. Today, electronics are made from silicon chips. We have already seen the landscape of a finished chip. During manufacturing, metal features would be built up by a

centuries-long drizzle of metal-atom rain, and hollows would be formed by a centuries-long submergence in an acid sea. From the perspective of our simulation, the whole process would resemble geology as much as manufacturing, with the slow layering of sedimentary deposits alternating with ages of erosion.

The term nanotechnology is sometimes used as a name for small-scale microtechnology, but the difference between molecular manufacturing and this sort of microlandscaping is like the difference between watchmaking and bulldozing. Today, chemists make molecules by solution chemistry. We have seen what a liquid looks like in our first simulation, with molecules bumping and tumbling and wandering around. Just as assemblers can make chemical reactions occur by bringing molecules together mechanically, so reactions can occur when molecules bump at random through thermal vibration and motion in a liquid.

Indeed, much of what we know today about chemical reactions comes from observing this process. Chemists make large molecules by mixing small molecules in a liquid. By choosing the right molecules and conditions, they can get a surprising measure of control over the results: only some pairs of molecules will react, and then only in certain ways. Doing chemistry this way, though, is like trying to assemble a model car by putting the pieces in a box and shaking. This will only work with cleverly shaped pieces, and it is hard to make anything very complex. Chemists today consider it challenging to make a precise, three-dimensional structure having a hundred atoms, and making one with a thousand atoms is a great accomplishment. Molecular manufacturing,

in contrast, will routinely assemble millions or billions. The basic chemical principles will be the same, but control and reliability will be vastly greater. It is the difference between throwing things together blindly and putting them together with a watchmaker's care.

Technology today doesn't permit thorough control of the structure of matter. Molecular manufacturing will. Today's technologies have given us computers, spacecraft, indoor plumbing, and the other wonders of the modern age. Tomorrow's will do much more, bringing change and choices.

SIMPLE MATTER, SMART MATTER

Today's technology mostly works with matter in a few basic forms: gases, liquids, and solids. Though each form has many varieties, all are comparatively simple. Gases, as we've seen, consist of molecules ricocheting through space. A volume of gas will push against its walls and, if not walled in, expand without limit. Gases can supply certain raw materials for nanomachines, and nanomachines can be used to remove pollutants from air and turn them into something else.

Gases lack structure, so they will remain simple. Liquids are somewhat like gases, but their molecules cling together to form a coherent blob that won't expand beyond a certain limit. Liquids will be good sources of raw materials for nanomachines because they are denser and can carry a wide range of fuels and raw materials in solution (the pipe in the molecular-processing hall contained liquid). Nanomachines can clean up polluted water as easily as air, removing and transforming noxious molecules. Liquids have more structure than gases, but nanotechnology will have its greatest application to solids.

Solids are diverse. Solid butter consists of molecules stronger than steel, but the molecules cling to one another by the weaker forces of molecular stickiness. A little heat increases thermal vibrations and makes the solid structure disintegrate into a blob of liquid. Butterlike materials would make poor nanomachines. Metals consist of atoms held together by stronger forces, and so they can be structurally stronger and able to withstand higher temperatures.

The forces are not very directional, though, and so planes of metal atoms can slip past one another under pressure; this is why spoons bend, rather than break. This ability to slip makes metals less brittle and easier to shape (with crude technology), but it also weakens them. Only the strongest, hardest, highest-melting-point metals are worth considering as parts of nanomachines.

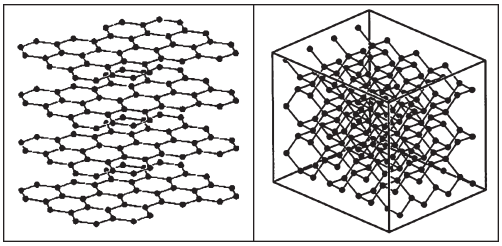


Fig. Carbon-Soft And Hard

On the left is graphite–the material called "lead" in pencils–made of carbon atoms. On right is diamond–the same atoms arranged in a different pattern. Diamond consists of carbon atoms held together by strong, directional bonds, like the bonds down the axis of a protein chain.

These directional bonds make it hard for planes of atoms to slip past one another, making diamond (and similar materials) very strong indeed–ten to a hundred times stronger than steel. But the planes can't easily slip, so when the material fails, it doesn't bend, it breaks.

Tiny cracks can easily grow, making a large object seem weak. Glass is a similar material: glass windows seem weak–and a scratch makes glass far weaker–yet thin, perfect glass fibres are widely used to make composite materials stronger and lighter than steel. Nanotechnology will be able to build with diamond and similar strong materials, making small, flawless fibres and components. In engineering today, diamond is just beginning to be used. Japan has pioneered a technology for making diamond at low pressure, and a Japanese company sells a speaker with excellent high-frequency response–the speaker cone is reinforced with a light, stiff film of diamond. Diamond is extraordinary stuff, made from cheap materials like natural gas. U.S. companies are scrambling to catch up.

All these materials are simple. More complex structures lead to more complex properties, and begin to give some hint of what molecular manufacturing will mean for materials. What if you strung carbon atoms in long chains with side-groups, a bit like a protein chain, and linked them into a big three-dimensional mesh? If the chains were kinked so that they couldn't pack tightly, they would coil up and flop around almost like molecules in a liquid, yet the strong bonds would keep the overall mesh intact. Pulling the whole network would tend to straighten the chains, but their writhing motions would tend to coil them back up. This sort of network has been made: it is called rubber.

Rubber is weak mostly because the network is irregular. When stretched, first one chain breaks, then another, because they don't all become taut at the same time to share and divide the load. A more regular mesh would be as soft as

rubber at first, but when stretched to the limit would become stronger than steel. Molecular manufacturing could make such stuff. The natural world contains a host of good materials–cellulose and lignin in wood, stronger-than-steel proteins in spider's silk, hard ceramics in grains of sand, and more. Many products of molecular manufacturing will be designed for great durability, like sand.

Others will be designed to fall apart easily for easy recycling, like wood. Some may be designed for uses where they may be thrown away. In this last category, nanotailored biodegradables will shine. With care, almost any sort of product from a shoe to computer-driven nanomachines can be made to last for a good long time, and then unzip fairly rapidly and very thoroughly into molecules and other bits of stuff all of kinds normally found in the soil. This gives only a hint of what molecular manufacturing will make possible by giving better control of the structure of solid matter. The most impressive applications will not be superstrong structural materials, improved rubber, and simple biodegradable materials: these are uniform, repetitive structures not greatly different from ordinary materials. These materials are "stupid."

When pushed, they resist, or they stretch and bounce back. If you shine light on them, they transmit it, reflect it, or absorb it. But molecular manufacturing can do much more. Rather than heaping up simple molecules, it can build materials from trillions of motors, ratchets, light-emitters, and computers. Muscle is smarter than rubber because it contains molecular machines: it can be told to contract. The products of molecular manufacturing can include materials

able to change shape, colour, and other properties on command. When a dust mote can contain a supercomputer, materials can be made smart, medicine can be made sophisticated, and the world will be a different place.

IDEAS AND CRITICISMS

We've just seen a picture of molecular manufacturing (of one sort) and of what it can do (in sketchy outline). Now let's look at the idea of nanotechnology itself: Where did it come from, and what do the experts think of it? This will have more to say on the latter point, presenting the thoughts of researchers who are advancing the field through their own work.

Origins

The idea of molecular nanotechnology, like most ideas, has roots stretching far back in time. In ancient Greece, Democritus suggested that the world was built of durable, invisible particles-atoms, the building blocks of solid objects, liquids, and gases. In the last hundred years, scientists have learned more and more about these building blocks, and chemists have learned more and more ways to combine them to make new things. Decades ago, biologists found molecules that do complex things; they termed them "molecular machines." Physicist Richard Feynman was a visionary of miniaturization who pointed towards something like molecular nanotechnology: on December 29, 1959, in an after-dinner talk at the annual meeting of the American Physical Society, he proposed that large machines could be used to make smaller machines, which could make still smaller ones, working in a top-down fashion from the

macroscale to the microscale. At the end of his talk, he painted a vision of moving individual atoms, pointing out, "The principles of physics, as far as I can see, do not speak against the possibility of maneuvering things atom by atom." He pictured making molecules, pointing clearly in the direction taken by the modern concept of nanotechnology: "But it is interesting that it would be, in principle, possible (I think) for a physicist to synthesize any chemical substance the chemist writes down. Give the orders, and the physicist synthesizes it. How? Put the atoms down where the chemist says, and so you make the substance."

Despite this clear signpost pointing to a potentially revolutionary area, no one filled the conceptual gap between miniature machines and chemical substances. There was no clear concept of making molecular machines able to build more such machines, no notion of controllable molecular manufacturing. With hindsight, one wonders why the gap took so long to fill. Feynman himself didn't follow it up, saying that the ability to maneuver atoms one by one "will really be useless" since chemists would come up with traditional, bulk-process ways to make new chemical substances.

For a researcher whose main interest was physics, he had contributed much just by placing the signpost: it was up to others to move forward. Instead, the idea of molecular machines for molecular manufacturing didn't appear for decades. From today's viewpoint, molecular nanotechnology looks more like an extension of chemistry than like an extension of miniaturization. A mechanical engineer, looking at nanotechnology, might ask, "How can machines be made so small?" A chemist, though, would ask, "How can molecules

be made so large?" The chemist has the better question. Nanotechnology isn't primarily about miniaturizing machines, but about extending precise control of molecular structure to larger and larger scales. Nanotechnology is about making (precise) things big.

Table. Macroscopic and Molecular Components

Technology	Function	Molecular Examples
Struts, beams, casins	Transmit force, hold positions	Cell walls, microtubules
Cables	Transmit tension	Collagen, silk
Fasteners, glue	Connect parts	Intermolecular forces
Solenoids, actuators	Move things	Muscle actin, myosin
Motors	Turn shafts	Flagellar motor
Drive shafts	Transmit torque	Bacterial flagella
Bearings	Support moving parts	Single bonds
Clamps	Hold workpieces	Enzymatic binding sites
Tools	Modify workpieces	Enzymes, reactive molecules
Production lines	Control devices ribosomes	Enzyme systems,
Numerical control systems	Store and read programmes	Genetic system

Nature gives the most obvious clues to how this can be done, and it was the growing scientific literature on natural molecular machines that led one of the present authors to propose molecular nanotechnology of the sort described here. A strategy to reach the goal was part of the concept: Build increasingly complex molecular machinery from simpler pieces, including molecular machines able to build more molecular machines.

And the motivation for studying this, and publishing? Largely the fear of living in a world that might rush into the new technology blindly, with ugly consequences. This concept and initial exploratory work started in early 1977 at MIT; the

first technical publication came in 1981 in the Proceedings of the National Academy of Sciences. For years, MIT remained the centre of thinking on nanotechnology and molecular manufacturing: in 1985, the MIT Nanotec-hnology Study Group was formed; it soon initiated an annual lecture series which grew into a two-day symposium.

The first book on the topic, Engines of Creation, was published in 1986. In 1988, Stanford University became the first to offer a course in molecular nanotechnology, sponsored by the Department of Computer Science. In 1989, this department hosted the first major conference on the subject, cosponsored by the Foresight Institute and Global Business Network. With the upcoming publication of a technical book describing nanotechnology–from molecular mechanical and quantum-mechanical principles up to assembly systems and products–the subject will be easier to teach, and more college courses will become available.

In parallel with the development and spread of ideas about nanotechnology and molecular manufacturing–ideas that remain pure theory, however well grounded–scientists and engineers, working in laboratories to build real tools and capabilities, have been pioneering roads to nanotechnology. Research has come a long way since the mid-1980s, as we'll see. But, as one might expect with a complex new idea that, if true, disrupts a lot of existing plans and expectations, some objections have been heard.

"It Won't Work"

Life might be much simpler if these ideas about nanotechnology had some fatal flaw. If only molecules

couldn't be used to form machines, or the machines couldn't be used to build things, then we might be able to keep right on going with our crude technologies: our medicine that doesn't heal, our spacecraft that don't open a new frontier, our oil crises, our pollution, and all the limits that keep us from trading familiar problems for strange ones. Most new ideas are wrong, especially if they purport to bring radical changes. It is not unreasonable to hope that these are wrong. From years of discussions with chemists, physicists, and engineers, it is possible to compile what seems to be a complete list of basic, critical questions about whether nanotechnology will work. The questioners generally seem satisfied with the answers.

"Will Thermal Vibrations Mess Things Up?"

The earlier scenarios describe the nature of thermal vibration and the problems it can cause. Designing nanomachines strong enough and stiff enough to operate reliably despite thermal vibration is a genuine engineering challenge. But calculating the design requirements usually requires only simple textbook principles, and these requirements can be met for everything described in this book.

"Will Quantum Uncertainty Mess Things Up?"

Quantum mechanics says that particles must be described as small smears of probability, not as points with perfectly defined locations. This is, in fact, why the atoms and molecules in the simulations felt so soft and smooth: their

electrons are smeared out over the whole volume of the molecule, and these electron clouds taper off smoothly and softly towards the edges.

Atoms themselves are a bit uncertain in position, but this is a small effect compared to thermal vibrations. Again, simple textbook principles apply, and well-designed molecular machines will work.

"Will Loose Molecules Mess Things Up?"

Chemists work with loose molecules in liquids, and they naturally tend to picture molecules as flying around loose. It is possible to build nanomachines and molecular-manufacturing systems that work in this sort of environment (biological mechanisms are an existence proof), but in the long run, there will be no need to do so. The Silicon Valley Faire simulation gives the right idea: Systems can be built with no loose molecules, making nanomechanical design much easier. If no molecules are loose inside a machine, then loose molecules can't cause problems there.

"Will Chemical Instability Mess Things Up?"

Chemists perform chemical reactions, which means that they tend to work with reactive, unstable molecules. Many molecules, though, can sit around in peace with their neighbors for millions of years, as is known both from chemical theory and from the study of molecules trapped in ancient rock.

Nanomachines can be built from the more stable sorts of structure. The only necessary exception is in molecular

assembly, where molecules must react, but even here the reactive molecules need not be turned loose. They can be applied just when and where they are needed in the construction process.

"Is It Too Complex, Like Biology?"

An easy way to explain molecular manufacturing is to say that it is somewhat like molecular biology: small, complex molecular devices working together to build things and do various jobs. The next point, however, is that molecular manufacturing is different in every detail and different in overall structure: compare the nanocomputers, assembler arms, and conveyor belts described above to the shaggy, seething living cell. Biology is complex in a strange and wonderful way. Engineers need not even understand life, much less duplicate it, merely to build a molecular-scale factory.

"I don't see anything wrong with it. But it's so interdisciplinary–couldn't there be a problem I can't see?" Nanotechnology is basically a shotgun marriage of chemistry and mechanical engineering, with physics (as always) presiding. This makes a complete evaluation difficult for most of today's specialists, because each of these fields is taught separately and usually practiced separately. Many specialists, having highly focused backgrounds, find themselves unequipped to evaluate proposals that overlap other disciplines. When asked to do so, they will state feelings of discomfort, because although they can't identify any particular problems, they can't verify the entire concept as sound. Scientists and engineers with multidisciplinary

backgrounds, or with access to specialists from other fields, can evaluate the idea from all sides.

It Will Work

When physicists, chemists, biologists, engineers, and computer scientists evaluate those parts of nanotechnology that fall within their disciplines, they agree: At no point would it require new principles or violate a physical law.

There may for many years be some experts offering negative off-the-cuff opinions, but the consensus among those who have taken the time to examine the facts is clear. Molecular nanotechnology falls entirely within the realm of the possible.

"It Would Work, but Isn't It a Bad Idea to Implement It?"

If this means, "These new technologies could easily do far more harm than good," then there is no argument, because no one seems to disagree.

If this means, "These new technologies will certainly do more harm than good," then we disagree: much good is possible, much harm is avoidable, and it would be too bold to declare any such outcome "certain." If this means, "These new technologies should be avoided," then we reply, "How, with what risks, and with what consequences?" It is conclude that it is safer to ride the beast than to hang on to its tail while others swarm aboard.

If this means, "Don't think about it or describe it," then we reply, "How else are we to understand it or make decisions?" Increased human abilities have routinely been used to damage the environment and to make war. Even the crude

technologies of the twentieth century have taken us to the brink. It is natural to feel exhilarated (or terrified) by a prospect that promises (or threatens) to extend human abilities beyond most past dreams (or nightmares). It is better to feel both, to meld and moderate these feelings, and to set out on a course of action that makes bad outcomes less likely. We're convinced that the best course is to focus on the potential good while warning of the potential evils.

"But Isn't It Unlikely to Arrive Within Our Lifetimes?"

Those in failing health may be justified in saying this; others are expressing an opinion that may well be wrong. It would be optimistic to assume that benefits are around the corner, and prudent to assume that they will be long delayed. Conversely, it would be optimistic to assume that dangers will be long delayed, and prudent to assume that they will arrive promptly.

Whatever good or ill may come of post-breakthrough capabilities, the turbulence of the coming transition will present a real danger. While we invite readers to take a "What if?" stance towards these technologies, it would be imprudent to listen to the lulling sound of the promise "not in our lifetimes."

Even today, public acceptance of man's coming exploration of space is slow. It is considered an event our children may experience, but certainly not one that we shall see.

PERSPECTIVE

We are still many years away from nanotechnology based on molecular manufacturing. It might even seem that such

vast, slow giants as ourselves could never make such small, quick machines. The following sections will describe how advances in science and technology are leading towards these abilities. We'll try to get some feel for the road ahead, for its length, and for how fast we're moving.

We are already surprisingly close to developing a crude molecular manufacturing technology, and getting visibly closer every week. The first, crude technology will enable the construction of molecular machines that can be used to build better molecular machines, climbing a ladder of capabilities that leads to general-purpose molecular assemblers as good or better than those described here. The opportunities then will be enormous.

If we haven't prepared, the dangers, too, will be enormous. Whether we're ready or not, the resulting changes will be disruptive, sweeping industries aside, upending military strategies, and transforming our ways of life.

THE MOLECULAR WORLD

Nanotechnology will be a bottom-up technology, building upward from the molecular scale. It will bring a revolution in human abilities like that brought by agriculture or power machinery. It can even be used to reverse many of the changes brought by agriculture or power machinery. But we humans are huge creations with no direct experience of the molecular world, and this can make nanotechnology hard to visualize, hence hard to understand. Scientists working with moleculas face this problem today. They can often calculate how molecules will behave, but to understand this behaviour, they need more than heaps of numbers: they need pictures,

movies, and interactive simulations, and so they are producing them at an ever-increasing pace. The U.S. National Science Foundation has launched a programme in "scientific visualization", in part to harness supercomputers to the problem of picturing the molecular world.

Molecules are objects that exert forces on one another. If your hands were small enough, you could grab them, squeeze them, and bash them together. Understanding the molecular world is much like understanding any other physical world: it is a matter of understanding size, shape, strength, force, motion, and the like–a matter of understanding the differences between sand, water, and rock, or between steel and soap bubbles. Today's visualization tools give a taste of what will become possible with tomorrow's faster computers and better "virtual realities," simulated environments that let you tour a world that "exists" only as a model inside the computer. Before discussing nanotechnology and how it relates to the technologies of today, let's try to get a more concrete understanding of the molecular world by describing a simulation embedded in a scenario. In this scenario, events and technologies described as dating from 1990 or before are historically accurate; those with later dates are either projections or mere scenario elements. The descriptive details in the simulation are written to fit designs and calculations based on standard scientific data, so the science isn't fiction.

EXPLORING THE MOLECULAR WORLD

In a scenario, we saw Joel Gregory manipulating molecules in the virtual reality of a simulated world using video goggles, tactile gloves, and a supercomputer. The early twenty-first

century should be able to do even better. Imagine, then, that today you were to take a really long nap, oversleep, and wake up decades later in a nanotechnological world. In the twenty-first century, even more than in the twentieth, it's easy to make things work without understanding them, but to a newcomer much of the technology seems like magic, which is dissatisfying. After a few days, you want to understand what nanotechnology is, on a gut level. Back in the late twentieth century, most teaching used dry words and simple pictures, but now—for a topic like this—it's easier to explore a simulated world. And so you decide to explore a simulation of the molecular world.

Looking through the brochure, you read many tedious facts about the simulation: how accurate it is in describing sizes, forces, motions, and the like; how similar it is to working tools used by both engineering students and professionals; how you can buy one for your very own home, and so forth. It explains how you can tour the human body, see state-of-the-art nanotechnology in action, climb a bacterium, etc. For starters, you decide to take an introductory tour: simulations of real twentieth-century objects alongside quaint twentieth-century concepts of nanotechnology. After paying a small fee and memorizing a few key phrases, you pull on a powersuit, pocket a Talking Tourguide, step into the simulation chamber, and strap the video goggles over your eyes. Looking through the goggles, you seem to be in a room with a table you know isn't really there and walls that seem too far away to fit in the simulation chamber.

But trickery with a treadmill floor makes the walk to the walls seem far enough, and when you walk back and thump

the table, it feels solid because the powersuit stops your hand sharply at just the right place.

You can even feel the texture of the carvings on the table leg, because the suit's gloves press against your fingertips in the right patterns as you move. The simulation isn't perfect, but it's easy to ignore the defects. On the table is an old 1990s silicon computer chip. When you pick it up, as the beginners' instructions suggest, it looks like Figure.

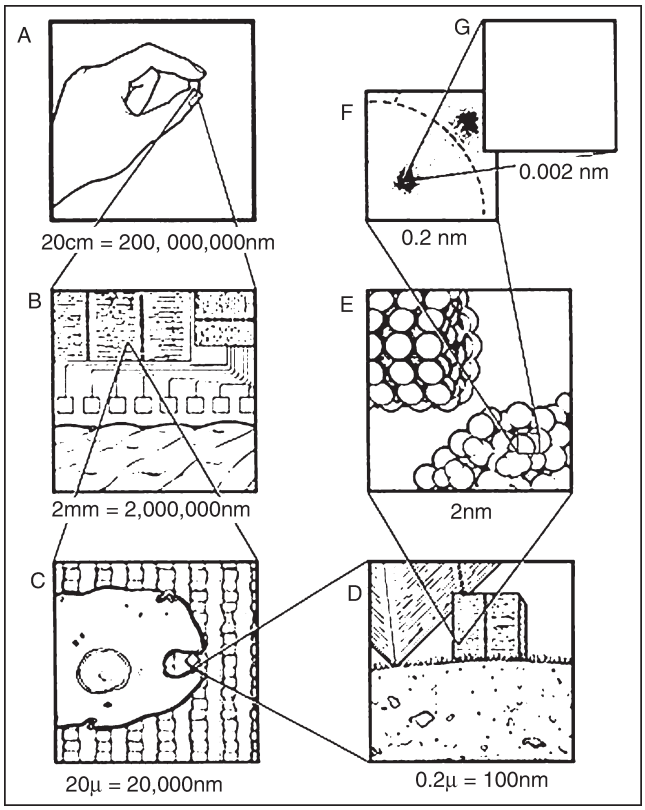


Fig. Power of Ten

Frame:

(A) Shows a hand holding a computer chip. This is shown magnified 100 times in

(B) Another factor of 100 magnification

(C) Shows a living cell placed on the chip to show scale. Yet another factor of 100 magnification

(D) Shows two nanocomputers beside the cell. The smaller (shown as block) has roughly the same power as the chip seen in the first view; the larger (with only the corner visible) is as powerful as mid-1980s mainframe computer. Another factor of 100 magnification

(E) Shows an irregular protein from the cell on the lower right, and a cylindrical gear made bymolecular manufacturing at top left. Taking a smaller factor of 10 jump,

(F) Shows two atoms in the protein, with electron clouds represented by stippling. A final factor of 100 magnification

(G) Reveals the nucleus of the atom as a tiny speck.

