A Polymorphic Hardware Platform

Paul Beckett
RMIT University, Melbourne, Australia
pbeckett@rmit.edu.au

Abstract

In the domain of spatial computing, it appears that platforms based on either reconfigurable datapath units or on hybrid microprocessor/logic cell organizations are in the ascendancy as they appear to offer the most efficient means of providing resources across the greatest range of hardware designs. This paper encompasses an initial exploration of an alternative organization. It looks at the effect of using a very fine-grained approach based on a largely undifferentiated logic cell that can be configured to operate as a state element, logic or interconnect – or combinations of all three. A vertical layout style hides the overheads imposed by reconfigurability to an extent where very fine-grained organizations become a viable option. It is demonstrated that the technique can be used to develop building blocks for both synchronous and asynchronous circuits, supporting the development of hybrid architectures such as globally asynchronous, locally synchronous.

1. Introduction

In many ways the discussion of "coarse-grained" versus "fine-grained" architectures for reconfigurable computing is reminiscent of the early CISC vs. RISC debate. This latter debate was largely about how a mapping from high-level language to machine code could be best achieved - was it better to provide "solutions", i.e. complex features in the ISA that a compiler could use, or would a better way be to provide "primitives" from which more complex instructions could be built?

In the spatial domain, many of the same arguments are re-emerging - this time focusing on the hardware mapping process. Now the question is: will high configuration and routing overheads [1] always favor coarse-grained architectures that provide operator-level configurable functional blocks and/or word-level datapaths [2] over fine-grained organizations offering only logic primitives and interconnect from which these blocks can be built?

If the debate was to be based only on current FPGA organizations, then it might be said that the argument has already been fought and won: by coarse-grain style architectures [2]. A large number of platforms based on reconfigurable datapath units of various granularities have been proposed along with a range of synthesis tools (e.g. [3], [4], [5], [6]) while increasingly, commercial FPGA vendors are producing hybrid architectures incorporating both standard microprocessors and reconfigurable logic on the one chip (examples include the Virtex-II Pro "platform FPGAs" from Xilinx® [7] and the "Excalibur" series from Altera®[8]).

However, it appears likely that continued scaling into the deep sub-micron (DSM) region and from there into nano-scale dimensions may change this situation. New circuit opportunities are becoming available as a result of scaling and even CMOS devices will exhibit novel behavior at these dimensions. Ideas such as chemically-assembled molecular electronics [9], nanotube and nanowire devices [10], [11], [12], quantum dot techniques [13], [14] and magnetic spin-tunneling devices [15] have all been proposed as the basis of future, nano-scale reconfigurable systems. What these ideas have in common is that they tend to be characterized by reduced fanout (i.e. low drive), low gain and poor reliability [16]. Thus it is highly likely that future reconfigurable systems will be characterized by arrays of simple cells with highly localized interconnect. Just how these reconfigurable platforms will influence future hardware designs is an area of active research.

In a previous paper [17], a very fine-grained topology was described in which thin-body, fully depleted (FD), double gate (DG) MOSFETs and resonant tunneling diodes (RTDs) were combined to form a compact cell that could be said to exhibit "polymorphism" in that the cells were easily configurable to operate as state elements, logic, interconnect, or combinations of all three. A vertical layout style was exploited to hide the overheads imposed by reconfigurability to an extent where very fine-grained organization becomes a viable option. In this paper, the idea is extended to demonstrate how all of the components of a reconfigurable computing system can be formed from such array of locally connected cells. These

0-7695-1926-1/03/$17.00 (C) 2003 IEEE
Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)

components can encompass both synchronous and asynchronous logic circuits as both exhibit simple logic organizations with local feedback paths. This will form the basis for an exploration of these types of fine-grained architectures and their application to future reconfigurable systems.

The remainder of the paper proceeds as follows. Firstly in Section 2, the limitations of current FPGA organizations are reviewed, in order to provide a framework for this work. In Section 3, the operation of the basic components - the double gate transistor and resonant tunneling RAM - are briefly outlined and some reconfigurable logic organizations illustrated. Section 4 demonstrates that a simple, locally connected array of these cells can be configured into the various components of a reconfigurable architecture and are applicable to both synchronous and asynchronous systems. Finally, the paper is briefly summarized and some directions for future work outlined.

2. Reconfigurable architectures: the FPGA

To date, the "workhorse" of reconfigurable architectures has been the FPGA. However, by their very nature, FPGA organizations trade flexibility for sub-optimal delay performance and low area-efficiency. In this section the effects of interconnect delay and area efficiency on FPGA performance are reviewed with a view to setting the context for the development of the proposed reconfigurable hardware platform.

2.1 FPGA interconnect delay

For FPGAs using DSM technology, interconnect and wiring delays already account for as much as 80% of the path delay [1]. As devices scale, the effect of distributed resistance and capacitance of both programmable interconnect switches and wiring will become worse. Estimates by De Dinechin [18] indicate that if FPGA organizations stay the same, their operating frequency will only increase $O(\lambda^{1/2})$ with reducing feature size (λ), further widening the performance gap relative to custom hardware.

