Pure Appl. Chem., Vol. 72, Nos. 1–2, pp. 11–20, 2000.
© 2000 IUPAC

Wires, switches, and wiring. A route toward a chemically assembled electronic nanocomputer*

James R. Heath†

Department of Chemistry and Biochemistry, UCLA, 405 Hilgard Ave., Los Angeles, CA 90095-1569 USA

Abstract: A Boolean logic, nonreversible computing machine should, in principle, be capable of 10^{18} bit operations per second at a power consumption of 1 W. In order to build such a machine that can even approach this benchmark for efficiency, the development of a robust quantum-state switch capable of ambient operation, as well as a bottom–up manufacturing technology, will be necessary. My group, in collaboration with Hewlett Packard, has developed much of the architecture for such a machine, which we call a chemically assembled electronic nanocomputer (CAEN). More recently, in a collaborative effort with Fraser Stoddart's group at UCLA, we have begun to build it. The fundamental unit of the machine is a field-programmable molecular switch, and the fundamental architecture is a hierarchical organization of wire/switch lattices called crossbars. Electronically, singly configurable molecular-based switch devices based on rotaxane molecular compounds have been fabricated in high yield. These switches were used to construct simple molecular-based logic structures and read-only memory elements.

INTRODUCTION

In the early 1960s Rolf Landauer began to consider the physics of information manipulation [1]. All modern computers are Boolean logic, nonreversible machines. This implies that the states '1' and '0' need to be energetically distinct from one another. Furthermore, in a nonreversible machine, information is lost. Consider, for example, a simple AND logic gate. That gate has two inputs and one output. The state of the output depends, of course, on the collective states of the inputs, so that if both inputs are high, then the output is high. However, if any or both of the inputs is low, then the output is low. Let's say that we do a simple calculation with an AND gate, and we produce a '0' on the output. Any one of three different input states could have given us this output, and we can't tell which one. Thus, the 'nonreversible' term used to describe the computing machine refers to the fact that information is lost during the calculation. In another example, say that we carry out the following addition on our calculation: 5 + 5. The calculator generates the number '10', and now we show that number to a colleague. Our colleague can't tell if your question was 5 + 5 = ?, or 2×5 = ?, or 3 + 7 = ?, etc. Thus, in this type of machine, the question is lost, and that has entropic consequences. Landauer demonstrated that the minimum energy required to carry out a single bit operation, considering both entropic and enthalpic costs, was $k_B T\ln 2$ [1]. This minimum energy, however, requires that the bit manipulation takes place very slowly. As any chemist knows, the faster a chemical reaction takes place, the larger the ΔG of reaction. This is also true for information manipulation, as both processes must obey the laws of

Pure Appl. Chem.* **72, 1–331 (2000). An issue of reviews and research papers based on lectures presented at the 1st IUPAC Workshop on Advanced Materials (WAM1), Hong Kong, July 1999, on the theme of nanostructured systems.
†E-mail: heath@chem.ucla.edu

thermodynamics. Feynman considered the bit operations to take place in a finite amount of time (at a particular clock frequency), and he demonstrated that the power consumption of this (slightly more realistic) process was $P = nk_BTd\nu^2c^{-1}$ [2]. In this equation, P is the power consumption (in watts), n is the number of parallel operations, ν is the clock speed, c is the speed of light, and d is a transmission distance. It is not clear just what the appropriate metric for optimizing a machine is, but it is reasonably straightforward to demonstrate that 10^{18} bit operations/second at a power consumption of 1 W is not unrealistic for a massively parallel machine. Feynman's equation implies several things. First, note that the power consumption increases linearly with the number of parallel operations, but quadratically with the clock speed! Thus, parallel computers are much more energy-efficient than serial machines. Second, d should be small, and so an energy-optimized computer is going to be characterized by small (nanometer) length scales. Finally, if the energy difference between a '0' and a '1' is k_BT times some factor, then a quantum-state switch is required to achieve such efficiency. In the talk preceding mine, Stan Williams discussed the difficulties involved in manufacturing such a machine with current microelectronics technology. First, complementary metal oxide semiconductor (CMOS) devices are not quantum-state switches, but instead are solid-state switches. Second, true quantum-state switch operation under ambient temperature requires length scales below 10 nm. The costs associated with bringing the CMOS (top–down) manufacturing technology down this length scale are truly unrealistic! This means that if one is to build a machine that operates near the thermodynamic limit for computation, then that machine is going to use a bottom–up manufacturing approach (chemistry), and the best quantum-state switches obtainable via chemical routes (or perhaps any other routes, for that matter) are molecules. In fact, molecules are quite natural quantum-state switches.