VISION AND MOTION

You feel as though you're falling towards the chip's surface, shrinking rapidly. In a moment, it looks roughly like Figure 1B, with your thumb still there holding it. The world grows blurrier, then everything seems to go wrong as you approach the molecular level. First, your vision blurs to uselessness—there is light, but it becomes a featureless fog. Your skin is tickled by small impacts, then battered by what feel like hard-thrown marbles.

Your arms and legs feel as though they are caught in turbulence, pulling to and fro, harder and harder. The ground hits your feet, you stumble and stick to the ground like a fly on flypaper, battered so hard that it almost hurts. You asked for realism, and only the built-in safety limits in the suit keep the simulated thermal motions of air molecules and of your own arms from beating you senseless. "Stop!" gives you

a rest from the suit's yanking and thumping, and "Standard settings!" makes the world around you become more reasonable. The simulation changes, introducing the standard cheats. Your simulated eyes are now smaller than a light wave, making focus impossible, but the goggles snap your vision into sharpness and show the atoms around you as small spheres. (Real nanomachinesare as blind as you were a moment ago, and can't cheat.)

You are on the surface of the 1990s computer chip, between a cell and two blocky nanocomputers like the ones in Figure 1D. Your simulated body is 50 nanometres tall, about 1/ 40,000,000 your real size, and the smaller nanocomputer is twice your height. At that size, you can "see" atoms and molecules, as in Figure 1E.

The simulation keeps bombarding you with air molecules, but the standard settings leave out the sensation of being pelted with marbles. A moment ago you were stuck tight to the ground by molecular stickiness, but the standard settings give your muscles the effective strength of steel—at least in simulation—by making everything around you much softer and weaker.

The tourguide says that the only unreal features of the simulation have to do with you—not just your ability to see and to ignore thermal shaking and bombardment, but also your sheer existence at a size too small for anything so complex as a human being. It also explains why you can see things move, something about slowing down everything around you by a factor of 10 for every factor of 10 enlargement, and by another factor to allow for your being made stronger and hence faster. And so, with your greater

strength and some adjustments to make your arms, legs, and torso less sticky, you can stand, see, feel, and take stock of the situation.

MOLECULAR TEXTURE

The ground underfoot, like everything around you, is pebbly with atom-sized bumps the size of your fingertips. Objects look like bunches of transparent grapes or fused marbles in a variety of pretty but imaginary colours. The simulation displays a view of atoms and molecules much like those used by chemists in the 1980s, but with a sharper 3-D image and a better way to move them and to feel the forces they exert. Actually, the whole simulation setup is nothing but an improved version of systems built in the late 1980s—the computer is faster, but it is calculating the same things. The video goggles are better and the whole-body powersuit is a major change, but even in the 1980s there were 3-D displays for molecules and crude devices that gave a sense of touching them.

The gloves on this suit give the sensation of touching whatever the computer simulates. When you run a fingertip over the side of the smaller nanocomputer, it feels odd, hard to describe. It is as if the surface were magnetic–it pulls on your fingertip if you move close enough. But the result isn't a sharp click of contact, because the surface isn't hard like a magnet, but strangely soft. Touching the surface is like touching a film of fog that grades smoothly into foam rubber, then hard rubber, then steel, all within the thickness of a sheet of corrugated cardboard.

Moving sideways, your fingertip feels no texture, no friction, just smooth bumps more slippery than oil, and a tendency

to get pulled into hollows. Pulling free of the surface takes a firm tug. The simulation makes your atom-sized fingertips feel the same forces that an atom would. It is strange how slippery the surface is—and it can't have been lubricated, since even a single oil molecule would be a lump the size of your thumb. This slipperiness makes it obvious how nano-scale bearings can work, how the parts of molecular machines can slide smoothly.

But on top of this, there is a tingling feeling in your fingers, like the sensation of touching a working loudspeaker. When you put your ear against the wall of the nanocomputer, you flinch back: for a moment, you heard a sound like the hiss of a twentieth—century television tuned to a channel with no broadcast, with nothing but snow and static—but loud, painfully loud. All the atoms in the surface are vibrating at high frequencies, too fast to see. This is thermal vibration, and it's obvious why it's also called thermal noise.

GAS AND LIQUID

Individual molecules still move too quickly to see. So, to add one more cheat to the simulation, you issue the command "Whoa!", and everything around seems to slow down by a factor of ten. On the surface, you now can see thermal vibrations that had been too quick to follow. All around, air molecules become easier to watch.

They whiz about as thick as raindrops in a storm, but they are the size of marbles and bounce in all directions. They're also sticky in a magnetlike way, and some are skidding around on the wall of the nanocomputer. When you grab one, it slips away. Most are like two fused spheres,

but you spot one that is perfectly round—it is an argon atom, and these are fairly rare. With a firm grip on all sides to keep it from shooting away like a watermelon seed, you pinch it between your steel-strong fingers.

It compresses by about 10 per cent before the resistance is more than you can overcome. It springs back perfectly and instantly when you relax, then bounces free of your grip. Atoms have an unfamiliar perfection about them, resilient and unchanging, and they surround you in thick swarms. At the base of the wall is a churning blob that can only be a droplet of water.

Scooping up a handful for a closer look yields a swarm of molecules, hundreds, all tumbling and bumbling over one another, but clinging in a coherent mass. As you watch, though, one breaks free of the liquid and flies off into the freer chaos of the surrounding air: the water is evapourating. Some slide up your arm and lodge in the armpit, but eventually skitter away. Getting rid of all the water molecules takes too much scraping, so you command "Clean me!" to dry off.

TOO SMALL AND TOO LARGE

Beside you, the smaller nanocomputer is a block twice your height, but it's easy to climb up onto it as the tourguide suggests. Gravity is less important on a small scale: even a fly can defy gravity to walk on a ceiling, and an ant can lift what would be a truck to us. At a simulated size of fifty nanometres, gravity counts for nothing. Materials keep their strength, and are just as hard to bend or break, but the weight of an object becomes negligible. Even without the

strength-enhancement that lets you overcome molecular stickiness, you could lift an object with 40 million times your mass–like a person of normal size lifting a box containing a half-dozen fully loaded oil tankers. To simulate this weak gravity, the powersuit cradles your body's weight, making you feel as if you were floating. This is almost like a vacation in an orbital theme park, walking with stickyboots on walls, ceilings, and whatnot, but with no need for antinausea medication.

On top of the nanocomputer is a stray protein molecule, like the one in Figure E. This looks like a cluster of grapes and is about the same size. It even feels a bit like a bunch of grapes, soft and loose. The parts don't fly free like a gas or tumble and wander like a liquid, but they do quiver like gelatin and sometimes flop or twist. It is solid enough, but the folded structure is not as strong as your steel fingers. In the 1990s, people began to build molecular machinery out of proteins, copying biology. It worked, but it's easy to see why they moved on to better materials.

From a simulated pocket, you pull out a simulated magnifying glass and look at the simulated protein. This shows a pair of bonded atoms on the surface at 10 times magnification, looking like Figure F. The atoms are almost transparent, but even a close look doesn't reveal a nucleus inside, because it's too small to see. It would take 1,000 times magnification to be able to see it, even with the head start of being able to see atoms with your naked eye.

How could people ever confuse big, plump atoms with tiny specks like nuclei? Remembering how your steel-strong fingers couldn't press more than a fraction of the way towards

the nucleus of an argon atom from the air, it's clear why nuclear fusion is so difficult. In fact, the tourguide said that it would take a real-world projectile over a hundred times faster than a high-powered rifle bullet to penetrate into the atomic core and let two nuclei fuse. Try as you might, there just isn't anything you could find in the molecular world that could reach into the middle of an atom to meddle with its nucleus. You can't touch it and you can't see it, so you stop squinting though the magnifying glass. Nuclei just aren't of much interest in nanotechnology.

PUZZLE CHAINS

Taking the advice of the tourguide, you grab two molecular knobs on the protein and pull. It resists for a moment, but then a loop comes free, letting other loops flop around more, and the whole structure seems to melt into a writhing coil. After a bit of pulling and wrestling, the protein's structure becomes obvious: It is a long chain–longer than you are tall, if you could get it straight—and each segment of the chain has one of several kinds of knobs sticking off to the side.

With the multicoloured, glassy-bead portrayal of atoms, the protein chain resembles a flamboyant necklace. This may be decorative, but how does it all go back together? The chain flops and twists and thrashes, and you pull and push and twist, but the original tight, solid packing is lost. There are more ways to go wrong in folding up the chain than there are in solving Rubik's Cube, and now that the folded structure is gone, it isn't even clear what the result should look like. How did those twentieth-century researchers ever solve the notorious "protein folding problem"? It's a matter of record that they started building protein objects in the late 1980s.

This protein molecule won't go back together, so you try to break it. A firm grip and a powerful yank straightens a section a bit, but the chain holds together and snaps back. Though unfolding it was easy, even muscles with the strength of steel—the strength of Superman—can't break the chain itself. Chemical bonds are amazingly strong, so it's time to cheat again. When you say, "Flimsy world–one second!" while pulling, your hands easily move apart, splitting the chain in two before its strength returns to normal.

You've forced a chemical change, but there must be easier ways since chemists do their work without tiny superhands. While you compare the broken ends, they thrash around and bump together. The third time this happens, the chain rejoins, as strong as before. This is like having snap-together parts, but the snaps are far stronger than welded steel. Modern assembler chemistry usually uses other approaches, but seeing this happen makes the idea of molecular assembly more understandable: Put the right pieces together in the right positions, and they snap together to make a bigger structure. Remembering the "Whoa!" command, you decide to go back to the properly scaled speed for your size and strength. Saying "Standard settings!," you see the thrashing of the protein chain speed up to hard-to-follow blur.

NANOMACHINES

At your feet is a ribbed, ringed cylindrical object about the size of a soup can—not a messy, loosely folded strand like the protein (before it fell apart), but a solid piece of modern nanotechnology. It's a gear like the one in Figure 1E. Picking it up, you can immediately feel how different it is from a

protein. In the gear, everything is held in place by bonds as strong as those that strung together the beads of the protein chain. It can't unfold, and you'd have to cheat again to break its perfect symmetry.

Like those in the wall of the nanocomputer, its solidly attached atomsvibrate only slightly. There's another gear nearby, so you fit them together and make the atomic teeth mesh, with bumps on one fitting into hollows on the other. They stick together, and the soft, slick atomic surfaces let them roll smoothly. Underfoot is the nanocomputer itself, a huge mechanism built in the same rigid style. Climbing down from it, you can see through the transparent layers of the wall to watch the inner works. An electric motor an arm-span wide spins inside, turning a crank that drives a set of oscillating rods, which in turn drive smaller rods.

This doesn't look like a computer; it looks more like an engineer's fantasy from the nineteenth century. But then, it is an antique design–the tourguide said that the original proposal was a piece of exploratory engineering dating from the mid-1980s, a mechanical design that was superseded by improved electronic designs before anyone had the tools to build even a prototype. This simulation is based on a version built by a hobbyist many years later. The mechanical nanocomputer may be crude, but it does work, and it's a lot smaller and more efficient than the electronic computers of the early 1990s. It's even somewhat faster. The rods slide back and forth in a blur of motion, blocking and unblocking each other in changing patterns, weaving patterns of logic.

This nanocomputer is a stripped-down model with almost no memory, useless by itself. Looking beyond it, you see the

other block–the one on the left in Figure D–which contains a machine powerful enough to compete with most computers built in 1990. This computer is a millionth of a metre on a side, but from where you stand, it looks like a blocky building looming over ten stories tall. The tourguide says that it contains over 100 billion atoms and stores as much data as a room full of books.

You can see some of the storage system inside: row upon row of racks containing spools of molecular tape somewhat like the protein chain, but with simple bumps representing the 1s and 0s of computer data.

These nanocomputers seem big and crude, but the ground you're now standing on is also a computer–a single chip from 1990, roughly as powerful as the smaller, stripped-down nanocomputer at your side. As you gaze out over the chip, you get a better sense for just how crude things were a few decades ago. At your feet, on the smallest scale, the chip is an irregular mess. Although the wall of the nanocomputer is pebbly with atomic-scale bumps, the bumps are as regular as tile. The chip's surface, though, is a jumble of lumps and mounds.

This pattern spreads for dozens of paces in all directions, ending in an irregular cliff marking the edge of a single transistor. Beyond, you can see other ridges and plateaus stretching off to the horizon. These form grand, regular patterns, the circuits of the computer. The horizon–the edge of the chip–is so distant that walking there from the centre would (as the tourguide warns) take days. And these vast pieces of landscaping were considered twentieth-century miracles of miniaturization?

CELLS AND BODIES

Even back then, research in molecular biology had revealed the existence of smaller, more perfect machines such as the protein molecules in cells. A simulated human cell–put here because earlier visitors wanted to see the size comparisons–its on the chip next to the smaller nanocomputer. The tourguide points out that the simulation cheats a bit at this point, making the cell act as though it were in a watery environment instead of air.

The cell dwarfs the nanocomputer, sprawling across the chip surface and rearing into the sky like a small mountain. Walking the nature trail around its edge would lead across many transistor-plateaus and take about an hour. A glance is enough to show how different it is from a nanocomputer or a gear: it looks organic, it bulges and curves like a blob of liver, but its surface is shaggy with waving molecular chains. Walking up to its edge, you can see that the membrane wrapping the cell is fluid (cell walls are for stiff things like plants), and the membrane molecules are in constant motion. On an impulse, you thrust your arm through the membrane and poke around inside.

You can feel many proteins bumping and tumbling around in the cell's interior fluid, and a crisscrossing network of protein cables and beams. Somewhere inside are the molecular machines that made all these proteins, but such bits of machinery are embedded in a roiling, organic mass. When you pull your arm out, the membrane flows closed behind. The fluid, dynamic structure of the cell is largely self healing. That's what let scientists perform experimental surgery on cells with the old, crude tools of the twentieth

century: They didn't need to stitch up the holes they made when they poked around inside. Even a single human cell is huge and complex. No real thinking being could be as small as you are in the simulation: A simple computer without any memory is twice your height, and the larger nanocomputer, the size of an apartment complex, is no smarter than one of the submoronic computers of 1990. Not even a bendable finger could be as small as your simulated fingers: in the simulation, your fingers are only one atom wide, leaving no room for the slimmest possible tendon, to say nothing of nerves. For a last look at the organic world, you gaze out past the horizon and see the image of your own, full-sized thumb holding the chip on which you stand. The bulge of your thumb rises ten times higher than Mount Everest. Above, filling the sky, is a face looming like the Earth seen from orbit, gazing down.

It is your own face, with cheeks the size of continents. The eyes are motionless. Thinking of the tourguide's data, you remember: The simulation uses the standard mechanical scaling rules, so being 40 million times smaller has made you 40 million times faster. To let you pull free of surfaces, it increased your strength by more than a factor of 100, which increased your speed by more than a factor of 10. So one second in the ordinary world corresponds to over 400 million here in the simulation. It would take years to see that huge face in the sky complete a single eyeblink.

2

Quantum Dots

The use of semiconductors has greatly increased in the last century. As new technologies start to rely more and more on semi-conductors, their shortcomings are more and more apparent. Traditional semi-conductor devices have been found to be too big and too slow.

As engineers search for a faster and more adaptable alternative to conventional semiconductors they have discovered quantum dots, a new form of semiconductors that model atoms. Being only nanometres in size, these pseudo-atoms take semi-conductors to a whole new level and can allow devices to work almost at the speed of light. Furthermore, quantum dots have numerous applications in optical technologies, mediums, and industries. This paper seeks to introduce the principle of quantum dots, their creation methods, and their applications. Modern electronics, as well as many other fields of science, rely on the use of

semi-conductors. Quantum dots (QDs) are particles that hold a droplet of free electrons which simulates "the ultimate miniaturized semiconductor." Any material that can conduct electricity better than an insulator but not as well as a conductor is considered a semi-conductor. What makes semi-conductors so important is that their unique structure allows different semi-conductors to carry current under different circumstances. This gives the user more control over the flow of current. Most semi-conductors are crystalline substances such as germanium and silicon. We can see its use from the basis of electronic parts such as diodes and transistors to biomedical processes.Conventional semi-conductors are used often in electrical circuits. However, they have limited ranges of tolerance for the frequency of the current they carry. The low tolerance of traditional semi-conductors often poses a problem to circuits, and many of its other applications. This is what makes the use of quantum dots so important. As they are fabricated artificially, different quantum dots can be made to tolerate different current frequencies through a much larger range than conventional ones.

The use of quantum dots as semi-conductors offers more freedom to just about everything involving the use of semi-conductors. Quantum dots can best be described as false atoms. The primary material that a quantum dot is made out of is called a "hole", or a substance that is missing an electron from its valence band giving it a positive charge. The primary material is extremely small, which is why it is called a dot, and at that size, electrons start to orbit it. Since quantum dots do not have protons or neutrons in the centre,

their mass is much smaller. Since the mass at the centre is smaller than that of an atom, quantum dots exert a smaller force on the orbiting electrons causing an orbit larger than that of a regular atom. Daneshvar, personal communication, Jul 15, 2005). With a mass that small, scientists are able to precisely calculate and change the size of the band-gap of the quantum dot by adding or taking electrons. The band-gap of a quantum dot is what determines which frequencies it will respond to, so being able to change the band-gap is what gives scientists more control and more flexibility when dealing with its applications.

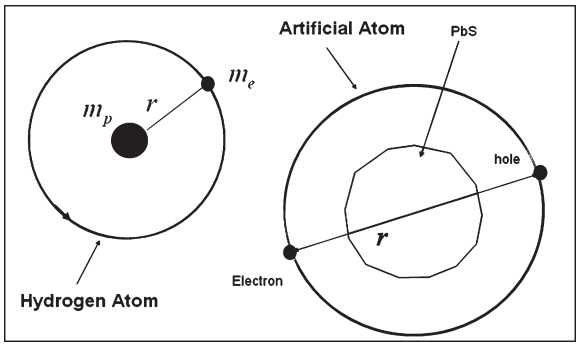


Fig. The Orbit of a Hydrogen Atom to that of a Quantum Dot.

One way to synthesize quantum dots is through molecular beam epitaxy. In this process, certain chemicals are evapourated and then sprayed to condense into small objects on a substrate. The condensation of the chemical on the substrate is similar to water on glass. If someone drops water on glass the water condenses into many balls.

As more layers are sprayed onto the substrate the size of the balls starts to build up into pyramid-shaped objects. Eventually, the balls build up to a specific size and they're quantum dots. This process has some downsides though. It is much harder to use quantum dots while they are still

attached to the substrate. While they are all attached together on the substrate they act as one solid which almost defeats the purpose of creating the quantum dots.

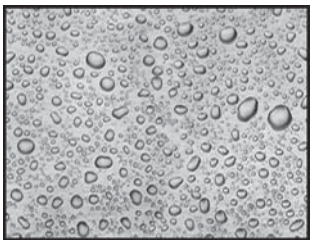


Fig. The Different Chemicals, once Sprayed onto the Substrate, Acts Almost like Water and form Together in "Balls", or Quantum Dots.

Another way to form quantum dots is through electron beam lithography. This process is a little like etching a chip. A mask is created with an electron beam that has many tiny holes in it. Then evapourated chemicals, similar to the ones used in epitaxy, are sprayed through the mask onto a substrate, creating many little balls.This process has some of the same shortcomings as epitaxy, mainly that the quantum dots are still connected to the substrate after synthesis.

Additionally, scientists have found it difficult to create such small masks that need to have holes just nanometres in diametre. Lithography was originally a very popular process for creating quantum dots; however, this process creates many defects and is slow compared to the other processes.

Colloidal synthesis is a process that involves creating quantum dots in a liquid. This is by far the best technique for the formation of quantum dots because the process can occur under "benchtop conditions," or in a normal laboratory setting.

When certain materials, primarily those from periodic groups two through four, are dissolved in a certain type of polymer solute the solution can enter into a phase where particles can come together to form quantum dots. Since the size is dependent on time, the longer the dots are left in the solution the bigger they get.

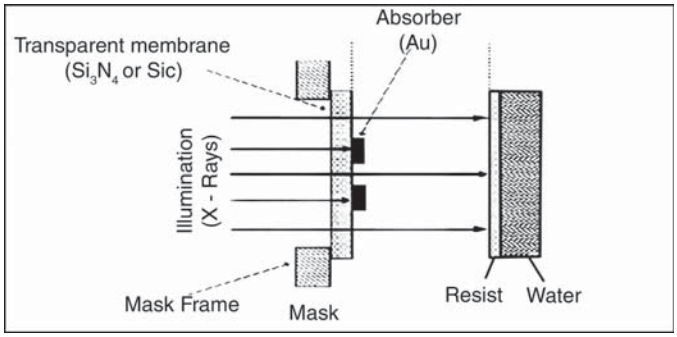


Fig. The X-ray Light Reacts with the Photo-resist on the Water to Create Quantum Dots.

This is in part what makes colloidal synthesis the most popular method. Scientists can use time to change the properties of the quantum dot, engineering it for certain light frequencies. This process, unlike lithography and epitaxy, synthesizes the quantum dots in such a way that they are suspended individually, making it easier for use in applications. As mentioned, QDs have many interesting characteristics that not only contribute to numerous applications but also display phenomena that are consequential to the fundamental study of Physics. Typical dimensions of QDs are between nanometres to a few microns. Due to their extremely small size, QDs can be controlled by a few electrons and this provides many advantages that can optimize devices. In addition to the nanoscale size that they exhibit, QDs enable superior transport and optical properties that has beneficial applications in the fields including

immunocytoc-hemistry and the study of lasers. In the biological field of science, QDs have become known to be very useful. Recent studies of QDs have resulted in developing new fluorescence immunocytochemical probes. A probe is a substance that is radioactively labeled or otherwise marked and used to detect or identify another substance in a sample. A fluorescence immunocytochemical probe is usually used to detect antigens in tissues.

In contrast to organic fluorophores, which are not photostable, QDs have properties of high brightness, photostability, narrow emission spectra and an apparent large Stokes' shift, thus they can replace the usage of organic fluorophores.

The current mode of detecting the antigens which takes from two to six days can speed up to a matter of hours using quantum dots.

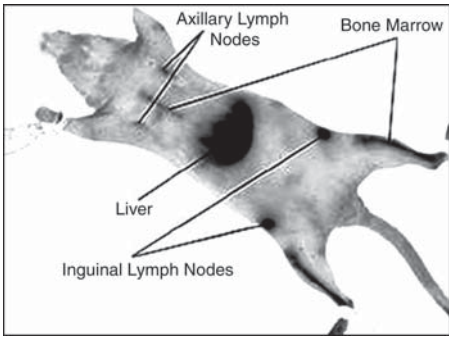


Fig. Immunocytochemical Probes are used in the Dead Rodent. The Probes in the Body have Circulated and now Show up under Florescent Light, Creating a much Safer Alternative to the X-ray.

Prior to the introduction of the QD, microelectronic technology has focused on reducing the size of transistors to produce increasingly smaller, faster and more efficient computers (Shrinking Information Storage to the Molecular Level). However, this method is reaching its physical limit

due to the restrictions placed by the laws of physics that do not allow these devices to operate below a certain size. With this advantageous feature of QDs, information storage can be brought down to the molecular level. Since no flow of electrons to transmit a signal is needed, electric current does not need to be produced and heat problems are avoided. Also, the quantum dot devices are sensitive enough to and can make a usage of the charges of single electrons.

With improvements in quantum-dot ordering and positioning, it is possible for us to hope in the near future to address and store information optically in a single quantum dot, thus opening the possibility of ultrahigh-density memory devices.

