EVALUATING TECHNOLOGY MAPPING METHODS FOR QCA DEVICES

Stéphano Gonçalves, Julio S. Domingues Jr, Melissa S.R. Colvara,
Leomar S. da Rosa Jr, Felipe S. Marques
Group of Architectures and Integrated Circuits – GACI
Technology Development Center - CDTec
Federal University of Pelotas – UFPel
Pelotas, RS, Brazil
{smmgoncalves, jsdomingues, mdsrcolvara, leomar, felipem}@inf.upel.edu.br

ABSTRACT
The CMOS technology is reaching its physical limits. The Quantum Cellular Automata (QCA) is an emerging computation technology with great potential to replace the CMOS technology. However, there is no complete mapping flow for this technology. This paper presents an analysis of technology mapping methods aiming QCAs devices. Based on that, we present some results that show a gap between the state of art on QCA synthesis and the regular mapping tools. Therefore, in the future, we intend to introduce an efficient mapping algorithm to handle QCA devices. In order to achieve it, we propose a generic mapping flow that is able to use different mapping procedures aiming a given technology. The crossover of these methods can result in a good mapping methodology.

Categories and Subject Descriptors
B.6.3 **[Logic Design]**: Design Aids – *automatic synthesis, optimization.*

General Terms
Algorithms, Design, Experimentation.

Keywords
Logic Synthesis, Technology Mapping, QCA.

1. INTRODUCTION

The design of VLSI circuits is very demanding since it is necessary to develop circuits with higher performance and lower power consumption. This is not a trivial task, and this is why designers need support tools that automate and reduce the time-to-market. A circuit design cycle has several steps. However, technology mapping plays an important role in the synthesis process, defining the main characteristics of a circuit, concerning area, power consumption and delay. Technology mapping define which cells will be used in the circuit design. The quality of the mapping is directly related to the quality of the cell library.

Currently, designers of VLSI circuit are facing another problem: the CMOS technology is reaching its physical limits. Therefore, it is difficult for designers to perform optimizations and improvements on the circuit design. Recent researches point out new technologies to replace the regular CMOS. Most of them are related to quantum computing. One of these alternatives is the QCA (Quantum Cellular Automata) that emerges as a strong candidate. The state of art works on QCA present some structures to build digital circuits. They are: the QCA-Inverter, QCA-Wire and QCA-Majority Gate. As far as we know, there is one straightforward method to perform technology mapping using QCA designs [1]. However, the state of art method, proposed in [5], presents better results than the method of [1]. In [2], a handmade set of 13 majority gates is presented and it is demonstrated that this small library can be used to map a circuit. A set of AOI (And Or Inverter) gates implemented with QCA technology is presented in [3]. These cells can produce satisfactory results in the existing tools. However, the cost of these cells is too expensive when compared to majority gates. A series of adders designs implemented through QCA cells are presented in [4]. Another recent work in this area [5] proposes a set of scripts to perform automated synthesis for QCA devices on SIS [14]. However, the method proposed in [5] is only able to obtain the minimal majority gate arrangement for a given Boolean function, but it does not achieve optimal results in the technology mapping, as we will show in the experiments section.

None of these works presents a complete automatic flow to efficiently map digital circuits using QCA designs. This paper presents an analysis focusing on technology mapping for QCA circuit designs. We intend to identify the main needs of a technology mapping flow to handle QCA cells. In order to analyze the impact of using different libraries, we have performed experiments using three libraries (based on previous works) using the standard cell mapping of the ABC tool [6].

In general, just like CMOS technology, the richer are the libraries (in terms of number of Boolean functions), the better are the mapping results (in terms of area - number of QCA devices). Since the number of Boolean functions increases exponentially according to the number of input variables, it is necessary to automate the building process of a QCA library that uses only Majority gates and inverters to implement arbitrary Boolean functions. The library conception is not discussed on this paper. Furthermore, some preliminary results show a gap between the state of art on QCA synthesis and the regular mapping tools. Based on that, in the future we intend to introduce an efficient mapping algorithm to handle QCA devices combining different mapping procedures in a generic mapping flow.

This paper is organized as follow. Section 2 presents some background information about QCA technology and technology mapping. Some experimental results are shown in section 3. Section 4 presents an idea for a generic technology mapping flow. Finally, section 5 presents our conclusions and points out some future works.

2. BACKGROUND

This section reviews some important concepts for better understanding the rest of the paper.

2.1. QCA Technology

A quantum cellular automata can be represented as a set of four containers of quantum loads or dots, positioned at the corners of a square shape. The cell contains two free electrons (located on opposite diagonal) that can move through quantum mechanics using a tunnel to one of two free points. The electrons are forced to the opposite diagonal by Coulombic repulsion. These positions represent the two possible logic states [4]: P = 1 representing the logic state '1 'and P = -1 state '0'. These representing the logic state '1 'and P = -1 state '0'. These states are shown in Fig.1.a.