At first glance, chemical manufacturing and microprocesser fabrication appear to be incommensurate with one another, for at least three reasons.

- First, modern microprocessors are perfect and complicated systems, whereas anything that is chemically fabricated tends to be either crystalline (or amorphous), and is characterized by imperfections due to reaction yields, lattice defects, etc. Thus, one challenge for developing a bottom-up manufacturing strategy is to find a way to extract perfect complexity from defective order.
- Second, any machine (even a human brain) has a very finite number of inputs (the keyboard, for example) and outputs (the screen), but a much larger number of computational resources (100 Mbytes RAM, Pentium Pro II, etc.). In a computing machine, the few inputs/outputs are connected to the resources through structures known as multiplexers and demultiplexers. The analogous structures for a chemically assembled machine will have to reach down from the micron or millimeter length scales into the nanometer or Angstrom length scales. An architecture for such a multiplexer has not yet been developed, and will not be discussed here.
- Third, any chemically assembled machine is likely to have a vast amount of resources. How does one take advantage of so many resources, and do so in an efficient manner? In a modern CMOS-based machine, the resources are laid out according to the following: Engineers know that, on a single cm^2 of Si wafer, the current lithography generation will enable them to build a certain number of wires and switches, amplifiers, etc. Those resources need to be arranged in such a fashion that they will be efficiently utilized for a general or specific purpose computing machine. Thus, an algorithm is written that can take all of these resources and attempt to find a ground-state solution for arranging them together. This algorithm is run several times, until everyone is satisfied that a reasonably optimal solution has been found. Here's the catch. This problem is known as an 'NP-hard' problem, which means that the difficulty in finding a solution scales exponentially with the number of resources. The more familiar 'traveling salesman' problem is similar. Thus, trying to find an optimal solution for arranging 10^{23} resources will take longer than the age of the universe! This means that an architecture must be found that makes solving this problem scale linearly with the numbers of resources, rather than exponentially.

THE ARCHITECTURE

The set of problems outlined above appears to be fairly significant, and we haven't yet decided on whether we need a two- or three-dimensional architecture, or even what the quantum-state switches will be! I will discuss the issue of dimensionality first, since all other solutions will hinge on that. In the early 1960s, Richard Rent of IBM developed a phenomenological rule known today Rent's rule [3]. A simple Rent's rule calculation is shown in Fig. 1. The point of Rent's rule is that within any portion of any given circuit there are a certain numbers of degrees of freedom. In order to take advantage of all those degrees of freedom, then a given fraction of the wires need to be devoted to input/output (I/O). In a perfectly designed system (that may do nothing interesting other than be perfectly designed), Rent's rule says that for n degrees of freedom, there must be $2n^{1/2}$ I/O wires. Here, the ½ exponent is called the Rent's rule exponent. When human design is taken into account, the Rent's rule exponent becomes something closer to 0.6 or 0.65. If the circuit has any defects, then additional wires must be added to route around those defects, and the Rent's rule exponent becomes 0.7 or higher. This is for a 2D circuit. For a 3D circuit, perfect design dictates a Rent's rule exponent of 2/3, and human design, defect, etc. will make that number 0.75–0.8 or so. The problem with a 3D circuit is that the surface area of a volume only scales, in the best case, to the 2/3 power of the volume, and so there is just enough surface area to utilize all of the resources. Once human design, defects, etc. are taken into account, there are not enough wires. This implies that either the system must be utilized in a *very* inefficient manner (remember, the scaling is in the exponent), or that the efficient use of 3D architectures is just not possible for a Boolean logic, reversible machine. Thus, the issue of 3D versus 2D was already settled by Richard Rent 40 years ago! We need a 2D architecture for our fundamental circuits. It turns out that it is possible to network in 3D, although not in a manner that is space-filling [4]. However, the issue of networking is beyond the scope of this paper.

