Opportunities and Challenges in Application-Tuned Circuits and Architectures Based on Nanodevices

Teng Wang
University of Massachusetts at Amherst
Amherst, Massachusetts, 01003
twang@ecs.umass.edu

Zhenghua Qi
University of Massachusetts at Amherst
Amherst, Massachusetts, 01003
zqi@ecs.umass.edu

Csaba Andras Moritz
University of Massachusetts at Amherst
Amherst, Massachusetts, 01003
addras@ecs.umass.edu

ABSTRACT

Nanoelectronics research has primarily focused on devices. By contrast, not much has been published on innovations at higher layers: we know little about how to construct circuits or architectures out of such devices. In this paper, we focus on the currently most promising nanodevice technologies, such as arrays of semiconductor nanowires (NWs) and arrays of crossed carbon nanotubes (CNTs). In contrast to general-purpose programmable fabrics (such as PLAs), we investigate nano-fabrics that, while also programmable and hierarchical, are more tuned towards an application domain (in this sense they resemble ASIC). Our goal is to achieve denser designs with better fabric utilization, efficient cascading of circuits, and routing of signals. We demonstrate detailed designs of several circuits and processor data-paths, and highlight associated challenges and opportunities for optimization.

Categories and Subject Descriptors

C.5.3 [**Computer Systems Organization**]: Computer System ImplementationMicrocomputers

General Terms

Design Performance

Keywords

NASIC, nanowire, microwire, FET, tile

1. INTRODUCTION

Semiconductor nanowires (NWs) and carbon nanotubes (CNTs) based device technologies are perhaps the most promising nanodevice technologies available today. This paper presents application-tuned nano-fabrics, circuits, and simple architectures, that can be realized using arrays of semiconductor nanowires and arrays of crossed carbon nanotubes.

CF'04, April 14–16,2004,Ischia, Italy.

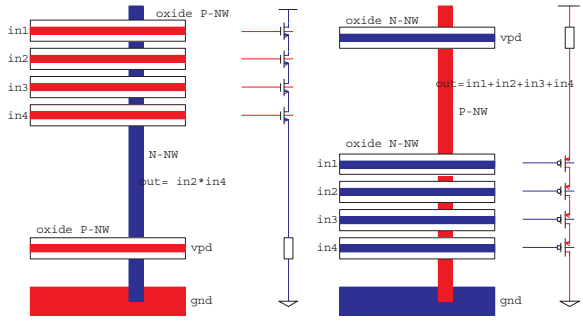

Figure 1: OR/AND logic implemented with SiNW FETs.

In contrast to general-purpose programmable fabrics (such as PLAs) our vision is to develop nano-fabrics that, while also programmable and hierarchical, are more tuned towards an application domain. Our goal is to achieve denser designs with better fabric utilization, efficient cascading of circuits, and routing of signals. We call these designs *NASICs*: *Nanoscale Application-Specific Integrated Circuits.* Density optimizations are critical despite that nanoelectronics based designs have an inherent density advantage compared to MOS. Much of this advantage could be lost, however, once fabrication constraints and fault-tolerance issues are taken into consideration. NASICs provide an opportunity to further optimize fabric density as compared to generic PLA-type of implementations.

NASIC designs are based on a hierarchical structure. Such a structure is necessary to effectively deal with the high defect rates expected in nanofabrics as well as with the additional device-specific topological, interconnect-related, and manufacturing constraints. The underlying nano-scale support is based on a grid of CNTs or NWs.

The grid junctions can be programmed either as FETs, P-N type diodes, or can be disconnected. Both diodes and FETs have been demonstrated based on NWs [15, 13] and CNTs [17]. Figure 1 (similar to [25]) illustrates simple 4-input OR and AND logic, implemented with SiNW FETs.