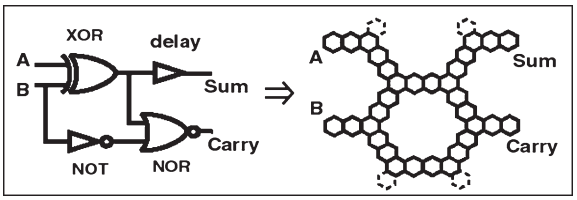
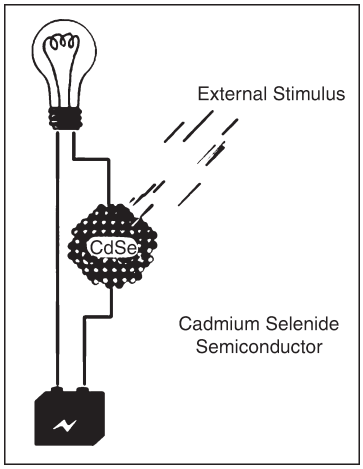


Fig. Nanocomputers might have a Completely new type of Structure made up of 'Cells'. One way of Building this
Structure would be using Quantum-Dots.

QDs also have other applications like quantum dot lasers which promises far more great advantages than quantum well lasers. Because QD lasers are less temperature-dependent and less likely to degrade under elevated temperature, it allows more flexibility for lasers to operate more efficiently.

Other beneficial features of QD lasers include low threshold currents, higher power, and great stability compared to the restrained performance of the conventional lasers. Respectively, the QD laser will play a significant role in optical data communications and optical networks. Optical switches

have been a major research objective in the scientific community. The use of optical switches would increase the rate at which data can be transferred. With regular switches, data can only travel as fast as the electrical current can, but optical switches can travel,



Almost as fast as the speed of light. The principle of optical switches is that semi-conductors will only allow certain levels of energy to pass through it. So if we place a quantum dot semi-conductor in a circuit, but supply a voltage below the acceptable range, current will not flow. However, if we shine a light on the quantum dot semi-conductor, it would put enough energy into the semi-conductor that it will allow current to flow. This idea is mainly for powering electronic devices, but using quantum-dots as receivers for electrical data is just a step up.

Quantum dots have applications outside of biology and engineering. An idea that may be instituted in the future would be against counterfeiting money. The treasury could engineer quantum dots to be responsive to a specific frequency of light and suspend them in ink that they would print onto money. Shining light with the same frequency

that the ink solution has been engineered for would reveal whether or not the money is real or counterfeit. This idea can be used for just about any substance that could be illegally duplicated. Another security application involves attaching quantum dots to dust. QDs can be engineered so that they have the same properties as dust and give off infrared radiation. In hostile areas, this "quantum dust" can be used to track wanted criminals or the movement of hostile activity. In urban areas, "quantum dust" can be used as a security device to set off alarms if the infrared radiation is detected. Though QDs are still under research for other possible applications and need more technological advancement in order to be put into use, the features introduced will grant far better optical communication, significant change in electronic devices, and even detection of antigens in the body tissues.

HAZY SHELLS OF COMPUTRONIUM THAT RING THE SUN

Concentric clouds of nanocomputers the size of rice grains, powered by sunlight, orbiting in shells like the packed layers of a Matrioshka doll – are still immature, holding barely a thousandth of the physical planetary mass of the system, but they already support a classical computational density of 10^{42} MIPS; enough to support a billion civilizations as complex as the one that existed immediately before the great disassembly. The conversion hasn't yet reached the gas giants, and some scant outer-system enclaves remain independent – Amber's Ring Imperium still exists as a separate entity, and will do so for some years to come – but the inner solar system planets, with the exception of Earth,

have been colonized more thoroughly than any dusty NASA proposal from the dawn of the space age could have envisaged.

From outside the Accelerated civilization, it isn't really possible to know what's going on inside. The problem is bandwidth: While it's possible to send data in and get data out, the sheer amount of computation going on in the virtual spaces of the Acceleration dwarfs any external observer. Inside that swarm, minds a trillion or more times as complex as humanity think thoughts as far beyond human imagination as a microprocessor is beyond a nematode worm. A million random human civilizations flourish in worldscapes tucked in the corner of this world-mind. Death is abolished, life is triumphant.

A thousand ideologies flower, human nature adapted where necessary to make this possible. Ecologies of thought are forming in a Cambrian explosion of ideas: For the solar system is finally rising to consciousness, and mind is no longer restricted to the mere kilotons of gray fatty meat harbored in fragile human skulls.

PROJECT

It has to be clearly stated that current operating speeds of nano-electromechanical single electron transistors (NEMSETs) are of the order of 1 GHz, which is not competitive with standard complimentary metal oxide semiconductors (CMOS_ As we have found in recent measurements self-excitation can be exploited to generate mechanical oscillations without any ac excitation. Hence, dc voltages are sufficient to operate the NMC. Basically, a dc voltage creates an electric

field to support mechanical oscillations of the nanopillars. A classical example is straightforward to construct. It has to be noted that onset of the mechanical oscillations is induced by a thermal fluctuation, which is found to be enhanced, if the electrical field is inhomogeneous.

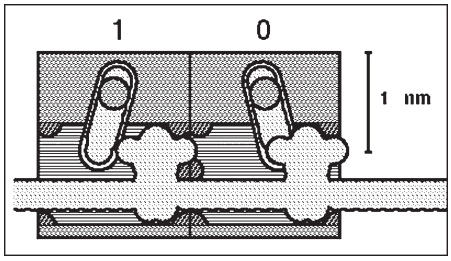
The current work that is described as nanomechanical, will still be using DC current. However, a mechanical piece the pillar controls the flow of current. We propose a fully mechanical computer based on nanoelectro-mechanical elements. Our aim is to combine this classical approach with modern nanotechnology to build a nanomechanical computer (NMC) based on nanomechanical transistors.

The main motivation behind constructing such a computer is three fold:

1. Mechanical elements are more robust to electromagnetic shocks than current dynamic random access memory (DRAM) based purely on complimentary metal oxide semiconductor (CMOS) technology,
2. The power dissipated can be orders of magnitude below CMOS and
3. The operating temperature of such an NMC can be an order of magnitude above that of conventional CMOS.

Drexler's work on nanomechanical computer concepts is not mentioned. They do discuss the potential for reversible computing implementation.

A summary of Nanosystems is here There was an analysis and simulation of the Drexler Nanocomputer architecture by Bryan Wagner The Drexler idea was based on nanoscale rod logic



Mechanism for two nanocomputer gates, initial position. One control rod with two gate knobs is seen laterally; two more rods with knobs are seen end on. Each rod with associated knobs is a single molecule

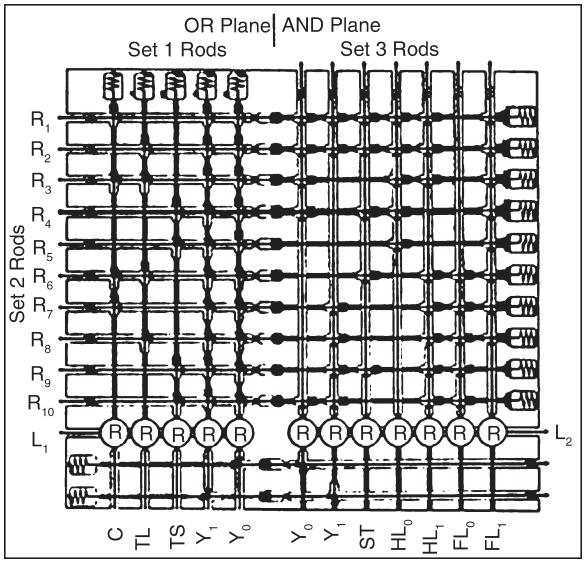


Fig. Schematic of Programmable Logic Array (PLA) Finite State Machine Implementing Rod Logic for a Nanomachanical Central Processing unit.

Drexler chose to model this cruder system to show that even simple and easy to define mechanical processes could have interesting performance at the nanoscale Robert Freitas's Nanomedicine book describes nanomechanicaland nanoelectronic computers, biocomputers and briefly examines the ultimate limits to computation including reversible and quantum computing.

3

Computational Nanotechnology

The concept of a Universal Computer, a device able to compute anything computable, dates back to Babbage in the early part of the 19th century , and to Turing, Church, and von Neumann in the 20th century[18].

The concept of a “Universal Constructor” is perhaps more recent. The concept was well understood by von Neumann in the 1940’s, who defined a “Universal Constructor” in a two-dimensional cellular automata world.

Such a model is a mathematical abstraction something like an infinite checkerboard, with several different types of “checkers” that might be on each square. The different pieces spontanesouly change and move about depending on what pieces occupy neighboring squares, in accordance with a pre-defined set of rules.

Von Neumann used the concept of a Universal Constructor in conjunction with a Universal Computer as the core

components in a self-replicating system . The possibility of fabricating structures by putting "...the atoms down where the chemist says..." was recognized by Feynman in 1959. Drexler recognized the value of the "assembler" in 1977.

The assembler is analogous to von Neumann's Universal Constructor, but operates in the normal three dimensional world and can build a large, atomically precise structures by manipulating atoms and small clusters of atoms. He published the concept in 1981 and in subsequent work has proposed increasingly detailed designs for such devices.

The basic design of Drexler's assembler consists of (1) a molecular computer, (2) one or more molecular positioning devices (which might resemble very small robotic arms), and (3) a well defined set of chemical reactions (perhaps one or two dozen) that take place at the tip of the arm and are able to fabricate a wide range of structures using site-specific chemical reactions.

It is now common for forecasts of future technical abilities to include the ability to fabricate molecular devices and molecular machines with atomic precision. While there continues to be debate about the exact time frame, it is becoming increasingly accepted that we will, eventually, develop the ability to economically fabricate a truly wide range of structures with atomic precision.

This will be of major economic value. Most obviously a molecular manufacturing capability will be a prerequisite to the construction of molecular logic devices. The continuation of present trends in computer hardware depends on the ability to fabricate ever smaller and ever more precise logic devices at ever decreasing costs.

The limit of this trend is the ability to fabricate molecular logic devices and to connect them in complex patterns at the molecular level. The manufacturing technology needed will, almost of necessity, be able to economically manufacture large structures (computers) with atomic precision (molecular logic elements). This capability will also permit the economical manufacture of materials with properties that border on the limits imposed by natural law.

The strength of materials, in particular, will approach or even exceed that of diamond. Given the broad range of manufactured products that devote substantial mass to load-bearing members, such a development by itself will have a significant impact. A broad range of other manufactured products will also benefit from a manufacturing process that offers atomic precision at low cost.

Given the promise of such remarkably high payoffs it is natural to ask exactly what such systems will look like, exactly how they will work, and exactly how we will go about building them. One might also enquire as to the reasons for confidence that such an enterprise is feasible, and why one should further expect that our current understanding of chemistry and physics (embodied in a number of computational chemistry packages) should be sufficient to explain the operating principles of such systems.

It is here that the value of computational nanotechnology can be most clearly seen. Molecular machine proposals, provided that they are specified in atomic detail (and are of a size that can be dealt with by current software and hardware), can be modeled using the tools of computational chemistry.

There are two modeling techniques of particular utility. The first is molecular mechanics, which utilizes empirical force fields to model the forces acting between nuclei[9, 10, 15, 16, 17]. The second is higher order ab initio calculations, which will be discussed in a few paragraphs.

A few links to internet computational chemistry resources are provided here to provide those new to the area with some feel for what's available.

MOLECULAR MECHANICS

Molecular mechanics allows computational modeling of the positions and trajectories of the nuclei of individual atoms without an undue computational load. Current packages available on personal computers can readily do energy minimizations on systems with thousands of atoms, while supercomputers can handle systems with hundreds of thousands of atoms or more.

More complex analyses, particularly analyses that involve searching through large configuration spaces, can limit the size of system that can be effectively handled. As will be discussed later the need to search through large configuration spaces (as when determining the native folded structure of an arbitrary protein) can be avoided by the use of relatively rigid structures (which differ from relatively floppy proteins and have few possible configurations).

The modeling of machine components in vacuum reduces the need to model solvation effects, which can also involve significant computational effort. In molecular mechanics the individual nuclei are usually treated as point masses. While quantum mechanics dictates that there must be a certain degree of positional uncertainty associated with each nucleus,

this positional uncertainty is normally significantly smaller than the typical internuclear distance. Bowen and Allinger[16] provide a good recent overview of the subject.

While the nuclei can reasonably be approximated as point masses, the electron cloud must be dealt with in quantum mechanical terms. However, if we are content to know only the positions of the nuclei and are willing to forego a detailed understanding of the electronic structure, then we can effectively eliminate the quantum mechanics.

For example, the H_2 molecule involves two nuclei. While it would be possible to solve Schrodinger's equation to determine the wave function for the electrons, if we are content simply to know the potential energy contributed by the electrons (and do not enquire about the electron distribution) then we need only know the electronic energy as a function of the distance between the nuclei.

That is, in many systems the only significant impact that the electrons have on nuclear position is to make a contribution to the potential energy E of the system. In the case of H_2, E is a simple function of the internuclear distance r. The function E(r) summarizes and replaces the more complex and more difficult to determine wave function for the electrons, as well as taking into account the inter-nuclear repulsion and the interactions between the electrons and the nuclei.

The two hydrogen nuclei will adopt a position that minimizes E(r). As r becomes larger, the potential energy of the system increases and the nuclei experience a restoring force that returns them to their original distance. Similarly, as r becomes smaller and the two nuclei are pushed closer

together, we also find that a restoring force pushes them farther apart, again restoring them to an equilibrium distance.

More generally, if we know the positions r_1, r_2,.... r_N of N nuclei, then E(r_1, r_2,.... r_N) gives the potential energy of the system. Knowing the potential energy as a function of the nuclear positions, we can readily determine the forces acting on the individual nuclei and therefore can compute the evolution of their position over time.

The function E is a newtonian potential energy function (not quantum mechanical), despite the fact that the particular value of E at a particular point could be computed from Schrodinger's equation. That is, the potential energy E is a newtonian concept, but the particular values of E at particular points are determined by Schrodinger's equation.

Often called the Born-Oppenheimer approximation[13,20], this approach allows a great conceptual and practical simplification in the modeling of molecular systems. While it would in principle be possible to determine E by solving Schrodinger's equation, in practice it is usual to use available experimental data and to infer the nature and shape of E by interpolation.

This approach, in which empirically derived potential energy functions are created by interpolation from experimental data, has spawned a wide range of potential energy functions, many of which are sold commercially. Because the gradient of the potential energy function E defines a conservative force field F, molecular mechanics methods are also called "force field" methods.

While it is common to refer to "empirical force field" methods, the more recent use of ab initio methods to provide data points to aid in the design of the force fields[17] makes this term somewhat inaccurate, though still widely used.

The utility of molecular mechanics depends crucially on the development of accurate force fields. Good quality force fields have been developed for a fairly broad range of compounds including many compounds of interest in biochemistry[9, 10, 15, 16, 17].

While we will not attempt to survey the wide range of force fields that are available, one particular subset of compounds for which good quality force fields are available involve H, C, N, O, F, Si, P, S, Cl (and perhaps a few others) when they are restricted to form chemically uncomplicated structures (e.g., bond strain is not too great, dangling bonds are few or absent, etc).

Many atomically precise structures which should be useful in nanotechnology fall in this class and can be modeled with an accuracy adequate to determine the behaviour of molecular machines.

A potential energy function of particular utility in modeling diamondoid structures was described by Brenner (*Empirical potential for hydrocarbons for use in simulating the chemical vapour deposition of diamond films*, Donald W.

Brenner, Phys Rev B, Vol. 42, No. 15, November 15 1990, pages 9458-9471).

Brenner's potential energy function, though limited to the elements carbon and hydrogen, has the great advantage that it will handle transition states, unstable structures, and the like.

Thus, given essentially arbitrary coordinates of the carbon and hydrogen atoms in a system, Brenner's potential will return the energy of the system. This permits molecular dynamical modeling of arbitrary hydrocarbon systems, including systems which use synthetic reactions involved in the synthesis of diamond.

A particular reaction of interest is the selective abstraction of a chosen hydrogen atom from a diamond surface. See *Surface patterning by atomically-controlled chemical forces: molecular dynamics simulations* by Sinnott et al., Surface Science 316 (1994), L1055-L1060; see also the work of Robertson et al. for an illustration of the use of Brenner's potential in modeling the behaviour of graphitic gears, including failure modes).

Brenner has commented on the utility of this potential for modeling proposed molecular machine systems. Of course, the "accuracy" of the force fields depends on the application. A force field which was accurate to (say) 10 kcal/ mole would be unable to correctly predict many properties of interest in biochemistry.

For example, such a force field would lead to serious errors in predicting the correct three-dimensional structure of a protein. Given the linear sequence of amino acids in a protein, the protein folding problem is to determine how it will fold in three dimensions when put in solution.

Often, the correct configuration will have an energy that differs from other (incorrect) configurations by a relatively modest amount, and so a force field of high accuracy is required. Consider, however, the bearing illustrated in figures 1 and 2.

Unlike the protein folding problem, where an astronomical range of configurations of similar energy are feasible, the bearing basically has only one configuration: a bearing. While the protein has many unconstrained torsions, there are no unconstrained torsions in the bearing.

To significantly change any torsion angle in the bearing would involve ripping apart bonds. Thus, the same force field which is of marginal utility in dealing with strands of floppy protein is quite adequate for solid blocks of stiff diamondoid material.

Not only will small errors in the force field still result in an accurate prediction of the global minima, (which in this case will be a single large basin in the potential energy surface) but also the range of possible structures is so sharply limited that little or no computational effort need be spent comparing the energies of different configurations. Long computational runs to evaluate the statistical properties of ensembles of configurations are thus eliminated.

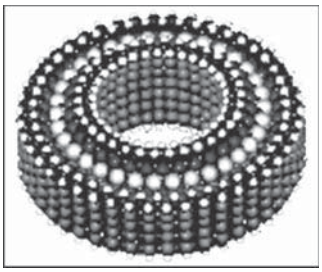


Fig. A Molecular Bearing.

This style of design has been called, only half in jest, "molecular bridge building" because bridges are also designed with large safety margins. This observation, that the same force field that is inadequate for one class of structures is quite adequate for the design and modeling of a different class of structures, leads to a more general principle.

Computational experiments generally provide an answer with some error distribution. If the errors produced by the model are of a similar size to the errors that would result in incorrect device function, then the model is unreliable.

On the other hand, if the errors in the model are small compared with the errors that will produce incorrect device function, then the results of the model are likely to be reliable. If a device design falls in the former category, i.e., small errors in the model will produce conflicting forecasts about device function, then the conservative course of action is to reject the proposed design and keep looking.

That is, *we can deliberately design structures which are indeed adequately handled by our computational tools*. As illustrated by the bearing, there are a wide range of basically mechanical structures whose stability is sufficiently clear and whose interactions with other structures is sufficiently simple that their behaviour can be adequately modeled with currently available force fields.

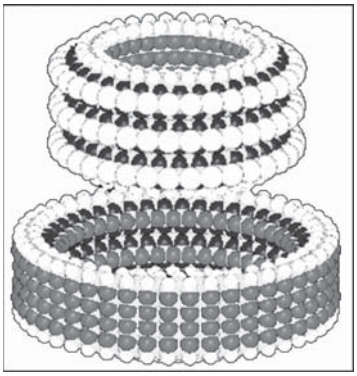


Fig. The Bearing Taken Apart.

An even stronger (although somewhat more subtle) statement is possible. The bearing illustrated in figures 1 and 2 is simply a single bearing from a very large class of bearings. The strain in the axle and the sleeve is proportional

to the diameter of the bearing. By increasing the diameter, we can reduce the strain. Thus, we can design bearings in this broad class in which the strain can be reduced to whatever level we desire. Because we are dealing with what amounts to a strained block of diamond, the fact that we can reduce strain arbitrarily means that we can design a bearing whose local structure bears as close a resemblance to an unstrained block of diamond as might be wished.

We can therefore be very confident indeed that some member of this class will perform the desired function (that of a bearing) and will work in accordance with our expectations.

Given that we have developed software tools that are capable of creating and modeling most of the members of a broad class of devices, then we can investigate many individual class members.

This investigation then lets us make rather confident statements about the functionality that members of this class can provide, even if there might be residual doubts about individual class members.

A moments reflection will show that the class of objects which are chemically reasonably inert, relatively stiff (no free torsions), and which interact via simple repulsive forces that occur on contact; can describe a truly vast class of machines. Indeed, it is possible to design computers, robotic arms and a wide range of other devices using molecular parts drawn from this class.

AB INITIO METHODS

While empirical force fields are sufficiently accurate to model the behaviour of chemically stable stiff structures

interacting with other chemically stable stiff structures, they do not (at present) provide sufficient accuracy to deal with chemical transitions. Thus, if we wish to model the manufacture of a molecular part (such as the axle or sleeve of the bearing of figures 1 and 2) then we must use higher order ab initio techniques (another paper in this issue illustrates what is meant by this in detail[14]).

These techniques impose severe constraints on the number of atoms that can be modeled (perhaps one or two dozen heavy atoms, depending on the hardware, software, and specific type of modeling being attempted), but can provide an accuracy sufficient to analyse the chemical reactions that must necessarily take place during the synthesis of large, atomically precise structures.

In [14], an analysis of the abstraction of a hydrogen atom from various structures, including isobutane (which serves as a model of the diamond (111) surface), has been carried out to illustrate the kind of reactions that are of interest. More generally, higher order ab initio techniques are sufficient to analyse the addition or removal of a small number of atoms from a specific site on a work piece.

Synthesis of a large object would then consist of repeated site specific applications of a small number of basic operations, where each basic operation changed the chemical structure of only a small number of atoms at a time.

Provided that we reject reaction mechanisms where the result predicted depends on errors that are smaller than can reasonably be modeled, the analysis of these basic operations can be satisfactorily carried out with current methods and hardware.

SUFFICIENCY OF CURRENT MODELING METHODS

In summary, it is quite possible to adequately model the behaviour of molecular machines that satisfy two constraints: (1) they are built from parts that are sufficiently stable that small errors in the empirical force fields do not raise significant questions about the shape or stability of the parts, and (2) the synthesis of the parts is done by using positionally controlled reactions, where the actual chemical reactions involve a relatively small number of atoms whose behaviour can be adequately modeled with higher order ab initio methods.

Clearly, not all molecular machines satisfy these constraints. Is the range of molecular machines which do satisfy these constraints sufficiently large to justify the effort of designing and modeling them? And in particular, can we satisfactorily model Drexler's assembler within these constraints?

The fundamental purpose of an assembler is to position atoms. To this end, it is imperative that we have models that let us determine atomic positions, and this is precisely what molecular mechanics provides. Robotic arms or other positioning devices are basically mechanical in nature, and will allow us to position molecular parts during the assembly process.

Molecular mechanics provides us with an excellent tool for modeling the behaviour of such devices. The second fundamental requirement is the ability to make and break bonds at specific sites. While molecular mechanics provides an excellent tool for telling us where the tip of the assembler

arm is located, current force fields are not adequate to model the specific chemical reactions that must then take place at the tip/work-piece interface involved in building an atomically precise part (though this statement must be modified in light of the work done with Brenner's potential, discussed above). For this, higher order ab initio calculations are sufficient.

The glaring omission in this discussion is the modeling of the kind of electronic behaviour that occurs in switching devices. Clearly, it is possible to model electronic behaviour with some degree of accuracy, and equally clearly molecular machines that are basically electronic in nature will be extremely useful.

It will therefore be desirable to extend the range of computational models discussed here to include such devices. For the moment, however, the relatively modest inclusion of electrostatic motors as a power source is probably sufficient to provide us with adequate "design space" to design and model an assembler.

While it might at first glance appear that electronics will be required for the computational element in the assembler, in fact molecular mechanical logic elements will be sufficient. Babbage's proposal from the 1800's clearly demonstrates the feasibility of mechanical computation (it is interesting to note that a working model of Babbage's difference engine has been built and is on display at the British Museum of Science.

They were careful to use parts machined no more accurately than the parts available to Babbage, in order to demonstrate that his ideas could have been implemented in the 1800's). Drexler's analysis of a specific molecular

mechanical logic element and further analysis of system design issues also make it clear that molecular mechanical computation is sufficient for the molecular computer required in an assembler.

Molecular electronic proposals for computation have not been worked out as clearly at the system level as the molecular mechanical concepts . Thus, at the moment, molecular mechanical proposals are better understood, at least in this particular context.