Now we come back to the issue of chemical assembly. We already stated that any chemically assembled system is likely to be crystalline and defective. The fact that it is crystalline, and that we are going to have a 2D architecture, implies that we need an architectural solution that *tiles* in 2D. A few

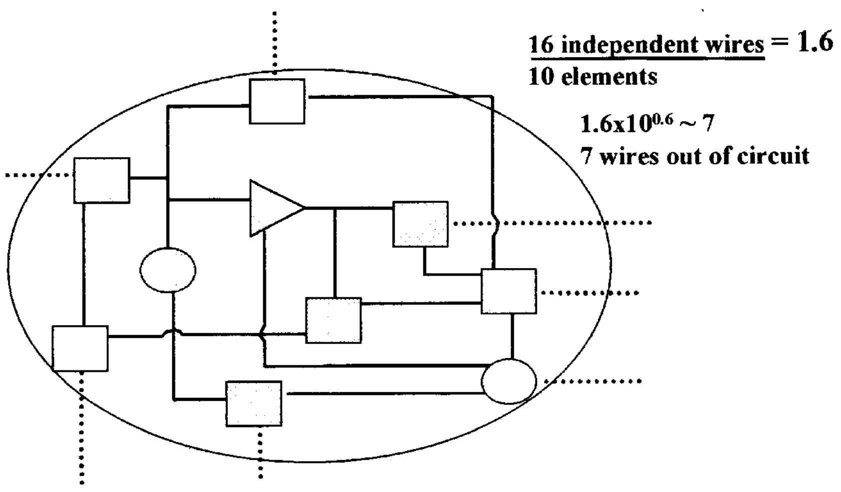

$\frac{\text{16 independent wires}}{\text{10 elements}} = 1.6$
$1.6\text{x}10^{0.6} \sim 7$
7 wires out of circuit

Fig. 1 A graphical depiction of Rent's rule. This drawing represents a circuit containing logic gates (squares, circles, and triangles), and wires. There are 10 gates and 16 wires, including the input and output wires. Note that not all of the wires are independent, and only the independent ones are counted. As shown in the figure, this circuit has a Rent's rule exponent of approximately 0.7.

years ago, I, along with two collaborators from Hewlett Packard Labs, Stan Williams and Phil Kuekes, began trying to reconcile the issues of chemistry with computer manufacturing, and we used a machine known as Teramac as a model for our discussion. Teramac is a massively parallel, custom-configurable supercomputer built at HP in the early 90s by Kuekes, Greg Snider, and others [5]. Teramac also was an extremely defective computer—it contained nearly a quarter million hardware defects, any one of which would completely disable a Pentium processor. Nevertheless, when Teramac was running, it was capable of 10^{12} logical operations/second, and it only had a clock speed of 1 MHz!

Teramac turned out to be very important in developing our thoughts for how to build a CAEN, and so it is worth spending some time discussing this machine. In Teramac, the fundamental device was a *configurable bit*, properly known as a field-programmable gate array (FPGA) [6]. The fundamental circuit was a square lattice of FPGAs, known as a *crossbar*, and the fundamental architecture was known as a *fat tree* [7]. It turns out that the concept of a crossbar of configurable bits actually solves the problem of how to deal with a defective manufacturing technology that produces only crystalline structures. The concept of the fat tree makes the problem of utilizing a vast number of resources one that scales roughly linearly with the numbers of devices, rather than exponentially [8]. Fat trees are extensions of earlier concepts, one example of which are known as Banyan networks [9].

Teramac was a chameleon of a computer. As built, it was nothing but wires and switches, with no ability to compute. Kuekes and his colleagues knew ahead of time that their machine would be riddled with defects, and so they designed the machine with this in mind, meaning that they designed circuits with Rent's rule exponents of 0.7 and higher. They wrote algorithms that sent electrical signals throughout the entire machine, similar to how some computer viruses work. Those algorithms established two data sets: the good resources and the defective ones. Once these data bases were in hand, it was possible to devise a way to efficiently utilize the good resources in order to build a computer. Remember, all they had were resources, but they had a lot of them, and so they could imagine using these resources to fabricate virtually any known type of computer. Rather than physically moving the good resources around on the circuit boards to assemble the computer, the entire assembly was done through software. Thus, the logical structure of the machine was downloaded upon the good hardware resources. This 'building' of the computer took a lot of time and was therefore expensive. However, it was done by a second computer that served as a tutor to Teramac. The software, coupled with the high Rent's rule exponent used in the circuit design, allowed for defective resources to be routed around [10].

