Logic in Wire: Using Quantum Dots to Implement a Microprocessor

Michael T. Niemier, University of Notre Dame
Peter M. Kogge, University of Notre Dame

Abstract

Despite the seemingly endless upwards spiral of modern VLSI technology, many experts are predicting a hard wall for CMOS in about a decade. Given this, researchers continue to look at alternative technologies, one of which is based on quantum dots, called quantum cellular automata. While the first such devices have been fabricated, little is known about how to design complete systems of them. This paper summarizes one of the first such studies, namely an attempt to design a complete, albeit simple, CPU in the technology. The projections are striking: a projected 10 to 1 increase in circuit density when compared to a CMOS equivalent, but a design approach which is radically different from conventional "logic" design, especially in timing considerations.

1 Introduction

As an alternative to CMOS VLSI, researchers have proposed an approach to computing with quantum dots, the quantum cellular automata (QCA). QCA is based on the encoding of binary information in the charge configuration within quantum dot cells. Computation power is provided by the Coulombic interaction between QCA cells. No current flows between cells and no power or information is delivered to individual internal cells. Local interconnections between cells are provided by the physics of cell-to-cell interaction due to rearrangement of electron positions [1].

A high-level diagram of a 4-dot QCA cell appears in Figure 1. Four quantum dots are positioned to form a square. Two mobile electrons exist in the cell and can move to different quantum dots in the QCA cell by means of electron tunneling. This tunneling is assumed to be completely controllable by potential barriers that can be raised and lowered between adjacent QCA cells by means of capacitive plates [1].

For an isolated cell there are two energetically minimal equivalent arrangements of the two electrons in the QCA cell, denoted cell polarization P = +1 and cell polarization P = -1. This concept is also illustrated graphically in Figure 1 [1].

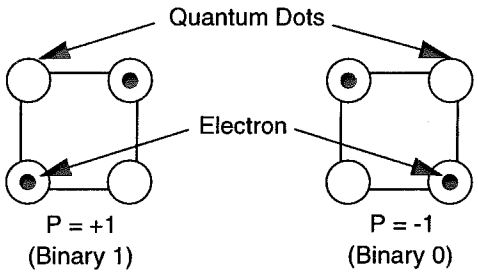

Figure 1: QCA cell polarizations/representation of binary 1 and binary 0

2 QCA Building Blocks

QCA cells perform computation by interacting coulombically with neighboring cells to influence each other's polarization. In the following section we review some simple, yet essential, QCA logical devices: a majority gate, a "wire", and more complex combinations of QCA cells [1].

The fundamental QCA logical circuit is the three input majority gate that appears in Figure 2. Computation is performed with the majority gate by driving the device cell (cell 4 in the figure) to its lowest energy state. This happens when it assumes the polarization of the majority of the three input cells. An input cell is defined as one where the tunneling barriers have been raised to the point where the electrons are "trapped" in a polarization. The device cell will always assume the majority polarization because it is this polarization where electron repulsion between the electrons in the three input cells and the device cell will be at a minimum.

Figure 3 illustrates how a binary value propagates down a QCA "wire". In this figure, the wire is a horizontal row of QCA cells. The binary signal propagates from left to right because of the Coulombic interactions between cells.

In Figure 3, cell 1 has polarization P = -1 and cell 2 has polarization P = +1. A binary 0 (from polarization P = -1) will propagate down the length of the wire because of the Coulombic interactions between cells. Initially, the electron repulsion caused by Coulombic interaction

0-7695-0104-4/99 $10.00

between cell 1 and cell 2 will cause cell 2 to change polarizations. Then, the electron repulsion between cell 2 and cell 3 will cause cell 3 to change polarizations. This process will continue down the length of the QCA "wire".

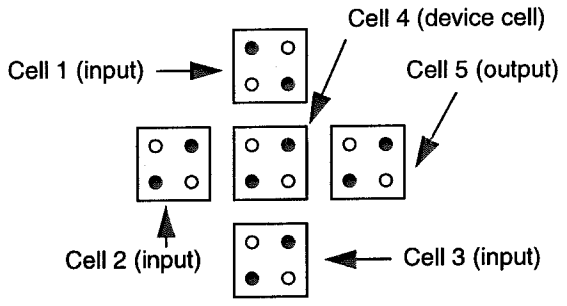

Figure 2: The fundamental QCA logical device - the majority gate

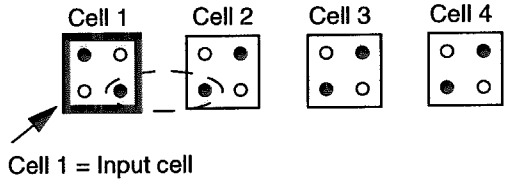

Figure 3: A QCA "wire"