This is essentially the same problem faced by ASIC designers and as a result, future interconnect topologies are likely to include "fat" (i.e. unscaled) global wires plus careful repeater insertion [19]. Liu and Pai [20] have shown that even at the 120nm node, with low-K dielectrics and copper traces, transistors with extreme width to length ratios (in the order of 100:1) would be required to drive any significant length of interconnect with acceptable performance (e.g. driving a 1mm line with less than 100ps delay [20]). As a result, architectural solutions such as the pipelining of interconnect as well as logic [21], [22] may become increasingly important in the future.

2.2 FPGA area efficiency

Low area efficiency in FPGAs may arise from a number of sources. One obvious problem is that all logic components must exist, and thus occupy space, whether they are used in a particular mapping or not. This is illustrated in Figure 1 for a typical logic cell in which a particular mapping could result in any of the D-type flip/flop, the 3-LUT or the carry-multiplexer structures remaining unused. Numerous cell organizations have been proposed in an attempt to minimize the effect of this wasted space. Generally these have involved decoupling the various parts of a logic cell in order to permit their simultaneous use by the mapping process, hopefully leading to an overall reduction in hardware area. However, problems of logic allocation as well as routing congestion ensure that this is not always possible so some components must inevitably remain unused. Indeed, users of standard FPGA devices have come to recognize that leaving a percentage of the area unused is mandatory if a routing solution is be found in reasonable time [23] and Hutton [24] has observed that the underutilization of resources such as wires, memory, etc. actually represents a key "feature" that allows a variety of designs to be implemented on the same generic device.

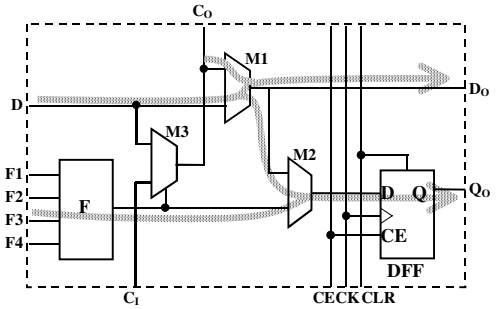


Figure 1. A Typical FPGA Logic Cell (from the XC5200 [7])

Inefficiencies in FPGA utilization may also occur at a more basic level. For example, a configurable 4-LUT can be seen to be an extremely poor implementation strategy if a single gate is all that is required [24]. It becomes an increasingly better strategy as logic complexity increases until the limit on the number of inputs and the effect of cascading starts to dominate [25]. Just what represents an "optimum" LUT size appears to be still an open question [26].

It could be argued that the area efficiency of the logic cells is unimportant in an FPGA as its total area is very much dominated by its routing structures. As a first order approximation, FPGA area is proportional to the number of configuration bits required to control the routing

0-7695-1926-1/03/$17.00 (C) 2003 IEEE
Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)

switches [1], [24]. This is one of the primary reasons that general-purpose FPGAs are poorly matched to standard datapath elements such as integer multipliers and floating-point operators – the regular structure of such operators ensures that they can always be implemented more compactly as purpose-built datapath units with optimal routing. It is this observation that is driving the move towards the inclusion of operational units into reconfigurable fabrics- from fixed-point multiplier blocks to entire CPUs. The tradeoff is that all of these units suffer exactly the same problems as conventional microprocessors: fixed word lengths [27] and worse-case performance ensure that in many cases they will be sub-optimally matched to the specific problem.

In summary, a "wish-list" of features for future reconfigurable systems might include the following items:

- flexible organizations that allow an area tradeoff to be made between the routing and logic
- an organization that reduces or hides the overhead imposed by reconfigurability;
- a very small footprint for logic and interconnect supporting a high density of components.
- structures that exhibits a simple timing model and that do not rely heavily on global interconnect.

3. Reconfigurable Technology

In this section, a reconfigurable platform based on double gate transistors is described that exhibits many of desirable features outlined above. The ultimate objective is to determine how homogeneous platforms such as this might be applied to future problems in reconfigurable systems – problems such as very large scale spatial computing [28], for example.

The fully depleted (FD) double gate (DG) transistor (Figure 2) is likely to be an important device technology as geometries move into the nano-scale region. It appears that that they will be ultimately scalable to gate lengths in the order of 10nm, although achieving the required level of dimensional control will be extremely difficult [29], as will overcoming device parasitics to reach acceptable performance targets.

One of the major advantages of DG technology is that the undoped channel region eliminates performance variations (in threshold voltage, conductance etc.) due to random dopant dispersion. Further, double gate transistors can be made very compact as they do not require the additional structures such as body contacts and wells that enlarge traditional CMOS layouts. The devices also exhibit a number of interesting characteristics that make them well suited to high-density reconfigurable architectures. They can theoretically be built on top of other structures in three-dimensional layouts and, most importantly for the application proposed in this paper, the second (back) gate offers a means of controlling the operation of the logic device in a way that decouples the configuration mechanism from the logic path.