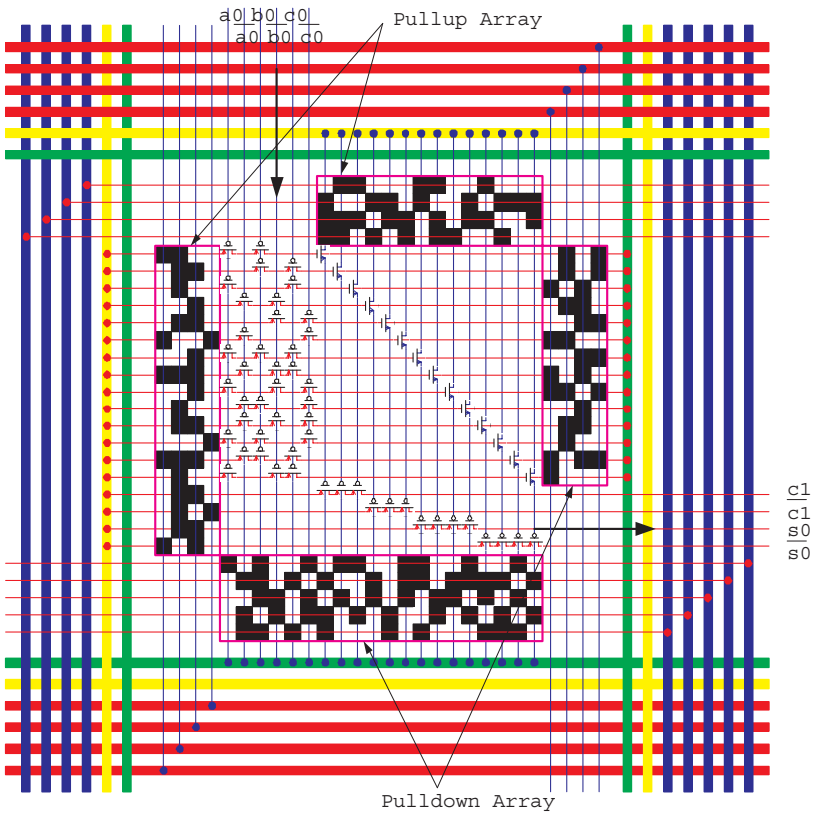

Figure 2: 1-bit adder realized with NW FETs.

The FETs are realized on the SiNW grid at junction points. On top of SiNW or CNT grids, we realize *NASIC tiles*. A NASIC tile consists of basic circuits such as adders, multiplexers, and flip-flops.

Ultimately NASIC circuits will have to be designed with *built-in redundancy* to provide a measure of local fault-tolerance, and used to implement application specific logic functions and registers.

The doping of nano-grid strips, the size of the NASIC tiles, the use of certain nano-scale (sub-lithographic) wires as interconnect between tiles and the micro-level interconnects are (to some extent) determined in an application / architecture-domain specific manner. These aspects determine a *NASIC fabric*.

These aspects are the key differentiators between PLA-type of nanoscale designs [24, 25] and NASICs. PLA-type of designs are also based on programming crosspoints in a nano-grid, but consist of alternating NOR-NOR (or other) logic planes of fixed sizes and wiring/routing between them.

Our preliminary work has demonstrated that there are considerable density advantages when such customization is done with a particular architectural/application domain (e.g., microprocessor data-paths) in mind. This advantage is primarily a result of more efficient circuits and cascading in NASICs. Our current work is based on static designs but we are also investigating basic components using dynamic circuit styles.

Several interconnected tiles form a larger *multi-tile*; connection between tiles is realized at the nano scale (with for example NWs or CNTs depending on the technology). Multi-tiles are assumed to be connected with microwires that provide efficient and reliable global communication. Additional microwires are used as address wires similar to [25] and provide the ability to program the logic in the tiles by determining the type (i.e., disconnect, FET, diode) of each crosspoint in the nanogrid. We envision that each level in the proposed hierarchical structure will have its own built-in fault tolerance, appropriate for that level. Our current strategy is to use built-in redundancy to provide fault-tolerance at the circuit level.

In subsequent sections we demonstrate detailed designs of several NASIC circuits, processor data-paths, a simple stream-processor with control, and highlight associated challenges and opportunities. We demonstrate possible optimizations, within the constraints of sub-lithographic fabrication, that improve fabric utilization. Our preliminary studies demonstrate the capabilities of NASIC designs by comparing them with conventional CMOS ones implemented in aggressive deep-submicron technology.

2. NANOELECTRONICS BACKGROUND

The basic structures available for nanocircuits are nanowires (NW) or nanotubes (NT). Both NTs and NWs are reported to have sizes down to a few nanometers and be capable of densities of 10^{11} to 10^{12} switches/cm^2 [15]. Both NTs and NWs have been produced in large numbers, but the electrical characteristics of NWs can be more reliably controlled [13].

Horizontal arrays of NT and NWs have been formed by fluidic alignment [13]. Both Samsung (Choi, patent #6,566,704)

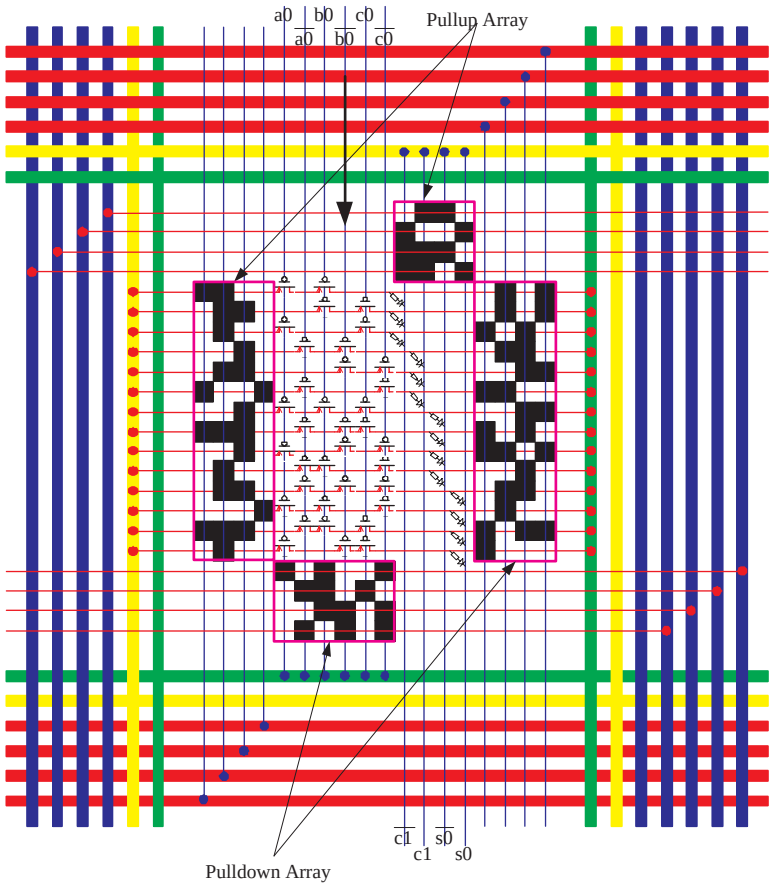

Figure 3: Optimized 1-bit adder realized with NOR-OR NW logic.

