Latching on the Wire and Pipelining in Nanoscale Designs

Csaba Andras Moritz and Teng Wang
Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, MA 01003
{andras,teng}@ecs.umass.edu

Abstract

In contrast to general-purpose programmable fabrics, such as PLAs, we 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 and efficient cascading of circuits. We call these designs *NASICs*: *Nanoscale Application-Specific Integrated Circuits*. We believe NASICs are a more natural fit to implement microprocessors, out of semiconductor nanowires and crossed carbon nanotubes, than PLA style designs. A key challenge in nanoscale designs in general is to preserve the density advantages of the fabric once topological, interconnect, and fault-tolerance constraints are considered. In this paper we demonstrate possible optimizations, within the constraints of sub-lithographic fabrication, that improve NASIC fabric utilization. We show designs that are based on dynamic evaluation approaches, rather than static ratioed logic, that can provide the scalability necessary to build larger-scale pipelined architectures. We describe NASIC dynamic circuit styles that allow pipelining and temporary storage on the wire, called nano-latches, without requiring explicit latching.

1 Introduction

Sequential circuits are key components of computer architectures. Unfortunately, as we will show, it is hard to design high-density sequential circuits at the nano scale, based on traditional MOS-like approaches. To address this problem, we develop *implicit nano-latches*, which provide high-density temporary storage at the nano scale, without requiring explicit latching. Our approach preserves the density advantages of nanodevice-based fabrics, while permitting the design of complex, high-density, pipelined structures. The work presented here is part of our effort to build *NASICs*: Nanoscale Application-Specific Integrated Circuits. This paper builds on our previous work on static-style NASICs [2].

The most promising underlying nano-technologies today are semiconductor nanowires (NW) and arrays of crossed carbon nanotubes (CNT): we use grids of CNTs or NWs, with the grid junctions acting, as required, as FETs or diodes, or be disconnected. Other researchers have built FETs and diodes out of NWs [7] and CNTs [12].

The elemental units in NASICs are the *tiles*. These are circuits for adders, multiplexors, and flip-flops. Individual tiles can then be connected with nanowires or microwires to form a larger, multi-tile structure. Microwires also permit the tiles to be programmed by appropriately setting the type of each crosspoint (e.g., disconnect, FET, or diode) in the nanogrid.

NASICs face design challenges not encountered in the world of traditional microelectronic devices. For example, the defect levels in nano-fabrics tend to be quite high: we have to design enough fault-tolerance to sustain functionality in the face of a substantial fraction of the circuits being faulty. The overhead of micro-wires used for global communication and programming can also be quite high. Moreover, density tends to be less when two-level, rather than multi-level, logic is used. Additionally, there are topological and device constraints which must be taken into account. For example, when logic is cascaded or when sequential circuits are used, only the diagonal portion of the logic area tends to be utilized: most of the rest is essentially wasted.

This paper addresses this latter constraint that is critical to building pipelined designs without loosing significant density. But, how much density can we afford to loose before NASIC-like fabrics would loose their advantages compared to aggressive CMOS technology? To gain some insight into this question, we build an analytical simulator for tiled architectures and compare speedups obtainable on 30-nm CMOS technology with speedups on NASICs.

1.1 Technical Background: Nanoscale Devices

Nanotubes (NTs) and nanowires (NWs) have sizes of the order of a few nanometers, and their density can be as high as 10^{12} switches/cm^2 [9]. The electrical characteristics of nanowires can be more reliably controlled than those of nanotubes [7].

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 [12],[18], and crossing NW or NTs [7]. 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 [7].

Rapid progress is being made 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 [7]. 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 [14]. Hewlett-Packard Research has developed a molecular crossbar latch (Kuekes, patent #6,586,965).

While there are many practical challenges remaining, it looks like that it will be possible to build regular nanoarrays from uniform length CNTs or NWs. For example, p-doped horizontal NWs and n-doped vertical NWs can form a nanoarray. At the junctions of NWs, p-n diodes or FETs can be produced. Fluidic alignment, which involves suspending nanowires or nanotubes in a solution (e.g., ethanol) and then making the fluid flow along precut channels on a surface, allows one to construct arrays. Both Samsung (Choi, patent #6,566,704) 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.