A QCA wire can also be comprised of cells orientated at 45 degrees as opposed to the 90-degree orientation discussed above. With the 45-degree orientation, as the binary value propagates down the length of the wire, it alternates between polarization P = +1 and polarization P = -1. To tap off a copy of the desired logical value from the 45-degree-wire, a cell with a 90-degree orientation must be placed in the proper location to "rip off" the value. Figure 4 illustrates how to correctly rip off an uncomplemented binary 1 from a 45-degree-wire originally transmitting a binary 1. In general, to rip off a copy of the original binary value placed on the wire, a cell with a 90-degree orientation must be placed in an odd numbered gap. To obtain the value's complement, a cell with a 90 degree orientation must be placed in an even numbered gap. Thus, a complemented copy can be ripped off the wire in Figure 4 by placing the ripper cells in gap 2.

The significant advantage of the 45-degree wire is that both a transmitted value and its complement can be obtained from a wire without the use of an explicit inverter!

QCA cells do not have to be in a perfectly straight line to transmit binary signals correctly. Cells with a 90 degree orientation can be placed next to one another, but off center, and a binary value will still be transmitted successfully.

Finally, QCA wires also possess the unique property

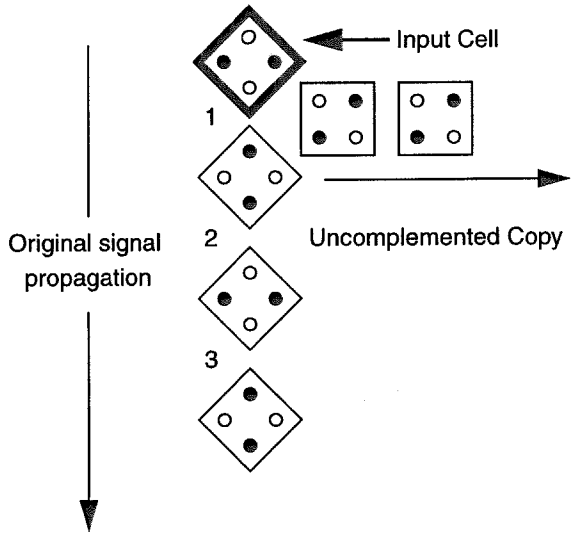

Figure 4: Ripping off a Binary 1

that they are able to cross in the plane without the destruction of the value being transmitted on either wire. However, this property holds only if the QCA wires are of different orientations (i.e. one wire is a 45-degree wire and the other is a 90-degree wire).

To implement more complicated logical functions, a subset of simple logical gates is required. For example, it would be impossible to implement a multiplexor, decoder, or adder in QCA without a logical AND gate, OR gate, or inverter. It has been demonstrated that a value's complement can be obtained simply by ripping it off a 45 degree wire at the proper location. Implementing the logical AND and OR functions is also quite simple.

The logical function for the majority gate is:

$$output = AB + BC + AC \quad (3.1)$$

The AND function can be implemented by setting one value (A, B, or C) in equation 3.1 to a logical 0. Similarly, the OR function can be implemented by setting one value (A, B, or C) in equation 3.1 to a logical 1. This results in the equations:

$$output = AB + B(0) + A(0) = AB \quad (3.2)$$

$$output = AB + B(1) + A(1) = A + B \quad (3.3)$$

More complex logical circuits can then be constructed from AND and OR gates.

3 A QCA Architecture

While there is still much work to be done, early results prove that QCA is a very viable alternative to CMOS. QCA wires and a QCA majority gate have been fabricated and tested successfully. However, the actual design of many of the circuits and devices required for a QCA microprocessor have not yet even been considered. What is required is a methodology for constructing

and designing such QCA circuits which is essential for a design such as a microprocessor. Furthermore, understanding how circuits are built should assist in the actual design of the devices themselves.

As a means for generating the QCA architecture an obvious first step is to translate existing CMOS designs directly into QCA majority gate logic. However, while such a translation is possible, the nature of QCA devices will require an architecture that is radically different from that of conventional CMOS.