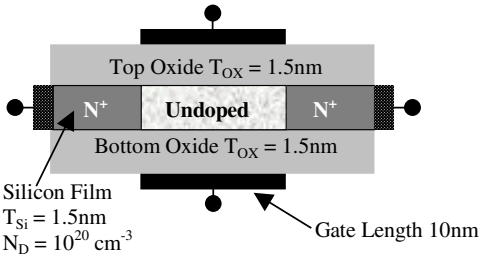


Figure 2. 10nm SOI-Si double gate NMOSFET (after [30])

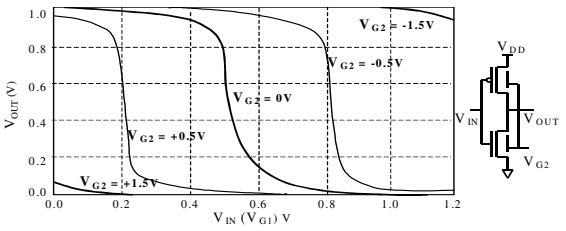
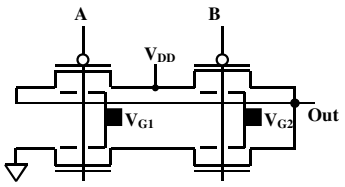
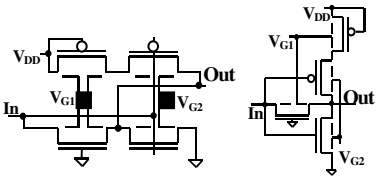


Figure 3. Configurable inverter example



Out	V_{G1}	V_{G2}
$\overline{A}$	0	2
$\overline{B}$	2	0
$\overline{A.B}$	0	0
1	-2	-2
0	2	2

Figure 4. A configurable 2-NAND gate



Out	V_{G1}	V_{G2}
$\overline{IN}$	-2	0
IN	2	-2
O/C	0	-2

Figure 5. Configurable inverting/non-inverting buffer structure

The basic idea for this reconfigurable system has been outlined in a previous paper [17] but is restated here for clarity. A simulation result for a simple double gate inverter circuit based on FDSOI MOSFET models [31] is illustrated in Figure 3. It can be seen that altering the bias on the back gate (V_{G2}) moves the voltage threshold of the p and n-type transistors such that the switching point of the inverter can be moved over the full logic range of the gate. At the two extremes, the output stays high (for V_{G2}<-1.5V) or low (for V_{G2}>1.5V) while for values of V_{G2} around 0V, the output switches symmetrically.

0-7695-1926-1/03/$17.00 (C) 2003 IEEE
Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)

Figure 4 and Figure 5 illustrate how this basic mechanism can be exploited to form more complex logic circuits. The circuit of Figure 4 is essentially a 2-NAND gate in which each complementary pair of transistors is controlled by an individual back gate bias voltage (V_A and V_B – shown as black squares on the diagram). The table outlines the enhanced set of logic functions that can be developed using this technique. In Figure 5, the same group of transistors has been reorganized into a structure that that can be configured to behave as either an inverting or non-inverting 3-state driver. Note that, as complementary operation is maintained in all cases, static power consumption will be minimized. Previous proposals for reconfigurable logic using carbon nanotube devices [12] and chemically assembled technology [9] have been based on nMOS-like structures, thereby relying on their inherent high resistance to ensure scalability.

To be useful, any configuration mechanism used for this system has to be able to develop the three bias voltages without occupying significant space or consuming excessive power. A plausible mechanism for this purpose can be based on resonant tunneling (RT), a mature technology that has been known and used for many years. The negative differential resistance (NDR) characteristics of RT devices directly support multi-valued logic [32] of the sort required by the reconfiguration system and a number of 3-state memory cells have already been proposed [33], [34], [35].

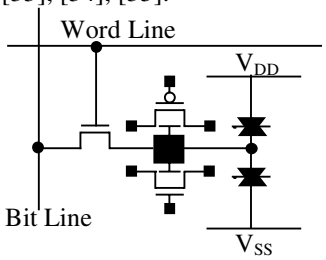


Figure 6. Leaf-Cell / RTD memory

Figure 6 shows a reconfigurable "leaf-cell" formed from three FDSOI transistors, and a RTD RAM of the type described in [34]. To merge the RAM and the logic mesh will involve matching the V_G values required to set the double gate transistors into their three operating regions with the RAM tunneling voltages which are, in turn, set by adjusting the thickness of each of the RTD layers [36]. While silicon interband tunnel diodes with adequate room temperature peak-to-valley current ratios have recently been reported [37], [38], it is possible that III-V technology may be more appropriate to this application as it may be easier to achieve the required operating voltages. It has already been shown [39] that uniform and reproducible III-V layers, that are also compatible with conventional (CMOS) integrated circuit processes, can be produced using molecular beam epitaxy.

The Nanotechnology Roadmap [40] predicts that by 2012, RTDs will scale to about 50nm and operate with peak currents in the 10 to 50pA range. At the limits of scaling for the FDSOI devices (~10nm), it is envisaged that these could be integrated into a compact vertical stack, such that the top of the lower RTD mesa forms the back gate of the complementary pair. The basic cell could then be replicated into a very large array – with potential densities in excess of 10^9 logic cells/cm^2. Even at this scale, the configuration circuits would be likely to consume less than 100mW of static power. Local interconnect to adjacent cells would complete the logic cell layout.