1.2 Static NASIC Circuits

We illustrate two NASIC adder designs based on static ratioed logic. Our designs are based on a programmed grid-structure of CNTs and NWs similar to [15]. Other examples of NASICs are shown in [2].

Figure 1 shows a FET-based 1-bit full adder. 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 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 product terms.

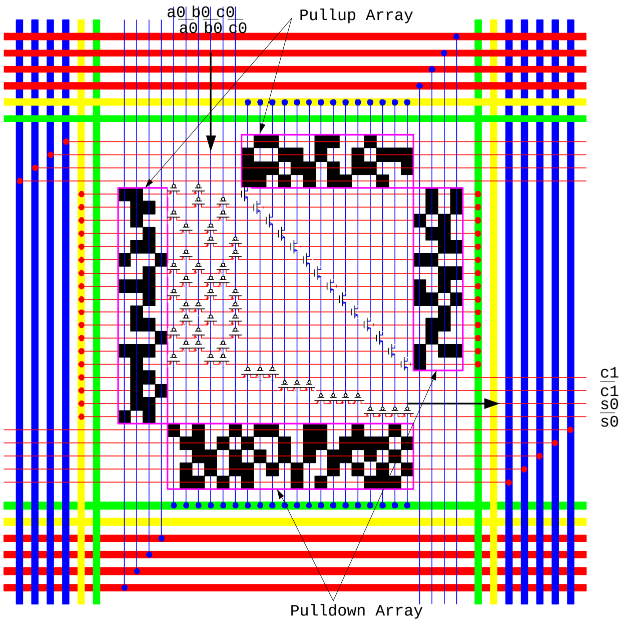

Figure 1: 1-bit adder realized with NW FETs. Note the impact of the buffer that moves the design into a diagonal shape.

The NOR logic on the left side of Figure 1 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.

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 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 2, 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 utilization. While this is a non-restoring

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

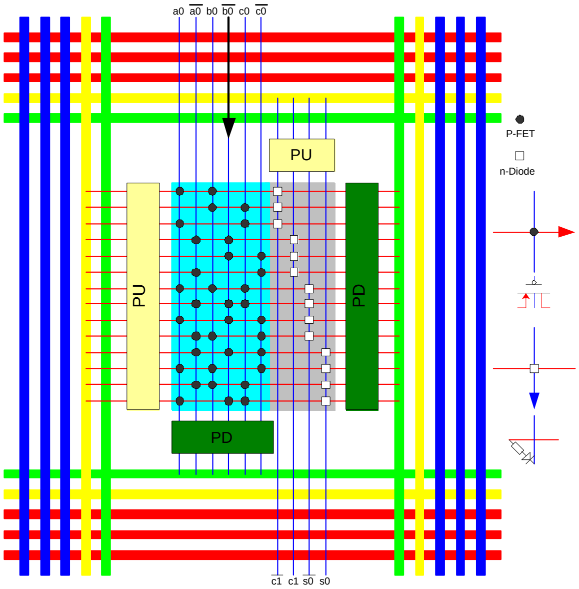

Figure 2: Optimized 1-bit adder realized with NOR-OR NW logic.

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.

2 Challenges in Designing Sequential Circuits

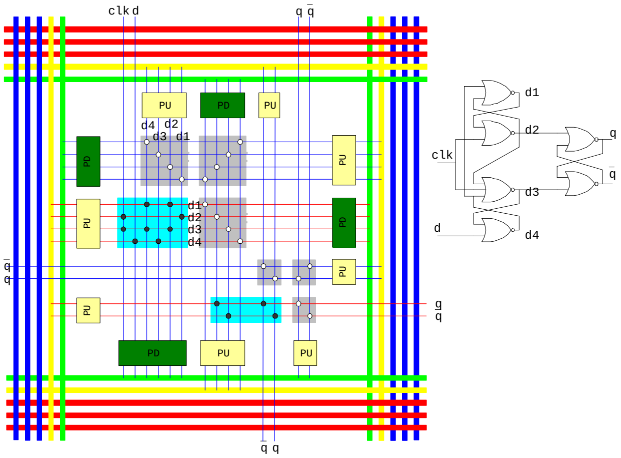

Figure 3: 1-bit Flip-flop design with NWs.

