Modeling QCA for Area Minimization in Logic Synthesis

Nadine Gergel
University of Virginia, Dept. of ECE
351 McCormick Road
PO Box 400743
Charlottesville, VA 22904
(434) 924-7504
neg3c@virginia.edu

Shana Craft
University of Virginia, Dept. of ECE
351 McCormick Road
PO Box 400743
Charlottesville, VA 22904
(434) 924-3446
scc9e@virginia.edu

John Lach
University of Virginia, Dept. of ECE
351 McCormick Road
PO Box 400743
Charlottesville, VA 22904
(434) 924-6086
jlach@virginia.edu

ABSTRACT

Concerned by the wall that Moore's Law is expected to hit in the next decade, the integrated circuit community is turning to emerging nanotechnologies for continued device improvements. While significant advancements in nanotechnology devices have been achieved, much work is required to integrate these technologies into the existing design methodologies. Given that the physical design paradigm of each nanotechnology will be significantly different than that of traditional silicon circuits, the underlying cost functions used in optimization algorithms throughout the design abstraction hierarchy must be altered. Because nanotechnologies are not as well developed and understood as silicon devices, abstraction will initially result in less accurate models. However, if models are developed and augmented as nanotechnologies continue to evolve, the transition from CMOS-based design to nano-based design will be relatively seamless.

This paper details the logic-level abstraction process for area minimization for one promising nanotechnology – quantum cellular automata (QCA). The model abstracts relative area costs, including interconnect area, for QCA devices, and it is integrated within existing multi-level logic synthesis techniques. Results validate the proposed approach of designing nano-based circuits with the traditional abstraction-based design methodology.

Categories and Subject Descriptors

B. 6. 1 [Logic Design]: Design Styles

General Terms

Algorithms, Design

Keywords

Nanotechnology, Logic Synthesis, QCA, CAD, Interconnect

1. INTRODUCTION

The logic density of integrated circuits has been rapidly increasing for the last few decades. However, due to physical restrictions, the limit of transistor sizes will soon be reached, forcing researchers to explore novel physical design paradigms in the form of various nanotechnologies. Given the significant structural and functional differences between nanotechnologies and traditional silicon-based CMOS circuits, the current design process will have to be changed for nano-based design.

It would be greatly beneficial to maintain the general design methodologies used today, with any nano-induced changes being as transparent to designers and design tools as possible. To account for the physical design paradigm differences between CMOS and various nanotechnologies, current design methodologies must be examined to identify the necessary, but ideally minimal, changes that must be made for nano-based design without affecting the overall design flow.

This paper examines changes to the logic-level area minimization process for one emerging nanotechnology – quantum cellular automata (QCA). While the *logic* minimization algorithms remain the same (i.e. minimizing the number of gates and gate inputs), overall *area* minimization, which includes finding the optimal number of logic levels for a function, can vary for CMOS and QCA. The primary change centers around a novel cost model for QCA logic and interconnect. When this model is incorporated into traditional logic synthesis techniques, the benefits of QCA-dependent logic synthesis are achieved while keeping the design methodology changes completely transparent to the designer and tools.

In the development of this model, many assumptions were made about QCA's structure and functionality due to the lack of knowledge the exists about this and other emerging physical design paradigms relative to the well-understood realm of CMOS abstraction. The continued development and understanding of nanotechnologies will undoubtedly lead to more accurate models, but this paper lays the groundwork for a minimal perturbation approach for incorporating nanotechnologies into the existing abstraction-based design methodology via physical-design-dependent models.

2. BACKGROUND AND MOTIVATION

2.1 QCA Structures

The basic unit in quantum cellular automata is a cell that contains four possible positions for electrons, which are called quantum dots. Each cell has electrons occupying two of its four dots. These two electrons repel each other and thus must be located in opposite corners of the cell – top left and bottom right, or top right and bottom left. As shown in Figure 1, the former configuration represents a logical '0', and the latter represents a logical '1'.

These cells can then be aligned in rows and columns to form logic structures [5]. Consider the five cell cross structure in Figure 2, which is designed to perform the logic function Out=A+B. Not only do the electrons in each cell repel each other, but the repulsion propagates to adjacent cells. Therefore, inputs A, B, and Prog are affecting the configuration of the center and Out cells. In this case, the majority of the inputs are set to '1', so the center and Out cells are forced into the '1' position, which is the proper result of A+B. Note that if A were set to '0', the majority of inputs would force Out to '0'. Similarly, if Prog were set to '0' (making the cross structure an AND gate), Out would be forced to '0'.

