

A System-level Design Approach to the Evaluation of Self-assembled Computer Architectures

Chris Dwyer

Dept. of Electrical and Computer Engineering, Duke University, Durham, NC 27708

Abstract— Advances in materials science, chemistry, and self-assembly are anticipated to enable the fabrication of massively parallel computer systems. An important aspect of this development is the design and evaluation of system-level interactions and performance based on the fundamental properties of emerging nanoscale devices on real computer applications.

This paper will present the design, evaluation, and fabrication theory for an emerging DNA self-assembly process as applied to a parallel computer architecture that is uniquely enabled by self-assembly. A brief evaluation of the architecture (a decoupled array multiprocessor or DAMP) will be presented in terms of peak runtime performance and power consumption.

1. Introduction

Nanoscale science and technology are advancing rapidly and the proof-of-concept demonstrations that fuel the development of these emerging devices have begun to find real applications [1-4]. Full system evaluation is becoming an important aspect of the development of new applications that are enabled by these new technologies. The clean abstractions that define conventional computer architecture are insufficient for dealing with the dramatic change between the behavior of nanoscale and microscale technology.

Several computer architectures have emerged that address the importance of the technological layer in system performance [5-13]. Even conventional scaled CMOS is beginning to experience an increase in the degree of coupling between device architecture, system or computer architecture, and the raw technology.

The degree of feedback between the architectural level and device level defines the abstract division between the two. That is, tight coupling between the computer architecture and the device technology blurs the line between them and makes independent optimization of either side unproductive. The linearity of a design and its optimization (or an ability to treat the entire system as a linear composition) is failing because of the changes in the technological layer and underlying assumptions about device performance and capabilities.

Nanoscale and molecular scale devices increase the coupling between design levels through: (i) non-uniform device properties, (ii) high defect rates, (iii) high fault generation rates, and (iv) a limited scale of device integration. A comprehensive list is

beyond the scope of this paper but all these aspects play an important role in any nanoscale computer architecture.

The response to this challenge is to evaluate all levels of the design simultaneously in a multi-scale fashion. This paper will describe a full system evaluation framework within the context of a DNA self-assembling technology. Section 2 provides the assumptions and background to the device and computer architecture evaluation frameworks, and section 3 describes a case study of the DAMP.

2. Self-Assembled Computer Evaluation Frameworks

In contrast to spatial or tile computing [14, 15] or scaffolding by DNA tiles [16] the device architecture described here adopts an approach where DNA self-assembly guides the placement of nanoelectronic components [17-19]. A framework for evaluating device properties is presented before describing some of the metrics used to evaluate computer architectures.

Device Architectures

The term “device” as used to describe the technology for implementing a computer system is an abstract concept that generally refers to a highly non-linear switch (e.g., a transistor). The development of new technological paradigms has expanded the notion of what a suitable device can be. And the differences in this definition can be large: from resonant tunneling diodes (RTDs) to nanoscale field-effect transistors (FETs) to the presence or absence of a protein.

Each device can be viewed in terms of the following characteristics:

(i) *Linear signal transduction*

The quality of how the device can be interconnected so that an information bearing signal can be propagated with linear loss (e.g. small protein diffusion through a cell membrane, Ohmic losses for electron/hole currents in metals, etc.) The time delay (at an operating frequency) for this transduction is important to performance evaluations.

(ii) *Non-linear signal modulation by another signal*

The non-linear modulation of a signal by a device is universally important to computer systems (analog or digital). The modulation can be synchronized in time (e.g., three terminal devices like transistors) or not (e.g., two terminal molecular switches). The quality and time delay of this modulation is relevant to noise immunity and performance (e.g., transconductance or transimpedance of an amplifier in an RLC environment).

(iii) *Signal amplification / restoration*

Unless the linear signal transduction mechanism *and* non-linear modulation are truly lossless, signals will degrade during propagation to other device elements.

Either the device must amplify, or restore (as in CMOS), the information bearing signal per stage or the circuit design must be sufficiently shallow that the signal degradation is acceptably low.

