gestions have invoked mixing due to rotational effects. However, we know from seismological measurements of the Sun that its core is essentially in solid-body rotation. This precludes rotational mixing as an important process at some distant time in the future when the Sun evolves into a giant star.

The elegant new insight provided by Eggleton *et al.* is that the distribution of $^3$He itself plays a key role in driving this additional mixing by establishing a distribution of $^3$He in the star that is susceptible to a Rayleigh-Taylor instability. This is similar to the well-known instability in everyday life where a dense fluid on top of a lighter fluid leads to the mixing of the two fluids. Even though the unstable gradient produced by $^3$He is tiny, the authors demonstrate, using three-dimensional, hydrodynamic stellar calculations, that this leads to rapid mixing and the destruction of $^3$He.

These calculations are some of the first results of an ambitious project led by Eggleton and Dearborn at the Lawrence Livermore National Laboratory (LLNL) to realistically model stars in three dimensions using a computer code named "Djehuty." The important difference between Djehuty and all other stellar evolution codes is the inclusion of three-dimensional hydrodynamics, which can account self-consistently for mixing and nonspherical effects. This is a step requiring a great deal of effort in both code development and computational brute force.

These results highlight the importance of improving the underlying physics in stellar models, especially because the driving physics in this particular case would be easy to ignore; this apparently minor consideration can be seen to have very important results.

Djehuty is still a work in progress: Impressive though their work already is, to model binary stars they will require an aspherical gravitational potential and will need to increase the number of hydrodynamic mesh points they use in the simulation by four orders of magnitude (*3*). Even the addition of rotation will cross another interesting threshold.

Stars cannot yet be genuinely evolved through their lifetimes with Djehuty, even with the computing resources at LLNL; in fact, the calculations presented by Eggleton *et al.* contain less than a day's worth of stellar evolution. To put this in perspective, for Eggleton's one-dimensional stellar evolution code, "an evolutionary sequence requires about 60 min from the main sequence to the helium flash"—and that was in 1971 (*4*). It will be a very long time before three-dimensional stellar evolution calculations can be done as rapidly and routinely as today's one-dimensional codes.

The example of the $^3$He problem and its resolution illustrates that, to understand the origin and evolution of the universe, we need to understand stars, and vice versa. In particular, the predictions of Big Bang nucleosynthesis have pointed to a problem in stellar modeling. On the other hand, there are many examples where an understanding of stars in the local universe would lead to a better understanding of the evolution of the universe and its constituents on the large scale. Besides the $^3$He problem, there is also a $^7$Li problem. The Big Bang prediction of the $^7$Li is at least a factor of 2 higher than is observed in metal-poor stars (*5*), and in this case extra mixing cannot provide a simple explanation. Undoubtedly, the resolution of this puzzle will provide new insights into the physics of stars, the Big Bang, or even both.

**References**


1. P. P. Eggleton, D. S. P. Dearborn, J. C. Lattanzio, *Science* **314**, 1580 (2006); published online 26 October 2006 (10.1126/science.1133065).
2. R. T. Rood, T. M. Bania, T. L. Wilson, *Nature* **355**, 618 (1992).
3. D. S. P. Dearborn, J. C. Lattanzio, P. P. Eggleton, *Astrophys. J.* **639**, 405 (2006).
4. P. P. Eggleton, *Mon. Not. R. Astron. Soc.* **151**, 351 (1971).
5. D. N. Spergel *et al.*, *Astrophys. J.*, in press; (astro-ph/0603449).


10.1126/science.1134924

CHEMISTRY

# Pulling Strings

A toolkit of DNA-based devices can be used for computational circuits.

**Walter Fontana**

Computing devices, like the information they process, are embodied in a material substrate constrained by the laws of physics (*1*). The design of modern computing devices has nevertheless succeeded to a remarkable extent in separating hardware from software and questions specific to physics from questions specific to computation. In such a setting, abstract formalisms of the kind envisioned by Turing (*2*) can justifiably ignore the nature of materials and issues such as energy dissipation and material stability.

However, this separation between hardware and software—and hence physics and computation—breaks down when device features approach atomic scales or when the devices that process information are of the same class as the information itself. How, then, can we realize computation in the world of molecules and their chemical reactions? On page 1585 of this issue, Seelig *et al.* (*3*) provide an answer to this question.