Figure 3 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 using latches or pipelined circuits is difficult.

One of the most common design elements in any processor design is the datapath. Figure 4 shows a simple datapath which combines an optimized adder circuit (left-upper corner) and the flip-flop (middle) together, and shows the routing of the outputs in the four directions, to enable inter-tile cascading. 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.

As seen, the flip-flop causes the design to move in a diagonal shape and significantly reduces the effective density of the design. Any additional logic in the tile would be pushed into a diagonal shape with much of the nanofabric density lost. To understand this question better we studied the impact of various effective fabric densities on overall speedups as compared to aggressive CMOS designs. This will give an idea of how important is to address the density impacting factors, e.g., due to sequential circuits, interconnects, or fault-tolerance.

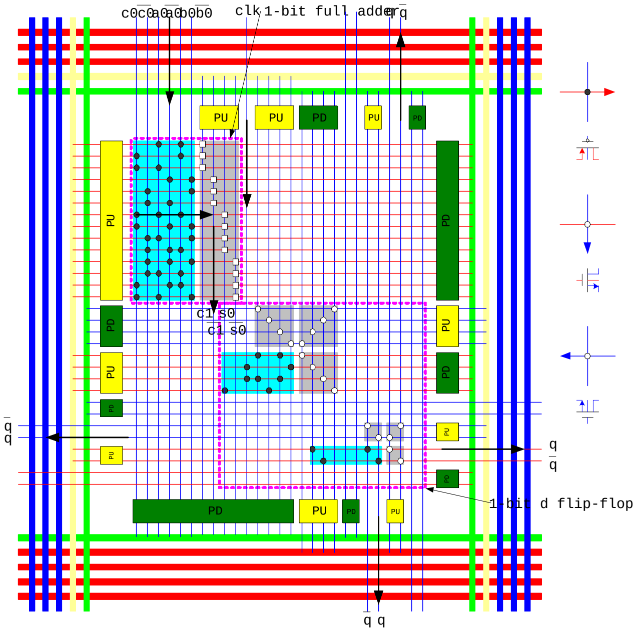

Figure 4: 1-bit data-path with NOR-OR NW logic.

2.1 Preliminary Density Study

To get some sense on how density impacts application speedups with NASICs, we have built a preliminary version of a high-level NASIC multi-tile simulator. Our objective is primarily to gain insight on when reduction in fabric density due to various effects in nanofabrics would start canceling out the advantages of NASICs versus aggressive CMOS designs.

To illustrate our point, we consider a multi-tile architecture based on a simple replicated single-issue datapath and simple storage. This primarily analytical model based simulation framework extends our multi-billion transistor CMOS architecture simulator described in [20], that was based on CMOS VLSI cost models. We modified this simulator to account for the characteristics of the nanogrid-based NASICs that we have shown earlier.

Our application model assumes that work is parallelized (with two levels of problem-size blocking) and distributed across the tiles similarly to [20]. Communication pattern/overhead is correctly modeled and so are the wire delays that impact communication latencies between nano-tiles. The actual latency thus also depends on the closeness of the neighboring tiles. We model nanowire delays based on *RC* delays. We approximate the capacitance similarly to [15] but designate the conductivity of the silicon nanowire as a variable (as it depends on the doping [17]). The parameter used in the simulation is the ratio of the nanowire resitivity to the resitivity of Al wires used in CMOS designs. The inter-tile nanowires used are 10^3 less conductive than the Al.

For each application studied, the simulator determines the area that is devoted to processing, local communication, local storage, and global communication as a function of application requirements. Space constraints prevent us from going into the details of the optimization framework [20] used in the simulator; rather, we focus here on the insights gained from the preliminary studies we have conducted.

To simplify our design, we assume that the processing in each tile is equivalent to just a simple single-issue datapath with a word size of 32 bits. We use our CMOS model [20] to calculate the number of transistors required to realize such functionality and then we scale to the requirements of a NASIC fabric. We base our study on the fact that it will be extremely difficult to achieve greater than 20% density [15]: thus we vary the effective utilization of our NASIC tiles from 1% to 20%.

We use this framework to evaluate the impact of the projected effective NASIC density on application speedups. As suggested in previous sections, there are many aspects that contribute to the effective density of a NASIC tile. For example, we have seen that sequential circuits impact NASIC density significantly.