and Iljin Nanotech (Lee, patent #6,350,488) have patented methods of growing vertically oriented arrays of nanotubes with separations of less than 10 nm. Current control in NW or NT is exerted using gates formed in various ways, or by forming diode junctions. FET behavior has been achieved using metallic gates [17],[28] and crossing NW or NTs [13]. By varying the amount of oxide grown at their intersection, crossing NT or NWs can be made such that one NW forms a diode with the other, or one acts as a FET gate to the other, or they do not couple [13].

Radial Modulation Doping has been demonstrated by Lauhon [18] to control the doping profile radially on nanowires by changing impurities present as a function of time. A SiNW banded with differentiated conduction and gateable regions has been shown in [19]. This enables differentiated and coded wires.

Rapid progress is being achieved in the development of feasible logic devices. Diode resistor logic was demonstrated. At the same time restoring logic was introduced with nanowire FET-resistor logic [13]. Avouris from IBM made important progress toward low power logic by developing complementary devices on the same NT and demonstrated a CMOS-like nano-inverter [20]. Hewlett-Packard Research has developed a molecular crossbar latch (Kuekes, patent #6,586,965).

3. NASIC FABRICS AND TILE DESIGNS

This section presents several NASIC designs based on a programmed grid-structure of CNTs and NWs similar to [24, 25]. However, those works assume that logic functions will be mapped into the nano-fabric similar to PLAs without providing further detail on the actual implementation or density of the logic. By contrast, we investigate and provide details on actual circuits and NASIC tile designs that could utilize such an underlying technology most efficiently, in an application-tuned manner.

Several factors affect NASIC fabric utilization. These include the overhead due to microwires used for programming and global communication between tiles. Additional difference is due to using two-level logic, topological and device constraints.

Figure 2 shows a FET-based 1-bit full adder. The thicker wires represent microwires. The thin wires are nanowires and their color[1] shows the doping. The doping of the wires along the source-drain of a FET transistor determines the type of the transistor.

This circuit demonstrates the difficulty of obtaining good fabric utilization due to the presence of buffers. These buffers are needed to turn the corner in order to couple the PFET-based NOR plane (the input plane is actually AND on inverted inputs) that derives the initial product terms in the adder, and the NOR plane that generates the sum of the

[1]Colored viewing preferred; seen in B/W the P-wires are the horizontal wires and the N-wires are the vertical ones in Figure 2.

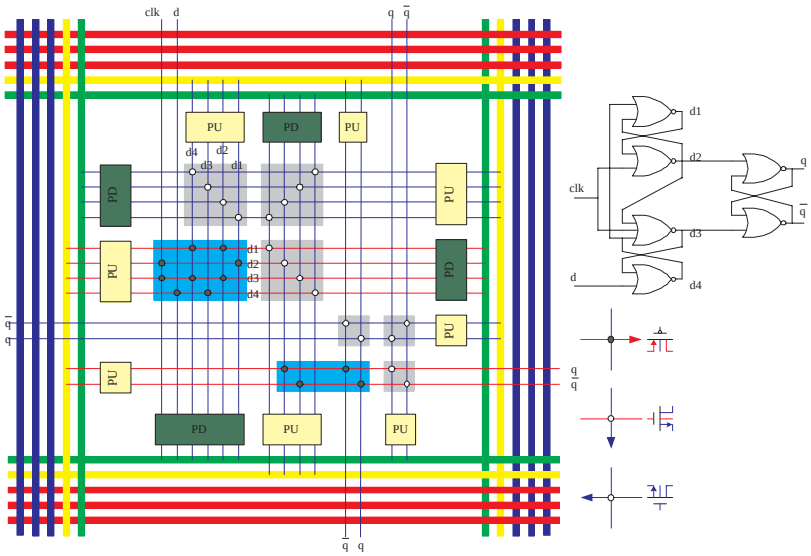

Figure 4: 1-bit Flip-flop design with NWs.

product terms.

The programming of the grid junctions is realized through a technique described in [25]. During nanoscale addressing, if we apply a voltage on the horizontal and vertical wires, we can program the state of the device at their intersection.

The interfacing between nanowires and microwires as well as the addressing of the junctions assumes a nanoscale decoder block similar to [24]. The decoders require at least $O(log(N))$ microwires (or more if redundancy is incorporated), where N is the number of the nanowires in one dimension, assuming a square NASIC tile.

An additional requirement is the decoder imprint pattern, as shown on the sides of the adder circuit (part of the pullup and pulldown arrays that are also used to connect to V_{dd} and Gnd) in Figure 2. Other schemes that do not require direct patterning (a weakness of this interfacing), e.g., based on stochastic self-assembly, are under development by several research groups [10, 26].