One way to formally express computation is in terms of rules that rewrite words. Molecules, like words, are combinatorial structures, and chemical reactions can provide the required rules, if we can program chemistry. To exert the necessary control over chemistry, we must be able to specify which components in a mixture of molecules interact when, and where. This control can be achieved by designing appropriate single-stranded DNA (or RNA) sequences that bind to each other like Velcro's hook and loop fasteners, but in an addressable manner. Based on this idea, Seelig *et al.* exploit a simple principle—strand displacement—to implement not just logic gates, but also a toolkit of devices for building molecular circuits of a digital kind.

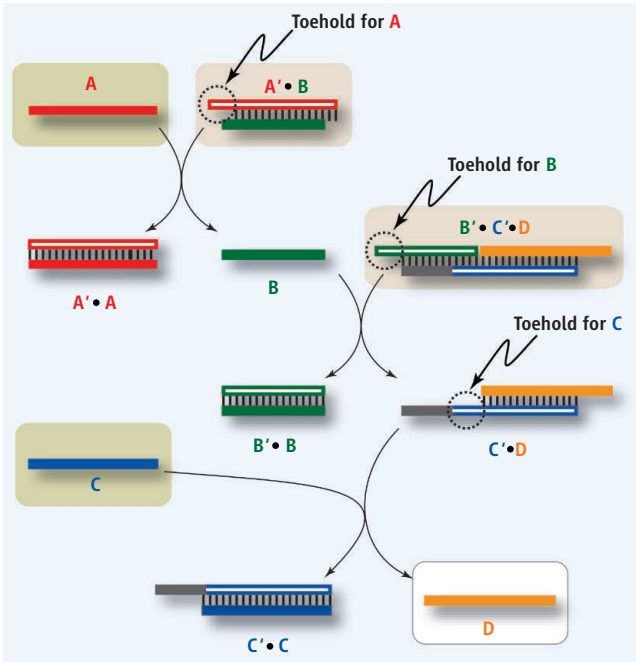
In the past decade, DNA (or RNA) sequences have been used to find solutions to combinatorial problems by self-assembly (*4*), to encode complete decision trees for simple games like tic-tac-toe (*5*), and to build programmable sensors of cellular states (*6*). DNA has also been used to build nanostructures and nanomechanical devices (*7, 8*), as well as two-dimensional grids that can function as frames of reference for placing such devices at specific locations (*9*). For example, Seeman and co-workers (*10*) have developed a rotary device that consists of two DNA strands woven into two pairs of helices, with a flexible hinge region in between. This device can act as a programmable, molecular-scale robot arm. On page 1583 of this issue, Ding and Seeman (*11*) report the deliberate, function-preserving placement of such a device in a two-dimensional array of DNA tiles.

There are two reasons for the versatility of

The author is in the Department of Systems Biology, Harvard University, Boston, MA 02115, USA. E-mail: walter@hms.harvard.edu

DNA as a structural, mechanical, and computational substrate. First, the Watson-Crick base-pairing rules provide a natural foundation for programming the interaction specificity of DNA sequences. Second, base pairing provides the free energy needed to deliberately change structures and move computation forward. A DNA strand will let go of its binding partner if a third strand offers base pairs that are energetically more favorable. A DNA sequence thus serves both as a specific instruction and as the fuel needed for its own execution.

To illustrate the case of Seelig *et al.*, consider two sequences A′ and B that form a complex A′B by virtue of complementary segments. In the presence of a strand A that offers more favorable base pairs to A′ than B, a displacement reaction $A + A'B \rightarrow AA' + B$ will occur (see the figure). The DNA complex A′B therefore stands for the statement "if A then B′, because it yields B when it encounters A. In this scheme, the gate "B AND C" translates into a statement like "if (B and C) then D", which is a DNA complex designed to release strand D as a result of two sequential displacement reactions that require the presence of B and C (see the figure).




**Gate composition.** The translator gate A′B (gray box in top row) exchanges strand A for B. B then triggers the "B AND C" gate (B′C′D, gray box in second row). Together, the two gates form an "A AND C" gate that emits a D-strand in the presence of inputs A and C (light green boxes). Open and filled bars represent complementary sequence segments. If D were A (or give rise to A, as accomplished by a translator D′A), the end product could reenter the cascade at the top, creating a feedback loop.