A new QCA architecture is essential for several reasons. First, all circuits require an inherent amount of pipelining that is derived from QCA's four-phased clock. At any given point, an array of QCA cells will be in the switch, hold, release, or relaxed phase (hence the four different clock phases) [2]. This clocking scheme allows one subarray to perform a certain calculation, have its state frozen by the raising of its interdot barriers, and have the output of that subarray act as the input to a successor array. For long QCA wires, multiple subarrays (and hence pipeline stages) are required, as the Coulombic interaction between adjacent QCA cells will only transmit a binary value successfully over a finite number of QCA cells before the signal begins to degrade. At certain times, the value must be held or "frozen" so that it will act upon only a finite number of cells. Thus, inherent pipeline registers must be built into most complex QCA circuits (particularly wires). Another reason for an inherently different QCA architecture is the functionally complete QCA logic set discussed in Section 2.

4 The (QCA) Simple 12

The inherent pipelining associated with QCA and the logical device and routing methodology discussed above are only several of the ways that QCA designs differ from conventional CMOS designs. To develop a library of design rules and hence the QCA architecture, we are designing and simulating a custom design of a microprocessor called Simple 12 entirely in QCA. The advantages of choosing Simple 12 are three fold. First, the processor IS simple. Simple 12 has 12-bit data words, an 8-bit addressable memory, and uses minimal hardware. Consequently, much of the physical layout can be performed by hand. Second, an actual processor will be designed with an instruction set that includes arithmetic instructions, loads, stores, and jumps. Therefore, solutions to the difficulties encountered in this design will apply to even more complex systems and processors. Third, we have completed and fabricated a two micron CMOS Simple 12. Thus, it will be possible to make comparisons to an existing design in a technology on which we are trying to improve.

5 Floorplanning

To explain the concept of QCA floorplanning the design of the QCA Simple 12 ALU will be discussed. The QCA ALU was designed by translating the logic of the transistor version of the ALU to an equivalent QCA representation. Essentially, equivalent QCA majority gate representations of the transistor logic were determined and implemented. The schematic for the first-cut of the QCA ALU appears in Figure 5. The vertical lines represent separation of the circuit into the different clocking zones/phases.

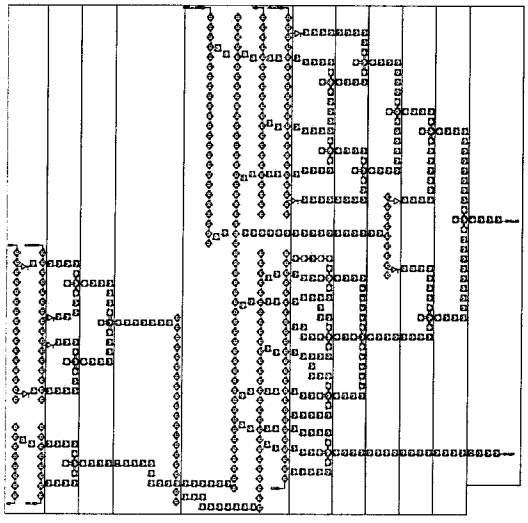

Figure 5: 1st cut of the Simple 12 ALU

The first-cut of the QCA ALU has three significant problems. First, the clocking zones have different widths. If their widths are not uniform, the clock rate will be limited to the width of the largest clocking zone which will be slower than for a narrower clocking zone. A second difficulty with the design appearing in Figure 5 is that there is a large number of QCA cells per clocking phase. If too many cells are included in a clocking phase, the clock rate will deteriorate. Also, for arrays of cells on the order of 1000, there will be a tendency for the system to settle in an excited state rather than a ground state because of thermodynamic effects [3]. Finally, and most importantly, no physical feedback exists in this design which is all but essential in most useful microprocessors and finite state machines.

Solving the problems of clocking zone widths and the number of cells "placed" in each clocking zone is simple - the schematic only needs to be adjusted to make them equal. The adiabatic clock rate can now increase. Furthermore, it is also possible to reduce the number of cells per clocking zone simply by adjusting the design and layout of the ALU. However, feedback still does not

exist in this schematic.

By examining Figure 5, it is easy to see that other problems exist. One such problem is a significant amount of wasted area because of the logic required to generate intermediate ALU data. To remedy this problem, "trapezoidal clocking" will be introduced. In Figure 6, QCA logic to generate intermediate ALU data is not placed in front of the computational logic but rather, below it. Instead of leaving large gaps (like those appearing in Figure 5), "trapezoids" containing computational logic and intermediate signal generation logic can be fit together to minimize wasted area.