This situation is likely to change, and when it does the electronic designs could be incorporated (where appropriate) into the design and modeling of an assembler. However, it is not entirely obvious that electronic designs will prove superior to molecular mechanical designs, particularly when device parameters such as size and energy dissipation are considered.

While it seems virtually certain that electronic devices will prove faster than molecular mechanical devices, it is less obvious that electronic devices must necessarily be either smaller or dissipate less energy (though confer more recent work on reversible logic).

Indeed, given the difficulty of localizing individual electrons, it seems possible that electronic devices will prove to be inherently larger than molecular mechanical devices. This might provide a long term role for molecular mechanical devices as high density memory elements, or as logic elements when space (or atom count) is particularly constrained.

Whether the assembler is designed and built with an electronic or mechanical computer is less significant than designing and building it. By way of example, a stored

programme computer can be built in many different ways. Vacuum tubes, transistors, moving mechanical parts, fluidics, and other methods are all entirely feasible.

Delaying the development of the Eniac because transistors are better than vacuum tubes would have been a most unwise course of action. Similarly, as we consider the design, modeling, and construction of an assembler, we should not hesitate to use simpler methods that we can understand in favour of better methods that are not yet fully in hand.

The methods of computational chemistry available today allow us to model a wide range of molecular machines with an accuracy sufficient in many cases to determine how well they will work. We can deliberately confine ourselves to the subset of devices where our modeling tools are good enough to give us confidence that the proposals should work.

This range includes bearings, computers, robotic arms, site-specific chemical reactions utilized in the synthesis of complex structures, and more complex systems built up from these and related components. Drexler's assembler and related devices can be modeled using our current approaches and methods.

MOLECULAR COMPILERS

Computational nanotechnology includes not only the tools and techniques required to model proposed molecular machines, it must also include the tools required to specify such machines. Molecular machine proposals that would require millions or even billions of atoms have been made.

The total atom count of an assember might be roughly a billion atoms. While commerically available molecular

modeling packages provide facilities to specify arbitrary structures, it is usually necessary to "point and click" for each atom involved.

This is obviously unattractive for a device as complex as an assembler with its roughly one billion atoms. It should be clear that molecular CAD tools will be required if we are to specify such complex structures in atomic detail with a reasonable amount of effort.

An essential development is that of "molecular compilers" which accept, as input, a high level description of an object and produce, as output, the atomic coordinates, atom types, and bonding structure of the object.

A simple molecular compiler has already been written at PARC, and was used to produce the bearing of figures 1 and 2. The specification of the axle was:

Scale	0.9
Tube	0 0. 75 0. 75 1. 75 0 0 0 17 – 17 0 5. 25 5. 25
Grid	0 0. 5 – 0. 5 0 1 1
Delete	1. 25 0 0
Grid	0 0. 25 0 0 0 0. 25
Change	O_3 to S_3 1. 5 0 0

The first line, "scale 0.9," is simply an instruction to shrink the size of the axle by 10% compared with the normal size. This allows the axle to be positioned inside the sleeve before joint minimization of the axle and sleeve is done (using MSI's PolyGraf, a commercially available molecular mechanics package that implements both MM2 and the Dreiding II force field[10]).

If the axle w were not done, then the atoms in the axle and the sleeve would be comingled, and minimization would

produce meaningless results. The second line begins with the "tube" specifier. This tells the programme to produce a tubular structure, rather than a "block" or a "ring." the "ring" specifier is used to produce toroidal structures, e.g., tubes that have been further bent into donuts.

The first triplet of numbers following "tube," "0 0.75 0.75," specifies the offset to be used for the crystal lattice. The second triplet, "1.75 0 0" specifies that the surface of the tube is (100), and that the thickness of the tube wall is 1. 75 lattice spacings. The third triplet, "0 17 – 17," specifies the direction and length of the circumference of the tube.

The direction [0 17 – 17] (or [0 1 – 1]) is at right angles to [1 0 0], the direction of the tube surface. The circumference of the tube is simply the length of the [0 17 – 17] vector. Finally, "0 5.25 5.25" gives the direction of the axis and the length of the tube. This does not fully specify the axle of the bearing, for the 100 surface has been cut circumferentially to produce grooves. The next command, "grid 0 0.5 – 0.5 0 1 1" specifies that a grid is to be laid down on the surface.

The grid is specified by two vectors, "0 0.5 – 0.5" and "0 1 1,"" which give the directions and lengths of the edges of the unit parallelogram from which the grid is composed. The next command, "delete 1.25 0 0"" specifies that the point at coordinates "1.25 0 0" is to be deleted, and further, that all points in the same location with respect to any unit parallelogram of the grid are also to be deleted. Thus, this "grid .... delete ...." cuts out the grooves that are visible on the outer surface of the axle.

Finally, the commands "grid 0 0.25 0 0 0 0.25," and "change O_3 to S_3 1.5 0 0" are used to lay down a new, very fine grid

that causes all oxygen atoms on the surface of the axle to be changed to sulfur. A similar set of commands was used to specify the sleeve of the bearing.

It is easy to specify bearings with different surfaces, surface orientations, circumferences, lengths, etc. To select a (111), (110), (312), or any other surface, it is sufficient to change the vector that specifies the surface, and the vectors that specify the tangent to the surface and the axis of the tube (which must be at right angles to the surface vector).

Thus, complex structures involving many atoms can be generated quickly and easily with a few lines of input specification. The C source code is available at URL ftp://ftp.parc.xerox.com/pub/nano/tube.c.

The programme will generate output in either PolyGraf format or in Brookhaven format, so the output should be readable by most computational chemistry packages.

The software required to design and model complex molecular machines is either already available, or can be readily developed over the next few years.

Many of the modeling issues can be dealt with by existing commercially available computational chemistry packages. The molecular compilers and other molecular CAD tools needed for this work can be implemented using generally understood techniques and methods from computer science.

Using this approach, it will be possible to substantially reduce the development time for complex molecular machines, including Drexler's assembler. This approach is similar in spirit to the computer aided design and modeling used to speed the development of many products today. The author was part of a general purpose computer start up which

successfully designed and built a new computer from scratch. This included the hardware, software, compilers, operating system, etc.

During this process, extensive use was made of computational models to verify each level of the design. The operating system was written in Pascal, and checked out on another computer. The compilers were written in Pascal, and also checked out on another computer.

The code produced by the compilers was checked out on an instruction set simulator. The microcode was checked out on a micro-instruction simulator. The logic design was checked out with logic level simulation tools, and circuit simulation packages were used to verify the detailed electronic design. All this work was done at the same time.

The software was written and debugged even though the machine on which it would eventually be executed didn't exist. The microcode was written and debugged before the hardware was available. When the hardware was finally made available, system integration went relatively rapidly.

Imagine, for a moment, how long it would have taken to develop this system had we carried out the development in the obvious sequential fashion. First, we would have implemented the hardware and only then begun work on the microengine. Later, with the hardware and microengine fully checked out and working, we could have developed and debugged the microcode. With this firmly in hand, we could then have written the compilers and verified the code they produced. Finally, we could have implemented and checked out the operating system. Needless to say, such a strategy would have been very slow and tedious.

Doing things in the simple and most obvious way often takes a lot longer than is needed. If we were to approach the design and construction of an assembler using the simple serial method, it would take a great deal longer than if we systematically attacked and simultanesouly solved the problems that arise at all levels of the design at one and the same time. That is, by using methods similar to those used to design a modern computer, including intensive computational modeling of individual components and sub-systems, we can greatly shorten the time required to design and build complex molecular machines.

This can be further illustrated by considering the traditional manner of growth of our synthetic capability over time (figure 3). Today, we find we are able to synthesize a certain range of compounds and structures.

As time goes by, we will be able to synthesize an ever larger range of structures. This growth in our ability will proceed on a broad front, and reflects the efforts of a broad range of researchers who are each pursuing individual goals without concern about the larger picture into which they might fit. As illustrated in figure 4, given sufficient time we will eventually be able to synthesize complex molecular machines simply because we will eventually develop the ability to synthesize just about anything.

However, if we wish to develop a particular kind of device, e.g., an assembler, then we can speed the process up (as illustrated in figure 5) by conducting computational experiments designed to clarify the objective and to specify more precisely the path from our current range of synthetic capabilities to the objective.

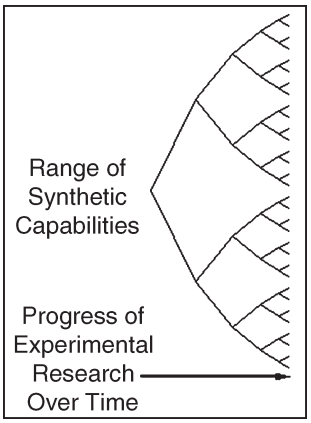


Fig. Experimental Advances in our Synthetic Capabilities

Such computational experiments are inexpensive, can provide very detailed information, are possible for any structure (whether we can or cannot synthesize it) and will, in general, reduce the "time to market" for the selected product.

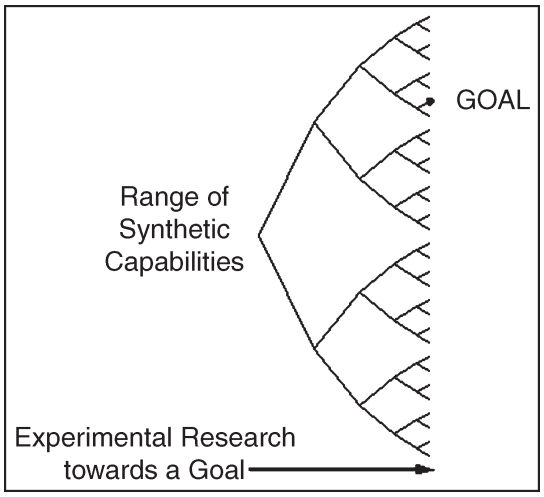


Fig. Experimental Progress Towards a Goal

Computational experiments let us examine structures quickly and easily, rejecting those which have obvious defects (a precursor to the bearing shown in figure 1, for example, was too strained.

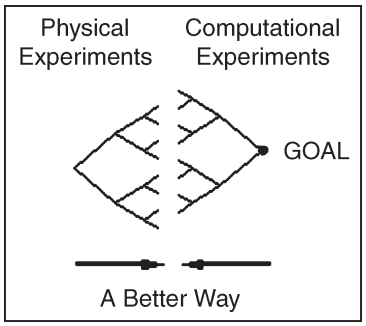


Fig. A more Efficient Method of Achieving a Goal

By modifying the design and again minimizing the structure, we found a design with an acceptable strain). This kind of examination of the "design space" is impossible with physical experiments today, but is easily done with computational experiments.

Computational experiments also provide more information. For example, molecular dynamics can literally provide information about the position of each individual atom over time, information which would usually be inaccessible in a physical experiment.

Of course, the major advantage of computational experiments over physical experiments in the current context is the simple fact that physical experiments aren't possible for molecular machines that we can't make with today's technology.

By using computational models derived from the wealth of experimental data that is available today, we can (within certain accuracy bounds) describe the behaviour of proposed systems that we plan to build in the future.

If we deliberately design systems that are sufficiently robust that we are confident they will work regardless of the small errors that must be incurred in the modeling process, we

can design systems today that we will not be able to build for some years, and yet still have reasonable confidence that they will work.

By fully utilizing the experience that has been developed in the rapid design and development of complex systems we can dramatically reduce the development time for molecular manufacturing systems. It is possible to debate how long it will be before we achieve a robust molecular manufacturing capability.

However, it is very clear that we'll get there sooner if we develop and make intelligent use of molecular design tools and computational models. These will let us design and check the blueprints for the new molecular manufacturing technologies that we now see on the horizon, and will let us chart a more rapid and more certain path to their development.

4

Digital Circuits

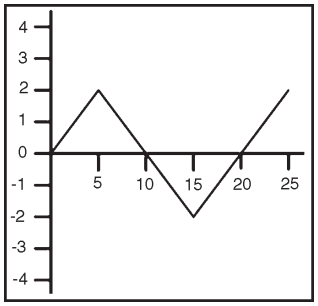
The quantities that are to be measured, monitored, recorded, processed and controlled are analog and digital, depending on the type of system used.

It is important when dealing with various quantities that we be able to represent their values efficiently and accurately. There are basically two ways of representing the numerical value of quantities: analog and digital.

ANALOG REPRESENTATION

Systems which are capable of processing a continuous range of values varying with respect to time are called analog systems. In analog representation a quantity is represented by a voltage, current, or meter movement that is proportional to the value of that quantity. Analog quantities such as those cited above have an important characteristic: they can vary over a continuous range of values.

Diagram of Analog Voltage vs Time



DIGITAL REPRESENTATION

Systems which process discrete values are called digital systems. In digital representation the quantities are represented not by proportional quantities but by symbols called digits. As an example, consider the digital watch, which provides the time of the day in the form of decimal digits representing hours and minutes (and sometimes seconds). As we know, time of day changes continuously, but the digital watch reading does not change continuously; rather, it changes in steps of one per minute (or per second).

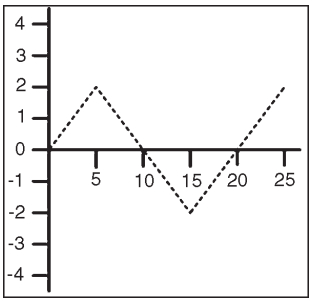
In other words, time of day digital representation changes in discrete steps, as compared to the representation of time provided by an analog watch, where the dial reading changes continuously.

Below is a diagram of digital voltage vs time: here input voltage changes from +4 Volts to -4 Volts; it can be converted to digital form by Analog to Digital converters (ADC). An ADC converts continuous signals into samples per second. Well, this is an entirely different theory.

Digital Voltage vs Time

The major difference between analog and digital quantities, then, can be stated simply as follows:

- Analog = continuous
- Digital = discrete (step by step)



ADVANTAGES OF DIGITAL TECHNIQUES

- Easier to design. Exact values of voltage or current are not important, only the range (HIGH or LOW) in which they fall.
- Information storage is easy.
- Accuracy and precision are greater.
- Operations can be programmed. Analog systems can also be programmed, but the available operations variety and complexity is severely limited.
- Digital circuits are less affected by noise, as long as the noise is not large enough to prevent us from distinguishing HIGH from LOW (we discuss this in detail in an advanced digital tutorial section).
- More digital circuitry can be fabricated on IC chips.

LIMITATIONS OF DIGITAL TECHNIQUES

- Most physical quantities in real world are analog in nature, and these quantities are often the inputs and outputs that are being monitored, operated on, and controlled by a system. Thus conversion to digital format and re-conversion to analog format is needed.

NUMBER SYSTEM

Many number systems are in use in digital technology. The most common are the decimal, binary, octal, and hexadecimal systems. The decimal system is clearly the most familiar to us because it is a tool that we use every day. Examining some of its characteristics will help us to better understand the other systems. In the next few pages we shall introduce four numerical representation systems that are used in the digital system.

There are other systems, which we will look at briefly:

- Decimal
- Binary
- Octal
- Hexadecimal

DECIMAL SYSTEM

The decimal system is composed of 10 numerals or symbols. These 10 symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Using these symbols as digits of a number, we can express any quantity. The decimal system is also called the base-10 system because it has 10 digits.

10^3	10^2	10^1	10^0		10^{-1}	10^{-2}	10^{-3}
=1000	=100	=10	=1	.	=0.1	=0.01	=0.001
Most Significant Digit			Decimal point			Digit	Least Significant Digit

Even though the decimal system has only 10 symbols, any number of any magnitude can be expressed by using our system of positional weighting.

Decimal Examples

- 3.14_{10}

- 52_{10}
- 1024_{10}
- 64000_{10}

BINARY SYSTEM

In the binary system, there are only two symbols or possible digit values, 0 and 1.

This base-2 system can be used to represent any quantity that can be represented in decimal or other base system.

2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
=8 Most Significant Digit	=4	=2	=1	. Binary point	=0.5	=0.25	=0.125 Least Significant Digit

BINARY COUNTING

The Binary counting sequence is shown in the table:

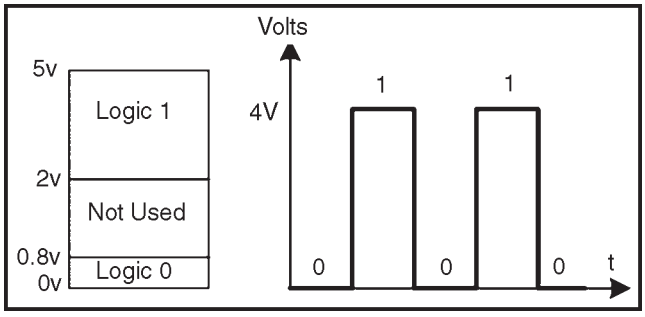
2^3	2^2	2^1	2^0	Decimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Representing Binary Quantities

In digital systems the information that is being processed is usually presented in binary form. Binary quantities can be represented by any device that has only two operating states or possible conditions. *E.g.*. a switch is only open or closed. We arbitrarily (as we define them) let an open switch represent binary 0 and a closed switch represent binary 1. Thus we can represent any binary number by using series of switches.

Typical Voltage Assignment

- *Binary* 1: Any voltage between 2V to 5V
- *Binary* 0: Any voltage between 0V to 0.8V
- *Not used*: Voltage between 0.8V to 2V in 5 Volt CMOS and TTL Logic, this may cause error in a digital circuit. Today's digital circuits works at 1.8 volts, so this statement may not hold true for all logic circuits.



We can see another significant difference between digital and analog systems. In digital systems, the exact voltage value is not important; eg, a voltage of 3.6V means the same as a voltage of 4.3V. In analog systems, the exact voltage value is important. The binary number system is the most important one in digital systems, but several others are also important. The decimal system is important because it is

universally used to represent quantities outside a digital system. This means that there will be situations where decimal values have to be converted to binary values before they are entered into the digital system. In additional to binary and decimal, two other number systems find wide-spread applications in digital systems. The octal (base-8) and hexadecimal (base-16) number systems are both used for the same purpose- to provide an efficient means for representing large binary system.

OCTAL SYSTEM

The octal number system has a base of eight, meaning that it has eight possible digits: 0,1,2,3,4,5,6,7.

8^3	8^2	8^1	8^0		8^{-1}	8^{-2}	8^{-3}
=512	=64	=8	=1	.	=1/8	=1/64	=1/512
Most Significant				Octal		Least	Digit
	point			Significant			
	Digit						

Octal to Decimal Conversion

- $237_8 = 2 \times (8^2) + 3 \times (8^1) + 7 \times (8^0) = 159_{10}$
- $24.6_8 = 2 \times (8^1) + 4 \times (8^0) + 6 \times (8^{-1}) = 20.75_{10}$
- $11.1_8 = 1 \times (8^1) + 1 \times (8^0) + 1 \times (8^{-1}) = 9.125_{10}$
- $12.3_8 = 1 \times (8^1) + 2 \times (8^0) + 3 \times (8^{-1}) = 10.375_{10}$

HEXADECIMAL SYSTEM

The hexadecimal system uses base 16. Thus, it has 16 possible digit symbols. It uses the digits 0 through 9 plus the letters A, B, C, D, E, and F as the 16 digit symbols.

16^3	16^2	16^1	16^0		16^{-1}	16^{-2}	16^{-3}
=4096	=256	=16	=1	.	=1/16	=1/256	=1/4096

Most Significant Digit	Hexa Decimal point	Least Significant Digit

Hexadecimal to Decimal Conversion

- $24.6_{16} = 2 \times (16^1) + 4 \times (16^0) + 6 \times (16^{-1}) = 36.375_{10}$
- $11.1_{16} = 1 \times (16^1) + 1 \times (16^0) + 1 \times (16^{-1}) = 17.0625_{10}$
- $12.3_{16} = 1 \times (16^1) + 2 \times (16^0) + 3 \times (16^{-1}) = 18.1875_{10}$

BINARY ARITHMETIC

NUMBERS VERSUS NUMERATION

It is imperative to understand that the type of numeration system used to represent numbers has no impact upon the outcome of any arithmetical function (addition, subtraction, multiplication, division, roots, powers, or logarithms). A number is a number is a number; one plus one will always equal two (so long as we're dealing with real numbers), no matter how you symbolize one, one, and two. A prime number in decimal form is still prime if it's shown in binary form, or octal, or hexadecimal? is still the ratio between the circumference and diameter of a circle, no matter what symbol(s) you use to denote its value. The essential functions and interrelations of mathematics are unaffected by the particular system of symbols we might choose to represent quantities. This distinction between numbers and systems of numeration is critical to understand.

The essential distinction between the two is much like that between an object and the spoken word(s) we associate with it. A house is still a house regardless of whether we call it by its English name house or its Spanish name casa. The first

is the actual thing, while the second is merely the symbol for the thing. That being said, performing a simple arithmetic operation such as addition (longhand) in binary form can be confusing to a person accustomed to working with decimal numeration only. In this lesson, we'll explore the techniques used to perform simple arithmetic functions on binary numbers, since these techniques will be employed in the design of electronic circuits to do the same. You might take longhand addition and subtraction for granted, having used a calculator for so long, but deep inside that calculator's circuitry all those operations are performed"longhand," using binary numeration. To understand how that's accomplished, we need to review to the basics of arithmetic.

BINARY ADDITION

Adding binary numbers is a very simple task, and very similar to the longhand addition of decimal numbers. As with decimal numbers, you start by adding the bits (digits) one column, or place weight, at a time, from right to left. Unlike decimal addition, there is little to memorize in the way of rules for the addition of binary bits:

```
0 + 0 = 0
1 + 0 = 1
0 + 1 = 1
1 + 1 = 10
1 + 1 + 1 = 11
```

Just as with decimal addition, when the sum in one column is a two-bit (two-digit) number, the least significant figure is written as part of the total sum and the most significant figure is “carried” to the next left column. Consider the following examples:

- 11 1← Carry bits → 11

- 1001101 1001001 1000111
- + 0010010 + 0011001 + 0010110
- ——————— ——————— ———————
- 1011111 1100010 1011101

The addition problem on the left did not require any bits to be carried, since the sum of bits in each column was either 1 or 0, not 10 or 11. In the other two problems, there definitely were bits to be carried, but the process of addition is still quite simple.

As we'll see, there are ways that electronic circuits can be built to perform this very task of addition, by representing each bit of each binary number as a voltage signal (either "high," for a 1; or "low" for a 0). This is the very foundation of all the arithmetic which modern digital computers perform.

NEGATIVE BINARY NUMBERS

With addition being easily accomplished, we can perform the operation of subtraction with the same technique simply by making one of the numbers negative. For example, the subtraction problem of 7–5 is essentially the same as the addition problem 7 + (–5).

Since we already know how to represent positive numbers in binary, all we need to know now is how to represent their negative counterparts and we'll be able to subtract. Usually we represent a negative decimal number by placing a minus sign directly to the left of the most significant digit, just as in the example above, with -5. However, the whole purpose of using binary notation is for constructing on/off circuits that can represent bit values in terms of voltage (2 alternative values: either "high" or "low"). In this context, we don't have

the luxury of a third symbol such as a "minus" sign, since these circuits can only be on or off (two possible states).

One solution is to reserve a bit (circuit) that does nothing but represent the mathematical sign:

- $101_2 = 5_{10}$ (positive)
- Extra bit, representing sign (0=positive, 1=negative)
- |
- $0101_2 = 5_{10}$ (positive)
- Extra bit, representing sign (0=positive, 1=negative)
- |
- $1101_2 = -5_{10}$ (negative)

As you can see, we have to be careful when we start using bits for any purpose other than standard place-weighted values. Otherwise, 1101_2 could be misinterpreted as the number thirteen when in fact we mean to represent negative five. To keep things straight here, we must first decide how many bits are going to be needed to represent the largest numbers we'll be dealing with, and then be sure not to exceed that bit field length in our arithmetic operations. For the above example, I've limited myself to the representation of numbers from negative seven (1111_2) to positive seven (0111_2), and no more, by making the fourth bit the "sign" bit. Only by first establishing these limits can I avoid confusion of a negative number with a larger, positive number.