(iv) *Signal noise immunity*

A signal will incur some form of noise (Johnson / thermal, shot, 1/f, etc.) if the device is operated above 0K. Noise suppression is related to signal restoration but a device must be evaluated in terms that capture the influence of noise on the information bearing signal.

(v) *Circuit patterning and interconnect*

Most interesting computer systems require many devices to be interconnected in non-planar circuit topologies. This means that devices must support fan-out (or multiple output destinations). This is related to the device's ability to amplify a signal for multiple sinks. The interconnection network is equally important (related to signal transduction) but must be evaluated in terms of pitch and minimum feature sizes as well as defect rates and effective interconnect density.

(vi) *Scale of device integration*

Many devices must be integrated to create a computer system. The number of devices that can be integrated into a common interconnection network determines the number of resources available for computing. The scale of the device and interconnection integration will impact the kinds of architectures that can be built upon the technology.

(vii) *Energy consumption*

The energy loss during transduction, modulation, and amplification/restoration are important to understand the system-level implications of the device. Further, the operational mode and environmental requirements of the device are fundamentally important in determining the full system energy budget. For example, cryogenic cooling consumes vastly greater quantities of energy than most multiprocessor clusters – high temperature (i.e., room temperature) operation is important.

The device characteristics listed above represent a large dimensional space and simulation tools are important in navigating the design tradeoffs. Device level simulators commonly model the operational modes of a device using analytical derivations or empirical data from laboratory devices. The results of these simulations are distilled into figures of merit (e.g., energy per transition, transition delay, etc.) and used in higher-level architectural simulators.

The case studies presented in section 3 rely on the detailed device simulation of a ring-gated FET (RG-FET) and a highly doped silicon rod by the PISCES IIb drift-diffusion simulator [19]. This level of modeling captures the behavior of the devices to describe all of the characteristics required to be suitable for a large-scale computer system.

Computer Architectures

The evaluation of a computer design requires behavioral simulators that can bridge the gap between device-level methodology and system-level methodology. The gap exists because of the large number of simulation resources required to accurately model (i.e., to simulate with single cycle accuracy) the performance of a large computer system.

Some of the commonly used metrics include execution time of an application or benchmark, cycles per instruction (and/or instructions per cycle) for a benchmark, and average power or total energy consumption. Although commonly misused, the system clock frequency plays an important but ultimately ancillary role in system performance.

The choice of benchmarks, or code fragments that represent real user applications, is important in making the simulation results meaningful to real end users. For example, an architecture that has been optimized to perform contrived code sequences may appear to be a breakthrough but in reality can never be applied.

For similar reasons the simulation framework must accurately model the architecture so that real instruction and data streams can be executed. During the simulated execution performance data is collected to calculate the values of the metrics. The performance data is modeled behaviorally using the device-level metrics collected during the detailed device simulations (e.g., delay per transition, energy consumed per transition, etc.) For example, when a signal in the system transitions from one value to the next, an energy penalty can be “charged” to it and accumulated over the execution of the program to estimate total energy consumption. A similar method is used to collect timing information.

3. Case Study: the DAMP

Tightly coupled systems have complex objective functions which make the process of optimizing their performance difficult. The architecture presented here is a brief description of such a system to illustrate the design approach [13, 20].

The device architecture used to model the DAMP is based on nanoscale rod/tube components assembled by DNA-guided self-assembly [19-21]. The assumption here is that each type of component can be individually assigned to a specific location in the circuit. Further, the DNA junctions between components can be metallized to create Ohmic contacts between the rod ends. Custom design tools were developed to enable the creation of circuit models that implement the logic and memory functions required by the architecture. Due to the conventional electronic behavior of the RG-FET and interconnect (verified using the PISCES IIb simulations) the requirements of the device outlined in section 2 are satisfied. What remains is to demonstrate that the architecture has good performance on a useful application.

The DAMP is a single-instruction multiple-data (SIMD) system design that uses simple processing elements (PEs) in parallel to solve optimization problems. However, the constraints placed on the design by the nanoscale self-assembly

technology changes the fundamental assumptions about how the system can work. For instance, the design relies on a single input and output channel that is shared between all PEs. This restrictive I/O design is indeed a performance bottleneck but eliminates the challenge of interfacing a high density nanoscale device array with microscale circuitry. Without feedback between the device and architecture levels unreasonable assumptions will create a design that can not be supported by the device technology.