Configuring the resources of Teramac into a computing machine was made vastly easier because of the fat tree architecture. A fat tree is a hierarchical arrangement of the resources, similar to a family tree in which kids, parents, grandparents, etc. are all at different levels. The difference between a fat tree and a family tree is that the fat tree has a breadth, not just a width. One example of a fat tree is the system of roads, boulevards, and freeways in Los Angeles. An example of how a fat tree is used is shown in Fig. 2. In this analogy, there are thin lines (neighborhood streets), thicker solid lines (boulevards), and even thicker triple striped lines (freeways). In a fat tree, all of the neighborhood streets are connected to each other and to the boulevards that intersect the neighborhood. All of the boulevards are connected to each other and to the intersecting freeways, and so on. Finding a pathway from one house to another house in the same neighborhood (short-distance communication) is relatively easy, and all pathways are relatively the same. Getting from neighborhood to neighborhood (intermediate distance communication) is relatively easy through the use of boulevards. Finally, getting from one part of the city to another (long-distance communication) is also relatively easy through the use of freeways. Many choices are nearly equivalent. It is not critical to find the shortest distance route, as a solution reasonably close to the 'ground-state' solution can always be found. Teramac had a 5-level fat tree, and any CAEN is likely to need a dozen or so levels to deal with the vast numbers of resources. Note that, without a map, finding an efficient route between the two points of Fig. 3 would be nontrivial. It is even possible that, given an order of magnitude more time to find a solution (but no map), no good solution would be found. This

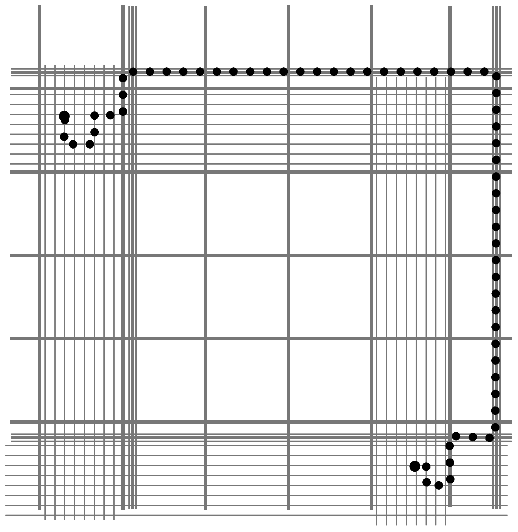


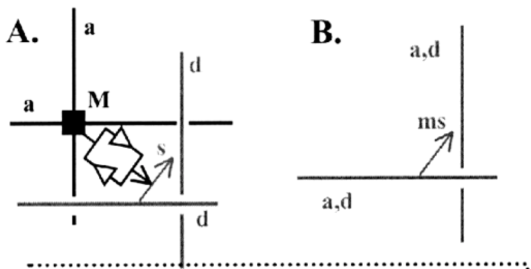
Fig. 2 A fat tree architecture modeled after a street map of an ideal town. Only a few of the 'minor roads' (thin grey lines) are shown, while all of the major boulevards (thick grey lines) and all of the freeways (triple grey lines) are shown. The object is to find a good way to connect the two dots, and one possible pathway (black dotted lines) is shown. The assumption that the freeways are better for long distance communication has been made, and the boulevards are better for intermediate distance communication. Note that many solutions are possible, but it is relatively easy to a nearly optimal one.

illustrates the usefulness of retaining, but not building in, the 3rd dimension. It is only upon looking down onto these resources from the 3rd dimension that one finds an efficient pathway between any two points. *If this map had contained twice the numbers of resources, it would have only taken approximately twice as long to find an optimal pathway between two points.* Such an experiment was run on the Teramac resources. Thus, the fat tree goes far toward solving the issue of scaling.

Note several things about the fat tree of Fig. 2. First, broken streets (wires) or disconnected intersections (switches) can be dealt with. There are always alternate routes that can be found which avoid defective components. In fact, the efficiency of a fat tree architecture probably does not become severely hampered until somewhere near the percolation threshold. Second, this is an architecture that appears crystalline (albeit with differing length scales for streets, boulevards, freeways, etc.), and it does tile in two dimensions [11].