Representing negative five as 1101_2 is an example of the *sign-magnitude* system of negative binary numeration. By using the leftmost bit as a sign indicator and not a place-weighted value, I am sacrificing the "pure" form of binary notation for something that gives me a practical advantage: the representation of negative numbers. The leftmost bit is

read as the sign, either positive or negative, and the remaining bits are interpreted according to the standard binary notation: left to right, place weights in multiples of two.

As simple as the sign-magnitude approach is, it is not very practical for arithmetic purposes. For instance, how do I add a negative five (1101_2) to any other number, using the standard technique for binary addition? I'd have to invent a new way of doing addition in order for it to work, and if I do that, I might as well just do the job with longhand subtraction; there's no arithmetical advantage to using negative numbers to perform subtraction through addition if we have to do it with sign-magnitude numeration, and that was our goal!

There's another method for representing negative numbers which works with our familiar technique of longhand addition, and also happens to make more sense from a place-weighted numeration point of view, called *complementation*. With this strategy, we assign the leftmost bit to serve a special purpose, just as we did with the sign-magnitude approach, defining our number limits just as before. However, this time, the leftmost bit is more than just a sign bit; rather, it possesses a negative place-weight value. For example, a value of negative five would be represented as such:

Extra bit, place weight = negative eight:

- |
- $1011_2 = 5_{10}$ (negative)
- $(1 \times -8_{10}) + (0 \times 4_{10}) + (1 \times 2_{10}) + (1 \times 1_{10}) = -5_{10}$

With the right three bits being able to represent a magnitude from zero through seven, and the leftmost bit representing either zero or negative eight, we can successfully

represent any integer number from negative seven (1001_2 = $-8_{10} + 7_{10} = -1_{10}$) to positive seven ($0111_2 = 0_{10} + 7_{10} = 7_{10}$).

Representing positive numbers in this scheme (with the fourth bit designated as the negative weight) is no different from that of ordinary binary notation.

However, representing negative numbers is not quite as straightforward:

Zero	0000		
Positive one	0001	Negative one	1111
Positive two	0010	Negative two	1110
Positive three	0011	Negative three	1101
Positive four	0100	Negative four	1100
Positive five	0101	Negative five	1011
Positive six	0110	Negative six	1010
Positive seven	0111	Negative seven	1001
		Negative eight	1000

Note that the negative binary numbers in the right column, being the sum of the right three bits' total plus the negative eight of the leftmost bit, don't "count" in the same progression as the positive binary numbers in the left column. Rather, the right three bits have to be set at the proper value to equal the desired (negative) total when summed with the negative eight place value of the leftmost bit.

Those right three bits are referred to as the *two's complement* of the corresponding positive number. Consider the following comparison:

Positive number	**Two's complement**
001	111
010	110
011	101
100	100
101	011

110	010
111	001

In this case, with the negative weight bit being the fourth bit (place value of negative eight), the two's complement for any positive number will be wthatever value is needed to add to negative eight to make that positive value's negative equivalent. Thankfully, there's an easy way to figure out the two's complement for any binary number: simply invert all the bits of that number, changing all 1's to 0's and visa-versa (to arrive at what is called the *one's complement*) and then add one! For example, to obtain the two's complement of five (101_2), we would first invert all the bits to obtain 010_2 (the "one's complement"), then add one to obtain 011_2, or -5_{10} in three-bit, two's complement form.

Interestingly enough, generating the two's complement of a binary number works the same if you manipulate *all* the bits, including the leftmost (sign) bit at the same time as the magnitude bits. Let's try this with the former example, converting a positive five to a negative five, but performing the complementation process on all four bits. We must be sure to include the 0 (positive) sign bit on the original number, five (0101_2). First, inverting all bits to obtain the one's complement: 1010_2. Then, adding one, we obtain the final answer: 1011_2, or -5_{10} expressed in four-bit, two's complement form.

It is critically important to remember that the place of the negative-weight bit must be already determined before any two's complement conversions can be done. If our binary numeration field were such that the eighth bit was designated as the negative-weight bit (10000000_2), we'd have to

determine the two's complement based on all seven of the other bits. Here, the two's complement of five (0000101_2) would be 1111011_2. A positive five in this system would be represented as 00000101_2, and a negative five as 11111011_2.

SUBTRACTION

We can subtract one binary number from another by using the standard techniques adapted for decimal numbers (subtraction of each bit pair, right to left, "borrowing" as needed from bits to the left). However, if we can leverage the already familiar (and easier) technique of binary addition to subtract, that would be better. As we just learned, we can represent negative binary numbers by using the "two's complement" method and a negative place-weight bit. Here, we'll use those negative binary numbers to subtract through addition. Here's a sample problem:

Subtraction: $7_{10}–5_{10}$ *Addition equivalent*: $7_{10} + (-5_{10})$

If all we need to do is represent seven and negative five in binary (two's complemented) form, all we need is three bits plus the negative-weight bit:

Positive seven = 0111_2
Negative five = 1011_2

Now, let's add them together:

```
 1111 <– Carry bits
 0111
+ 1011
 ————
 10010
 |
Discard extra bit.
Answer = 0010₂
```

Since we've already defined our number bit field as three bits plus the negative-weight bit, the fifth bit in the answer (1) will be discarded to give us a result of 0010_2, or positive

two, which is the correct answer. Another way to understand why we discard that extra bit is to remember that the leftmost bit of the lower number possesses a negative weight, in this case equal to negative eight.

When we add these two binary numbers together, what we're actually doing with the MSBs is subtracting the lower number's MSB from the upper number's MSB. In subtraction, one never "carries" a digit or bit on to the next left place-weight.

Let's try another example, this time with larger numbers. If we want to add -25_{10} to 18_{10}, we must first decide how large our binary bit field must be. To represent the largest (absolute value) number in our problem, which is twenty-five, we need at least five bits, plus a sixth bit for the negative-weight bit. Let's start by representing positive twenty-five, then finding the two's complement and putting it all together into one numeration:

$+25_{10} = 011001_2$ (showing all six bits)

One's complement of $11001_2 = 100110_2$

One's complement + 1 = two's complement = 100111_2

$-25_{10} = 100111_2$

Essentially, we're representing negative twenty-five by using the negative-weight (sixth) bit with a value of negative thirty-two, plus positive seven (binary 111_2).

Now, let's represent positive eighteen in binary form, showing all six bits:

$18_{10} = 010010_2$.

Now, let's add them together and see what we get:.

11 <— Carry bits

100111

+ 010010

———————

111001

Since there were no "extra" bits on the left, there are no bits to discard. The leftmost bit on the answer is a 1, which means that the answer is negative, in two's complement form, as it should be. Converting the answer to decimal form by summing all the bits times their respective weight values, we get:

$(1 \times -32_{10}) + (1 \times 16_{10}) + (1 \times 8_{10}) + (1 \times 1_{10}) = -7_{10}$

Indeed -7_{10} is the proper sum of -25_{10} and 18_{10}.

MINIMIZATION OF BOOLEAN FUNCTIONS

The most obvious way to simplify Boolean expressions is to manipulate them in the same way as normal algebraic expressions are manipulated. With regards to logic relations in digital forms, a set of rules for symbolic manipulation is needed in order to solve for the unknowns.

A set of rules formulated by the English mathematician George Boole describe certain propositions whose outcome would be either true or false. With regard to digital logic, these rules are used to describe circuits whose state can be either, 1 (true) or 0 (false). In order to fully understand this, the relation between the AND gate, OR gate and NOT gate operations should be appreciated. A number of rules can be derived from these relations as demonstrates.

Boolean Postulates:

- *P*1: X = 0 or X = 1
- *P*2: 0. 0 = 0
- *P*3: 1 + 1 = 1

- *P*4: 0 + 0 = 0
- *P*5: 1. 1 = 1
- *P*6: 1. 0 = 0. 1 = 0
- *P*7: 1 + 0 = 0 + 1 = 1

LAWS OF BOOLEAN ALGEBRA

Table shows the basic Boolean laws. Note that every law has two expressions, (a) and (b). This is known as *duality*. These are obtained by changing every AND(.) to OR(+), every OR(+) to AND(.) and all 1's to 0's and vice-versa.

It has become conventional to drop the . (AND symbol) *i.e.* A.B is written as AB.

Boolean Laws

*T*1: Commutative Law

(a) $A + B = B + A$

(b) $A\,B = B\,A$

*T*2: Associate Law

(a) $(A + B) + C = A + (B + C)$

(b) $(A\,B)\,C = A\,(B\,C)$

*T*3: Distributive Law

(a) $A\,(B + C) = A\,B + A\,C$

(b) $A + (B\,C) = (A + B)\,(A + C)$

*T*4: Identity Law

(a) $A + A = A$

(b) $A\,A = A$

*T*5:

(a) $A\bar{B} + AB = A$

(b) $(A + B)(A + \bar{B}) = A$

*T*6: Redundance Law

(a) $A + A\,B = A$

(b) $A\ (A + B) = A$

*T*7:

(a) $0 + A = A$

(b) $0\ A = 0$

*T*8:

(a) $1 + A = 1$

(b) $1\ A = A$

*T*9:

(a) $\bar{A} + A = 1$

(b) $\bar{A}\,A = 0$

*T*10:

(a) $A + \bar{A}B = A + B$

(b) $A(\bar{A} + B) = AB$

*T*11: De Morgan's Theorem

(a) $(\overline{A + B}) = \bar{A}\,\bar{B}$

(b) $(\overline{AB}) = \bar{A} + \bar{B}$

Examples

*Prove T*10: (a) $A + \bar{A}B = A + B$

- *Algebraically:*

$$\begin{aligned} A + \bar{A}B &= A1 + \bar{A}B && T7(a) \\ &= A(1 + B) + \bar{A}B && T7(c) \\ &= A + AB + \bar{A}B && T3(a) \\ &= A + B\left(A + \bar{A}\right) && T3(a) \\ &= A + B && T(8) \end{aligned}$$

- *Using the truth table*:

A	**B**	**A+B**	$\bar{A}B$	$A + \bar{A}B$
0	0	0	0	0
0	1	1	1	1
1	0	1	0	1
1	1	1	0	1

Using the laws given above, complicated expressions can be simplified.

$$Z = (A + \bar{B} + \bar{C})(A + \bar{B}C)$$
$$Z = AA + A\bar{B}C + A\bar{B} + \bar{B}\bar{B}C + A\bar{C} + \bar{B}C\bar{C}$$
$$Z = A(1 + \bar{B}C + \bar{B} + \bar{C}) + \bar{B}C + \bar{B}C\bar{C} \text{ From laws TBb and Tb}$$
$$Z = A + \bar{B}C \text{ From laws T8b and T9b}$$

KARNAUGH MAPS

So far we can see that applying Boolean algebra can be awkward in order to simplify expressions. Apart from being laborious (and requiring the remembering all the laws) the method can lead to solutions which, though they appear minimal, are not. The Karnaugh map provides a simple and straight-forward method of minimising boolean expressions. With the Karnaugh map Boolean expressions having up to four and even six variables can be simplified. .

The Karnaugh map (K map) provides a systematic method for simplifying a Boolean expression or a truth table function. When used properly, the K map will produce the simplest SOP or POS expression possible. Familiarity with the law and rules of Boolean algebra is not required. Instead, simplification is done graphically using the K mapping technique. The K map is a table consisting of $N = 2^n$ cells, where n is the number of input variables. For a SOP expression each cell represents one particular combination of the variables in product form. The table format is such that there is a single variable change between any adjacent cells.

This is the characteristic the will determine adjacency. To illustrate the above points let us consider an example where

n = 2 and N = 4. Assuming the input variable are A and B then the K map illustrating the four (4) possible variable combinations $\overline{A}\overline{B}, \overline{A}B, A\overline{B}$ and AB is show below in Table.

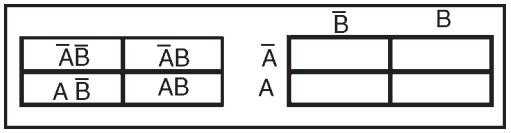


Fig. Two Variable K map Format

If we extend our example to consider the case where n = 3 and N = 8 then assuming that our input variables are A, B and C the associated K Map is shown in Table.

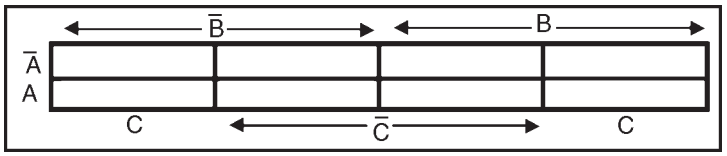


Fig. Three Variable K-Map

For n>5 the K map technique becomes impractical unless implemented on computer.

Simplifying Using the K-Map

To simplify a SOP for of a Boolean expression using a K map, first identify all the input combinations that produce an output of logic level 1 and place them in their appropriate K map cell. Consequently, all other cells must contain zero (0). Second, group the adjacent cells that contain 1 in a manner that maximises the size of the groups but also minimises the total number of groups.

All 1's in the output must be included in a group even if the group is only one cell. Third, as each SOP term represents an AND expression, each (AND) grouping is written with only the input variables that are common to the group. Finally, the simplified expression is formed by ORing each of the

(AND) groups. To illustrate lets consider the function, $X = \bar{A}BC + AB\bar{C} + ABC$ whose truth table and K map are illustrated in Tables.

Table. Truth Table of $X = \bar{A}BC + AB\bar{C} + ABC$

Input			Output
A	**B**	**C**	**X**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

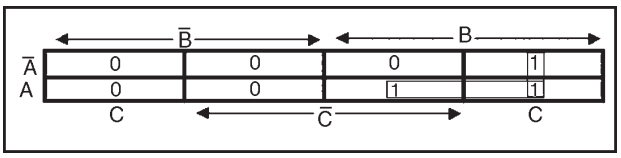


Fig. K-Map for $X = \bar{A}BC + AB\bar{C} + ABC$

By examination of the K map, the simplified expression for $X = \bar{A}BC + AB\bar{C} + ABC$ is $X = BC + AB$

Let us now use the K map technique to simplify the SOP Boolean expression $X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC$. The associated truth table and K map are presented in Table 1-19 and Table, respectively.

Table. Truth Table of,

$$X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC$$

Input			Output
A	**B**	**C**	**X**
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

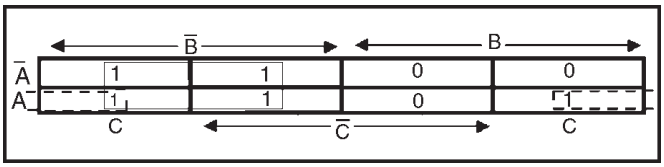


Fig. K-Map for $X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC$

By examination of the K map, the simplified expression for $X = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC$ is $X = \bar{B} + AC$.

Suppressed Variables

Sometimes a SOP expression does not have a complete set of variables in each of its terms. For example, the expression $X = A\bar{B}\bar{C} + B\bar{C} + ABC$ contains a term that does not include the variable A or its complement. A missing variable in a SOP expression is called a *suppressed* variable.

Before plotting an expression with suppressed variables on a K-MAP we must first expand out all of the shortened terms to include the missing variables and its complement.

To illustrate let us consider use our example above:

$$B\bar{C} = B\bar{C}(1)$$
$$B\bar{C} = B\bar{C}\left(A + \bar{A}\right)$$
$$B\bar{C} = AB\bar{C} + \bar{A}B\bar{C}$$

Then our expanded expression becomes,

$$X = A\bar{B}\bar{C} + AB\bar{C} + \bar{A}B\bar{C} + ABC$$

and we may now use the truth table and the associated K map, as illu,strated in Tables to simplify.

Table. Truth Table of $X = A\bar{B}\bar{C} + AB\bar{C} + \bar{A}B\bar{C} + ABC$

	Input		**Output**
A	**B**	**C**	**X**
0	0	0	0
0	0	1	0
0	1	0	1

0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

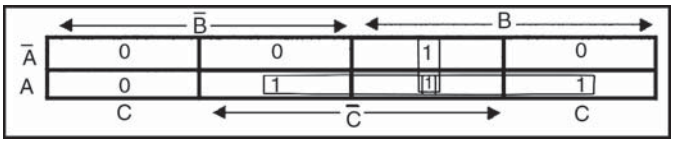


Fig. K-Map for $X = A\bar{B}\bar{C} + AB\bar{C} + \bar{A}B\bar{C} + ABC$

From the three groupings shown in the K map we form the expression $X = B\bar{C} + A\bar{C} + AB$.

DON'T CARE STATES

Truth table specifications for a logic function may not to include all possible combinations of the input binary digits for the input variables, yet they may still be complete specifications of the logic function for the prescribed application. In these situations certain input combinations will not occur due to the nature of the application. When the input combinations are irrelevant or cannot occur, the output states are in the Truth table and the K map are filled with an X and are referred to as don't care states.

When simplifying K maps with don't care states, the contents of the undefined cells (1 or 0) are chosen according to preference. The aim is to enlarge group sizes thereby eliminating as many input variables from the simplified expression as possible. Only those X's that assist in simplifying the function should be included in the groupings. No additional X's should be added that would result in additional terms in the expression.

To illustrate let us consider the function specified by Table 1-23 and its corresponding K map shown in Table 1-24. Note that the two groupings determine that the simplified expression is expressed as $J = C + \bar{A}\bar{B}$

Table. Truth Table of the Function J

Input			Output
A	**B**	**C**	**X**
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	X
1	1	1	X

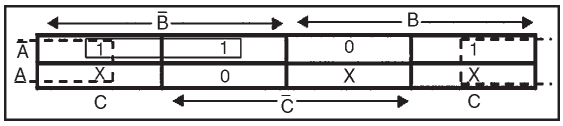
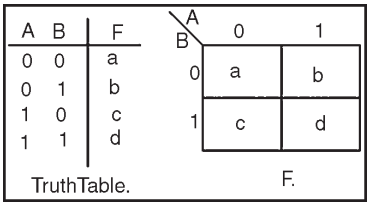


Fig. K-Map for the Function J

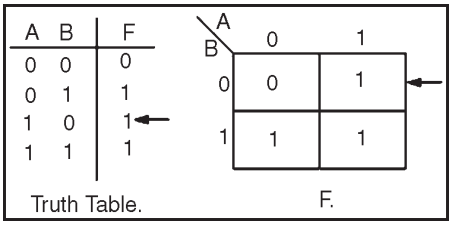
A Karnaugh map provides a pictorial method of grouping together expressions with common factors and therefore eliminating unwanted variables. The Karnaugh map can also be described as a special arrangement of a truth table.

The diagram below illustrates the correspondence between the Karnaugh map and the truth table for the general case of a two variable problem.



The values inside the squares are copied from the output column of the truth table, therefore there is one square in the map for every row in the truth table. Around the edge of the Karnaugh map are the values of the two input variable.

A is along the top and B is down the left hand side. The diagram below explains this:



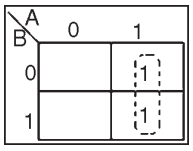
The values around the edge of the map can be thought of as coordinates. So as an example, the square on the top right hand corner of the map in the above diagram has coordinates A=1 and B=0. This square corresponds to the row in the truth table where A=1 and B=0 and F=1. Note that the value in the F column represents a particular function to which the Karnaugh map corresponds.

EXAMPLES

Example 1:

Consider the following map. The function plotted is:

$Z = f(A,B) = A\bar{B} + AB$



- Note that values of the input variables form the rows and columns. That is the logic values of the variables A and B (with one denoting true form and zero denoting false form) form the head of the rows and columns respectively.
- Bear in mind that the above map is a one dimensional type which can be used to simplify an expression in two variables.

- There is a two-dimensional map that can be used for up to four variables, and a three-dimensional map for up to six variables.

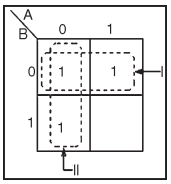
Using algebraic simplification:

$$Z = A\overline{B} + AB$$
$$Z = A(\overline{B} + B)$$
$$Z = A$$

Referring to the map above, the two adjacent 1's are grouped together. Through inspection it can be seen that variable B has its true and false form within the group. This eliminates variable B leaving only variable A which only has its true form. The minimised answer therefore is Z = A.

Example 2:

Consider the expression $Z = f(A,B) = \overline{A}\overline{B} + A\overline{B} + \overline{A}B$ *plotted on the Karnaugh map*:



Pairs of 1's are *grouped* as shown above, and the simplified answer is obtained by using the following steps:

Note that two groups can be formed for the example given above, bearing in mind that the largest rectangular clusters that can be made consist of two 1s. Notice that a 1 can belong to more than one group.

The first group labelled I, consists of two 1s which correspond to A = 0, B = 0 and A = 1, B = 0. Put in another way, all squares in this example that correspond to the area of the map where B = 0 contains 1s, independent of the value of A. So when B = 0 the output is 1. The expression of the output will contain the term $\overline{B}$

For group labelled II corresponds to the area of the map where A = 0. The group can therefore be defined as $\bar{A}$. This implies that when A = 0 the output is 1. The output is therefore 1 whenever B = 0 and A = 0

Hence the simplified answer is Z = $\bar{A} + \bar{B}$

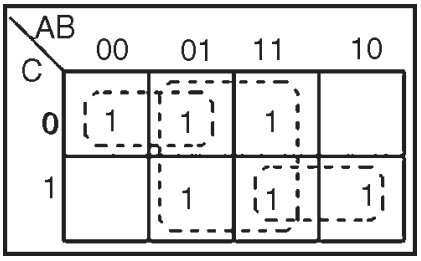
Problems

Minimise the following problems using the Karnaugh maps method.

$$Z = f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}B + AB\bar{C} + AC$$

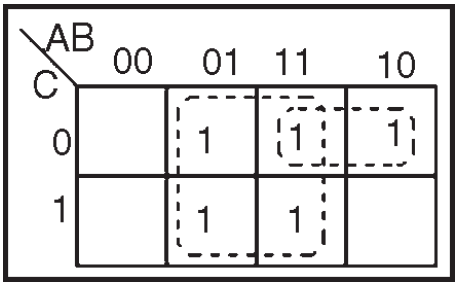
$$Z = f(A,B,C) = \bar{A}B + B\bar{C} + BC + A\bar{B}\bar{C}$$

$$Z = f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}B + AB\bar{C} + AC$$



By using the rules of simplification and ringing of adjacent cells in order to make as many variables redundant, the minimised result obtained is B + AC+ $\bar{A}\bar{C}$

$$Z = f(A,B,C) = \bar{A}B + B\bar{C} + BC + A\bar{B}\bar{C}$$



By using the rules of simplification and ringing of adjacent cells in order to make as many variables redundant, the minimised result obtained is B + A$\bar{C}$

TABULAR METHOD OF MINIMISATION

In order to understand the tabular method of minimisation, it is best you understand the numerical assignment of Karnaugh map cells and the incompletely specified functions also known as the can't happen conditions. This is because the tabular method is based on these principles.

The tabular method which is also known as the Quine-McCluskey method is particularly useful when minimising functions having a large number of variables, *e.g.* The six-variable functions. Computer programmes have been developed employing this algorithm. The method reduces a function in standard sum of products form to a set of prime implicants from which as many variables are eliminated as possible. These prime implicants are then examined to see if some are redundant.

The tabular method makes repeated use of the law $A + \bar{A} = 1$. Note that Binary notation is used for the function, although decimal notation is also used for the functions. As usual a variable in true form is denoted by 1, in inverted form by 0, and the abscence of a variable by a dash (–).