The basic elements of the QCA logic are the QCA-Majority Gate, QCA-Inverter and QCA-Wire. These elements are show respectively in Fig.1 (a,b,c,d). The QCA-Majority gate is the main structure and is composed of five QCA cells, as is illustrated in Fig.1.b. This structure is able to represent the logic function M (a, b, c) = ab + ac + bc. It has three input operators that can be used to implement different Boolean according to the inputs.

2.2. Data Structures and Technology Mapping

Graphs are very useful data structures able to represent digital circuits. Generally, these circuits are represented through trees or DAGs (Directed Acyclic Graphs). Trees are simple data structures that represent logic cones. This way they only represent fragmented portions of the circuit. This is a good feature when the best local solution is been looked for. On the other hand, in most of the problems, it cannot reach the best global solution [7]. Currently, most of the methods can achieve better synthesis using DAGs. In this case, this graphs can represent the whole circuit, and avoid local minima. Among different DAG types, the most popular for digital circuit representation is the AIG (And-Inverter Graph). It has a very simple structure defined in [8] that makes most of the synthesis process easier.

As mentioned before, technology mapping is an important stage of the logic synthesis. It can be divided into three steps: decomposition, matching and covering [7]. In the decomposition step, a more complex circuit representation is decomposed in a more simple structure such as an AIG. The matching phase compares portions of the circuit with a bound library. This way, the mapping procedure can identify those library cells that can implement a portion of the circuit. The last stage is called covering. In this step, the mapping procedure chooses the best set of cells to implement the circuit among those identified in the matching phase, and based on some criteria. A cell library is defined by finite set of cells that can implement different logic functions. Either the library may be pre-characterized (where the characteristics of each cell are previously identified and computed) or it may be dynamic (where cells designed according to the needs of the circuit).

3. EXPERIMENTS

Currently, the main goal of QCA synthesis is to achieve smallest number of majority gates and inverters needed to implement a given digital circuit. In order to analyze how good is the synthesis achieved by the state of art technology mapping algorithms considering QCA devices, we have run a set of experiments on the ABC tool.

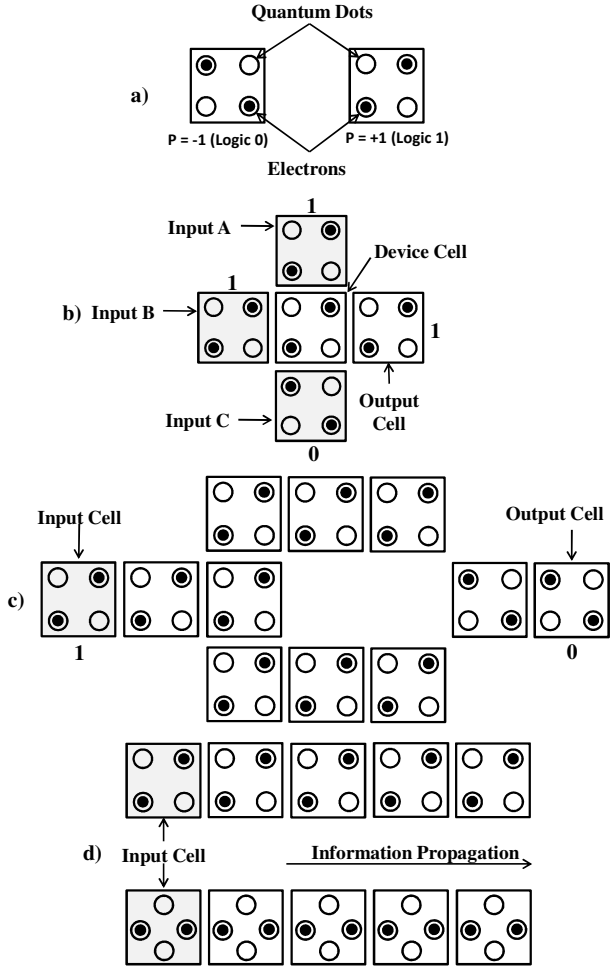

Figure1 - Basic elements of QCA technology

For these experiments, a subset of the ISCAS benchmarks were used. We have run the library mapping method of ABC using three different libraries for each circuit. The library mapping is performed through the command map. The algorithm [11] which implements the mapping is based on a simplified cut-based Boolean matching, lossless synthesis and supergates. The command map with the –a flag (indicating area oriented mapping only) were run ten times for each circuit (it is an iterative incremental process). Each mapped circuits was written into a Netblif file. Through this, we could identify the relative costs of each circuit. The cost of a circuit is the sum of the costs of all majority gates and inverters contained in the circuit. Based on the relation proposed in [3], the costs three and five were assigned to majority gates and inverters, respectively. These costs are proportional to the size of the single quantum cell. It means that the majority gate (inverter) is three (five) times bigger than a single quantum cell.