To compare with CMOS technology we assume that the average density achievable in CMOS designs is equal to the density of a 2-input NAND gate with a fanout of 2, an approximation often used in practice for processors. We use a 30-nm process technology. In our nanoscale design we use 90-nm pitched microwires. We assume an 1.8 cm^2 total die area for both NASIC and CMOS designs. The die size selected is equivalent to an area required for 1 billion CMOS transistors with our average CMOS density assumptions.

Figure 5 shows the impact of the NASIC fabric utilization on application speedup, for four parallel kernels: *fft, lcs, Jacobi Relaxation, and Nbody* commonly found in scientific codes. A speedup of 1 corresponds to the baseline performance achievable with our standard CMOS fabric. We assume the same speed for NASIC as for the CMOS implementations[2].

The relatively poor speedup for Nbody reflects Nbody's more random local communication pattern that is the primary limiting factor even at higher fabric utilization rates. Another insight from this study is that while speedups compared to aggressive CMOS designs are possible, system-level architectures would need to be able to localize communication as much as possible, otherwise no benefits will be obtained even for high NASIC fabric densities.

Note that except Nbody, the other applications show that an NASIC fabric density of 5% still gives a good speedup relative to CMOS. Achieving 5% density might in fact prove to be challenging once topological, doping, programming, and fault-tolerance issues are addressed. This result further underlines the need to carefully optimize and consider all aspects that can preserve the density of nano fabrics.

In the next section we present new techniques that can improve NASIC density considerably by using implicit nano latches rather than explicit sequential circuits in pipelined designs.

3 Proposed Dynamic Designs, Nano-latches, and Pipelining

A key issue when building NASIC tiles, is the low effective density of latch circuits. This is due to the difficulty of building sequential circuits on a nanogrid. Moreover, latches require regions of differently doped nanowires along a dimension. While scientists have demonstrated differential doping [13], this could still be a limiting aspect.

Moreover, the circuits shown earlier are all based on static ratioed logic. There are several problems with these circuits. First, static ratioed circuits require careful sizing of devices for correct logic, that

[2] Current NASIC speed is limited by NW-microwire contact resistance [15] that is likely to improve with better manufacturing.

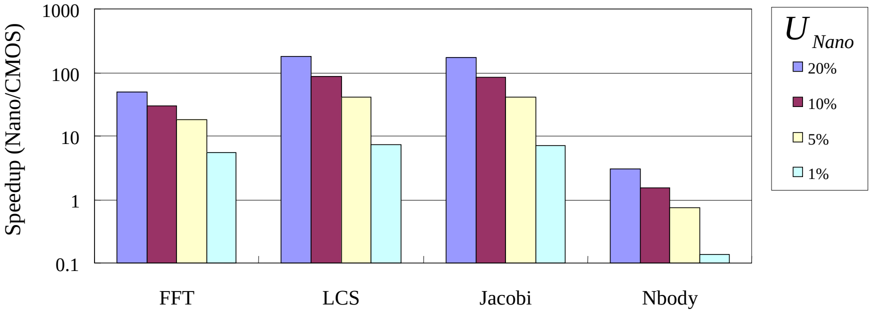

Figure 5: Impact of NASIC density on application speedup.

sets additional constraints on manufacturing. In addition, static power consumption due to direct-path currents would limit the scaling of NASIC tiles to arbitrary sizes. In this section, we show how to use dynamically cascaded circuits adapted to the nano-grid to alleviate these problems.

Conventional MOS designs apply various techniques, such as adding an inverter between the cascaded dynamic circuits, also called Domino logic, to guarantee functional correctness. However, we have found that the solutions that have been proposed for MOS (see [1] for an overview) are not suitable or difficult to realize once the constraints of the nano-grid (e.g., the two-level logic, the grid layout and doping related) are taken into consideration.

We propose a dynamic circuit style which requires only one type of doping in each dimension. This design also enables register-like behavior on the nanowires, without explicit latching, reducing the need for using the latch circuit presented earlier, in many designs, and enabling pipelined circuits.