4. Polymorphic Hardware

Having created what is, in essence, an undifferentiated leaf-cell, the question remains as to the best way to deploy it. An example of a reconfigurable array constructed using this technique is shown in Figure 7. In this case the basic logic block is arranged as a 6-input, 6-output NAND array with each (horizontal) output terminated in a configurable inverter/3-state driver of the type described in Figure 5 (only one set is shown). The latter circuit serves a number of purposes. In its off-state, it decouples adjacent cells and determines the direction of logic flow. Configured as an inverting driver, it supports the creation of more complex logic functions and, just as importantly, provides a buffer that will allow any output line to be used as a data feed-through from an adjacent cell. Finally it can be set up as a simple pass-transistor connection to the neighboring cell.

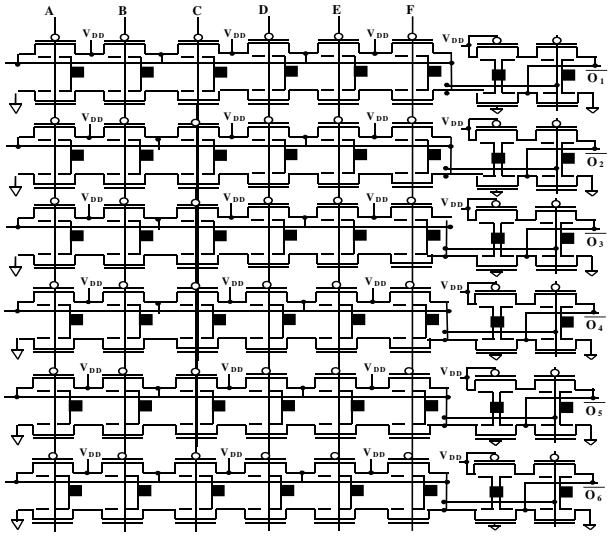


Figure 7. An example reconfigurable cell based on a 6x6 NAND organization.

From the outside, the reconfiguration array appears as a simple (albeit multi-valued) 8x8 RAM block and would

0-7695-1926-1/03/$17.00 (C) 2003 IEEE

be controlled by set of (multi-valued) RAM drivers surrounding the array and forming a link to a reconfiguration bit stream. In this organization, each block requires 128 bits reconfiguration data – in the same order (on a function-for-function basis) as the several hundred bits required by typical CLB structures and their associated interconnects in FPGA devices.

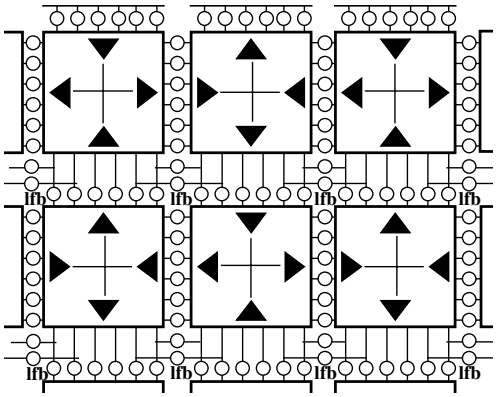


Figure 8. Partial array layout showing the orientation of adjacent logic cells

In Figure 8, the NAND cells are organized into an array with adjacent connections in the vertical and horizontal direction. The white circles represent the 3-state drivers, while the black arrows indicate the potetial I/O directions of each cell (although this will depend on whether a particular connection is configured or not). Note that adjacent cells are rotated by 90° such that the outputs from each cell abut the inputs of the two adjacent cells. Pairs of cells, configured together, represent the equivalent of a small LUT with 6 inputs, 6 outputs and 6 product-terms. The two local connection lines (labeled *lfb* in Figure 8) support the feedback necessary to develop state functions such as flip-flops and latches as well as allowing a small amount of logic cascading. Because of the regularity of the structure and the adjacent connectivity, the array has the potential to be very dense – a pair of LUT cells could occupy less than $400\lambda^2$, for example. This can be contrasted with estimates in which the area of a "typical" 4-input LUT could be as high as $600K\lambda^2$ if the programmable interconnect and configuration memory are included [1].

In Figure 9, one functional pathway in a typical FPGA has been implemented as a way of illustrating how the logic mapping in the proposed scheme compares to that of a conventional FPGA (the dots in this figure represent the leaf-cells that have been enabled – the remainder are configured *off*). Four of the NAND-cells form a 3-LUT (2 cells) plus an edge-triggered D-type flip-flop (2 cells). As the right-most LUT cell uses only four NAND-term lines, the remainder of that cell is used to bring in the reset line connection and to develop the complementary clock signals. Standard asynchronous state machine techniques can be used to develop logic equations for the edge-triggered flip-flop while an alternative, level-triggered (transparent) latch circuit can be built using the same number of cells. It is clear from the layout that the FPGA components that are not needed for this particular logic decomposition, are simply not instantiated – including, of course, the static configuration multiplexers.