Rules of Tabular Method

Consider a function of three variables f(A, B, C):

$\bar{A}BC$ is represented by $011 \leftarrow$ Binary notation, where $A = 0, B = 1$ and $C = 1$

$A\bar{B}\bar{C}$ is represented by 100

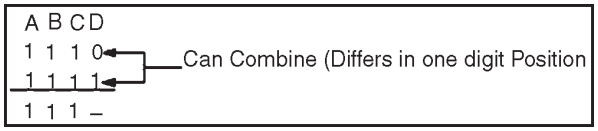
$A\bar{C}$ is represented by 1 – 0

BC is represented by – 11

Consider the function:

$$F(A,B,C,D) = \sum(1110,1111) = ABC\bar{D} + ABCD = ABC$$

Listing the two minterms shows they can be combined,



Now consider the following:

$$F(A,B,C,D) = \sum(1101,1110) = AB\bar{C}D + ABC\bar{D}$$

Note that these variables cannot be combined,

This is because the *FIRST RULE* of the Tabular method for two terms to combine, and thus eliminate one variable, is that they must differ in only one digit position.

Bear in mind that when two terms are combined, one of the combined terms has one digit more at logic 1 than the other combined term. This indicates that the number of 1's in a term is significant and is referred to as its index.

For example: f(A, B, C, D)

0000.....Index 0

0010, 1000.....Index 1

1010, 0011, 1001.....Index 2

1110, 1011.....Index 3

1111.....Index 4

The necessary condition for combining two terms is that the indices of the two terms must differ by one logic variable which must also be the same.

Examples

Example 1:

Consider the function:

$$Z = f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C$$

To make things easier, change the function into binary notation with index value and decimal value.

$$f(A,B,C) = \sum(000,001,100,101) — \text{Binary notation}$$

$$0 \quad 1 \quad 1 \quad 2 — \text{Index}$$

$$0 \quad 1 \quad 4 \quad 5 — \text{Decimal value}$$

Tabulate the index groups in a columm and insert the decimal value alongside.

	First List	Second List	Third List
	A B C	A B C	A B C
Index 0 →	0 → 0 0 0 ✓	0,1 0 0 – ✓	0,1,4,5 – 0 –
	1 → 0 0 1 ✓	0,4 – 0 0 ✓	0,4,1,5 – 0 –
Index 1 →	4 → 1 0 0 ✓	1,5 – 0 1 ✓	
Index 2 →	5 → 1 0 0 ✓	4,5 1 0 – ✓	

From the first list, we combine terms that differ by 1 digit only from one index group to the next. These terms from the first list are then seperated into groups in the second list. Note that the ticks are just there to show that one term has been combined with another term. From the second list we can see that the expression is now reduced to:

$$Z = \overline{A}\overline{B} + \overline{B}\overline{C} + \overline{B}C + A\overline{B}$$

From the second list note that the term having an index of 0 can be combined with the terms of index 1. Bear in mind that the dash indicates a missing variable andmust line up in order to get a third list. The final simplified expression is:

$$Z = \overline{B}$$

Bear in mind that any unticked terms in any list must be included in the final expression (none occured here except from the last list). Note that the only prime implicant here is $Z = \overline{B}$. The tabular method reduces the function to a set of prime implicants. Note that the above solution can be derived algebracially. Attempt this in your notes.

Example 2:

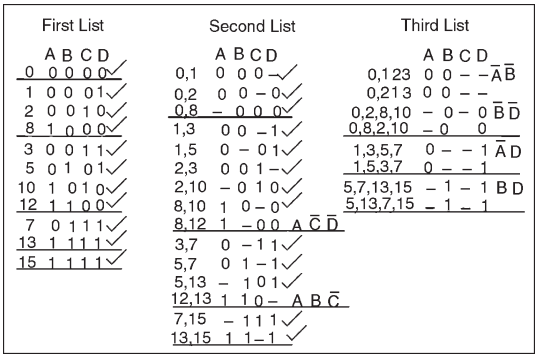
Consider the function:

f(A, B, C, D) = $\sum$(0,1,2,3,5,7,8,10,12,13,15), note that this is in decimal form.

$\sum$(0000,0001,0010,0011,0101,0111,1000,1010,1100,1101,1111)

in binary form.

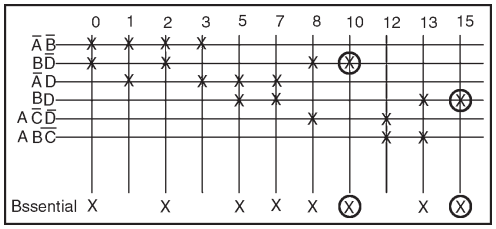
(0,1,1,2,2,3,1,2,2,3,4) in the index form.



The prime implicants are:

$$\bar{A}\bar{B} + \bar{B}\bar{D} + \bar{A}D + BD + A\bar{C}\bar{D} + AB\bar{C}$$

The chart is used to remove redundant prime implicants. A grid is prepared having all the prime implicants listed at the left and all the minterms of the function along the top. Each minterm covered by a given prime implicant is marked in the appropriate position.



From the above chart, BD is an essential prime implicant. It is the only prime implicant that covers the minterm decimal 15 and it also includes 5, 7 and 13 $\bar{B}\bar{D}$ is also an essential prime implicant. It is the only prime implicant that covers the minterm denoted by decimal 10 and it also includes the

terms 0, 2 and 8. The other minterms of the function are 1, 3 and 12. Minterm 1 is present in $\bar{A}\bar{B}$ and $\bar{A}D$. Similarly for minterm 3. We can therefore use either of these prime implicants for these minterms. Minterm 12 is present in $A\bar{C}\bar{D}$ and $AB\bar{C}$, so again either can be used.

Thus, one minimal solution is:

$$Z = \bar{B}\bar{D} + BD + \bar{A}\bar{B} + A\bar{C}\bar{D}$$

Problems

1. *Minimise the function below using the tabular method of simplification*:

$$Z=f(A,B,C,D)=$$

$$\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + \bar{A}BCD + \bar{A}BC\bar{D} + \bar{A}\bar{B}CD$$

2. Using the tabular method of simplification, find all equally minimal solutions for the function below.

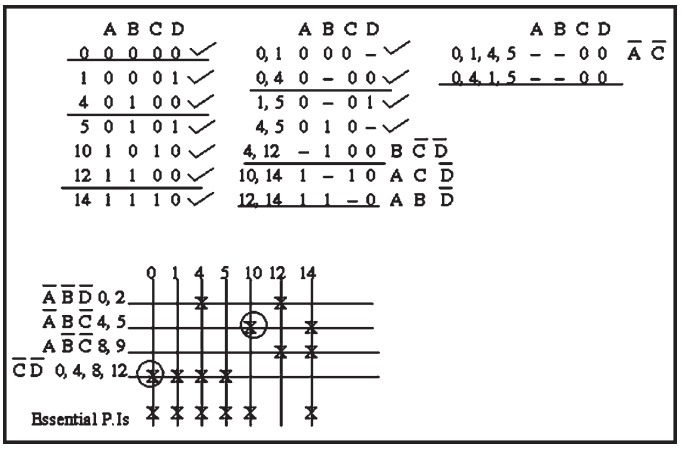
$Z = f(A,B,C,D) = \sum(1,4,5,10,12,14)$

3. *Consider the function*:

$$Z=f(A,B,C,D)=$$

$$\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + \bar{A}BCD + \bar{A}BC\bar{D} + \bar{A}\bar{B}CD$$

Convert to decimal and binary equivalents:



$Z = f(A,B,C,D) = \sum(0,2,4,5,8,9,12)$ - decimal equivalent

Z = f(A,B,C,D) = $\sum$(0000,0010,0100,0101,1000,1001,1100) - binary equivalent

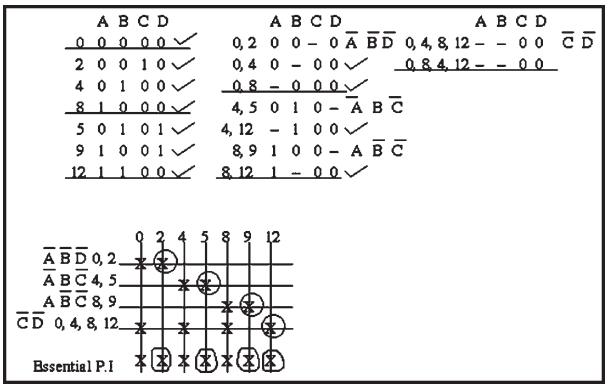
(0, 1, 1, 2, 1, 2, 2) - index values

The simplified answer is:

Z = $Z = \bar{A}\bar{B}\bar{D} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + \bar{C}\bar{D}$

$f(A,B,C) = \sum(0,1,4,5,10,12,14,) = \sum(0000,0001,0101,1010,1100,1110)$

$= \text{Index}: 0 \quad 1 \quad 2 \quad 2 \quad 2 \quad 3$



Note that the remaining term 12 is covered by $B\bar{C}\bar{D}$ and $AB\bar{D}$,

One simplified answer is:

$$Z = \bar{A}\bar{C} + AC\bar{D} + B\bar{C}\bar{D}$$

Another answer is:

$$Z = \bar{A}\bar{C} + AC\bar{D} + AB\bar{D}$$

CODE CONVERSIONS

Converting from one code form to another code form is called code conversion, like converting from binary to decimal or converting from hexadecimal to decimal.

BINARY-TO-DECIMAL CONVERSION

Any binary number can be converted to its decimal equivalent simply by summing together the weights of the various positions in the binary number which contain a 1.

Binary	Decimal
11011_2	
$2^4+2^3+0^1+2^1+2^0$	=16+8+0+2+1
Result	27_{10}

and,

Binary	Decimal
10110101_2	
$2^7+0^6+2^5+2^4+0^3+2^2+0^1+2^0$	=128+0+32+16+0+4+0+1
Result	181_{10}

You should have noticed that the method is to find the weights (*i.e.*, powers of 2) for each bit position that contains a 1, and then to add them up.

DECIMAL-TO-BINARY CONVERSION

There are 2 Methods

1. Reverse of Binary-To-Decimal Method
2. Repeat Division

Reverse of Binary-To-Decimal Method

Decimal	Binary
45_{10}	=32 + 0 + 8 + 4 +0 + 1 $=2^5+0+2^3+2^2+0+2^0$
Result	$=101101_2$

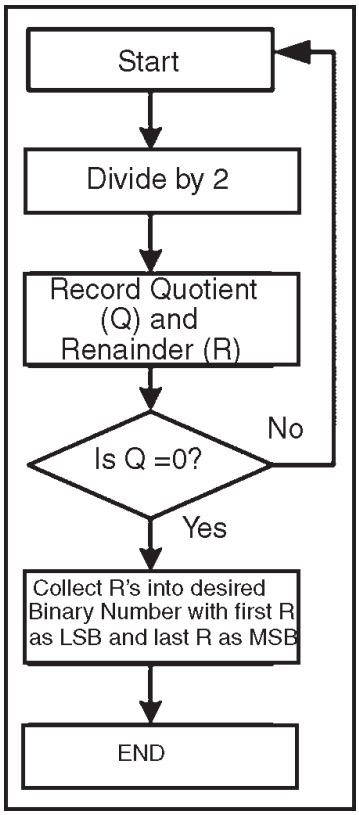
Repeat Division-Convert Decimal to Binary

This method uses repeated division by 2.

Convert 25_{10} to binary

Division	Remainder	Binary
25/2	= 12+ remainder of 1	1 (Least Significant Bit)
12/2	= 6 + remainder of 0	0
6/2	= 3 + remainder of 0	0
3/2	= 1 + remainder of 1	1
1/2	= 0 + remainder of 1	1 (Most Significant Bit)
Result	25_{10}	= 11001_2

The Flow chart for repeated-division method is as follows:



BINARY-TO-OCTAL/ OCTAL-TO-BINARY CONVERSION

Octal Digit	0	1	2	3	4	5	6	7
Binary Equivalent	000	001	010	011	100	101	110	111

Each Octal digit is represented by three binary digits.

Example:

$100\ 111\ 010_2 = (100)\ (111)\ (010)_2 = 4\ 7\ 2_8$

Repeat Division-Convert Cecimal to Octal [SH

This method uses repeated division by 8.

Example: Convert 177_{10} to octal and binary

Division	Result	Binary
177/8	= 22+ remainder of 1	1 (Least Significant Bit)
22/ 8	= 2 + remainder of 6	6
2/ 8	= 0 + remainder of 2	2 (Most Significant Bit)
Result	177_{10}	= 261_8
Binary		010110001_2

HEXADECIMAL TO DECIMAL/DECIMAL TO HEXADECIMAL CONVERSION

Example:

$2AF_{16} = 2 \times (16^2) + 10 \times (16^1) + 15 \times (16^0) = 687_{10}$

Repeat Division- Convert Decimal to Hexadecimal

This method uses repeated division by 16.

Example: convert 378_{10} to hexadecimal and binary:

Division	Result	Hexadecimal
378/16	= 23+ remainder of 10	A (Least Significant Bit)23
23/16	= 1 + remainder of 7	7
1/16	= 0 + remainder of 1	1 (Most Significant Bit)
Result	378_{10}	= $17A_{16}$
Binary		= 0001 0111 1010_2

BINARY-TO-HEXADECIMAL/HEXADECIMAL-TO-BINARY CONVERSION

Hexadecimal Digit	0	1	2	3	4	5	6	7
Binary Equivalent	0000	0001	0010	0011	0100	0101	0110	0111v

Hexadecimal Digit	8	9	A	B	C	D	E	F
Binary Equivalent	1000	1001	1010	1011		1100	1101	1110	1111

Each Hexadecimal digit is represented by four bits of binary digit.

Example:

$1011\ 0010\ 1111_2 = (1011)\ (0010)\ (1111)_2 = B\ 2\ F_{16}$

OCTAL-TO-HEXADECIMAL HEXADECIMAL-TO-OCTAL CONVERSION

- Convert Octal (Hexadecimal) to Binary first.
- Regroup the binary number by three bits per group starting from LSB if Octal is required.
- Regroup the binary number by four bits per group starting from LSB if Hexadecimal is required.

Example: Convert $5A8_{16}$ to Octal.

Hexadecimal	Binary/Octal
5A816	= 0101 1010 1000 (Binary)
	= 010 110 101 000 (Binary
Result	= 2 6 5 0 (Octal)

DEMORGANS THEOREM

For N variables, DeMorgan's theorems are expressed in the following formulas:

$$\overline{ABC...N} = \bar{A} + \bar{B} + \bar{C} + ... + \bar{N}(1-2)$$

That is, the complement of the product is equivalent to the sum of the complements

$$\overline{A + B + C + ... + N} = \bar{A}\bar{B}\bar{C}....\bar{N}(1-3)$$

Similarly, the complement of the sum is equivalent to the product of the complements

Applications of DeMorgan's theorems range from deriving the composite of functions to providing alternative design to logic circuits. The objective of using these theorems in circuit design is to minimise the number of ICs required in a logic circuit.

EXAMPLES OF DEMORGAN'S THEOREM

Applying Demorgan's theorem to the expression $\overline{A+B+C+...+N} = \bar{A}\bar{B}\bar{C}....\bar{N}(1-3)$ we get:

$$\overline{(A+B+C)D} = \overline{(A+B+C} + \bar{D}$$
$$= \bar{A}\bar{B}\bar{C} + \bar{D}$$

Applying Demorgan's theorem to the expression $ABC + DEF$ *we get*:

$$\overline{ABC + DEF} = (\overline{ABC})(\overline{DEF})$$
$$= (\bar{A} + \bar{B} + \bar{C})(\bar{D} + \bar{E} + \bar{F})$$

First, recall that a binary variable may be either in its true form (*A*) or its complement ($\bar{A}$). Second, recall that for *n*variables, the maximum number of input variable combinations is given by $N = 2^n$.

Then considering the AND gate, each of the N logic expressions formed is called a *standard product* or *minterm.* As indicated in Table 1-13, binary digits '1' and '0' are taken to represent a given variable (*e.g. A*) and its complemented (*e.g.* $\bar{A}$), respectively. Also from Table 1-13 note that each minterm is assigned a symbol (P_j) each where j is the decimal equivalent to the binary number of the minterm designated.

Similarly, if we consider an OR gate, each of the *N* logic expressions formed is called a *standard sum* or *maxterm.* In this case binary digits '1' and '0' are taken to represent a given complemented variable (*e.g.* $\bar{A}$) and its true form (e.g.*A*), respectively.

As shown in Table, a symbol (S_j) is assigned to each maxterm where j is the decimal equivalent to the binary number of the maxterm designated. Also observe that each maxterm is the complement of its corresponding minterm, and vice versa.

Table. The Complement of Each Term may be Determined using DeMorgan's Theorem

Input			Minterms		Maxterms	
A	B	C	Terms	Designation	Terms	Designation
0	0	0	$\overline{A}\,\overline{B}\,\overline{C}$	P_0	$A+B+C$	S_0
0	0	1	$\overline{A}\,\overline{B}\,C$	P_1	$A+B+\overline{C}$	S_1
0	1	0	$\overline{A}\,B\,\overline{C}$	P_2	$A+\overline{B}+C$	S_2
0	1	1	$\overline{A}\,B\,C$	P_3	$A+\overline{B}+\overline{C}$	S_3
1	0	0	$A\,\overline{B}\,\overline{C}$	P_4	$\overline{A}+B+C$	S_4
1	0	1	$A\,\overline{B}\,C$	P_5	$\overline{A}+B+\overline{C}$	S_5
1	1	0	$A\,B\,\overline{C}$	P_6	$\overline{A}+\overline{B}+C$	S_6
1	1	1	$A\,B\,C$	P_7	$\overline{A}+\overline{B}+\overline{C}$	S_7

Table above shows the Minterms and Maxterms for three Binary Variables

SIGNIFICANCE OF MINTERMS AND MAXTERMS

In short, minterms and maxterms may be used to define the two standard forms for logic expressions, namely the sum of products (SOP), or sum of minterms, and the product of sums (POS), or product of maxterms. These standard forms of expression aid the logic circuit designer by simplifying the derivation of the function to be implemented.

Sum of Products

The SOP expression is the equation of the logic function as read off the truth table to specify the input combinations when the output is a logical 1. To illustrate, let us consider Table. Observe that the output is high for the rows labelled 3, 5 and 6.

The SOP expression for this circuit is thus given any of the following:

1. $X = \overline{A}BC + A\overline{B}C + AB\overline{C}$
2. $X = P_3 + P_5 + P_6$ or
3. $X(A,B,C) = \sum(3,5,6)$

Table. Truth of $X = \overline{A}BC + A\overline{B}C + AB\overline{C}$

Row Number	Input A	Output B
C	X	0
0	0	0
0	1	0
0	1	0
2	0	1
0	0	3
0	1	1
1	4	1
0	0	0
5	1	0
1	1	6
1	1	0
1	7	1
1	1	0

Product of Sums

The POS expression is the equation of the logic function as read off the truth table to specify the input combinations when the output is a logical 0. To illustrate, let us again consider Table. Observe that the output is low for the rows labelled 0, 1, 2, 4 and 7. The POS expression for this circuit is thus given by any of the following:

1. $X = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + C) \quad (\overline{A} + \overline{B} + \overline{C})$
2. $X = S_0S_1S_2S_4S_7$
3. $X(A, B, C) = \prod(0,1,2,4,7)$

Further Examples

Derive the SOP and POS expressions for the truth tables shown below:

Table. Truth Describing the Output X

Row Number	Input A	Input B	Output X
0	0	0	1
1	0	1	0

2	1	0	1
3	1	1	0

SOP

$$X = P_0 + P_2 \text{ or}$$

$$X = \bar{A}\bar{B} + A\bar{B}$$

POS

$$X = S_1 S_3 \text{ or}$$

$$X = (A + \bar{B})(\bar{A} + \bar{B})$$

Recall from an earlier lecture that we stated that the NAND and NOR were universal logic gates. Using DeMorgan's theorem and the Rules and laws of Boolean algebra proving this should be an easy task. Figure shows the equivalency between the basic logic gates and their NAND logic circuits counterpart. Similarly, Figure shows the equivalency between the basic logic gates and their NOR logic circuits counterpart.

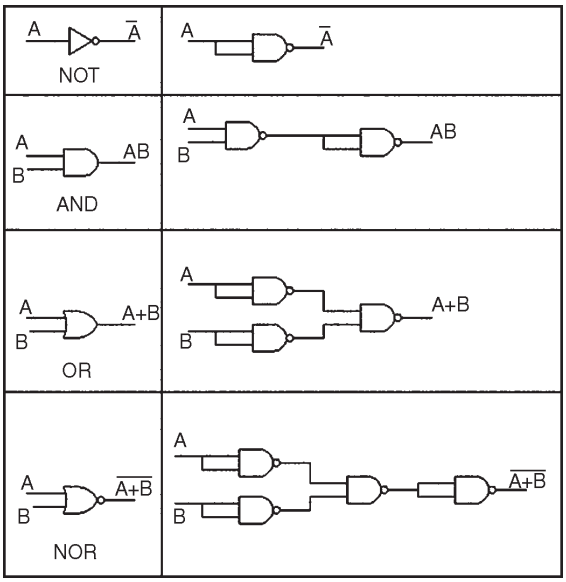


Fig. NAND Equivalent Circuits

Note that $AB = \overline{\overline{AB}}$ i.e the complement of "A NAND B"

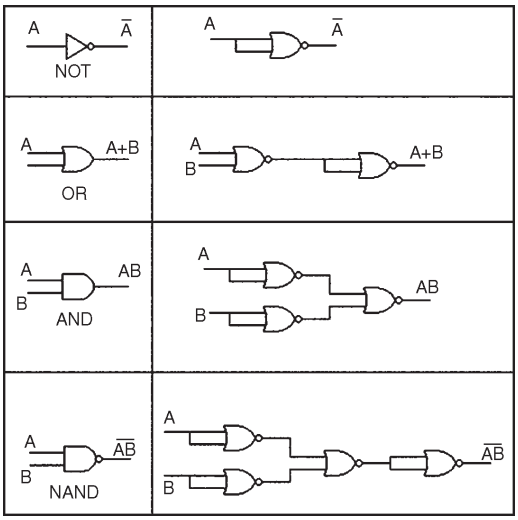


Fig. NOR Equivalent Circuits

BINARY CODES

Binary codes are codes which are represented in binary system with modification from the original ones.

Below we will be seeing the following:

- Weighted Binary Systems
- Non Weighted Codes

WEIGHTED BINARY SYSTEMS

Weighted binary codes are those which obey the positional weighting principles, each position of the number represents a specific weight. The binary counting sequence is an example.

Decimal	8421	2421	5211	Excess-3
0	0000	0000	0000	0011
1	0001	0001	0001	0100
2	0010	0010	0011	0101
3	0011	0011	0101	0110
4	0100	0100	0111	0111
5	0101	1011	1000	1000
6	0110	1100	1010	1001

7	0111	1101	1100	1010
8	1000	1110	1110	1011
9	1001	1111	1111	1100

8421 Code/BCD Code

The BCD (Binary Coded Decimal) is a straight assignment of the binary equivalent. It is possible to assign weights to the binary bits according to their positions. The weights in the BCD code are 8,4,2,1.

Example: The bit assignment 1001, can be seen by its weights to represent the decimal 9 because.

$$1\times8+0\times4+0\times2+1\times1 = 9$$

2421 Code

This is a weighted code, its weights are 2, 4, 2 and 1. A decimal number is represented in 4-bit form and the total four bits weight is 2 + 4 + 2 + 1 = 9.

Hence the 2421 code represents the decimal numbers from 0 to 9.

5211 Code

This is a weighted code, its weights are 5, 2, 1 and 1. A decimal number is represented in 4-bit form and the total four bits weight is 5 + 2 + 1 + 1 = 9. Hence the 5211 code represents the decimal numbers from 0 to 9.

Reflective Code

A code is said to be reflective when code for 9 is complement for the code for 0, and so is for 8 and 1 codes, 7 and 2, 6 and 3, 5 and 4. Codes 2421, 5211, and excess-3 are reflective, whereas the 8421 code is not.

Sequential Codes

A code is said to be sequential when two subsequent codes, seen as numbers in binary representation, differ by one. This greatly aids mathematical manipulation of data. The 8421 and Excess-3 codes are sequential, whereas the 2421 and 5211 codes are not.