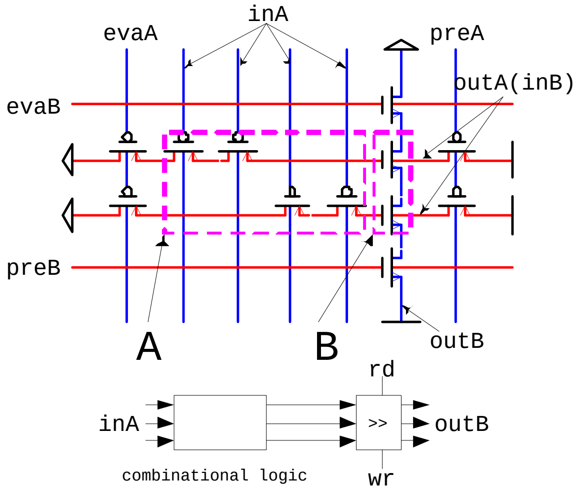

Figure 6: Dynamic combinational circuit and nano-latch

As shown in Figure 6, instead of using static pullup and pulldown arrays, we add dynamic precharge and evaluate transistors to control a precharge and an evaluate phase for each clock cycle. A logic function is implemented with a combination of an OR-NAND logic that are evaluated successively. The OR part corresponds to A and the NAND part to section B in Figure 6. Note that both A and B are using a precharge-evaluate sequence to generate an output, but they together form a pipeline consisting of two stages. A pipelined NASIC circuit (see Figure 8) can be built by cascading several circuits in this fashion. A novel aspect of our design for dynamic NASIC circuits, is the addition of the *hold phase* that is used to enable correct cascading. In addition, this phase also enables temporary storage of output values between circuits in NASIC tiles. Figure 7 shows a waveform that includes the precharge-evaluate-hold phases for both A and B. For example, the hold phase for A requires both preA and evaA to be high, to switch-off the corresponding transistors used for precharging and evaluation.

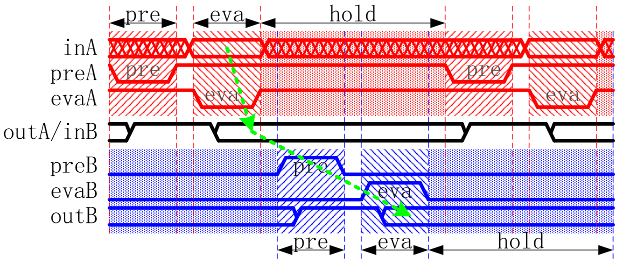

Figure 7: Waveform showing the signals necessary to control a dynamic NASIC circuit and the nano-latch.

In the precharge phase, the value of outA (that is the same as inB) is precharged to high voltage. In the evaluate phase, the circuit is discharging depending on the logic inputs inA. In the hold phase, the value of outA (or inB) is kept, because the input of the successive circuit is capacitively loaded, and preA and evaA are switched off.