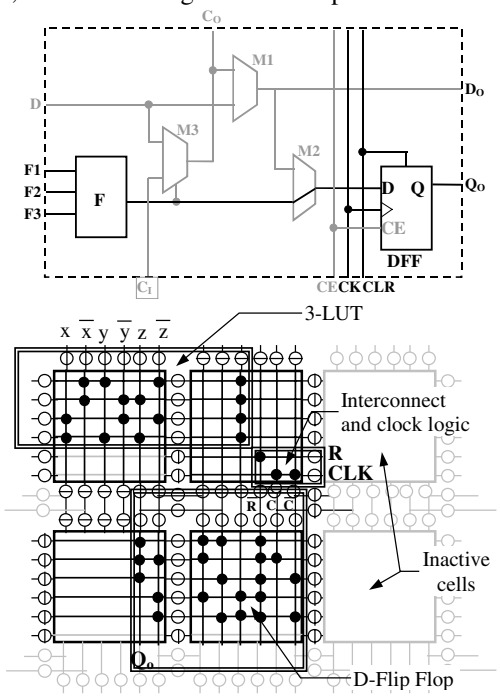


Figure 9. A Configured Logic Cell forming a 3-LUT and Flip-Flop. The 3-LUT logic shown is $x \oplus y \oplus z$

A partial view of an example datapath instantiation is shown in Figure 10. The sharing of terms between the sum and carry allows a full adder to be implemented in just five terms and if the two horizontal connections between adjacent cells are used to transfer the ripple carry between bits of the adder, each bit will fit within one 6-NAND cell pair. In a standard random logic environment such as a standard cell based ASIC or even a commercial FPGA, decomposition to the level of NAND gates would make little sense as it would be likely to result in a very inefficient (i.e. interconnect dominated) structure. The scheme proposed here is reminiscent of the sort of layout derived from a module generator targeting a "sea-of-gates" style implementation [41] and takes advantage of the regularity of these datapath structures. Of course, specialized support hardware such as fast carry chains will not be not available in this system. However, there is already some evidence (e.g. [42]) that functionality of this sort will be less effective when interconnection delay dominates and alternative techniques such as bit-serial

0-7695-1926-1/03/$17.00 (C) 2003 IEEE
Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)

arithmetic and asynchronous logic design may offer equivalent or better performance at these dimensions.

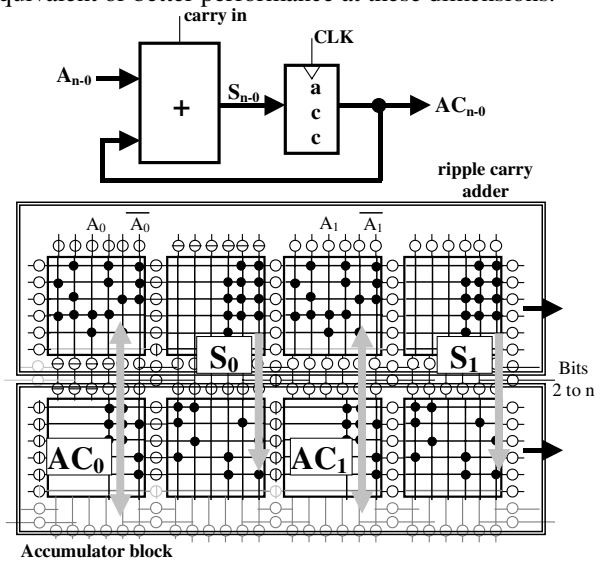


Figure 10. Datapath example (2 bits shown)

4.1 Asynchronous logic

The power consumed by global clock generation and distribution is already a major issue with current high performance (synchronous) processors [43] and is already impacting larger reconfigurable systems. Numerous asynchronous design techniques (e.g. [44]) have been proposed to eliminate the need for such global clocks. While it is still unclear as to whether totally asynchronous design styles offer actual improvements in overall performance, they are at least as good as conventional synchronous approaches and the removal of the global clock will, on its own, result in significant power savings.

An interesting concept that is likely to be important in the future is *globally asynchronous, locally synchronous (GALS)* where a system is partitioned into many clock domains and "asynchronous wrappers" [45] are provided for modules (probably in the range of 50-100K gates [19]) across which the synchronous clock delay is considered to be acceptable. The partitioning of a hardware platform into such modules immediately raises a problem that is somewhat analogous to the choice of page size in a hierarchical memory system in which fixed page sizes can lead to inefficient memory allocation and fragmentation problems. Ideally, the selection of module sizes would be entirely unconstrained - especially in dynamically reconfigurable systems [46]. Overall, this argues for a fine-grained configurable platform that exhibits the flexibility to be arranged into variable sized computational modules based on both asynchronous and synchronous logic elements.