In such a system, logic gates are molecular (DNA) complexes that execute their logic through reactions. Gates constructed in this way can be concatenated, because the output string released by one gate can react with another gate in the mixture, much like in biological signaling cascades. For example, an "A AND C" gate can be implemented by using a "B AND C" gate in conjunction with an "if A then B" construct that exchanges A for B (see the figure). This is analogous to address forwarding in a Web browser.

A test tube typically contains many copies of a given gate complex that undergo displacement reactions in accordance with the binding preferences programmed into their DNA sequences. Because these reactions yield a noisy output-strand concentration, digitization of the output yield as "high" (true) or "low" (false) is required to interpret a DNA gate as a logic operation.

Seelig *et al.* provide a toolkit of DNA-based reactions for such digital signal processing. The tools include thresholds to remove leaks and amplifiers to restore signal strength. For example, an amplifier permits one input strand to cause the release of more than one output strand copy. This can be achieved by a feedback construct involving two gates that mutually trigger each other ("if A then B + if B then A") as soon as input strand A appears in the mixture. Alternatively, the input strand has been used as a catalyst for refolding a metastable DNA complex in a process that also releases an output strand (*12*). In this way, the same input strand can help to refold several complexes, leading to output amplification.

It can be difficult to design sequences that make up large circuits. Complementary regions in a DNA sequence can cause a strand to fold back upon itself, potentially blocking further computation. Accidental complementarities across sequences can lead to interference between computations, in analogy to cross-talk in biological signaling systems. Seelig *et al.* use a computational optimization procedure to design sequences that minimize the likelihood of such complications. They validate their architecture and design tools with a dazzling circuit of 11 gates and six inputs.

What might this prototype technology be good for? The authors envision analytical applications in systems biology, such as the in situ detection, quantification, or amplification of microRNAs and transcription patterns. But this scalable molecular programming language may also provide a means for choreographing the assembly and operation of future nanometer-scale devices.

Unlike electronic circuit elements, DNA gates and their inputs are used up as the computation unfolds through chemical reactions; hardware and software are one and the same. Yet, what appear to be limitations may turn out to be intriguing opportunities. As gates are transformed by the very computations they control, can new gates assemble as by-products? Could one devise a computational gate "metabolism" that maintains an ensemble of gates through a catalytic cycle?

Milner has devised a calculus (*13*) that views every component of a distributed computational system as an interactive process, whose channels are consumed upon communication. Seelig *et al.* may unknowingly have come close to implementing design aspects of that calculus in chemistry. Theoretical computer scientists may find inspiration in a chemical model of an influential abstraction. In return, modifications of this calculus may become useful in the design and analysis of DNA gate systems.

Over the past half-century, the idea has taken hold that physical processes, particularly in biological systems, can be understood as computation. A back-and-forth between transparent experimental models of molecular computation and the development of formal tools for reasoning about concurrent behavior might lead to a better appreciation of what it means for cells to "compute," "organize," or "process information" and, perhaps, evolve.

**References**

1. R. Landauer, *Phys. Today* **44**, 23 (1991).
2. A. Turing, *Proc. London Math. Soc.* **42**, 230 (1936).
3. G. Seelig, D. Soloveichik, D. Y. Zhang, E. Winfree, *Science* **314**, 1585 (2006).
4. L. M. Adleman, *Science* **266**, 1021 (1994).
5. M. N. Stojanovic, D. Stefanovic, *Nat. Biotechnol.* **21**, 1069 (2003).
6. Y. Benenson, B. Gil, U. Ben-Dor, R. Adar, E. Shapiro, *Nature* **429**, 423 (2004).
7. P. W. K. Rothemund, *Nature* **440**, 297 (2006).
8. B. Yurke, A. J. Turberfield, A. P. Mills Jr., F. C. Simmel, J. L. Neumann, *Nature* **406**, 605 (2000).
9. E. Winfree, F. Liu, L. A. Wenzler, N. C. Seeman, *Nature* **394**, 539 (1998).
10. H. Yan, X. Zhang, Z. Shen, N. C. Seeman, *Nature* **415**, 62 (2002).
11. B. Ding, N. C. Seeman, *Science* **314**, 1583 (2006).
12. G. Seelig, B. Yurke, E. Winfree, *J. Am. Chem. Soc.* **128**, 12211 (2006).
13. R. Milner, *A Calculus of Communicating Systems* (Springer, Berlin/New York, 1980).

10.1126/science.1135101