A Defect Tolerant Self-organizing Nanoscale SIMD Architecture

Jaidev P. Patwardhan†, Chris Dwyer‡, and Alvin R. Lebeck†
{jaidev,alvy}@cs.duke.edu, dwyer@ee.duke.edu.

†Department of Computer Science
Duke University
Durham, NC 27708

‡Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708

1 Introduction

Manufacturing defects, power density, process variability, transient faults, bulk silicon limits, rising test costs and multibillion dollar fabrication facilities are some of the challenges facing the continued scaling of CMOS. While architectural modifications (e.g., multicore) can provide some short-term relief, the semiconductor industry recognizes the importance of these issues and the need to explore long term alternatives to CMOS devices and fabrication techniques [5]. One promising alternative is DNA-based self-assembly [11] of nanoscale components using inexpensive laboratory equipment to achieve tera to peta-scale integration. Although much of this technology is in its infancy (i.e., demonstrated in research lab experiments), by studying its potential uses for building computing systems, architects can gain a deeper understanding of its limitations and opportunities while providing important feedback to the scientists developing the new technologies. While our work is motivated by DNA-based self-assembly, it is applicable to any technology with similar characteristics (e.g., scaled CMOS with high process variability, high defect rates and point-to-point links between relatively small compute nodes).

We previously proposed an assembly process [9] to place electronic circuits on a DNA grid [14,15]. DNA-based fabrication produces precise control within a small area (e.g., 9 μm^2) enabling the construction of a large number (~10^9-10^{12}) of small nodes (computational circuits with ~10^4 transistors) that can be linked together using self-assembly. This produces a random network of nodes, due to the lack of control over placement and orientation of nodes, that contains defective nodes and links. A computing system built from this random network must: a) tolerate node and interconnect defects, b) not rely on underlying network structure, c) compose more powerful computational blocks from simple nodes, d) minimize communication overheads, and e) achieve performance that is at least comparable to future CMOS based systems. Several research projects have examined building computing systems with a subset of these goals, including self-organization [1,13], routing and resiliency in the face of defects [1,4] and the ability to compose complex computational units from simpler blocks [7], but we face added challenges because of the extremely limited computational capabilities available in nodes. We previously developed the nanoscale active network architecture (NANA) [8], which is a general purpose architecture designed with a similar set of goals, but it fails to match the performance of conventional CMOS systems.

We present a SIMD architecture designed to address these challenges. The fundamental building block in our architecture is a relatively small node (e.g., 1-bit ALU with 32 bits of storage and communication support for four neighbors) that operates asynchronously. A configuration phase at startup isolates defective nodes and allows groups of nodes to self-organize into SIMD processing elements (PEs). Simulations using conservative estimates for node size and device speed show that the proposed design can match the performance of aggressively scaled architectures for 8 out of 9 benchmarks tested. Furthermore, this performance is achieved with a very low power density of 6.5 W/cm^2 (vs. >75 W/cm^2 for modern cores) while conservatively assuming that about 90% of the devices in the system switch every nanosecond. Finally, we show that our system can tolerate up to 30% defective nodes. Our results demonstrate the potential of this technology for building high performance architectures despite high defect rates and loss of precise control during fabrication. Further improvements are possible as the technology scales to allow more complex nodes, better inter-node connectivity, and faster devices.

2 System Architecture

To efficiently utilize large numbers (>10^9-10^{12}) of nodes we implement a SIMD architecture and focus on data parallel workloads. Our proposed system - called the “Self-organizing SIMD Architecture” (SSA) - supports a three operand register-based ISA with predicated execution and explicit PE-Shift instructions to move data between PEs and communicate with an external controller. Each SSA instruction has between 39 and 44 bits and contains: 1) a 16-bit fully-decoded opcode microinstruction, 2) a 20-bit register specifier microinstruction, and 3) a 3-bit “synch” microinstruction with an optional 5-bit synch repeat counter. Each microinstruction can be independently broadcast and includes 2 bits of control overhead to select a control register as a destination. We assume that the external controller has access to a conventional memory system.