Current programmable systems tend not support hazard-free logic implementations [47]. Nor do they include special functions such as arbiters and synchronizers. In the archetypal asynchronous organization described by Sutherland [48] (Figure 11), a series of Muller C-elements control the data flow between pipeline registers (called "event controlled storage elements" by Sutherland). A C-element exhibits the logic form: $c = a.b + a.c' + b.c'$ [44] where a and b are the inputs (the *Req* and *Ack* signals derived from adjacent control cells in this case) and c' is its current output. In common with most asynchronous logic building blocks, both the C-element and the pipeline registers can be described in terms of small asynchronous state machines of a form that is directly supported by the array organization outlined in Figure 8. This is illustrated in Figure 12 for a single bit of a pipeline register and indicates that applying fine-grained organizations of this sort will provide a workable approach to the design of asynchronous and GALS style microarchitectures.

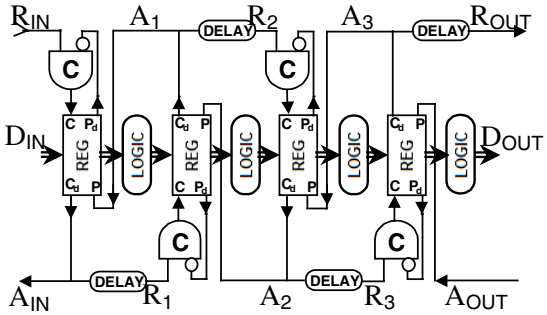


Figure 11. Micropipeline organization (from [48]).

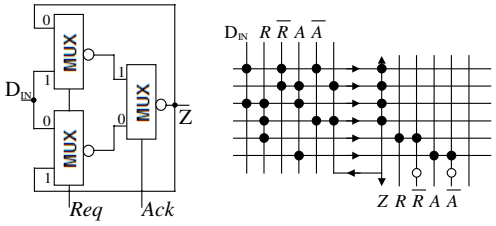


Figure 12. Event-controlled storage element (from [48]) and its implementation using reconfigurable blocks

5. Conclusions

In the domain of spatial computing, it seems that the high configuration and routing overheads associated with current FPGA architectures are favoring coarse-grained organizations that provide operator-level configurable functional blocks and/or word-level datapaths. In the context of current FPGA technology, this is an entirely reasonable approach – it would make little sense to spend six transistors to configure a four-transistor 2-NAND gate, for example.

However, it is possible that the low current drive, low gain and poor reliability of future DSM and nano-scale

0-7695-1926-1/03/$17.00 (C) 2003 IEEE
Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'03)

devices may reverse this trend. As a first step in an investigation into the way that future nano-scale device characteristics may affect reconfigurable systems, a very fine-grained reconfigurable platform, based on complementary, fully depleted dual-gate thin-film transistors has been described. While the technology challenges are manifold, such devices offer a number of tangible benefits, not the least of which is a plausible migration path from conventional CMOS.

It has been demonstrated that this reconfigurable technique can be used to develop simple combinational logic and asynchronous state machines thereby supporting a wide range of digital logic circuits. It is a fairly straightforward matter to generate layouts that are more-or-less equivalent to current FPGA components (LUTs, registers, multiplexers etc.). Further, as components that are not needed for a particular logic decomposition are not instantiated, the configuration mechanism is "hidden" by a vertical layout style, and the same components can be used interchangeably for logic and interconnection, the technique can lead to substantial reduction in the overall implementation size – possibly as large as three orders of magnitude over current FPGA devices.

Interconnection performance will be an important issue determining the operation of architectures at nano-scale dimensions – especially with device technologies such as single-electron and molecular electronics. Locally connected, highly pipelined organizations appear to be a good match to these characteristics but further work on the development of better models for the expected characteristics of the devices will be necessary before this is verified one way or the other. However, it already appears that interesting designs can be constructed from entirely locally connected building blocks. Future work will look at effect of these local interconnect constraints on system architecture as well as higher-level issues such as the performance of serial vs. parallel design styles and the comparative performance of synchronous, asynchronous and hybrid organizations.

6. References

[1] A. DeHon, "Reconfigurable Architectures for General-Purpose Computing," MIT, Massachusetts, A.I. Technical Report 1586, October, 1996.

[2] R. Hartenstein, "The Microprocessor is No Longer General Purpose: Why Future Reconfigurable Platforms Will Win," presented at Second Annual IEEE International Conference on Innovative Systems in Silicon, IEEE, pp 2 -12, 1996.

[3] R. Hartenstein, M. Herz, T. Hoffmann, U. Nageldinger, "Mapping Applications onto Reconfigurable KressArrays," presented at 9th International Workshop on Field Programmable Logic and Applications, FPL '99, Glasgow, UK, 1999.

[4] T. J. Callahan, Wawrzynek, J., "Adapting Software Pipelining for Reconfigurable Computing," presented at International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, CASES, 2000.

[5] C. Ebeling, Cronquist, D. C. Franklin, P., "RaPiD Reconfigurable Pipelined Datapath," presented at Field-Programmable Logic: Smart Applications, New Paradigms, and Compilers. 6th International Workshop on Field-Programmable Logic and Applications, 1996.

[6] H. Singh, G. Lu, E. Filho, R. Maestre, Ming-Hau Lee, F. Kurdahi, N. Bagherzadeh, "MorphoSys: Case Study of a Reconfigurable Computing System Targeting Multimedia Applications," presented at Proceedings of the 37th Conference on Design Automation, Los Angeles, CA USA, pp 573 - 578, 2000.