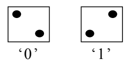

Figure 1. Logical representations of QCA cells

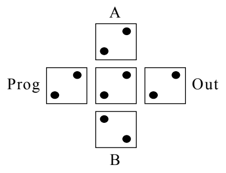

Figure 2. 2-input QCA OR Gate

These basic gates can be used to form complex circuitry. While early work focused on the development of basic logic gates [3], recent advances have explored the design of QCA-based circuits as complex as microprocessors [4]. While the work to date on QCA circuit design was done manually, future QCA-based design must be automated to handle the tremendous complexity of modern circuits. Therefore, accurate models are required to abstract the design process.

2.2 Input Dependence

When using CMOS transistors to build integrated circuits, the addition of an input to an OR or AND gate simply increases the number of transistors by two. However, the same relation does not hold for QCA, as the area per input increases with the number of inputs. For example, consider a 2-input and a 4-input OR gate. While the number of transistors in a 4-input CMOS OR gate is twice that of a 2-input OR gate (excluding the inverter), the number of QCA cells goes from five (as shown in Figure 2) to eighteen (as shown in Figure 3) [2].

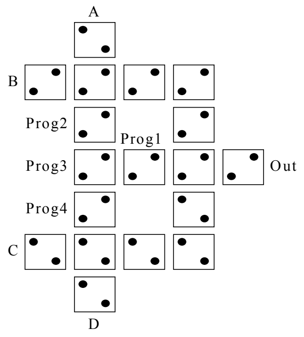

Figure 3. 4-input QCA OR Gate

This provides a major difference between the logic-level area models for the two physical design paradigms. Thus, when minimizing area for QCA, the number of gate inputs must be considered differently than in CMOS. Before considering interconnect, a 4-input OR function should be implemented in a series of three 2-input OR gates in QCA (for a total of fifteen cells). Such an example shows that the model for finding the optimal number of logic levels for a given function (i.e. the implementation that requires the least area) is different for QCA and CMOS.

2.3 Interconnect Dependence

QCA interconnect is formed using the same base cells, arranging them so that the signal propagates and drives subsequent logic gate inputs. The shaded blocks in Figure 4 are interconnect cells, connecting the outputs of one level of gates to the inputs of the succeeding level.

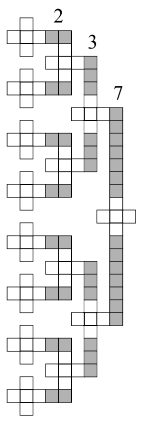

Figure 4. 4-level, 16-input QCA gate

Given that the base QCA interconnect and logic cells are the same, QCA interconnect has a tremendous impact on the size of the circuit. It is easy to see in Figure 4 that the number of interconnect cells is far from trivial in the overall area. Therefore, interconnect must be included in the model for logic-level area minimization. While the exact amount of interconnect will not be known until post-layout, estimates based on averages can be abstracted up to the logic-level. Such interconnect consideration is not currently involved in most CMOS logic-level models, providing another key difference between the two physical design paradigms[1].

3. QCA LOGIC-LEVEL AREA MODEL

This section details the development of the QCA logic-level gate and interconnect area models to be integrated within existing logic synthesis area minimization algorithms. These models are based on the current understanding of the QCA physical design paradigm. Given the immaturity of the field, the models are likely to change with increased understanding. However, these models and their seamless integration into the existing design methodology demonstrate the possibility of abstraction-based nano design.

It is also important to note that the relative area of various implementations is more important than individual magnitudes for search-based optimization algorithms, as "best" is a relative term. Therefore, while these models do provide individual magnitudes, it is the relative analysis that impacts area minimization. As a result, changes to the models will not necessarily yield different implementations, as relative area rankings will not be affected by across-the-board percentage magnitude adjustments.

[1] However, shrinking CMOS device sizes are increasing the relative area impact of interconnect. Therefore, including interconnect in the area model for CMOS logic synthesis could provide significant benefits.

3.1 QCA Gate Area

As stated above, the QCA gate area model is a function of the number of gate inputs, but the function differs greatly from CMOS. To derive this function, the minimum AND/OR gate area was established for gates up to thirteen inputs. This required exploring many different implementations of the same gate. For example, the optimal implementation of an *n*-input QCA gate may actually be two $\frac{n}{2}$ -input gates followed by one 2-input gate or some other variation. In such cases, internal signal interconnect area is also considered.

The following assumptions were made:

- QCA cells placed at least one cell apart do not affect each other's electron configuration or induce any propagation.
- Only the number of cells is counted, not the total area that is used. This is because the area estimations are to be used to compare relative areas, not to determine absolute area.