The design approach involved modifying (severely) a typical assumption about I/O connectivity because the underlying technology did not readily support such connectivity. The scale of device integration supported by self-assembly is another example of a technological constraint that has dramatic implications on the computer architecture. Reduced levels of integration constrain the logical size of a PE i.e., fewer devices per PE mean simpler functionality per PE. The only way this can make sense is if there are many PEs, even if they are simple. Self-assembly has the potential to organize molar-scale numbers of components which amount to 10^{12} to 10^{19} assembled devices (at 100 μM concentrations) depending on process yield and defect rates. This requires feedback between the device layer and the architecture or the final design will be impractically large for a single PE.

Each PE in the DAMP uses about 1500 RG-FETs to implement a register file, and accumulator, and a set of simple bit-serial operations as illustrated in figure 1.

The peak performance of the full DAMP (with 10^{12} PEs) on simple, local (i.e., no I/O involved) ALU operations is compared to some other machines in table 1.

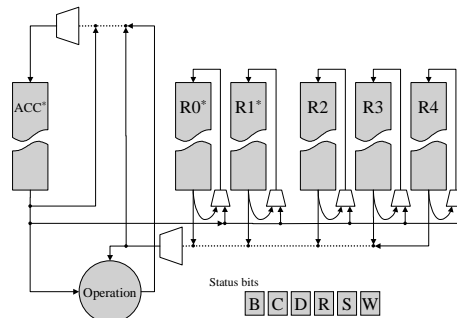


Figure 1: Schematic structure of a single DAMP processing element.

Table 1. Peak performance comparison

Machine	Peak ALU (16-bit) operations per second
DAMP	$\sim 10^{18}$
IBM BlueGene /L	$\sim 10^{15}$
NEC Earth Simulator	$\sim 10^{15}$
Intel P4	$\sim 10^{11}$

4. Conclusions

Nanoscale self-assembled technologies are blurring the clean abstractions between device and computer architectural levels. The consequence of this is an increase in the

coupling between device level design and full system design. The optimization of the full system is more difficult because there is little independence between the levels.

Full system evaluation requires a framework of metrics and device characterization that spans the divide between the abstract computer architecture and device architecture. A comprehensive framework must use detailed simulation at all levels but do so in a computationally efficient manner. The multi-scale simulations described here are commonly used to bridge this gap. The detailed simulation of devices is used to extract values for metrics that apply to the next level of the design that will ultimately be used by behavioral simulations of the full system. Full system metrics of the system performance are the end goal of the process to justify further study and implementation of the system.

Acknowledgements

This work was supported primarily by the NSF (CCR-0326157, EIA-9972879), through equipment donations from IBM and Intel, and the AFRL (FA8750-05-2-0018).

References

- [1] T. Rueckes, K. Kim, E. Joselevich, G. Y. Tseng, C.-L. Cheung, and C. M. Lieber, "Carbon Nanotube-Based Nonvolatile Random Access Memory for Molecular Computing," *Science*, vol. 289, pp. 94-97, 2000.
- [2] P. J. Kuekes and R. S. Williams, "Demultiplexer for a molecular wire crossbar network." United States Patent 6,256,767, July 2001.
- [3] J. M. Tour, L. Cheng, D. P. Nackashi, Y. Yao, A. K. Flatt, S. K. S. Angelo, T. E. Mallouk, and P. D. Franzon, "NanoCell Electronic Memories," in *Journal of the American Chemical Society*, vol. 125, 2003, pp. 13279-13283.
- [4] R. K. Venkatesan, A. S. AL-Zawawi, and E. Rotenberg, "Tapping ZettaRAM for Low-Power Memory Systems," in *Proceedings of the 11th International Symposium on High-Performance Computer Architecture (HPCA-11)*: IEEE Computer Society, 2005.
- [5] W. Porod, C. S. Lent, P. D. Tougaw, and G. H. Bernstein, "Quantum Cellular Automata," *Nanotechnology*, vol. 4, pp. 49-57, 1993.
- [6] J. R. Heath, P. J. Kuekes, G. S. Snider, and R. S. Williams, "A Defect-Tolerant Computer Architecture: Opportunities for Nanotechnology," *Science*, vol. 280, pp. 1716-1721, 1998.
- [7] T. J. Fountain, M. J. B. Duff, D. G. Crawley, C. D. Tomlinson, and C. D. Moffat, "The Use of Nanoelectronic Devices in Highly-Parallel Computing Systems," in *IEEE Transactions on VLSI Systems*, vol. 6, 1998, pp. 31-38.
- [8] J. C. Ellenbogen and J. C. Love, "Architectures for Molecular Electronic Computers: 1. Logic Structures and an Adder Designed from Molecular Electronic Diodes," in *Proceedings of the IEEE*, vol. 88, 2000, pp. 386-426.
- [9] S. C. Goldstein and M. Budiu, "NanoFabrics: Spatial Computing Using Molecular Electronics," in *Proceedings of the 28th Annual International Symposium on Computer Architecture (ISCA '01)*, 2001, pp. 178-191.
- [10] S. Frost, A. Rodrigues, A. Janiszewski, R. Raush, and P. M. Kogge, "Memory in motion: A study of storage structures in QCA," presented at the First Workshop on Non-Silicon Computing, 2002.