Careful node design is critical in maximizing system performance. Due to limited node size, designing the node architecture involves a trade-off between maximizing functionality (compute, communicate, and self-organize) and performance while minimizing circuit size. To avoid the area and power overhead of routing clock signals and to mitigate the effects of device parameter variation, instruction execution and sequencing within a node are asynchronous. Each node has a 1-bit ALU with a small register file and connects to other nodes with (up to four) single wire links. Each link supports low bandwidth asynchronous communication that transfers 1 data bit per handshake. To support deadlock-free routing, we add support for three virtual channels (1 bit each). The random network of nodes is organized at two levels during a configuration phase. First, since a node is too small to hold a PE, we group sets of nodes to form a PE. Second, PEs are linked in a logical ring providing programmers a simplified system view to reason about inter-PE communication. Communication with external circuitry occurs through metal junctions (“vias”) which are controlled by “*anchor*” nodes. Figure 1 shows a small random network configured to form three 8-bit PEs.

The configuration process, initiated from an anchor, maps out defective nodes and connects functional nodes in a broadcast tree using a variant of the “reverse path forwarding” (RPF) algorithm

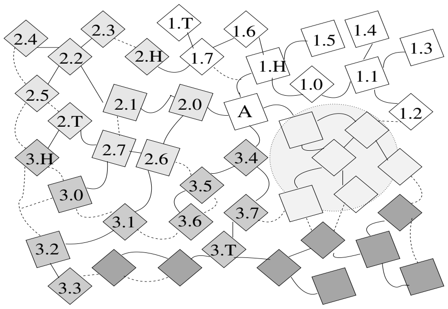

FIGURE 1. Configured system with 3 8-bit processing elements (PEs), with one anchor node (denoted 'A').

[3,10]. The system can be configured in two ways: a) as a monolithic system, all PEs on one logical ring (one "cell"), or b) as multiple, independent logical rings (multiple "cells"). In case (b), we achieve space partitioning by running the configuration algorithm from multiple anchors to create independent cells.

3 Evaluation

We evaluate SSA using a custom, event-driven simulator and use results from simulating smaller systems to extrapolate the behavior of larger systems. The simulator models each node in detail to obtain a detailed view of system execution. The latency of all activity in a node is a multiple of a base "time quantum", which we conservatively assume to be 1 nanosecond. Experimental devices are expected to operate at frequencies exceeding 100 GHz [2] with demonstrated frequencies over 10GHz [12] (time quantum of 0.1 ns), and asynchronous handshakes at high speeds have been demonstrated [6]. We compare the performance of SSA to a Pentium 4 (P4), an ideal out-of-order superscalar (I-SS), an ideal 16-way CMP (16-CMP) and an ideal implementation of SSA (I-SSA) that uses the same instruction set, but assumes unit instruction execution latencies, and no communication overhead. We use nine benchmarks - matrix multiplication, image filters (3), sorting, searching, bin-packing and data encryption (2).

We find that SSA achieves good performance on benchmarks that have data parallelism (all except sort). For a configuration with more than 64K PEs, SSA matches the performance of an ideal 16-way CMP. *Thus, despite SSA's severe limits on node computational power, network bandwidth and connectivity, and low control over the fabrication process, it matches the performance of idealized conventional architectures, with lower device switching speeds and a lower power density.* We also show that SSA can tolerate high node defect rates. For the encryption benchmarks, performance gracefully degrades as the fraction of defective nodes increases to 30%. For the other benchmarks, by over-provisioning the system, SSA tolerates up to 20% defective nodes with a small (<10%) degradation in performance.

Our results highlight SSA's flexibility in configuring independent cells to improve system utilization and throughput. SSA provides higher throughput than the P4 and I-SS while using the same area. Coupled with the ability to tolerate a significant defect rate, SSA shows potential in harnessing the higher device densities that emerging technologies promise to deliver.