The adder circuit in Figure 2 assumes the availability of complementary inputs. We have found that it is often more efficient to create complementary outputs (to be able to provide complementary inputs when cascading circuits) rather than invert inputs locally.

If complementary inputs were not available, inverting would require dedicated PFET transistors, and could waste an entire rectangular area vertically under the inverting PFET transistors. As we will show in the next section, providing complementary outputs enables us to cascade a PFET-based NOR plane with a P-N diode-based OR plane to achieve a more efficient circuit.

The NOR logic on the left side of Figure 2 composes product terms (equivalent to AND logic) using PFET NW transistors. The right side of the design uses NFET transistors to turn the corner and directs the terms to a PFET-based NOR plane to calculate the sum of the product terms for the addition logic.

This design is somewhat similar to a PLA architecture, but shows the use of a buffer plane in between the NOR planes and a circuit-specific layout, and the sizes of the various planes, including the position of the pullups and pulldowns. The horizontal wires (after the PFET transistors) on the left side are the product term outputs, e.g, the first horizontal line is the logic function $\overline{(a_0 + b_0)}$ that is equivalent to $\overline{a_0} * \overline{b_0}$. Note that the fourth horizontal line uses the complementary inputs directly to achieve the logic function $a_0 * b_0$ at the input of the NFET buffer.

The outputs of the adder in Figure 2 are $s_0, \overline{s_0}, c_1, \overline{c_1}$. These are the sum of the product terms calculated with the NW PFET transistors in the bottom-right NOR plane. Note that both complementary and non-complementary outputs are derived. On each wire, current is flowing between the pullup and pulldown blocks. The NOR logic assumes a static-load ratioed design similar to the one presented in [25] where performance is mainly limited by the micro-nano contact resistance that is on the order of mega-ohms [13].

3.1 Optimized Adder Circuit with NOR-OR

Despite the high density of NW devices, the NW-FET based logic arrays require buffers between the (conceptual) AND and OR planes. This is because, unlike in PLAs, it is impossible to provide ground lines interleaved with the nano-wires (for pull-down evaluation). As a result, large portions of the circuit, perpendicular to the diagonal defined by the AND/NOR and OR/NOR planes cannot be utilized for active devices. This "diagonal" effect, however, can be avoided by replacing the NOR plane with diode-based OR logic and avoiding the buffers previously needed to turn the corner.

The optimized adder shown in Figure 3, replaces the NFET buffer and PFET logic that were used previously to calculate the sum of the product terms, with an equivalent diode-logic based plane. The input NOR part of the circuit remains the same as in the previous design. The use of the diode-logic plane effectively removes the area overhead of the NFET buffer and moves the final sum calculation into a vertically positioned logic block that significantly improves fabric uti-

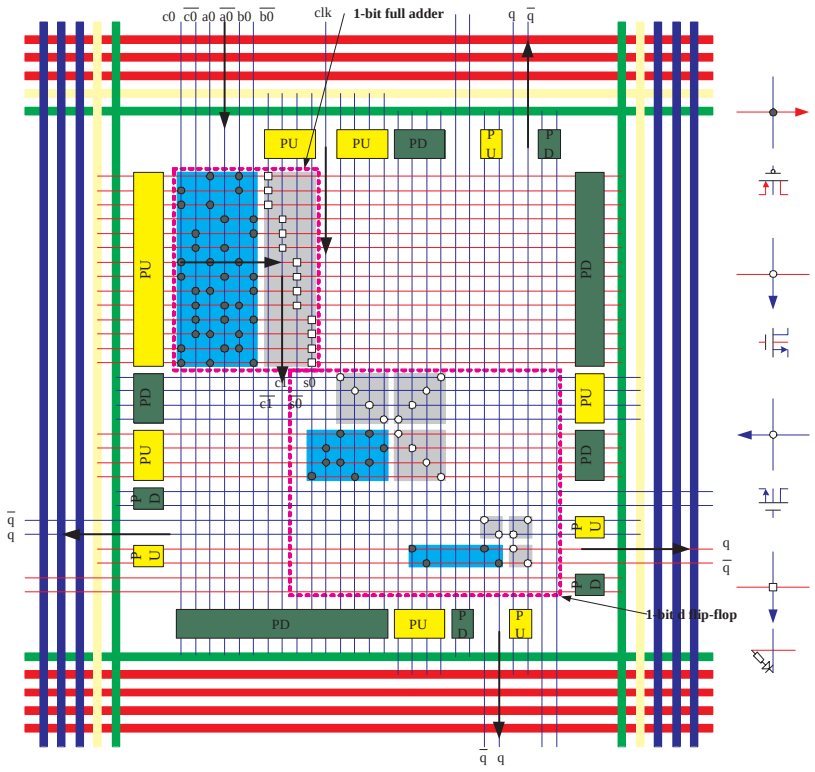

Figure 5: 1-bit data-path with NOR-OR NW logic.

lization.