THE HARDWARE

We have taken a stepwise approach toward building a CAEN. As a first attempt, we focused on building a single 'neighborhood' of switches and wires, using lithographically defined wires, but molecular-based switches. Since the switches are the new component here, I will discuss them in some detail. As stated above, the Teramac machine utilized FPGAs as configurable bits. The logical structure of a single FPGA is shown in Fig. 3. Note that this is not a simple structure! It contains 4 wires—two data lines (d) and two address lines (a), plus a memory bit (m) and a switch (s). It contains a total of 6 transistors, 4 of which are used in the inverting circuit to keep the device nonvolatile. Even though it might be possible to find molecular-based components that can reproduce all of the properties of a single FPGA, it is probably not worth it. What is much simpler, however, is to come up with a molecular-based system that



	CMOS	MolElec.
# wires	4	2
#switches	6 transistors	1 molecule
Gain?	Yes	No

Fig. 3 A direct comparison with a field-programmable gate array (A) with a molecular electronic switching junction (B).

reproduces certain of the most critical qualities of an FPGA: namely, high on/off switching ratios; nonvolatility; and the ability to electronically configure the switch, and then read its state.

An advantage of molecules is that they are voltage-addressable. This means that different voltages can be utilized to carry out different functions. Imagine a molecule that is sandwiched between two conducting electrodes. This molecule is essentially a tunneling barrier, since the only way that current is going to get from one electrode to the other is via tunneling through the molecule. Since tunneling is a quantum mechanical process, virtually anything that is done to the molecule is going to have an exponential effect on the tunneling rate between the two electrodes. Let's say that we prepare this device, and we measure the junction resistance at 0 applied volts to be **x** Ohms. We then oxidize the molecule at +1 V. When we look again at the junction resistance at 0 V, it is now **y** Ohms. If we have found the right molecule, then **x** and **y** are widely separated values. Since we read the device at 0 V, the memory is nonvolatile. In addition, we have a device in which the two wires are essentially address lines when used at +1V, but they are data lines when used at 0 V. Finally, this device is based on a molecular property, and so should scale down to single molecule/nanowire junctions.

One point that we have treated rather casually in the above discussion is the following: If we *read* the device at 0 V, and *write* it by oxidizing the molecule at +1 V, then what is to keep the device from reducing back to the unoxidized form when we lower the voltage back to 0 V? That would then give us the exact same junction resistance that we had before we oxidized the junction. It turns out that this will most certainly be a problem, and there exist two possible solutions. First, we can make the oxidation process irreversible, but in that case we can only use the switch one time. While this does not make the device useless, a much more interesting device would be one that was reconfigureable. Thus, the most attractive option is to find a way to build hysteretic character into the molecular junction voltammograms. This second option would make the operation of our devices very similar to cross-point ferromagnetic memory devices that are currently being developed by a number of companies [12]. In those devices, known as Mag-RAM devices, there are giant magnetoresistive (GMR) materials that act as 'spin valves.' If the GMR material is electrically poled in a particular direction, then the junction resistance of the crosspoint element is reduced. This is similar to a magnetization vs. field poling process, such as that used in hard drive data storage. The difference is that it is electrically addressable, rather than addressed with a read-head.

There are various classes of molecules that might serve as attractive candidates for these devices. In particular, however, molecules that have a large amount of structural reorganization that accompanies a redox process are chief considerations, since such molecules should be characterized by activation barriers to the actual redox process. Such activation barriers are necessary for generating hysteric voltammograms [13]. One set of candidates are the molecular–mechanical compounds known as