Trapezoidal clocking does not only provide a means for minimizing total area. It can also be used to implement a feedback path in QCA circuits. In Figure 6, the four clocking phases are each given a number (1, 2, 3, and 4). If the top "trapezoid" is computational logic, data can be fed back to the input (assumed to be clocking zone 1 at the far left) after "switching" in clocking zone 1 at the far right. It can be easily seen that the clocking phases are traversed in the proper order (i.e. 1, 2, 3, 4). Furthermore, a signal can start a given point and a path exists to return to that point - the definition of feedback.

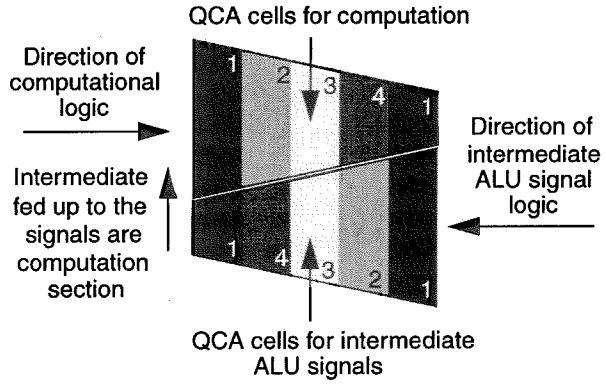

Figure 6: Trapezoidal clocking

The next question to be asked is whether or not a "universal clocking pattern" can be designed. A universal clocking pattern would allow multiple loops and feedback paths through the QCA clocking phases. As seen in Figure 7, such a universal clocking floorplan does exist and it allows extremely complicated paths of QCA wires to be constructed without violating the condition that the clocking zones must be traversed in the order 1, 2, 3, 4.

The universal clocking floorplan solves the problem of physical feedback in QCA circuitry. Furthermore, it provides an extremely efficient manner to run data signals and control signals. Data signals can be run horizontally while control signals are run vertically for example.

6 Conclusions

At present, designs for the Simple 12 ALU and a more complicated latch design have been completed. Also, extensive efforts have been made to study floorplanning and QCA circuits. A viable universal clocking floorplan has been developed that creates an efficient and functional layout of QCA clocking zones. Furthermore, this floorplan can be used as a basis for any QCA circuit design. Also, the floorplan, along with the ALU and advanced latch design have been integrated together to form a four-bit Simple 12 ALU along with a register for the output of each bit.

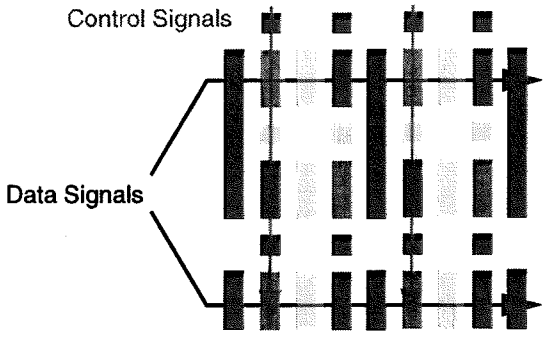

Figure 7: A Universal Clocking Floorplan

Using the CMOS Simple 12, area measurements were taken for both designs (with the CMOS numbers scaled to a 0.07 micron process - the end of the CMOS curve). It was determined that QCA offers at least an order of magnitude area density increase over the equivalent CMOS design. The current four-bit QCA design will next be simulated to verify both its logical and physical correctness. As more components of the QCA Simple 12 are constructed, they too will be simulated for logical and physical correctness and integrated into the design.

Preliminary results lead to the following conclusions: the layout of a QCA circuit will affect its timing and the timing of a QCA circuit will affect its layout. These ideas are essentially intertwined as evidenced by the universal clocking floorplan and the inherent pipelining of QCA circuits. This fundamental design rule will be used as the basis to develop the complete set of design rules for the QCA architecture and more complicated logical circuits.

References

[1] P. D. Tougaw, C. S. Lent, "Logical Devices Implemented Using Quantum Cellular Automata", Journal of Applied Physics, 75, 1818, 1994.

[2] P. D. Tougaw, C. S. Lent, "A Device Architecture for Computing with Quantum Dots", Proceedings of the IEEE, 85, 541, 1997.

[3] D. Berzon, T. Fountain, "Computer Memory Structures using QCAs", Report No. 98/1.