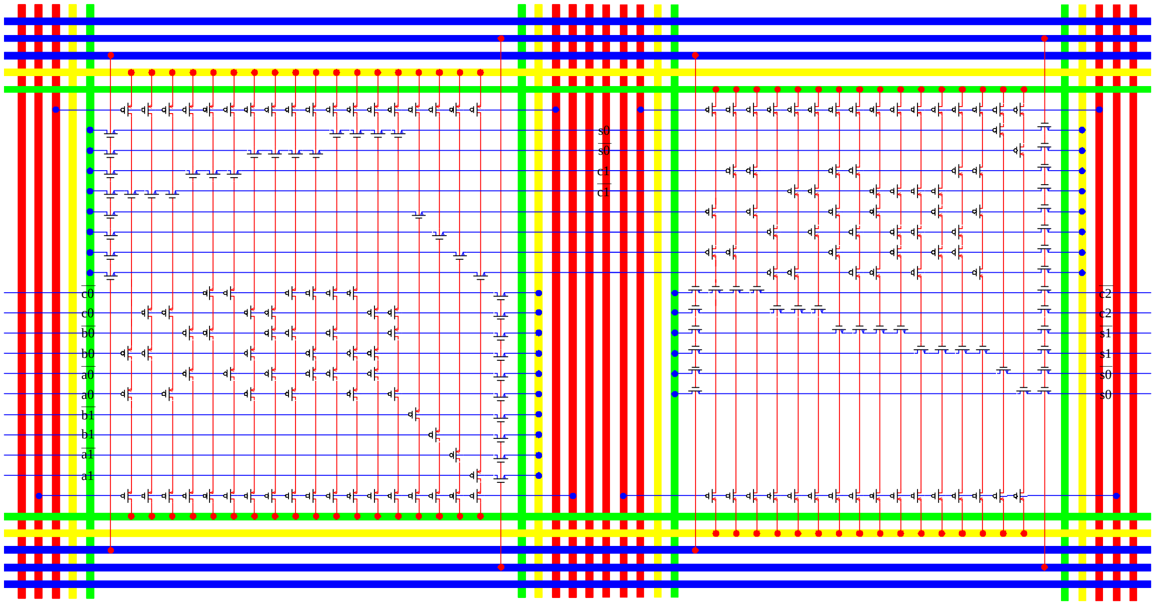

Figure 9: Pipelined 2-bit Full Adder.

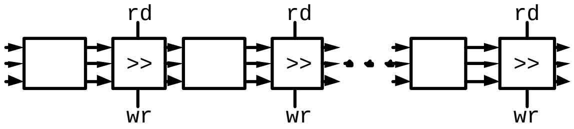

Figure 8: Block diagram of a pipelined circuit with stages separated by nano-latches.

There are several advantages to our proposed dynamic style for NASIC designs. First, we can implement an arbitrary 2-level logic with a simpler doping process. In these designs all horizontal and vertical nanowires will have the same doping. Second, and perhaps the most interesting aspect is that *we can latch data on nano-wires* without explicitly adding sequential circuits. We call this temporary storage on the wire the *Nano-Latch*. This is very useful as it can be a way to realize a pipelined structure without the problems we mentioned earlier with static and sequential circuit styles.

As a matter of fact the proposed dynamic circuit has the benefit that it can keep a value in a certain node for several cycles. Referring to Figure 7, if setting both preA and evaA to high, section A could stay in a hold phase storing a previously computed output for several cycles. The output of A is preserved. We can read this value by driving the precharge and evaluate transistors again. We can note that the precharge and evaluate signals of section A are write-enable controls and those of section B are read-enable controls for the nano-latch between A and B. Of course the value stored in a nano-latch (e.g., outA) could be lost due to leakage currents if section A stays in a hold phase for too long.

In conventional MOS technology, keeper devices are used to prevent the values stored in DRAM cells to be lost. We are currently investigating such circuits for the nano latches. An alternative, is to realize the keeper devices at microscale. Since a keeper device can serve a large number of nano devices, this strategy doesn't impact much on the area density.

3.1 Pipelined Adder

The example in Figure 9 shows a 2-bit pipelined full adder that avoids using sequential circuit based latches. Our objective is to show a simple example of a pipelined circuit with the proposed dynamic design. Refering to Figure 9, we demonstrate how to split a 2-bit ripple adder into 2 pipeline stages by using the Nano Latch concept and dynamic circuit style at nano scale. Signals *a0,b0,c0* are calculated first in the first stage, but *a1,b1* are shifted instead to the next pipeline stage. In the second stage, *a1,b1* and *c1* (from stage 1) are calculated. *s0* is also pipelined and input to the second stage.

4 Conclusion

With CMOS technology approaching fundamental limits, the focus will increasingly shift to nanoelectronics based architectures. Recent academic and industrial activity and successes at the device level support this trend. This paper presents several ideas to improve the density of NW and CNT based designs. In particular, the paper addresses approaches to reduce the density overhead of sequential circuits in pipelined nansocale architectures. We describe NASIC dynamic circuit styles that allow pipelining and temporary storage on the wire, called nano-latches, without requiring explicit latching.

References

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

[2] Teng Wang, Csaba Andras Moritz, "Exploring Nanoscale ICs and Architectures", to appear in ACM Computing Frontiers, Italy, April, 2004.

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

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

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

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

[7] 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).

[8] 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).

[9] 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).

[10] A. DeHon, "Array-Based Architecture for Molecular Electronics," *First Workshop on Non-Silicon Computation* **NSC-1**, (2002).

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

[12] 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).

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

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

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

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

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

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

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

[20] 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.*

[21] 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.

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

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