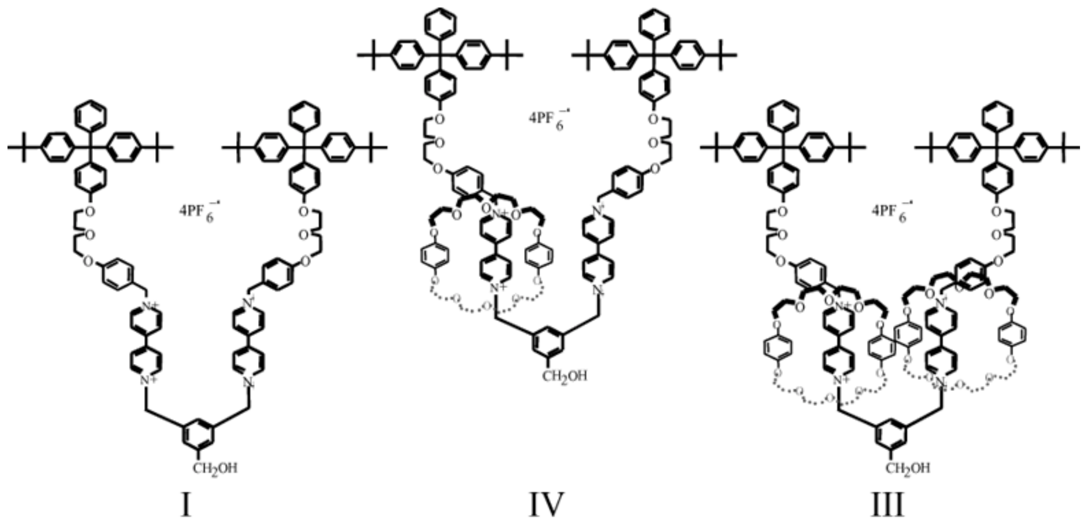


Fig. 4 The three molecular compounds used in this study. Only (B) and (C) are properly rotaxanes, since only those two molecules contain mechanically interlocked components.

rotaxanes, catenanes, and pseudo-rotaxanes. These classes of molecules had been developed by Fraser Stoddart and his group over a nearly 20-year period starting in 1981. Fraser's group had even shown (in the solution-phase) electrochemically addressable switching [14] and even simple logic [15] operations from certain of these systems [16]. Equally important was that his group had demonstrated that these various classes of molecular compounds were highly modular [17,18]. This means that it might be possible to have components that switch, components that aid assembly, and components that add chemical stability—all in the same molecular compound.

The molecular compounds that we began to work with, in collaboration with Stoddart's group, are shown in Fig. 3. These molecular compounds come from the class of molecules mentioned above that are known as rotaxanes [19]. Rotaxanes consist of two or more components mechanically interlocked with one another: one of the components is dumbbell-shaped, and the remaining components are rings that become trapped on the dumbbell component, encircling part of the dumbbell. In the unique class of rotaxanes used here, the dumbbell component contains two bipyridinium units. Only the two compounds of Fig. 4 that actually contain the rings are proper rotaxanes. The encircling ring(s) is (are) bis-para-phenylene-34-crown-10.

Based on both their structure and their solution-phase properties, these rotaxanes are not expected to exhibit bi-stable switching. This means that they are not candidates for a reconfigurable switch. However, all of the compounds irreversibly oxidize, which means that they are candidates for singly configurable switches. The reduction process on the unoxidized systems is reversible. The point of making these devices was to investigate the following questions: (1) Can we successfully incorporate monolayers of rotaxane molecules into a sandwich-type device within a crossbar structure? (2) Can we make a solid state, singly configurable switch from these molecules? (3) If we can make such a switch, can we correlate those switching properties with solution-phase voltammetry measurements? And (4) if we can make such switches in high yield, can we electronically configure several of these devices together to make wired-logic devices? The answers to all these questions turned out to be 'yes', and much of that work was recently published [20]. Only some of the highlights of that work will be presented here. In Fig. 5 we present the actual structure of the devices that were fabricated, and we present