To determine the intra-gate interconnect cost associated with chaining cells, tree structures were generated, and a cost was derived for each level of interconnect. Consider the 4-level, 16-input gate shown in Figure 4, with the internal interconnect cells shaded. This implementation is actually eight 2-input gates (Level 1) $\rightarrow$ four 2-input gates (Level 2) $\rightarrow$ two 2-input gates (Level 3) $\rightarrow$ one 2-input gate (Level 4). Note that the internal signal interconnect cost is level-dependent. The internal signals going to gate Levels 2, 3, and 4 require two, three, and seven cells, respectively.

This model was used to determine the cost of a variety of gate implementations with any number of inputs. It was found that the minimum cost implementations of 2-, 3-, and 4-input gates are five, ten, and eighteen cells respectively [2, 5]. Figure 5 shows the possible configurations for a 5-input gate. There could be more than one gate configuration for a given number of inputs that have the same area, but this should not affect final layouts. Combining gate area costs with the level-dependent internal signal interconnect cost model, the optimal implementation of the 5-input gate was one 3-input gate $\rightarrow$ one 3-input gate. This approach was repeated until cost models were developed for QCA gates with up to thirteen inputs.

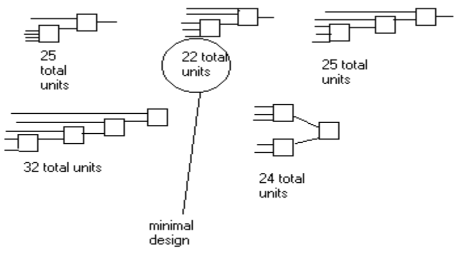

Figure 5. Various implementations of a 5-input QCA gate

Figure 6 shows the approximated gate area function, $A = (6 \times I) - 8.5$, where *A* is the number of QCA cells in the optimal area configuration and *I* is the number of gate inputs for that configuration. The standard deviation of this linear function from the actual data points is about 8.5%.

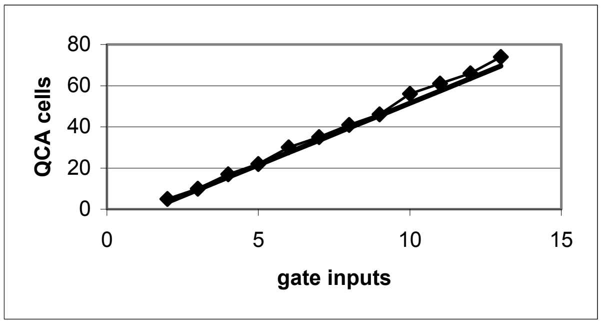

Figure 6. QCA gate area model

3.2 QCA Interconnect Area

As with intra-gate interconnect area, gate-to-gate interconnect area cost is level-dependent. Starting with Level 2 connections (i.e. inputs to Level 2 gates), the fanin interconnect costs were determined for gates with two to ten inputs. As one would expect, the relationship between interconnect area and number of inputs is nearly linear, as show in Figure 7. The approximate function for the cost model is $A = (4.93 \times I) - 8.16$.

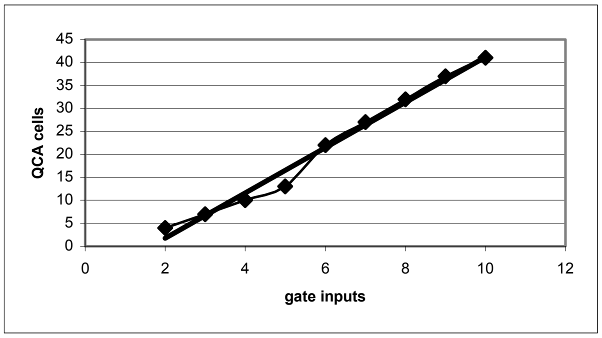

Figure 7. Level 2 interconnect costs

This process was extended for up to six levels of logic. The results are shown in Table 1. The following 2-variable formula was derived from this data to form the basis of the interconnect cost model: $A = L \times I^{0.3 \times L + 0.7}$, where L is the fan-in gate level. The standard deviation of this function from the actual data points is about 14%.