NON WEIGHTED CODES

Non weighted codes are codes that are not positionally weighted. That is, each position within the binary number is not assigned a fixed value.

Excess-3 Code

Excess-3 is a non weighted code used to express decimal numbers. The code derives its name from the fact that each binary code is the corresponding 8421 code plus 0011(3).

Example: 1000 of 8421 = 1011 in Excess-3

Gray Code

The gray code belongs to a class of codes called minimum change codes, in which only one bit in the code changes when moving from one code to the next. The Gray code is non-weighted code, as the position of bit does not contain any weight. The gray code is a reflective digital code which has the special property that any two subsequent numbers codes differ by only one bit. This is also called a unit-distance code. In digital Gray code has got a special place.

Decimal Number	Binary Code	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110

5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

BINARY TO GRAY CONVERSION

- Gray Code MSB is binary code MSB.
- Gray Code MSB-1 is the XOR of binary code MSB and MSB-1.
- MSB-2 bit of gray code is XOR of MSB-1 and MSB-2 bit of binary code.
- MSB-N bit of gray code is XOR of MSB-N-1 and MSB-N bit of binary code.

ERROR DETECTING AND CORRECTING CODES

For reliable transmission and storage of digital data, error detection and correction is required. Below are a few examples of codes which permit error detection and error correction after detection.

ERROR DETECTING CODES

When data is transmitted from one point to another, like in wireless transmission, or it is just stored, like in hard disks and memories, there are chances that data may get corrupted. To detect these data errors, we use special codes, which are error detection codes.

Parity

In parity codes, every data byte, or nibble (according to how user wants to use it) is checked if they have even number of ones or even number of zeros. Based on this information an additional bit is appended to the original data. Thus if we consider 8-bit data, adding the parity bit will make it 9 bit long.

At the receiver side, once again parity is calculated and matched with the received parity (bit 9), and if they match, data is ok, otherwise data is corrupt.

There are two types of parity:

- *Even parity*: Checks if there is an even number of ones; if so, parity bit is zero. When the number of ones is odd then parity bit is set to.
- *Odd Parity*: Checks if there is an odd number of ones; if so, parity bit is zero. When number of ones is even then parity bit is set to.

Check Sums

The parity method is calculated over byte, word or double word. But when errors need to be checked over 128 bytes or more (basically blocks of data), then calculating parity is not the right way. So we have checksum, which allows to check for errors on block of data. There are many variations of checksum.

- Adding all bytes
- CRC
- Fletcher's checksum
- Adler-32

The simplest form of checksum, which simply adds up the asserted bits in the data, cannot detect a number of types of errors. In particular, such a checksum is not changed by:

- Reordering of the bytes in the message
- Inserting or deleting zero-valued bytes
- Multiple errors which sum to zero

Example of Checksum: Given 4 bytes of data (can be done with any number of bytes): 25h, 62h, 3Fh, 52h

- Adding all bytes together gives 118h.
- Drop the Carry Nibble to give you 18h.
- Get the two's complement of the 18h to get E8h. This is the checksum byte.

To Test the Checksum byte simply add it to the original group of bytes. This should give you 200h. Drop the carry nibble again giving 00h. Since it is 00h this means the checksum means the bytes were probably not changed.

ERROR-CORRECTING CODES

Error-correcting codes not only detect errors, but also correct them. This is used normally in Satellite communication, where turn-around delay is very high as is the probability of data getting corrupt.

ECC (Error correcting codes) are used also in memories, networking, Hard disk, CDROM, DVD etc. Normally in networking chips (ASIC), we have 2 Error detection bits and 1 Error correction bit.

HAMMING CODE

Hamming code adds a minimum number of bits to the data transmitted in a noisy channel, to be able to correct every possible one-bit error. It can detect (not correct) two-bits errors and cannot distinguish between 1-bit and 2-bits inconsistencies. It can't - in general - detect 3(or more)-bits

errors. The idea is that the failed bit position in an n-bit string (which we'll call X) can be represented in binary with $\log_2(n)$ bits, hence we'll try to get it adding just $\log_2(n)$ bits.

First, we set $m = n + \log_2(n)$ to the encoded string length and we number each bit position starting from 1 through m. Then we place these additional bits at power-of-two positions, that is 1, 2, 4, 8..., while remaining ones (3, 5, 6, 7...) hold the bit string in the original order.

Now we set each added bit to the parity of a group of bits. We group bits this way: we form a group for every parity bit, where the following relation holds:

position(bit) AND position(parity) = position(parity)

(Note that: AND is the bit-wise boolean AND; parity bits are included in the groups; each bit can belong to one or more groups.) So bit 1 groups bits 1, 3, 5, 7... while bit 2 groups bits 2, 3, 6, 7, 10..., bit 4 groups bits 4, 5, 6, 7, 12, 13... and so on. Thus, by definition, X (the failed bit position defined above) is the sum of the incorrect parity bits positions (0 for no errors).

To understand why it is so, let's call X_n the n^{th} bit of X in binary representation. Now consider that each parity bit is tied to a bit of X: parity1 $\rightarrow X_1$, parity2 $\rightarrow X_2$, parity4 $\rightarrow X_3$, parity8 $\rightarrow X_4$ and so on–for programmers: they are the respective AND masks –. By construction, the failed bit makes fail only the parity bits which correspond to the 1s in X, so each bit of X is 1 if the corresponding parity is wrong and 0 if it is correct.

Note that the longer the string, the higher the throughput n/m and the lower the probability that no more than one bit fails. So the string to be sent should be broken into blocks

whose length depends on the transmission channel quality (the cleaner the channel, the bigger the block). Also, unless it's guaranteed that at most one bit per block fails, a checksum or some other form of data integrity check should be added.

5

Quantum Cellular Automata

A Quantum Cellular Automata (QCA) is another plausible nanodevice. This is a design which is based upon a theory of using electrons and their energy states for computing. This is a topic which has been studied for over 40 years, back to the work of von Neumann. Rather than being an individual device of a computer circuit scaled to a nanometre size, the QCA is a complete entity.

As described by researchers at Notre Dame, "quantum-dot cellular automata (QCA), parallels conventional computing in using a binary representation of information (not qubits) yet realises functionality in ways which are well-suited to the properties of single molecules - using molecules not as current switches but as structured charge containers."[LENT] The QCA is a single molecule which will contain a representation of the binary values 0 or 1 based upon the static electric charge of the electron. The cells have

not been found at this point in time to be viable at room temperature, but rather only at very cold degrees in the Kelvin range. Needless to say, operating at such low temperatures is not practical for any working device, and this is a serious limitation to the usefulness of QCAs.

The QCA fabrication does differentiate from other devices that we have seen because the fabrication uses lithography. "Molecular QCA cells must be attached to surfaces in ordered arrays. We are using electron-beam lithography to burn narrow "tracks" along surfaces."

The temperature limitations of Quantum Cellular Automatas have been given a new perspective through a study of Magnetic QCA described by Cowburn and Welland. Their presentation uses interacting submicron magnetic dots to perform logic operations and propagate information. [BECKETT] Although these devices will not be in the nanometre range, they will use less power than current CMOS.

CMOL

Some researchers are presenting a possible bridge from CMOS to the next generation of molecular computing by formulating a hybrid circuit. One such hybrid circuit is dubbed a CMOL, a play on the combination of CMOS and molecular. The CMOL is "a circuit [which] combines an advanced CMOS subsystem with two, mutually perpendicular, arrays of parallel nanowires and similar molecular devices formed at each crosspoint of the nanowires." [LIKHAREV]

This approach is important in several ways. First, it's unlikely that the computing industry will not be able to make

the leap from its top-down lithography methodology to a bottom up approach using nanowires and integrating with some molecular components without some intermediate step. The CMOL maintains the stability of the silicon chip and takes a small step forward by using bottom up fabrication. The CMOS circuit maintains the level of functionality that we are used to. As mentioned, the system bus can be an area in which slowness can occur. Experiments with the CMOL circuits have operated faster than regular computers with better power dissipation.

The creator of the CMOL hybrid circuit claims, 'The development of CMOL technology (especially the high-yield molecular self-assembly) will certainly require a major industrial effort and substantial time period, probably not less than 10 to 15 years. However, this timing may be still acceptable to prevent the impending crisis of Moore's law, provided that we start right now."

MARKOV RANDOM NETWORK

Researchers at Brown University have a proposed architecture based on Markov Random Fields (MRF). A MRF is a concept based on Markov chain, which is a "graph of stochastic variables, where each variable has the property that is independent of all the others (future and past) given its two neighbors" A Markov Random Field network allows circuits to behave in a relatively independent fashion, allowing for re-configurability and a good level of fault tolerance.

The researchers feel that carbon nanotubes are one of the most promising devices which have been built at the Nanoscale. Some issues are inherent with these nanodevices though; these researchers point out that the use of carbon

nanotubes circuits will increase failure rates and bring about heat dissipation issues. In order to deal with possible failure rates upwards of 10% and heating issues at the thermal limit, it is necessary to configure a network of circuits which can handle a high fault level and can somehow dissipate heat throughout the circuit network.

It has been found through experiments that the use of a Markov Random Field Network in conjunction with the Gibbs formulation for dissipation of heat can be a viable solution, because as the authors state, "its operation does not depend o perfect devices or perfect connections. [...] Successful operation only requires that the energy of correct states is lower than the energy of errors"

CELL MATRIX

The Cell Matrix architecture is presented by the Cell Matrix Corporation in Salt Lake City. This piece of research differs from the previous sections in the sense that this is a proposal of a true architecture as opposed to a design of a nano-sized component.

This architecture is currently being designed and tested in the silicon chip domain, but all research, configuration and design is for the coming of nanostructures. Currently, the Cell Matrix Corporation is developing an atomic cell unit, which is repeated to form a multidimensional matrix of cells to make a highly scalable architecture, which differs from other proposals discussed thus far.

The architecture is similar to the field programmable gate arrays (FPGAs) like the Nanofabric described in an earlier section, and it also has similarities to the cellular automaton

proposed by Von Neumann. Several aspects of a cell matrix distinguish it from a Von Neumann cellular automaton. First, the cell matrix has the same three dimensional nature.

The cell matrix can only be in a certain number of states it can be in at any given time, just like a cellular automaton. Second, a cell matrix is fine grained and reconfigurable, and third it is programmed like a digital circuit, and not like a cellular automaton. The benefits which cell matrix architecture provides are many.

It can efficiently handle a large number of switches due to the fact that its controlling system is highly scalable, and can handle any addition of cells to the system. This architecture also promises to provide highly parallel large scale computing. This is the result of the fact that each cell contains all necessary functionality, and can compute across the matrix at any point in a parallel fashion.

This group has demonstrated this highly parallel computing ability through an experiment with a search space problem. The cell matrix architecture relies on a very simple hardware definition mixed with a complex programming of each individual cell. Simple logic functions can be handled by individual cells, whereas more complex functions are handled by a collection of cells, which are spatially close to each other. In these collections, cells are set up to perform a subset of the work of the entire circuit.

These cells communicate with one another in an asynchronous fashion. Each cell also has the ability to reconfigure itself, allowing for self-testing and fault tolerance, although experimentation and testing in this arena is still taking place.

The Cell Matrix Corporation believes that their design can provide dynamic circuitry, allowing each of the cells to change their original behaviour or the behaviour of cells around them, or to migrate into new areas within the matrix. They also see this as being a mechanism to handle faulty cells in order to make their circuits fault tolerant.

Defect tolerance is always an important issue with any computer architecture due to the cost which defects add to the final cost of a circuit. Due to the fact that silicon chips are discarded if their fail rate is too high, a new architecture can find cost savings in total manufacturing costs if the new circuitry can fin a way to be fault tolerant by working around bad cells. The Cell Matrix has implemented defect tolerance and self testing in their architecture.

They have designed a test driver which has the following goals:

- Permit reporting of faults with a high resolution
- Permit access to a region despite failed regions near it.
- It should be easy to extend to a decentralized, parallel, distributed fault testing process with as small a footprint as possible.
- Have a driver which can share the hardware with other tasks so it can perform its functions on subcomponents of a critical system while that system is running. [DURBECK01]

The Cell Matrix architecture outline is well planned and several experiments on digital circuits have been promising. However, no experiments have yet been performed on any microscopic components. In theory the layout may be scalable

and successful; more incorporation of current nano-component research should be considered.

NANOPRISM

It has already been determined that fault tolerance, while something not so prevalent in the construction of silicon chips, will need to be an important part of any architecture presented for a nanocomputer. One way in which many manufacturers provide better reliability is by implementing techniques to increase redundancy.

However, it is known that adding redundancy cannot always increase reliability of a device due to the fact that the redundant device can also contain faults. A paper from Virginia Tech is attempting to determine the level at which fault tolerance and redundancy can produce a reliable nano-architecture.

As stated by the researchers, "The questions we try to answer in this paper is, what level of granularity and what redundancy levels result in optimal reliability for specific architectures. In this paper, we extend previous work on evaluating reliability-redundancy trade-offs for NAND multiplexing to granularity vs. redundancy vs. reliability trade-offs for other redundancy mechanisms, and present our automation mechanism using the probabilistic model checking tool PRISM."

The NANOPRISM tool is based on another probabilistic model checking tool called PRISM built at the University of Binghamton, which is designed for conventional CMOS architectures. The NANOPRISM tool will be a valuable resource for building a new architecture for a nanocomputer.

Redundancy can be a very important yet delicate technique to employ, due to the fact that employing too much can cause a lessening of reliability, not to mention the fact that there are several levels of granularity at which redundancy can be implemented. Because this tool automates the checking of a defect tolerant architecture, areas in which trade off levels are achieved can be identified much more quickly. The number of circuits and cells are going to increase exponentially as nanodevices scale up to the level they are supposed to be at. If we currently have almost 1 billion transistors on a single silicon chip, the number of devices we can just imagine the numbers present in a nanocomputer.

PROPOSED NANOCOMPUTER ARCHITECTURE DESIGN

After looking at all of these different nanodevices and proposed networks and architectures, how can we determine what is the best direction to go in order to create the optimal computer architecture for the next generation? The first thing to look at is what will be the major issues and limitations with the devices we wish to use as the basic building blocks for our computer.

The focus will not necessarily be on how these devices will be built whether that is through lithography or self assembly, it is immaterial, and concern about cost will be factored in, but not necessarily a focal point. Rather, we are more concerned with what qualities the finished product and the surrounding architecture will need to possess in order to be successful.

What are some qualities are interested in?:

- Fault Tolerance,
- Self testing/Re-configuration,
- Heat Tolerance and Power Dissipation,
- Independent and parallel processing,
- Improved communication, handling current pervasive bottlenecks in interconnections,
- Scalability.

We will use these important qualities as a guide to what each nanocomputer component will contain, as we lay out our proposed nanocomputer architecture from the bottom up.

CHIP DESIGN

The chip design of the nanocomputer would be best suited to have a carbon base which is both self-testing and reconfigurable. Because we know that faults will be at levels upwards of 10% for any chip built in the nanometre range, it will be necessary for these features to be part of any solution we choose. Also, it's important to remember the breadth of our failure rate. A Nanocomputer will contain not just one billion but several billion chips. If we have a computer with 5 billion chips, we will have 500 million chips which are defective with a 10% error rate. It will be an important cost factor to be able to repair these chips after they have been fabricated.

Of the research we have gone over thus far, the best proposed replacement for the CMOS chips is the NanoBlock, based on the FPGA, the Field Programmable Gate Array. These blocks are carbon based. These devices have been created from a bottom up assembly; they are fault tolerant and reconfigurable.

Bad devices or blocks of devices can be swapped out if necessary Another advantage to the NanoBlock is that it is chemically assembled, allowing more flexibility with reconfiguration. The devices can be manipulated so that changes can be made to improve any faults which may make an entire block unusable.

The argument for NanoBlocks also stems from the need to insure heat and power dissipation. The NanoBlocks are low power devices, and this makes it a good candidate as a base for building a nanocomputer.

INSTRUCTION SET DESIGN

Very little research in nanotechnology has addressed the future of instruction set design. We know that our hardware systems will become more complex and more powerful. We can infer that the instruction set design should then move back from a reduced instruction set (RISC) design to a more powerful complex instruction set (CISC).

This way, we can take advantage of the improvements in the hardware and write less complex compilers. The instructions can also perform more work in a single clock cycle, improving the speed operations are performed in the system. However, the one architecture that we have looked at which took instruction set design into consideration only used a simple accumulator. One would think that a more complex design would be beneficial and more productive, but it is possible that a simple approach will be the best way to manage so many million components.

RISC was chosen as the instruction set for this generation of components due to cost, and it's conceivable that the next generation will choose a similarly simple instruction set due

to size constraints and sheer numbers. The instruction set will presumably be one of the last design components to consider, but designers must take it into consideration when building components.

INTERCONNECTIONS AND COMMUNICATION POINTS

Interconnections are another vital piece to consider. If there can be devices in the multi-billion in a nanocomputer, these nano-connectors will also number in the multi-billion and will need to be efficient and allow for both local and global communication.

The development of nanowires as communication pathways between neighboring cells is the most well developed form of nano-sized interconnects. The carbon based nanowires are the most promising format because of their stability.

These devices also allow for self-testing, reconfiguration, and self-assembly. Each of these qualities makes it an attractive component in terms of costs, reliability, and practicality. It is also presumed that the reduction in size of these wires can improve density and we can move more data through interconnections at a faster rate. This can reduce starvation which can sometimes occur in a system due to bus latency and lack of bandwidth.

While carbon nanowires have been produced in the lab, the nanowires which we must use in our nanocomputer must improve their resistance. Arranging these nanowires in a Markov Random Circuit as one group of researchers has proposed would be a valuable arrangement, and would be an important algorithm to employ in a nanocomputer.

MAIN MEMORY, MULTIPROCESSING AND INSTRUCTION LEVEL PARALLELISM

Memory management is and will continue to be a very difficult topic facing all computer designers. Virtual memory demands which are currently limited by space will be unlimited in a nanocomputer architecture. Memory will be more plentiful, cheaper and faster in a nanocomputer, and we can expect to move from the 1 gigabyte of memory standard on modern computers to numbers two to five times more powerful. However, these advantages will bring a new host of issues.

Instruction level parallelism may be more difficult because there will be so much more data to handle. Algorithms will need to be developed to insure that critical sections of code are executed properly. However, we do know that we will have many more CPUs which we can cluster together with much less space and cost requirements. It is likely that instruction level parallelism will fall away to a new paradigm of handling large scale multiprocessing.

Perhaps machines will be split apart so that certain CPUs are specifically for certain jobs, or certain tasks within a computer. Nanocomputer research has not reached the point of coming up with a memory management scheme, but the chances that it is exactly as we handle memory today is unlikely.

STORAGE

A nanocomputer will have the possibility of an unlimited amount of storage space. Currently we have computers which can hold up to 150 gigabytes of information. We expect a

nanocomputer to be able to hold terabytes of information. These storage disks will also be easily searchable and can also be partitioned.Presumably we can hold volumes of data, and backup tapes or requiring repositories to be placed on a network of computers to make up a server can be a thing of the past. Smaller embedded devices will be able to hold much more data as well. Our entire medical history can exist on a key, or entire music collection can exist on a device the size of a credit card or smaller.

CONCLUSION

The future of nanotechnology will bring exciting change to the computing industry. We are promised machines which are faster, cheaper, and smaller. This promise will come with incredible challenges and sacrifice, and the computing industry must be flexible and agile enough to meet the demands of manufacturing these new devices and designing the machines which will contain them.

The industry must also change the current manufacturing processes and current design paradigms to meet the differences between this new generation of devices and our generation.

These new devices will require a completely new approach in manufacturing, moving from lithography top down design to bottom up design. It may also require self-assembly of components in order to be economically viable. This will require a new set of manufacturing tools and processes. Computer manufacturers will have to change their plants and assembly lines completely in order to build these new components.

A nanocomputer will also require a new instruction set, and innovative ways to handle the new challenges when dealing with molecular components must be considered. Hardware will be less reliable and self-testing and self-configuration must be built in to any nanocomputer. This may make manufacturing more expensive at first, and more time consuming.

The end of hardware improvements predicted by Gordon Moore is upon the computing industry, and lab experiments to build the next generation of hardware must take shape to keep pace with the demanding computing needs of the world. Valuable research has been made in many areas, but we are still in the primitive phases.

The industry has about ten or twenty years left before this paradigm shift must occur. It can be gleaned that we could experience some setbacks in the new generation of devices were they may not perform as reliably or as quickly as we are used to with CMOS based computing devices. However, one could say that it would be worth taking a few steps back in order to move many hundreds of steps forward, which is what nanotechnology is promising us. Below are some pictorial representations of some of the nanodevices and architectures we have discussed in this paper.

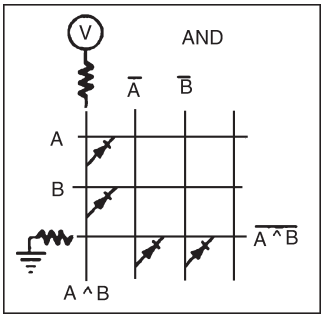


Fig. A Two Input and Gate Implemented in a CAEN Grid

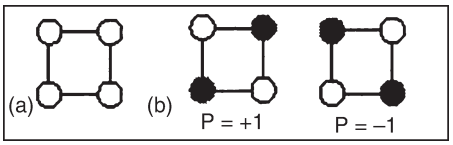


Fig. QCA Four Dot cell

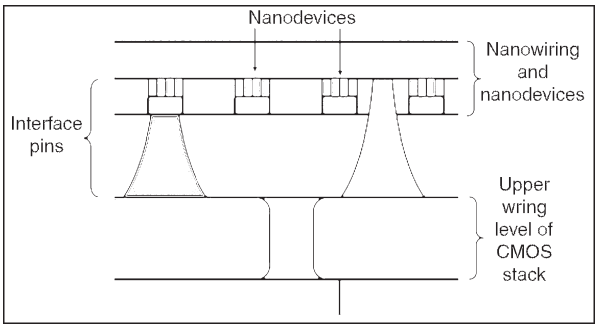


Fig. Structure of a CMOL Circuit

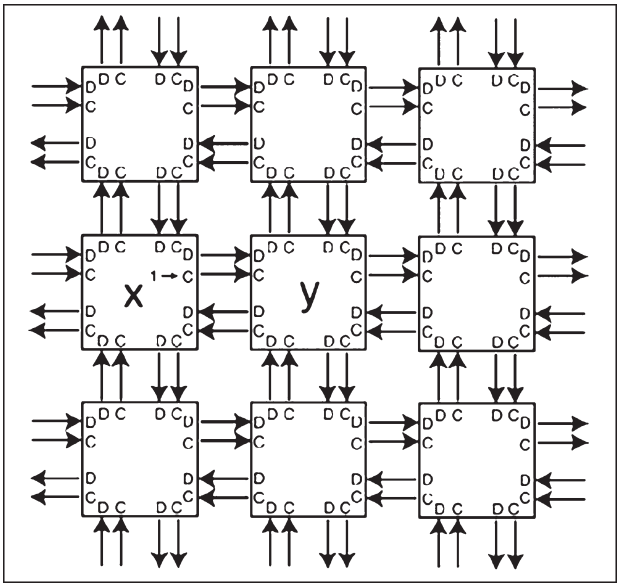


Fig. 3 x 3 Cell Matrix

6

Nanocomputers

While the first, primitive assemblers were controlled by changing what molecules are in the solution around the device, getting the speed and accuracy wanted for large-scale manufacturing takes real computation. Carl's setup uses a combination of special-purpose molecule processors and general-purpose assemblers, all controlled and orchestrated by nanocomputers. Computers back in the 1990s used microelectronics. They worked by moving electrical charge back and forth through conducting paths—wires, in effect—using it to block and unblock the flow of charge in other paths. With nanotechnology, computers are built from molecular electronics. Like the computers of the 1990s, they use electronic signals to weave the patterns of digital logic.