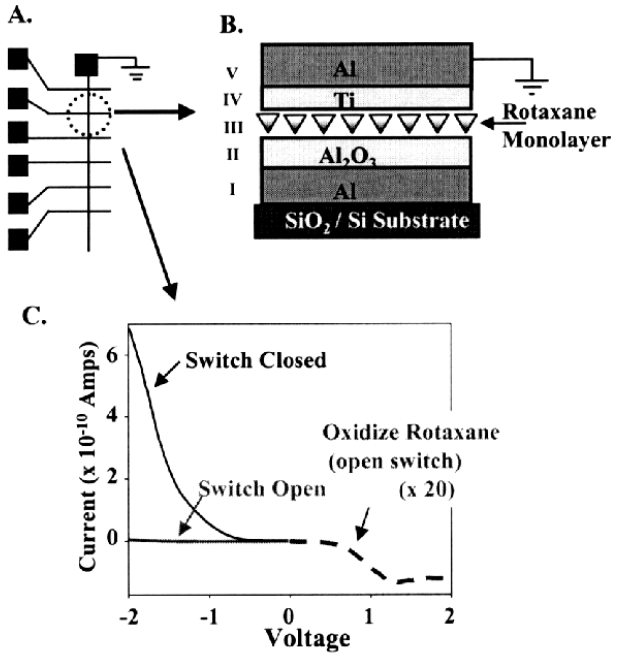


Fig. 5 (A.) A diagram of the linear array arrangement of devices utilized to make wired logic-gates. (B). A cross-sectional view depicting the assembly of the devices. A SiO_2 substrate is patterned with a an aluminum wire (I) that is characterized by approximately 1 nm of Al_2O_3 native oxide (II). On top of this bottom electrode a single monolayer of rotaxanes molecules (III) is transferred as a Langmuir–Blodgett film. The second electrode is deposited as (IV) 50 nm of Ti metal followed by (V) 1 µm of Al metal. The Fermi levels of both electrodes are essentially identical, so that all diode behavior observed in the devices originates from the molecules. (C). The switching properties of these molecular junctions is shown in this plot. The device, as originally prepared, represents the switch in its closed state, and it is read at negative (reducing voltages). Reduction of this junction is a reversible process, and the device may be read many times without significant degradation. The switch is 'opened' by writing the device at positive voltages. This serves to irreversibly oxidize the molecule. When the opened switch is read at negative biases, the current through the junction is now reduced by a factor of approximately 10^2.

the operation of one of these switches. Similar devices had been previously fabricated and characterized, and modeled in my group [21,22]. The structure shown in Fig. 5A is obviously a linear array arrangement of devices, not a crossbar. However, crossbars were fabricated as well, and devices in those structures performed in a manner similar to the linear arrays. It was also possible to correlate the electrochemical properties of the individual solid-state devices with solution-phase voltammetry [23]. Finally, a critical question that we were seeking to answer related to the issue of the electrical response of the switches—i.e., the difference between 'open' and 'closed'. The Mag-RAM devices previously discussed are made in a similar fashion (as a crosspoint memory), but those junctions show only a very small difference between a '1' state and a '0' state (about 15%) [24]. While that is sufficient for Mag-RAM devices to be utilized for memory applications, it is not sufficient to use them for logic. In a wired-logic gate, several devices are strung together, and the entire circuit is dissipative. In Fig. 5C, we observe a factor of approximately 10^2 between a '0' and '1' state, and thus logic devices could be fabricated from our switching junctions. The truth table of a wired AND gate is shown in Fig. 6. In this gate, the junction labeled 'L' was an open switch, while the two junctions labeled 'A' and 'B' were open

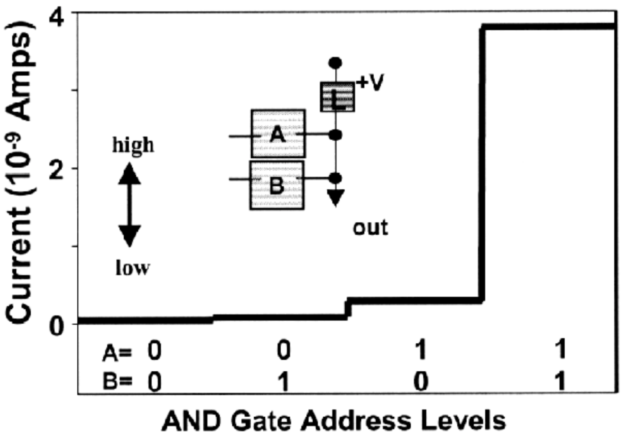


Fig. 6 Experimentally measured truth table of a wired-AND gate constructed from these molecular junction arrays. The hi/lo levels of the truth table are separated by a factor of 15.

switches. The truth table was measured at –1.5V, while the 'L' switch was opened at about +1 V. In our recent paper we were also able to demonstrate that the concept of electronically configuring these devices into logic gates was defect-tolerant.

CONCLUSIONS

In this paper, I have tried to outline a route for fabricating significant components of a computing machine that can operate near the thermodynamic limit for computing. A simple proof-of-concept for at least part of the outlined architecture was presented, but the realization of the goal of an entire machine is still a long way off. Two notable targets for immediate research include the development of a reconfigureable molecular switch, as well as the integration of molecular switches with true nanowires, such as chemically synthesized Si nanowires [25], or metallic single-walled carbon nanotubes [26]. A longer-term target is the development of a multiplexer/demultiplexer scheme that is amenable to chemical synthesis.

ACKNOWLEDGMENTS

The ideas described in this paper were developed over many conversations with various colleagues, and the experiments were carried out by a first-class group of graduate students and postdocs. In particular, I would like to acknowledge (in alphabetical order) Dr. Pat Collier, Mr. Phil Kuekes, Prof. Fraser Stoddart, Dr. Francisco Raymo, Dr. Eric Wong, and Dr. Stan Williams. This work was funded by an NSF-GOALI grant, the Office of Naval Research, and the Defense Advanced Research Projects Agency.