4 Limitations, Future Work and Conclusion

Our performance evaluation highlights the fact that SSA is not a general purpose architecture, and is unlikely to match the performance of conventional processors on most general purpose workloads. SSA is likely to achieve good performance on data parallel programs that require little inter-PE communication, nearest neighbor communication or regular and unidirectional dataflow, but not on programs that require all-to-all communication. There are a number of avenues for further research. We plan to extend SSA to speed up floating point operations, exploit multiple anchors to increase system I/O bandwidth, and to handle transient faults through redundant execution or by extending PEs to perform simple checksum/parity computations. We are also looking at extending the software toolchain to explore compiler optimizations.

The expected rise in defect rates for both top-down and bottom-up manufacturing techniques makes it imperative for architects to develop defect tolerant architectures to exploit the full potential of emerging nanoscale devices. To this end, we have developed SSA, a self-organizing SIMD architecture built from a random network of simple self-assembled computational nodes. Despite high defect rates, low bandwidth and lack of underlying physical structure we show that, for data parallel workloads, SSA is able to perform better than conventional microprocessors, while operating at a lower speed and consuming much less power. While SSA does not solve all problems encountered with self-assembled architectures, it is a step towards realizing defect tolerant computing systems that may provide inexpensive terascale integration.

Acknowledgments

This work is supported by an NSF ITR grant CCR-0326157, the Duke University Provost's Common Fund, AFRL contract FA8750-05-2-0018, and equipment donations from IBM and Intel.

References

[1] H. Abelson et al. Amorphous Computing. *Communications of the ACM*, 43(5):74–82, 2000.

[2] P. J. Burke. Carbon Nanotube Devices for GHz to THz Applications. *Proc. of SPIE*, 5593:52–61, 2004.

[3] Y. K. Dalal and R. M. Metcalfe. Reverse Path Forwarding of Broadcast Packets. *Comm. of the ACM*, 21(12):1040–48, 1978.

[4] C. Intanagonwiwat et al. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Mobile Computing and Networking*, pages 56–67, 2000.

[5] International Technology Roadmap for Semiconductors, 2005.

[6] A. Lines. Asynchronous interconnect for synchronous SoC design. *IEEE Micro*, 24:32–41, Jan/Feb 2004.

[7] K. Mai et al. Smart Memories: A Modular Reconfigurable Architecture. In *Proc. of the 27th Annual International Symposium on Computer Architecture*, June 2000.

[8] J. Patwardhan et al. NANA: A Nanoscale Active Network Architecture. ACM Journal on Emerging Technologies in Computing Systems (JETC), Vol. 2, No. 1, Pages 1-30, Jan 2006.

[9] J. P. Patwardhan et al. Circuit and System Architecture for DNA-Guided Self-Assembly of Nanoelectronics. In *Foundations of Nanoscience: Self-Assembled Architectures and Devices*, pages 344–358, Apr. 2004.

[10] J. P. Patwardhan et al. Evaluating the Connectivity of Self-Assembled Networks of Nano-scale Processing Elements. In *IEEE Intl. Workshop on Design and Test of Defect-Tolerant Nanoscale Architectures (NANOARCH '05)*, May 2005.

[11] B. H. Robinson and N. C. Seeman. The design of a biochop: a self-assembling molecular-scale memory device. *Protein Engineering*, 1:295–300, Aug. 1987.

[12] S. Rosenblatt et al. Mixing at 50GHz using a Single-Walled Carbon Nanotube Transistor. *Applied Physics Letters*, 87:153111, Oct. 2005.

[13] M. D. Schroeder et al. Autonet: A High-speed, Self-Configuring Local Area Network Using Point to Point Links. *IEEE Journal on Selected Areas in Communications*, 9(8), Oct. 1991.

[14] E. Winfree et al. Design and Self-Assembly of Two-Dimensional DNA Crystals. *Nature*, 394:539, 1998.

[15] H. Yan et al. DNA Templated Self-Assembly of Protein Arrays and Highly Conductive Nanowires. *Science*, 301(5641):1882–1884, Sept. 2003.