While this is a non-restoring logic, the outputs eventually drive the next stage input NOR plane where the product terms are fully restored by a pullup or a pulldown network. The resulting configuration allows us to eliminate the local "diagonal effect", so that the (conceptual) AND and OR planes can be placed next to each other, as shown in Figure 2. However, the diagonal effect remains on the global scale, between logic arrays that implement given complex logic functions (see Figures 4,5), where such an effect is unavoidable.

3.2 Design Issues in Sequential Circuits

Figure 4 presents an example of a flip-flop circuit that could be used to provide small storage locally. The design illustrates the challenges when incorporating sequential circuits with NW/CNT-based devices in a grid-based fabric. The relatively low area utilization is due to a feedback path that is realized with three non-inverting NFET blocks that take the $d1, d2, d3, d4$ signals back to the input of the flip-flop. The figure also shows an additional area (see logic in the right corner) that is required to route the outputs $q, \overline{q}$ (bottom-right corner) of the flip-flop in each dimension – often necessary in practice. One insight from this is that larger scale SRAM-like memories[2] may be difficult to build.

Our initial strategy to provide storage in nano-architectures is to use flip-flops such as those shown in Figure 4. Such a solution may not result in dense memories, but can be adapted to implement small local storage elements. In our previous work [38], we have found that cost-performance optimal billion-transistor multiprocessor-on-a-chip designs would require just hundreds of bytes of storage per processing element for most of the scientific applications studied. NASIC designs could be orders of magnitude more fine-grained, suggesting that the required local memory per (processing) element will likely to be even smaller.

Another possibility we are investigating is the use of dynamic evaluation approaches where temporary storage might be possible to achieve without requiring explicit latching.

[2]Dense ROMs are easy to build as they are similar to PLAs

3.3 Design of One-bit Datapaths

Figure 5 shows a simple datapath which combines the optimized adder circuit (left-upper corner) and the flip-flop (middle). It shows the routing of the outputs in the four directions, to enable inter-tile cascading. Datapaths are one of the most common design elements one needs to address in a processor design.

Note that the area is defined primarily by the flip-flop. There is an additional area (right-bottom corner in the figure) required to realize the wiring necessary to route the outputs in several directions. In practice, a better utilization is possible if the combinational part of the tile is more complex. This will be shown in the next section, where an adder is combined with a multiplier and a multiplexer circuit and the logic is optimized together in a NOR-OR fashion similar to the NOR-OR adder presented earlier.

3.4 A Simple Stream-Processor

When several logic functions are implemented in the same NASIC tile additional logic optimizations can be performed. This section shows an example where combinational circuits in a NASIC tile are combined and optimized jointly rather

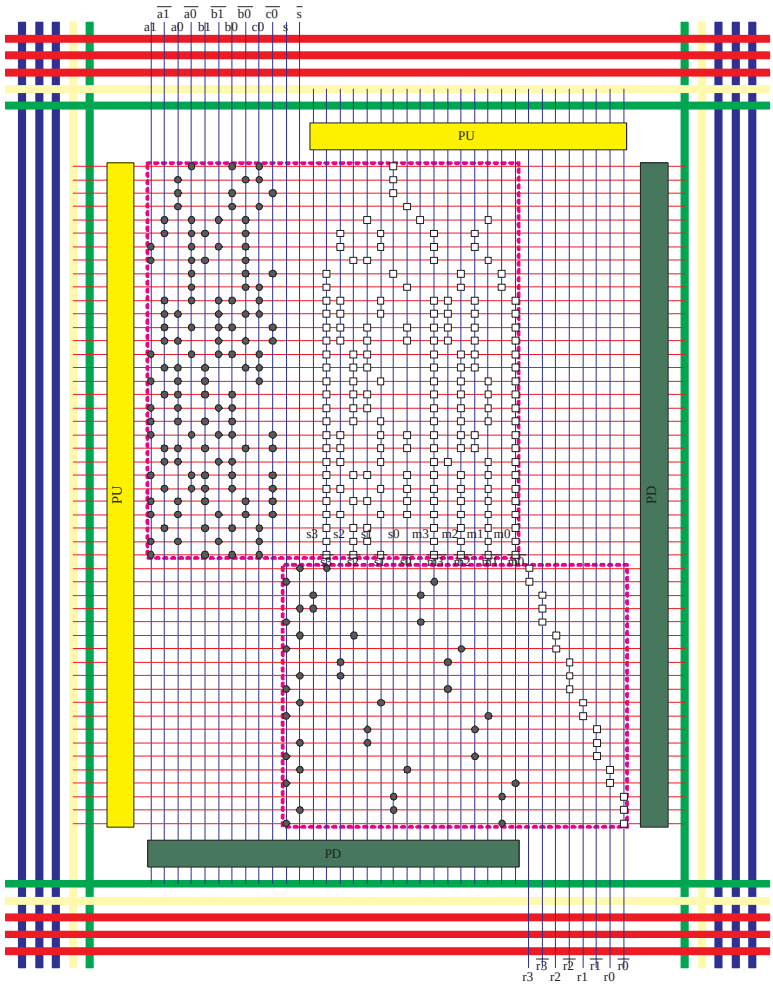

Figure 6: A simple stream-processor realized with NW FETs. The processor demonstrates the capability of selecting between a multiplier and an adder with a control input. The adder and multiplier logic are combined. The design includes a multiplexer at the bottom of the figure.

than individually. A simple example of a stream-processor is demonstrated where one of the inputs is a control bit that selects the functionality of the processor for a given set of data-inputs, and the results are streamed out on the other end of the design.