REFERENCES

1. R. Landauer. *IBM J. Res. Develop.* **3**, 183 (1961).
2. R. P. Feynman. *Lectures in Computation*, A. J. G. Hey and R. W. Allen (eds.), Addison-Wesley, Menlo Park (1996).
3. Rent's rule was never published by Richard Rent, but subsequent work has confirmed its validity: B. S. Landman and R. L. Russo. *IEEE Trans. on Comp*. **C20**, 1469 (1971).
4. P. J. Kuekes, private communication.
5. W. B. Culbertson, R. Amerson, R. J. Carter, G. S. Snider, P. J. Kuekes. *Proc. of the IEEE Symp. on FPGA's for Custom Computing Machines* (1997).

6. An excellent recent discussion of configurable computing can be found in: J. Villasenor and W. Mangione-Smith. *Sci. Am.* **276**, 68 (June 1997).
7. C. E. Leiserson. *IEEE Trans. On Comp.* **C34**, 892 (1985).
8. J. R. Heath, P. J. Kuekes, G. S. Snider, R. S. Williams. *Science* **280**, 1716 (1998).
9. L. R. Goke and G. L. Lipovski. "Banyan Networks for Partitioning Microprocessor Systems," *Proc. First Annual Symp. on Computer Architecture*, pp. 21–28, IEEE, New York (1973).
10. W. B. Culbertson and P. J. Kuekes. "Apparatus and Method for Configuring a Reconfigureable Electronic System Having Defective Resources," U.S. Patent # 5,791,771 (Aug. 4, 1998).
11. G. S. Snider. "Tileable Gate Array Cell for Programmable Logic Devices and Gate Array Having Tiled Gate Array Cells," U.S. Patent # 5,519,629 (May 21, 1996).
12. See, for example: (a). E. J. Torok, R. E. Lund, W. J. Simon. "Oligatomic Ferromagnetic Film Memory System Utilizing Field Stabilized Domains", U.S. Patent # 3,964, 034 (June 15, 1976). (b). W. J. Gallagher, J. H. Kaufman, S. S. Parkin, R. E. Scheuerlein. "Magnetic Memory Array Using Magnetic Tunnel Junction Devices in the Memory Cells", U.S. Patent, #5,640,343 (June 17, 1997).
13. One such example of a molecule with a hysteretic voltammogram is nickelocene. J. D. L. Holloway and W. E. Geiger, Jr. *J. Am.Chem. Soc.* **101**, 2038 (1998).
14. M. Asakawa, J. F. Stoddart, et al. *Angew. Chem. Int. Ed. Eng*. **37**, 333 (1998).
15. A. Credi, V. Balzani, S. J. Langford, J. F. Stoddart. *J. Am. Chem. Soc*, **119**, 2679 (1997).
16. A. Credi, V. Balzani, S. J. Langford, J. F. Stoddart. *J. Am. Chem. Soc.* **119**, 2679 (1997).
17. M.-V. Martínez-Díaz, N. Spencer, J. F. Stoddart. *Angew. Chem. Int. Ed. Engl*. **36**, 1904 (1997).
18. M. Asakawa, J. F. Stoddart, et al. *Chem. Eur. J*. **3**, 1992 (1997).
19. V. Balzani, M. Gomez-Lopez, J. F. Stoddart. *Acc. Chem. Res*. **31**, 405 (1998).
20. C. P. Collier, E. W. Wong, M. Belohradsky, F. J. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams, J. R. Heath. *Science* **285**, 391 (1999).
21. G. Markovich, D. V. Leff, S.-W. Chung, H. M. Soyez, B. Dunn, J. R. Heath. *Appl. Phys. Lett*. **70**, 3107 (1997).
22. S.-H. Kim, G. Markovich, S. Rezvani, S. H. Choi, K. L. Wang, J. R. Heath. *Appl. Phys. Lett.* **74**, 317 (1999).
23. This calculation was included as an appendix to ref. 20, and may be found on the *Science* web site at: www.sciencemag.org/feature/data/1039735.shl.
24. J. Brugg, private communication.
25. A. M. Morales, C. M. Lieber. *Science* **279**, 208–211 (1998).
26. S. Iijima. *Nature* **354**, 56–58 (1991).