- [11] J. Han and P. Jonker, "A Defect- and Fault-Tolerant Architecture for Nanocomputers," *Nanotechnology*, vol. 14, pp. 224-230, 2003.
- [12] J. P. Patwardhan, C. Dwyer, A. R. Lebeck, and D. J. Sorin, "Circuit and System Architecture for DNA-Guided Self-Assembly of Nanoelectronics," in *Foundations of Nanoscience: Self-Assembled Architectures and Devices*, 2004, pp. 344-358.
- [13] C. Dwyer, J. Poulton, R. M. Taylor, and L. Vicci, "DNA Self-assembled Parallel Computer Architectures," *Nanotechnology*, vol. 15, pp. 1688-1694, 2004.
- [14] H. Yan, L. Feng, T. H. LaBean, and J. H. Reif, "Parallel Molecular Computations of Pairwise Exclusive-Or (XOR) Using DNA "String Tile" Self-Assembly," *Journal Of The American Chemical Society (Communication)*, vol. 125, pp. 14246-14247, 2003.
- [15] M. Cook, P. W. K. Rothemund, and E. Winfree, "Self-assembled circuit patterns," *DNA Computing Lecture Notes in Computer Science*, vol. 2943, pp. 91-107, 2004.
- [16] B. H. Robinson and N. C. Seeman, "The design of a biochip: a self-assembling molecular-scale memory device," *Protein Engineering*, vol. 4, pp. 295-300, 1987.
- [17] B. R. Martin, D. J. Dermody, B. D. Reiss, M. Fang, L. A. Lyon, M. J. Natan, and T. E. Mallouk, "Orthogonal Self-Assembly on Colloidal Gold-Platinum Nanorods," in *Advanced Materials*, vol. 11, 1999, pp. 1021-1025.
- [18] J. K. N. Mbindyo, B. D. Reiss, B. R. Martin, C. D. Keating, M. J. Natan, and T. E. Mallouk, "DNA-directed Assembly of Gold Nanowires on Complementary Surfaces," in *Advanced Materials*, vol. 13, 2001, pp. 249-254.
- [19] C. Dwyer, L. Vicci, and R. M. Taylor, "Performance Simulation of Nanoscale Silicon Rod Field-Effect Transistor Logic," *IEEE Transactions on Nanotechnology*, vol. 2, pp. 69-74, 2003.
- [20] C. Dwyer, L. Vicci, J. Poulton, D. Erie, R. Superfine, S. Washburn, and R. M. Taylor, "The Design of DNA Self-Assembled Computing Circuitry," *IEEE Transactions on VLSI*, vol. 12, pp. 1214-1220, 2004.
- [21] C. Dwyer, V. Johri, J. P. Patwardhan, A. R. Lebeck, and D. J. Sorin, "Design Tools for Self-assembling Nanoscale Technology," *Nanotechnology*, vol. 15, pp. 1240-1245, 2004.