Table 1. Interconnect costs for up to 6 levels with 10 inputs

	QCA cells for fan-in interconnect				
inputs	level 2	level 3	level 4	level 5	level 6
2	4	6	14	30	62
3	7	17	33	61	127
4	10	24	58	120	246
5	13	41	67	185	375
6	22	54	130	276	562
7	27	77	173	373	749
8	32	96	238	494	1074
9	37	135	331	615	1255
10	41	160	366	782	1582

4. COST MODEL IMPLICATIONS

While the cost model magnitudes will likely change as understanding of the QCA physical design paradigm increases, the models provide reasonably accurate relative size data, which is all that is necessary when searching for the minimum area implementation at the logic level. For example, the local search heuristics that are traditionally used in logic synthesis use cost functions to find the best implementation of a function in a defined neighborhood. This search does not rely on absolute values but rather relative comparisons. Consider the following generic local search pseudocode:

```
local_search(solution)
{
  next_sol=solution;
  best_cost=cost(next_sol)
  for each (temp_sol=move(solution, radius))
  {
    temp_cost=cost(temp_sol);
    if (temp_cost<best_cost) {
      next_sol=temp_sol;
      best_cost=temp_cost;
    }
  }
  if (next_sol=solution) BREAK;
  else local_search(next_sol);
}
```

Each recursive call to the algorithm starts with a new base solution. The "cost" subroutine evaluates each solution based on a defined cost function. "move" finds all of the possible solutions within the base solution's neighborhood, the size of which is defined by "radius". For area minimization purposes, valid moves are increasing or decreasing the number of logic levels. The "for each" loop continues until all of the neighborhood's solutions have been evaluated and the best (based on the cost model) solution emerges, providing the base solution for the next recursive call.

This algorithm forms the basis for many CMOS-based logic synthesis tools, and it need not be altered for QCA-based design. The only change is in the "cost" subroutine. Instead of the well-known cost models used in CMOS designs, the combined gate and interconnect area models detailed in Section 3 will be invoked. Thus, the general flow of the algorithm (and the overall design methodology) remains unchanged, and the switch to QCA-based design is transparent to the designers and tools at this level of abstraction.

Table 2 shows the application of this algorithm to several simple functions using three different area models in the "cost" subroutine: standard CMOS gate area model, QCA with interconnect, and QCA without interconnect.

Table 2. Minimum area implementations

	levels in minimum cost implementation		
2-level function	CMOS	QCA w/ int	QCA w/o int
abcd+efgh+ijkl+mnop	2	2	2
abcd+afgh+ijkl+mnop	2	2	2
abcd+afgh+ajkl+mnop	4	2	4
abcd+afgh+ajkl+anop	3	3	3
abcd+abgh+ajkl+anop	5	3	5
abcd+abgh+abkl+anop	5	3	5
abcd+abgh+abkl+abop	3	3	3
abcd+abch+abkl+abop	4	4	4
abcd+abch+abcl+abop	4	4	4
abcd+abch+abcl+abcp	2	2	2

This data shows that the CMOS and QCA results are quite similar up to about four levels of logic, at which point CMOS can still provide area efficient implementations but the QCA area starts to explode. It is therefore reasonable to assume that highly complex functions that would be implemented in a large number of levels for CMOS are more area efficient in fewer levels for QCA. We cannot back these model-based findings with experimental data because QCA devices have not yet been physically realized.

It is also interesting to note that the minimum cost QCA implementations based on the area model without interconnect are exactly the same as the CMOS implementations, revealing the tremendous importance that interconnect plays in QCA-based circuit area. As mentioned in Section 2.3, shrinking CMOS device sizes are causing an increase in the relative area of interconnect. As a result, significant area savings could be achieved using interconnect-conscious CMOS logic synthesis.

5. CONCLUSIONS

This paper introduced the first logic-level area models for QCA-based area minimization. Integrating these and other nanotechnology models into the traditional abstraction-based design methodology will help make the transition to nano-based design as transparent to circuit designers and tools as possible. As the development and understanding of nanotechnologies continues, these models will become more accurate, thus enabling effective abstraction-based design of nanotechnology circuits**.**

6. REFERENCES

[1] Deng, X., Hanyu, T., and Kameyama, M., "Multiple-valued logic network using quantum-device-oriented superpass gates and its minimization," *IEE Proceedings- Circuits Devices Systems*, vol. 142, pp. 299-305, October 1995.

[2] Gin, A., Williams, S., Meng, H., and Tougaw, P.D., "Hierarchical design of quantum-dot cellular automata devices," *Journal of Applied Physics*, vol. 85, pp. 3713-3720, April 1999.

[3] Lent, C.S., and Tougaw, P.D., "A device architecture for computing with quantum dots," *Proceedings of the IEE*, vol. 85, pp. 541-557, April 1997.

[4] Niemier, M.T., Kontz, M.J., and Kogge, P.M., "A design of and design tools for a novel quantum dot based microprocessor," Deign Automation Conference, pp. 227-232, June 2000.

[5] Tougaw, P.D., and Lent, C.S., "Logical devices implemented using quantum cellular automata," *Journal of Applied Physics*, vol. 75, pp. 1818-1825, February 1994.