[7] Xilinx, www.xilinx.com.

[8] Altera, www.altera.com.

[9] S. C. Goldstein, M. Budiu, "NanoFabrics: Spatial Computing Using Molecular Electronics," presented at 28th International Symposium on Computer Architecture, Goteborg, Sweden, pp 178 - 189, 2001.

[10] T. Rueckes, K. Kim, E. Joselevich, G. Y. Tseng, C-L. Cheung, C. M. Lieber, "Carbon Nanotube-Based Nonvolatile Random Access Memory for Molecular Computing," *Science*, vol. 289, pp. 94 - 97, 2000.

[11] A. Bachtold, P. Hadley, T. Nakanishi, C. Dekker, "Logic Circuits with Carbon Nanotube Transistors," *Science*, vol. 294, pp. 1317-1320, 2001.

[12] A. DeHon, "Array-Based Architecture for Molecular Electronics," presented at First Workshop on Non-Silicon Computation (NSC-1), Cambridge, Massachusetts, 2002.

[13] C. S. Lent, Tougaw, P. D., Porod, W., Bernstein, G. H., "Quantum Cellular Automata," *Nanotechnology*, vol. 4, pp. 49-57, 1993.

[14] M. T. Niemier, Arun F. Rodrigues, Peter M. Kogge, "A Potentially Implementable FPGA for Quantum Dot Cellular Automata," presented at First Workshop on Non-Silicon Computation, NSC-1, Cambridge, Massachusetts, 2002.

[15] R. Richter, H. Boeve, L. Bär, J. Bangert, G. Rupp, G. Reiss, J. Wecker, "Field Programmable Spin-Logic Realized with Tunnelling-Magnetoresistance Devices," *Solid-State Electronics*, vol. 46, pp. 639-643, 2002.

[16] R. Ronen, A. Mendelson, K. Lai, Shih-Lien Lu, F. Pollack, J. P. Shen, "Coming Challenges in Microarchitecture and Architecture," *Proceedings of the IEEE*, vol. 98, pp. 325 - 340, 2001.

[17] P. Beckett, "A Fine-Grained Reconfigurable Logic Array Based on Double Gate Transistors," presented at IEEE International Conference on Field-Programmable Technology, FPT2002, Hong Kong, 2002.

[18] F. de Dinechin, The Price of Routing in FPGAs, http://citeseer.nj.nec.com/dedinechin99price.html, 1999.

[19] D. Sylvester, Keutzer, K., "Impact of Small Process Geometries on Microarchitectures in Systems on a Chip," *Proceedings of the IEEE*, vol. 89, pp. 467 - 489, 2001.

[20] R. Liu, C-S. Pai, "Interconnect Technology for Giga-Scale Integration," presented at 5th International Conference on Solid-State and Integrated Circuit Technology, Beijing, China, pp 17 - 20, 1998.

0-7695-1926-1/03/$17.00 (C) 2003 IEEE

[21] A. Singh, A. Mukherjee, M. Marek-Sadowska, "Interconnect Pipelining in a Throughput-Intensive FPGA Architecture," presented at Ninth International Symposium on Field programmable Gate Arrays, Monterey, CA, pp 153 - 160, 2001.

[22] P. D. Singh, Stephen D. Brown, "The Case for Registered Routing Switches in Field Programmable Gate Arrays," presented at Ninth International Symposium on Field Programmable Gate Arrays, Monterey, CA, pp 161 - 169, 2001.

[23] A. DeHon, "Balancing Interconnect and Computation in a Reconfigurable Computing Array (or, Why You Don't Really Want 100% LUT Utilization)," presented at ACM/SIGDA Seventh International Symposium on Field Programmable Gate Arrays, Monterey, California, United States, pp 69 - 78, 1999.

[24] M. Hutton, "Interconnect Prediction for Programmable Logic Devices," presented at International Workshop on System-Level Interconnect Prediction, SLIP'01, Sonoma, California, United States, pp 125 - 131, 2001.

[25] J. Rose, R. J. Francis, D. Lewis, P. Chow, "Architectures of Field-Programmable Gate Arrays: The Effect of Logic Functionality on Area Efficiency," *IEEE Journal of Solid-State Circuits*, vol. 25, pp. 1217 - 25, 1990.

[26] A. Yan, Rebecca Cheng, Steven J.E. Wilton, "On the Sensitivity of FPGA Architectural Conclusions to Experimental Assumptions, Tools, and Techniques," presented at ACM/SIGDA International Symposium of Field-Programmable Gate Arrays, FPGA'02, Monterey, California, USA., pp 147-156, 2002.

[27] M. Budiu, Sakr, M., Walker, K., Goldstein, S. C., "BitValue Inference: Detecting and Exploiting Narrow Bitwidth Computations," presented at 2000 Europar Conference, 2000.

[28] A. DeHon, "Very Large Scale Spatial Computing," presented at Third International Conference on Unconventional Models of Computation, UMC'02, 2002.