The top-left block in Figure 6 shows the logic required to implement a 2-bit adder and a 2-bit multiplier with PFET-based NORs and diode ORs. The signal outputs of the OR plane are restored in the multiplexer that is shown in the right-bottom corner of the design.

The logic optimization is performed with the two level logic minimizer tool *Espresso*, and the output is hand-mapped into transistor junctions and diode ORs to create an equivalent NASIC circuit. Note that good NASIC density is achieved on the combinational part. The OR logic enables us to compact the results of the add-multiply circuit so that a relatively dense circuit is achieved even for the multiplexer.

We have also implemented this circuit in multi-level CMOS and scaled it to BPTM 30-nm technology. The design has been synthesized with the Synopsys design compiler. The CMOS area corresponds to $12.23\mu m^2$. The nano-tile area with NWs ignoring the effect of microwires is $0.17\mu m^2$ or roughly 100 fold denser than CMOS. We estimate that around $2\times$ in density is lost to the difference between using two-level versus multi-level logic.

Assuming 90-nm pitched microwires (a design point that might be achievable by 2013 [22]) also accounted in the tile area, the area becomes $1.298\mu m^2$. In practice, we expect the logic implemented in a NASIC tile to be much more complex, reducing the overhead of the microwires on the tile utilization.

It is easy to show (also mentioned in [25]), that a grid area of roughly $1{,}000\times1{,}000$ NWs would reduce the microwire overhead to only 100%, giving a $50\times$ effective density ratio between the CMOS and nano implementations in our example.

Note that an important fraction of the overhead is due to the addressing microwires used for the programming of crosspoints. Better interfacing between the programming microwires and the nanowires would enable using smaller NASIC tiles for the same microwire overhead. This would, for example, use $K * log(N)$ microwires similar to [26], instead of $\sqrt{N}$ as is assumed above and in [24].

A state-of-the-art 32-bit ARM10TDMI core with six

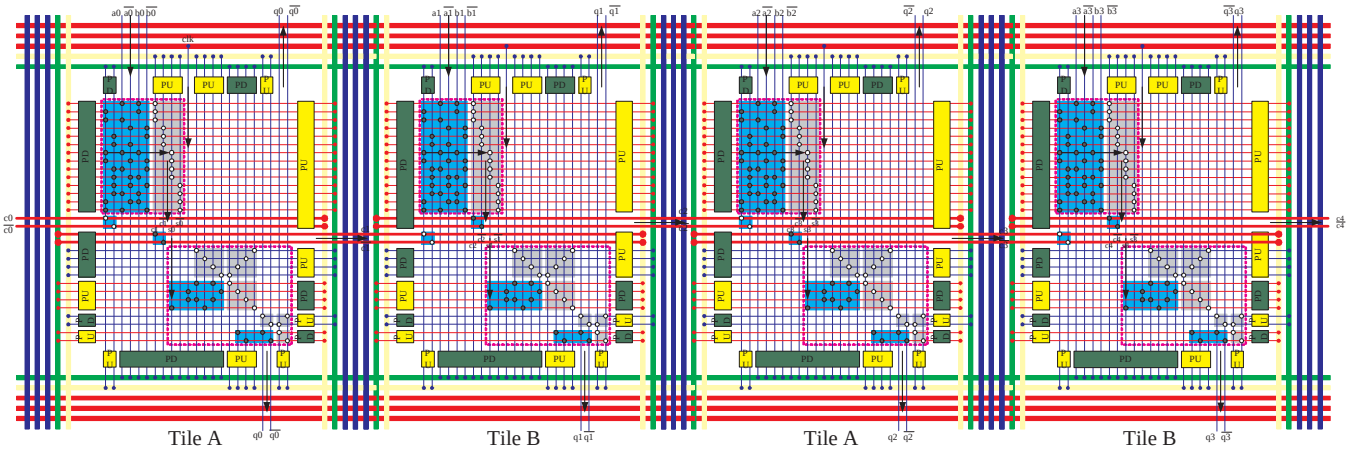

Figure 7: 4-bit adder realized with NW FETs in a 4-tile cascaded design. Two different tile types are necessary to achieve good fabric utilization.

pipeline stages requires 100,000 transistors when implemented in CMOS technology. A very simple single-issue datapath could be built with roughly 15,000 CMOS transistors.

Assuming 20% fabric utilization and a 2× density decrease due to the use of two-level logic, such a datapath could be realized in a NASIC tile of the size of around 400×400 NWs. Additional overhead might be required at the circuit-level for fault-tolerance purposes, as will be discussed in Section 5.

We expect CNT-based designs to be potentially much denser than NW-based ones, as CNTs could have 1-nm diameters compared to 3-nm for NWs. In our calculations above we used 3-nm NWs and 10-nm spacing.

4. STRATEGY FOR MULTI-TILE DESIGNS

We have completed initial multi-tile designs to investigate interconnection issues between tiles. Efficient nanoscale inter-tile interconnection and cascading of circuits are necessary before any larger-scale system architecture can be proposed.