Being made of molecular components, though, they are built on a much smaller scale than 1990s computers, and work much faster and more efficiently. On the scale of our

simulated molecular world, 1990s computer chips are like landscapes, while nanocomputers are like individual buildings. Carl's desktop PC contains over a trillion nanocomputers, enough to out-compute all the microelectronic computers of the twentieth century put together. Back in the dark ages of the 1980s, an exploratory engineer proposed that nanocomputers could be mechanical, using sliding rods instead of moving electrons as shown in Figure. These molecular mechanical computers were much easier to design than molecular electronic computers would have been. They were a big help in getting some idea of what nanotechnology could do.

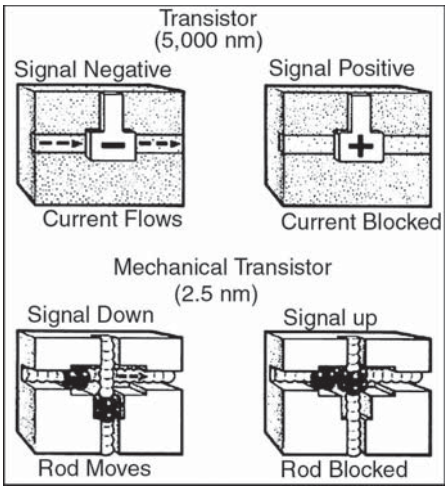


Fig. Mechanical Transistor

An electronic transistor lets current flow when a negative electric charge is applied and blocks current when a positive charge is applied. The mechanical "transistor" lets the horizontal rod move when the vertical rod is down, and blocks the horizontal rod when the vertical rod is up. Either device can be used to build logic gates and computers. Even back then, it was pretty obvious that mechanical computers would be slower than electronic computers.

Carl's molecular electronic PC would have been no great surprise, though nobody knew just how to design one. When nanotechnology actually arrived and people started competing to build the best possible computers, molecular electronics won the technology race. Still, mechanical nanocomput-erscould have done all the nanocomputing jobs at Desert Rose: ordinary, everyday molecular manufacturing just doesn't demand the last word in computer performance.

For Carl, the millions of nanocomputers in the milky waters of his building ponds are just extensions of machines on his desk, machines there to help him run his business and deliver products to his customers—or, in the case of the Red Cross emergency, to help provide time-critical emergency supplies. By reserving those three separate ponds, Carl can either build three different kinds of equipment for the Red Cross or use all the ponds to mass-produce the first thing on the Red Cross list: emergency shelters for ten thousand people. The software is ready, the plumbing is fine, the drums of building materials are all topped up, the Special Mix for this job is loaded: the build is ready to start. "Okay," Carl tells the computer, "build Red Cross tents." Computer talks to nanocomputers. In all three pools, nanocomputers talk to assemblers. The build begins.

ASSEMBLING PRODUCTS

Some of the building done at Desert Rose Industries uses assemblers much like the ones we saw in the first hall of the plant tour, back in the simulated molecular world of the Silicon Valley Faire. As seen in simulation, they are big, slow, computer-controlled things moving molecular tools. With the

right instructions and machinery to keep them supplied with molecular tools, these general-purpose assemblers can build almost anything. They're slow, though, and take a lot of energy to run. Some of the building uses special-purpose assembly systems in the molecule-processing style, like the systems in the basement we saw in the tour of a simulated molecular factory. The special-purpose systems are all moving belts and rollers, but no arms. This is faster and more efficient, but for quantity orders, cooling requirements limit the speed.

It's faster to use larger, prefabricated building blocks. Desert Rose uses these for most of their work, and especially for rush orders like the one Carl just set up. Their underground warehouse has room-sized bins containing upward of a thousand tons of the most popular building blocks, things like structural fibres.

They're made at plants on the West Coast and shipped here by subway for ready use. Other kinds are made on site using the special-purpose assemblers. Carl's main room has several cabinet-sized boxes hooked up to the plumbing, each taking in raw materials, running them through this sort of specialized molecular machinery, and pumping out a milky syrup of product. One syrup contains motors, another one contains computers, and another is full of microscopic plug-in light sources. All go into tanks for later use.

Now they're being used. The mix for the Red Cross tent job is mostly structural fibre stronger than the old bulletproof-vest materials. Other building blocks also go in, including motors, computers, and dozens of little struts, angle brackets, and doohickies. The mix would look like someone had stirred

together the parts from a dozen toy sets, if the parts were big enough to see. In fact, though, the largest parts would be no more than blurry dots, if you saw one under a normal optical microscope.

The mix also contains block-assemblers, floating free like everything else. These machines are big, about like an office building in our simulation view with the standard settings. Each has several jointed arms, a computer, and several plugs and sockets. These do the actual construction work.

To begin the build, pumps pour the mix into a manufacturing pond. The constant tumbling motions of microscopic things in liquids would be too disorganized for building anything so large as a tent, so the block-assemblers start grabbing their neighbors. Within moments, they have linked up to form a framework spread through the liquid. Now that they are plugged together, they divide up jobs, and get to work. Instructions pour in from Carl's desktop computer. The block-assemblers use sticky grippers to pull specific kinds of building blocks out of the liquid.

They use their arms to plug them together. For a permanent job, they would be using blocks that bond together chemically and permanently. For these temporary tents, though, the Red Cross design uses a set of standard blocks that are put together with amazingly ordinary fasteners: these blocks have snaps, plugs, and screws, though of course the parts are atomically perfect and the threads on the screws are single helical rows of atoms.

The resulting joints weaken the tent's structure somewhat, but who cares? The basic materials are almost a hundred times stronger than steel, so there is strength to waste if it

makes manufacturing more convenient. Fibre segments snap together to make fabrics. Some segments contain motors and computers, linked by fibres that contain power and data cables. Struts snap together with more motors and computers to make the tent's main structures. Special surfaces are made of special building blocks.

From the human perspective, each tent is a lightweight structure that contains most of the conveniences and comforts of an apartment: cooking facilities, a bathroom, beds, windows, air conditioning, specially modified to meet the environmental demands of the quake-stricken country. From a builder's perspective, especially from a nanomachine's point of view, the tent is just structure slapped together from a few hundred kinds of prefab parts.

In a matter of seconds, each block-assembler has put together a few thousand parts, and its section of the tent is done. In fact, the whole thing is done: many trillions of hands make light work. A crane swings out over the pond and starts plucking out tent packages as fresh mix flows in. Maria's concern has drawn her back to the plant to see how the build is going. "It's coming along," Carl reassures her. "Look, the first batch of tents is out." In the warehouse, the first pallet is already stacked with five layers of dove-gray "suitcases": tents dried and packed for transport. Carl grabs a tent by the handle and lugs it out the door. He pushes a tab on the corner labeled "Open," and it takes over a minute to unfold to a structure a half-dozen paces on a side. The tent is big, and light enough to blow away if it didn't cling to the ground so tightly. Maria and Carl tour the tent, testing the appliances, checking the construction of furniture:

everything is extremely lightweight compared to the bulk-manufactured goods of the 1990s, tough but almost hollow.

Like the other structures, the walls and floors are full of tiny motors and struts controlled by simple computers like the ones used in twentieth-century cars, televisions, and pinball machines. They can unfold and refold. They can also flex to produce sound like a high-quality speaker, or to absorb sound to silence outdoors racket. The whole three-room setup is small and efficient, looking like a cross between a boat cabin and a Japanese business hotel room. Outside, though, it is little more than a box. Maria shakes her head, knowing full well what architects can do these days when they try to make a building really fit its site. Oh well, she thinks, These won't be used for long.

"Well, that looks pretty good to me," says Carl with satisfaction. "And I think we'll be finished in another hour." Maria is relieved. "I'm glad you had those pools freed up so fast." By three o'clock, they've shipped three thousand emergency shelters, sending them by subway. Within half an hour, tents are being set up at the disaster site.

BEHIND THE SCENES AND AFTERWARD

Desert Rose Industries and other manufacturers can make almost anything quickly and at low cost. That includes the tunneling machines and other equipment that made the subway system they use for shipping. Digging a tunnel from coast to coast now costs less than digging a single block under New York City used to. It wasn't expensive to get a deep-transit terminal installed in their basement. Just as the tents aren't mere bundles of canvas, these subways aren't

slow things full of screeching, jolting metal boxes. They're magnetically levitated to reach aircraft speeds—as experimental Japanese trains were in the late 1980s—making it easy for Carl and Maria to give their customers quick service. There's still a road leading to the plant, but nobody's driven a truck over it for years.

They only take in materials that they will eventually ship out in products, so there's nothing left over, and no wastes to dump. One corner of the plant is full of recycling equipment. There are always some obsolete parts to get rid of, or things that have been damaged and need to be reworked. These get broken down into simpler molecules and put back together again to make new parts.

The gunk in the manufacturing ponds is water mixed with particles much finer than silt. The particles—fasteners, computers, and the rest—stay in suspension because they are wrapped in molecular jackets that keep them there. This uses the same principle as detergent molecules, which coat particles of oily dirt to float them away.

Though it wouldn't be nutritious or appetizing, you could drink the tent mix and be no worse for it. To your body, the parts and their jackets, and even the nanomachines, would be like so many bits of grit and sawdust. Carl and Maria get their power from solar cells in the road, which is the only reason they bothered having it paved. In back of their plant stands what looks like a fat smokestack. All it produces, though, is an updraft of clean, warm air.

The darkly paved road, baking in the New Mexico sun, is cooler than you might expect: it soaks up solar energy and makes electricity, instead of just heat. Once the power is

used, it turns back into heat, which has to go somewhere. So the heat rises from their cooling tower instead of the road, and the energy does useful work on the way. Some products, like rocket engines, are made more slowly and in a single piece. This makes them stronger and more permanent. The tents, though, don't need to be superstrong and are just for temporary use.

A few days after the tents go up, the earthquake victims start to move out into new housing (permanent, better-looking, and very earthquake resistant). The tents get folded and shipped off for recycling. Recycling things built this way is simple and efficient: nanomachines just unscrew and unsnap the connectors and sort the parts into bins again. The shipments Desert Rose gets are mostly recycled to begin with. There's no special labeling for recycled materials, because the molecular parts are the same either way.

For convenience (and to keep the plant small), Carl and Maria get most of their parts prefabricated, even though they can make almost anything. They can even make more production equipment. In one of their manufacturing ponds, they can put together a new cabinet full of special-purpose assemblers. They do this when they want to make a new type of part in-house.

Like parts, the part-assemblers are made by special-purpose assemblers. Carl can even make big vats in medium-size vats, unfolding them like tents. If Desert Rose Industries needed to double capacity, Carl and Maria could do it in just a few days. They did this once for a special order of stadium sections. Maria got Carl to recycle the new building before its shadow hurt their cactus garden.

FACTORY

In the Desert Rose Industries scenario, manufacturing has become cheap, fast, clean, and efficient. Using fast, precise machines to handle matter in molecular pieces makes it easy for nanotechnology to be fast, clean, and efficient. But for it to be cheap, the manufacturing equipment has to be cheap.

The Desert Rose scenario shows how this can work. Molecular-manufacturing equipment can be used to make all the parts needed to build more molecular manufacturing equipment. It can even build the machines needed to put the parts together. This resembles an idea developed by NASA for a self-expanding manufacturing complex on the Moon, but made faster and simpler using molecular machines and parts.

Replicators

In the early days of nanotechnology, there won't be as many different kinds of machines as there are at Desert Rose. One way to build a lot of molecular manufacturing equipment in a reasonable time would be to make a machine that can be used to make a copy of itself, starting with special but simple chemicals.

A machine able to do this is called a "replicator." With a replicator and a pot full of the right fuel and raw materials, you could start with one machine, then have two, four, eight, and so on. This doubling process soon makes enough machines to be useful.

The replicators—each including a computer to control it and a general-purpose assembler to build things—could then be used to make something else, like the tons of specialized machines needed to set up a Desert Rose manufacturing

plant. At that point, the replicators could be discarded in favour of those more efficient machines. Replicators are worth a closer look, though, because they show how quickly molecular manufacturing systems can be used to build more manufacturing equipment.

Figure shows a design described in Stanford University course CS 404 in the spring of 1988. If we were in one of our standard simulation views, the submicroscopic device at the top of the picture would be like a huge tank, three stories tall when lying on its side.

Most of its interior is taken up by a tape memory system that tells how to move the arm to build all the parts of the replicator, except the tape itself. The tape gets made by a special tape-copying machine. At the right-hand end of the replicator are pores for bringing in fuel and raw-material molecules, and machinery for processing them. In the middle are computer-controlled arms, like the ones we saw on the plant trip. These do most of the actual construction.

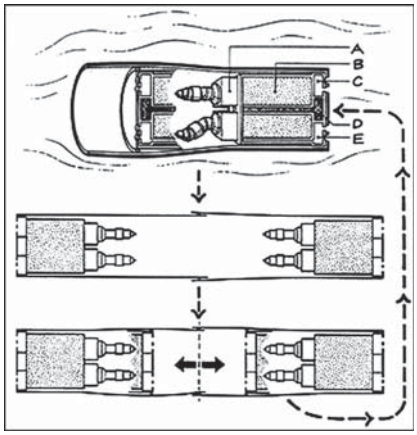


Fig. Replicator

A replicator would be able to build copies of itself when supplied with fuel and raw materials. In the diagram,

(A) Contains a nanocomputer,
(B) A library of stored instructions,
(C) Contains machinery that takes in fuel and produces electric power,
(D) Is a motor, and
(E) Contains machinery that prepares raw materials for use.

(All volumes follow calculations presented in a class at Stanford.) The lower diagrams illustrate steps in a replication cycle, showing how the working space is kept isolated from the external liquid, which provides the needed fuel and raw-material molecules. Replicators of this sort are useful as thought experiments to show how nanomachines can product more nanomachines, but specialized manufacturing equipment would be more efficient in practice.

The steps in the cycle—using a copy to block the tube, beginning a fresh copy, then releasing the old one—illustrate one way for a machine to build a copy of itself while floating in a liquid, yet doing all its construction work inside, in vacuum. (It's easier to design for vacuum, and this is exploratory-engineering work, so easier design is better design.) Calculations suggest that the whole construction cycle can be completed in less than a quarter hour, since the replicator contains about a billion atoms, and each arm can handle about a million atoms per second. At that rate, one device can double and double again to make trillions in about ten hours.

Each replicator just sits in a chemical bath, soaking up what it needs and making more replicators. Eventually, either the special chemicals run out or other chemicals are added

to signal them to do something else. At that point, they can be reprogrammed to produce anything else you please, so long as it can be extruded from the front. The products can be long, and can unfold or be pieced together to make larger objects, so the size of these initial replicators—smaller than a bacterium—would be only a temporary limitation.

General Assemblers

From the molecular manipulators and primitive assemblers described, the most likely path to nanotechnology leads to assemblers with more and more general capabilities. Still, efficiency favours special-purpose machines, and the Desert Rose scenario didn't make much use of general assemblers. Why bother making general-purpose assemblers in the first place?

To see the answer, turn the question around and ask, Why not build such a tool? There is nothing outstandingly difficult about a general assembler, as molecular machinery goes. It will just be a device with good, flexible positional control and a system to feed it a variety of molecular tools. This is a useful, basic capability. General-purpose assemblers could always be replaced by a lot of specialized devices, but to build those specialized devices in the first place, it makes sense to come up with a more flexible, general-purpose system that can just be reprogrammed.

So, general purpose machines are likely to find use in making short production runs of more specialized devices. Ralph Merkle, a computers and security expert at Xerox Palo Alto Research Centre, sees this as paralleling the way manufacturing works today: "General purpose devices could do many tasks, but they'll do them inefficiently. For any given

task, there will be one or a few best ways of doing it, and one or a few special purpose devices that are finely tuned to do that one task.

Nails aren't made by a general-purpose machine shop, they're made by nail-making machines. Making nails with a general-purpose machine shop would be more expensive, more difficult, and more time-consuming. Likewise, in the future we won't see a proliferation of general-purpose self-replicating systems, we'll see specialization for almost every task."

WHAT WILL THESE CAPABILITIES MAKE POSSIBLE

We've surveyed a lot of devices: assemblers of various flavours, nanocomputers, disassemblers, replicators, and others. What's important about these is not the exact distinctions between them, but the capabilities that they will give and the effects they will have on human lives. Again, we are suspending discussion of potential misapplications until later. If we tease apart the implications of what we've seen in the Desert Rose scenario, we can analyse some of the key impacts of molecular manufacturing in industry, science, and medicine.

Technology and Industry

At its base, nanotechnology is about molecular manufacturing, and manufacturing is the basis of much of today's industry. This is why Desert Rose made a good starting point for describing the possibilities of a nanotechnological world. From an industrial perspective, it makes sense to think of nanotechnology in terms of products and production.

New Products

Today, we handle matter crudely, but nanotechnology will bring thorough control of the structure of matter, the ability to build objects to atom-by-atom specifications. This means being able to make almost anything. By comparison, even today's range of products will feel very limited. Nanotechnology will make possible a huge range of new products, a range we can't envision today. Still, to get a feel for what is possible, we can look at some easily imagined applications.

Reliable Products

Today, products often fail, but for failures to occur-for a wing to fall off an airplane, or a bearing to wear out-a lot of atoms have to be out of place. In the future, we can do better. There are two basic reasons for this: better materials and better quality control, both achieved by molecular manufacturing. By using materials tens of times stronger than steel, as Desert Rose did, it will be easy to make things that are very strong, with a huge safety margin. By building things with atom-by-atom control, flaws can be made very rare and extremely small-nonexistent, by present standards. With nanotechnology, we can design in big safety margins and then manufacture the design with near-perfection. The result will be products that are tough and reliable. (There will still be room for bad designs, and for people who wish to take risks in machines that balance on the edge of disaster.)

Intelligent Products

Today, we make most things from big chunks of metal, wood, plastic, and the like, or from tangles of fibres. Objects

made with molecular manufacturing can contain trillions of microscopic motors and computers, forming parts that work together to do something useful. A climber's rope can be made of fibres that slide around and reweave to eliminate frayed spots. Tents can be made of parts that slide and lock to turn a package into a building. Walls and furniture can be made to repair themselves, instead of passively deteriorating. On a mundane level, this sort of flexibility will increase reliability and durability. Beyond this, it will make possible new products with abilities we never imagined we needed so badly. And beyond even this, it will open new possibilities for art.

Inexpensive Production

Today, production requires a lot of labour, either for making things or for building and maintaining machines that make things. Labour is expensive, and expensive machines make automation expensive, too. In the Desert Rose scenario, we got a glimpse of how molecular manufacturing can make production far less expensive than it is today. This is perhaps the most surprising conclusion about nanotechnology, so we'll take a closer look.

Clean Production

Today, our manufacturing processes handle matter sloppily, producing pollution. One step puts stuff where it shouldn't be; the next washes it off the product and into the water supply. Our transportation system worsens the problem as unreliable trucks and tankers spill noxious chemicals over the land and sea. Everything is expensive, so companies skimp on even the half-effective pollution controls

that we know how to build. Nanotechnology will mean greater control of matter, making it easy to avoid pollution.

This means that a little public pressure will go a long way towards a cleaner environment. Likewise, it will make it easy to increase efficiency and reduce resource requirements. Products, like the Red Cross tents at Desert Rose, can be made of snap-together, easily recyclable parts. Sophisticated products could even be made from biodegradable materials. Nanotechnology will make it easy to attack the causes of pollution at their technological root.

Nanotechnology will have great applications in the field of industry, much as transistors had great applications in the field of vacuum tube electronics, and democracy had great applications in the field of monarchy. It will not so much advance twentieth-century industry as replace it—not all at once, but during a thin slice of historical time.

Science

Chemistry

Today, chemists work with huge number of molecules and study them using clever, indirect techniques. Making a new molecule can be a major project, and studying it can be another. Molecular manufacturing will help chemists make what they want to study, and it will help them make the tools they need to study it. Nanoinstruments will be used to prod, measure, and modify molecules in a host of ways, studying their structures, behaviours, and interactions.

Materials: Today, materials scientists make new superconductors, semiconductors, and structural materials by mixing and crushing and baking and freezing, and so

forth. They dream of far more structures than they can make, and they stumble across more things than they plan. With molecular manufacturing, materials science can be much more systematic and thorough. New ideas can be tested because new materials can be built according to plan (rather than playing around, groping for a recipe). This need not rule out unexpected discoveries, since experiments—even blind searches—will go much faster. A few tons of raw materials would be enough to make a billion samples, each a cubic micron in size. In all of history so far, materials scientists have never tested so many materials. With nanoinstruments and nanocomputers, they could. One laboratory could then do more than all of today's materials scientists put together.

Biology: Today, biologists use a host of molecular devices borrowed from biology to study biology. Many of these can be viewed as molecular machines. Nanotechnology will greatly advance biology by providing better molecular devices, better nanoinstruments. Some cells have already been mapped in amazing molecular detail, but biology still has far to go. With nanoinstruments (including molecule-by-molecule disassemblers), biologists will at last be able to map cells completely and study their interactions in detail. It will become easy not only to find molecules in cells, but to learn what they do. This will help in understanding disease and the molecular requirements for health, enormously advancing medicine.

Computation

Today, computers range from a million to a billion times faster than an old desktop adding machine, and the results

have been revolutionary for science. Every year, more questions can be answered by calculations based on known principles of physics. The advent of nanocomputers—even slow, miserable, mechanical nanocomputers—will give us practical machines with a trillion times the power of today's computers (essentially by letting us package a trillion computers in a small space, without gobbling too much money or energy.) The consequences will again be revolutionary.

Physics

The known principles of physics are adequate for understanding molecules, materials, and cells, but not for understanding phenomena on a scale that would still be submicroscopic if atoms were the size of marbles. Nanotechnology can't help here directly, but it can provide manufacturing facilities that will make huge particle accelerators economical, where today they strain national budgets. More generally, nanotechnology will help science wherever precision and fine details are important. Science frequently proceeds by trying small variations in almost identical experiments, comparing the results.

This will be easier when molecular manufacturing can make two objects that are identical, molecule by molecule. In some areas, today's techniques are not only crude, but destructive. Archaeological sites are unique records of the human past, but today's techniques throw away most information during the dig, by accident. Future archaeologists, able to sift soil not speck by speck but molecule by molecule, will be grateful indeed to those archaeologists who today leave some ground undisturbed.

Medicine

Of all the areas where the ability to manufacture new tools is important, medicine is perhaps the greatest. The human body is intricate, and that intricacy extends beyond the range of human vision, beyond microscopic imaging, down to the molecular scale. "Molecular medicine" is an increasingly popular term today, but medicine today has only the simplest of molecular tools. As biology uses nanoinstruments to learn about disease and health, we will learn the physical requirements for restoring and maintaining health. And with this knowledge will come the tools needed to satisfy those requirements-tools ranging from improved pharmaceuticals to devices able to repair cells and tissues through molecular surgery.

Advanced medicine will be among the most complex and difficult applications of nanotechnology. It will require great knowledge, but nanoinstruments will help gather this knowledge. It will pose great engineering challenges, but computers of trillionfold greater power will help meet those challenges. It will solve medical problems on which we spend billions of dollars today, in hopes of modest improvements. Today, modern medicine often means an expensive way to prolong misery. Will nanomedicine be more of the same?

Any reader over the age of, say, thirty knows how things start to go wrong: an ache here, a wrinkle there, the loss of an ability. Over the decades, the physical quality of life declines faster and faster-the limits of what the body can do become stricter-until the limits are those of a hospital bed. The healing abilities we have when young seem to fade away. Modern medical practice expends the bulk of its effort on

such things as intensive care units, dragging out the last few years of life without restoring health.

Truly advanced medicine will be able to restore and supplement the youthful ability to heal. Its cost will depend on the cost of producing things more intricate than any we have seen before, the cost of producing computers, sensors, and the like by the trillions. To understand the prospects for medicine, like those for science and industry, we need to take a closer look at the cost of molecular manufacturing.