[29] Z. Ren, Ramesh Venugopal, Supriyo Datta, Mark Lundstrom, "Examination of Design and Manufacturing Issues in a 10 nm Double Gate MOSFET using Nonequilibrium Green's Function Simulation," 2001.

[30] Z. Ren, R. Venugopal, S. Datta, M. Lundstrom, D. Jovanovic, J. Fossum, Idealized SOI-Si Double Gate NMOSFET Device, Rev. 12-8-00, falcon.ecn.purdue.edu: 8080/mosfet/10nmstructure.pdf, 2000.

[31] J. G. Fossum, Chong, Y., "Simulation-Based Assessment of 50 nm Double-Gate SOI CMOS Performance," presented at IEEE International SOI Conference, Stuart, FL, USA, pp 107 -108, 1998.

[32] T. Waho, Chen, K.J., Yamamoto, M., "A Novel Multiple-Valued Logic Gate Using Resonant Tunneling Devices," *IEEE Electron Device Letters*, vol. 17, pp. 223-225, 1996.

[33] S.-J. Wei, Lin, H.C., "Multivalued SRAM Cell Using Resonant Tunneling Diodes," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 212-216, 1992.

[34] J. P. A. van der Wagt, "Tunnelling-Based SRAM," *Nanotechnology*, vol. 10, pp. 174-186, 1999.

[35] R. H. Mathews, Sage, J.P., Sollner, T.C.L.G., Calawa, S.D., Chang-Lee Chen, Mahoney, L.J., Maki, P.A., Molvar, K.M., "A New RTD-FET Logic Family," *Proceedings of the IEEE*, vol. 87, pp. 596 - 605, 1999.

[36] A. C. Seabaugh, Y.-C. Kao, H.-T. Yuan, "Nine-state Resonant Tunneling Diode Memory," *IEEE Electron Device Letters*, vol. 13, pp. 479 -481, 1992.

[37] K. D. Hobart, P. E. Thompson, S. L. Rommel, T. E. Dillon, P. R. Berger, D. S. Simons, P. H. Chi, ""P-on-N" Si Interband Tunnel Diode Grown by Molecular Beam Epitaxy," *Journal of Vacuum Science and Technology B*, vol. 19, pp. 290-293, 2001.

[38] N. Jin, Paul R. Berger, Sean L. Rommel, Phillip E. Thompson, Karl D. Hobart, "A PNP Si Resonant Interband Tunnel Diode with Symmetrical NDR," *Electronics Letters*, vol. 37, pp. 1412-1414, 2001.

[39] A. Seabaugh, X. Deng, T. Blake, B. Brar, T. Broekaert, R. Lake, F. Morris, G. Frazier, "Transistors and Tunnel Diodes For Analog/Mixed-Signal Circuits and Embedded Memory," presented at International Electron Devices Meeting, San Francisco, 1998.

[40] R. Compano (ed.), *Technology Roadmap for Nanoelectronics*, 2nd ed., European Commission IST Programme - Future and Emerging Technologies, 2000.

[41] E. Goetting, D. Schultz, D. Parlour, S. Frake, R. Carpenter, C. Abellera, B. Leone, D. Marquez, M. Palczewski, E. Wolsheimer, M. Hart, K. Look, M. Voogel, G. West, V. Tong, A. Chang, D. Chung, W. Hsieh, L. Farrell, W. Carter, "A Sea-of-Gates FPGA," *IEEE International Solid-State Circuits Conference*, vol. XXXVIII, pp. 110 - 111, 1995.

[42] V. Agarwal, Stephen W. Keckler, Doug Burger, "The Effect of Technology Scaling on Microarchitectural Structures," University of Texas at Austin, Austin, Technical Report TR2000-02, 2000.

[43] P. E. Gronowski, Bowhill, W.J., Preston, R.P., Gowan, M.K., Allmon, R.L., "High-Performance Microprocessor Design," *IEEE Journal of Solid-State Circuits*, vol. 33, pp. 676 - 686, 1998.

[44] S. Hauck, "Asynchronous Design Methodologies: An Overview," *Proceedings of the IEEE*, vol. 83, pp. 69 - 93, 1995.

[45] J. Muttersbach, T. Villiger, W. Fichtner, "Practical Design of Globally-Asynchronous Locally-Synchronous Systems," presented at Sixth International Symposium on Advanced Research in Asynchronous Circuits and Systems, ASYNC 2000, pp 52 - 59, 2000.

[46] A. DeHon, "DPGA-Coupled Microprocessors: Commodity ICs for the Early 21st Century," presented at FCCM '94 -IEEE Workshop on FPGAs for Custom Computing Machines, 1994.

[47] S. B. Hauck, S.; Borriello, G.; Ebeling, C.. "An FPGA for Implementing Asynchronous Circuits," *IEEE Design & Test of Computers*, vol. 11, pp. 60, 1994.

[48] I. E. Sutherland, "Micropipelines," *Communications of the ACM*, vol. 32, pp. 720 - 738, 1989.

0-7695-1926-1/03/$17.00 (C) 2003 IEEE