Figure 7 shows an example of a 4-bit cascaded adder (i.e., a 4-bit ripple-carry adder) realized across four NASIC tiles. Realizing the same circuit in one tile would be highly area-inefficient because the output from each 1-bit full adder would need to be used as input for the next full-adder and so on, and would move the design into a diagonal shape. The area above and below the diagonal would be unused, resembling the problem with cascading the adder and the flip-flop in the 1-bit datapath shown earlier. One interesting insight with our design is that we use *two slightly different tiles* in our cascaded adder to accommodate the signals that are necessary for cascading. Every other tile is exactly the same.

More work is required to understand the requirements of communication in more complex multi-tile organization taking into consideration the involved fabric-specific constraints. Additional work we are currently completing cascades tiles in such a way that signals are routed in multiple directions. These designs are beyond the scope of this paper.

Once multi-tile NASIC designs are created, various architecture styles could be investigated and the implementation-level feasibility studied. We are currently building a multi-tile architecture-level simulation framework to enable architecture-level studies and reveal architecture-level trade-offs.

5. CONCLUSIONS AND FUTURE WORK

With CMOS technology approaching fundamental limits, the focus will increasingly shift to nanoelectronics based architectures. In this paper we have shown several circuits and multi-tile designs based on nanoscale devices within the constraints of bottom-up self-assembly manufacturing. We have identified several challenges as well as optimizations in larger-scale designs and in building combinational logic that also involves sequential circuits. These issues are key to understand the capabilities of nanoscale designs as computational systems and before specific architectural styles could be investigated. More work is required to understand the capabilities and limitations of larger-scale multi-tile NASIC architectures.

Another problem we are currently addressing is the high density of *defects* in the nano fabric which arise from a bottom-up, chemical assembly of nano wires and devices. Our current strategy is to use built-in redundancy to provide fault tolerance at the circuit level. Note that if we require that all inputs arrive in true and complemented form (and similarly, all outputs will be generated in true and complemented form), this already provides some level of redundancy also known as dual-rail redundancy.

6. REFERENCES

[1] J. Jacob, A. Mishchenko, "Unate Decomposition of Boolean Functions", Proc. Intl. Workshop on Logic Synthesis, 2001, pp. 66-71.

[2] Z. Hasan, D. Harrison, and M. Ciesielski, "Fast Partitioning Method for PLA-Based FPGAs", *IEEE Design & Test of Computers*, pp. 34-39, Dec. 1992.

[3] M. Ciesielski and S. Yang, "PLADE: A Two-Stage PLA Decomposition", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 943-954, Aug. 1992.

[4] R. Purri, A. Bjorksten, and T. Rosser, "Logic Optimization by Output Phase Assignment in

Dynamic Logic Synthesis", Proc. Intl. Conference on CAD, ICCAD'96, 1996.

[5] G. DeMichelli, A. Sangiovanni-Vincentelli, "Pleasure: a Computer Program for Simple/Multiple Constrained Unconstrained Folding of Programmable Logic Arrays", pp 263 - 270, Proc. ACM, IEEE Design Automation Conference, 1988.

[6] Jan M. Rabaey, "Digital Integrated Circuits - A Design Perspective", Prentice Hall, 1996.

[7] G. Hachtel, R. Newton and A. Sangiovanni-Vincentelli, "An Algorithm for Optimal PLA Folding," IEEE Trans. on CAD, pp. 1023-1028, 1980.

[8] D. E. Culler and J. Pal Singh and A. Gupta, "Parallel Computer Architecture: A Hardware/Software Approach," *ISBN 1-55860 343-3*, 1998.

[9] P. Wong, "Panel Session on Nanotechnology," *IEEE International Symposium on Signal Processing*, San Diego, 2002.

[10] *http://www.hpl.hp.com/2002/oct-dec/beyond_silicon.html*.

[11] R.F. Service, "Molecules Get Wired," *Science* **294**, 2442 (2001).

[12] G.Y. Tseng and J.C. Ellenbogen, "Toward Nanocomputers," *Science* **294**, 1293 (2001).

[13] Y. Huang, X. Duan, Y. Cui, L.J. Lauhon, K-Y. Kim, and C.M. Lieber, "Logic Gates and Computation from Assembled Nanowire Building Blocks," *Science* **294**, 1313 (2001).

[14] J.A. Hutchby, G.I. Bourianoff, J.V. Zhirnov, and J.E. Brewer, "Extending the Road Beyond CMOS," *IEEE Circuits and Devices* **18**, 28 (2002).

[15] 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* **289**, 94 (2000).

[16] S.C. Goldstein and M. Budiu, "Nanofabrics: Spatial Computing Using Molecular Electronics," *Proceedings of the 28th Annual International Symbosium on Computer Architecture* (2001).

[17] R. Martel, V. Derycke, J. Appenzeller, S. Wind, and Ph. Avouris, "Carbon Nanotube Field-Effect Transistors and Logic Circuits," *DAC 2002* New Orleans, ACM (2002).

[18] L.J. Lauhon, M.S. Gudiksen, D. Wang, and C. M. Lieber, "Epitaxial core-shell and core-multi-shell nanowire heterostructures", Nature 420, 57, 2002.

[19] M.S. Gudiksen, L.J. Lauhon, J.F. Wang, D.C. Smith, and C.M. Lieber, "Growth of nanowire supperlattice structures for nanoscale photonics and electronics", Nature 415, 617, 2002.

[20] Ph. Avouris R. Martel, V. Derycke, J. Appenzeller, "Carbon Nanotube Transistors and Logic Circuits," PhysicaB, Vol. 323, pp6-14, October 2002.

[21] Ph. Avouris, "Carbon Nanotube Electronics", Talk, http://www.research.ibm.com/nanoscience/.

[22] International Technology Roadmap for Semiconductors, 2001.

[23] S.C. Goldstein, and M. Budiu, "NanoFabrics: Spacial Computing Using Molecular Electronics," Proc. International Symposium on Computer Architecture, ISCA, 2001.

[24] A. DeHon, "Array Based Architectures for Molecular Electronics," First Workshop on Non-Silicon Computations (NSC-1), 2002.

[25] A. DeHon, "Array-Based Architecture for FET-Based, Nanoscale Electronics," IEEE Transactions on Nanotechnology, Vol. 2, No. 1, March 2003.

[26] A. DeHon, "Stochastic Assembly of Sublitographic Nanoscale Interfaces" IEEE Transactions on Nanotechnology, Vol. 2, No. 3, Sept 2003.

[27] Yi Cui, X. Duan, and C. Lieber, " Doping and Electrical Transport in Silicon Nanowires", The Journal of Physical Chemistry, Vol. 104, June, 2000.

[28] A. Bachtold, P. Hadley, T. Nakanishi, C. Dekker, "Logic Circuits with Carbon Nanotube Transistors," Science 294, 1317, 2001.

[29] A. DeHon, "Sub-litographic Semiconductor Computing Systems," HotChips 15, Stanford University, Aug 2003.

[30] M. Butts, A. DeHon, and S.C. Goldstein, "Molecular Electronics: Devices, Systems and Tools for Gigagate, Gigabit Chips," Proc. Intrnational Conf. on Computer-Aided Design, ICCAD'02, 2002.

[31] M. Mishra, and S.C. Goldstein, "Scalable Defect Tolerance for Molecular Electronics", First Workshop on Non-Silicon Computations (NSC-1), 2002.

[32] W.B. Culbertson, R. Amerson, R. Carter, P. Kuekes, and G. Snider, "Defect Tolerance on the TERAMAC Custom Computer," Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 116-123, Aprip 1997.

[33] E. Waingold, M. Taylor, D. Srikrishna, V. Sarkar, W. Lee, V. Lee, J. Kim, M. Frank, P. Finch, R. Barua, J. Babb, S. Amarasinghe, and A. Agarwal "Baring it all to Software: Raw Machines," *IEEE Computer*, September 1997, pp. 86-93.

[34] C. A. Moritz, M. Frank, "LoGPC: Modeling Network Contention in Message-Passing Programs," *IEEE Transactions on Parallel and Distributed Systems*, April, 2001

[35] D. Culler, R. Karp, D. Patterson, A. Sahay, K. Schauser, E. Santos, R. Subramonian, and T. Eicken, "LogP: Towards a Realistic Model of Parallel Computation," *Proc. of Fourth ACM SIGPLAN Symp. on Principles and Practices of Parallel Programming,* May 1993.

[36] A. Alexandrov, M. Ionescu, K. E. Schauser, and C. Scheiman. "LogGP: Incorporating Long Messages into the LogP Model" *Proc. of the SPAA'95*, Santa Barbara, CA, July 1995.

[37] D. Yeung, W. J. Dally, A. Agarwal. "How to Choose the Grain Size of a Parallel Computer". *MIT/LCS Technical Report, MIT-LCS-TR-739.*

[38] C. A. Moritz, D. Yeung, A. Agarwal, "SimpleFit: A Framework for Analyzing Design Tradeoffs in Raw Architectures," *IEEE Transactions on Parallel and Distributed Systems, June 2001.*

[39] J. C. Eble III. "A Generic System Simulator with Novel On-Chip Cache and Throughput Models For Gigascale Integration," *PhD thesis, Georgia Institute for Technology*, November 1998.

[40] A San Francisco Consulting Group Report. "Seven of the Most Promising Nanotechnology Companies -Report," Voted by Attendees Nanotech Venture Fair 2002 2002

[41] H. B. Bakoglu, "Circuits, Interconnects and Packaging for VLSI," *Reading, MA: Addison Wesley* , 1990.

[42] H. T. Kung. Memory Requirements for Balanced Computer Architectures, *IEEE* 1986. pp. 49-54.

[43] T. J. Holman and L. Snyder. "Architectural Tradeoffs in Parallel Computer Design," *Advanced Research in VLSI, Proceedings of the 1989 Decennial Caltech Conference*, The MIT Press. Cambridge, Massachusetts, March 1989. pp. 317-334.

[44] S.Y. Kuo and W. Kent Fuchs, "Fault Diagnosis and Spare Allocation for Yield Enhancement in Large Reconfigurable PLA's," *IEEE Trans. on Computers,* Vol. 41, pp. 221-226, Feb. 1992.