Table 1 shows a comparison between the ABC mapping and the numbers of the Kong's synthesis [5]. The "Basic" column corresponds to a simple library that is composed by a majority

gate, an inverter, an AND2 (due to restrictions on ABC - this cell can be implemented through a Majority gate) and the constants ZERO and ONE. The third column corresponds to an expanded library which contains the thirteen functions defined in [2]. The fourth column has forty primitive functions which can be implemented by a single majority gate. The library proposed in [5] contains only three functions included in the library of [2]. This library is the same library used in the experiments performed in [5]. The last column corresponds to results presented by Kong [5]. The circuits mapped with the basic library presents highest costs. This is expected since it is a very simple library. In most cases the larger libraries tended to have the better results. However, it does not happen in all cases when comparing circuits mapped with the libraries proposed by [2] and [5]. The library proposed in [2] has more complex cells then the cells presented in [5].

Table 1 - Comparison between ABC mapping ans Kong's method.

Circuit	ABC			Kong's method [4]
	Basic	[1]	[4]	
9symml	1306	844	630	256
alu2	2288	1496	1305	1555
apex6	4230	3426	2168	2776
cht	1054	464	474	405
cm150a	318	163	163	238
cm151a	201	101	91	119
cm152a	168	68	78	98
cm162a	214	161	159	193
cm163a	206	153	156	199
cm42a	121	121	71	84
cm82a	127	24	77	51
cm85a	214	177	144	78
cmb	216	184	158	104
cu	247	182	187	225
decod	110	110	110	114
frg1	691	500	386	620
i2	1730	1656	641	627
k2	5062	4720	3458	4998
ldd	386	341	234	321
majority	48	33	18	18
mux	308	208	158	198
pcle	384	289	189	271
pcler8	469	364	234	315
pm1	204	170	160	185
term1	842	509	575	568
ttt2	938	573	571	695
unreg	871	426	386	417
vda	2316	2244	1574	2810
x2	243	201	153	186
z4ml	229	36	154	57

Even though the library in [2] has a smaller number of functions, it can achieve better results depending on the circuit structure. In most of the cases, the Kong's library [5] presents better results. When compared to the Kong's method, the ABC method is worse in few cases. It shows that even compared to the state of the art tool, the Kong's method has some tricks that leads to better results. As conclusion, we can say that the ABC method is not good enough, and it can be improved and optimized to achieve better solutions.

Fig. 2 presents a graphic that summarizes the results of Table 1. It shows the sums of the costs of all circuits for a given library. It is clear that the ABC mapping is more efficient for the whole set of circuit.

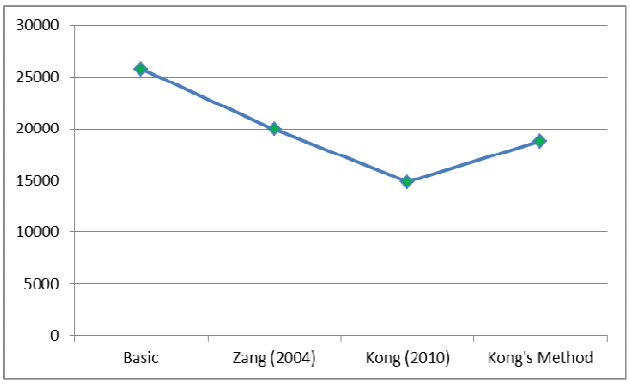

Figure 2 – Table 1 graphic summarization.

4. GENERIC MAPPING FLOW

The experimental results show us that the regular technology mapping could be improved to achieve better results on QCA technology. In fact, it could be something that can be done to any other technology. Until now, the methodology proposed in ABC is based on a bunch of logic synthesis concepts and algorithms that leads to good practical results either on FPGA or standard cell (mainly CMOS) synthesis. Kong [5] has used the same algorithms applied in a different methodology, and it has been shown that his proposition can achieve good results in some cases.

At the end, the difference between ABC mapping and the Kong's method is the order and the way that they apply a bunch of logic synthesis algorithms on their methods. Therefore, it would be nice to have an environment where we could specify a mapping methodology in a high abstraction level. It would allow us to quickly prototype different mapping methods and eventually apply it on different technologies.

The ABC tool is an open source academic software that allows the development of new methods using C/C++ programming language. Although it has a pretty good documentation, further developments on this environment are not going to result in a quickly prototyping. Thus we intent to present a generic mapping flow that should provide an environment to quickly prototype new mapping methodologies. Moreover, it should give some didactic support for educational purposes.

In an early stage of development, we do have a baseline flow that is been validated. Currently, it is able to read AIGER [7], BLIF (Berkeley Logic Interchange Format) and EQN (Equation format) files. These descriptions are represented through a unified AIG based data structure. Using this structure, we are able to perform either DAG or tree mapping. The matching phase relies on k-cut enumeration [13] and the Boolean matching algorithm presented in [10]. The covering procedure is based on [9][12].

The next step is the definition of a software engineering model that will be the basis of our generic technology mapping flow. The word "generic" means that we would like to specify a technology

mapping method in a high abstraction level and ease the integration processes of other algorithms. This way, we could create new methodologies combining different algorithms for matching and covering. In addition, we could also specify different objective functions for a given methodology. All these features would allow us to quickly prototype new technology mapping methods.

5. CONCLUSIONS AND FUTURE WORK

This paper presents a comparison between the ABC logic synthesis tool [6] and the state of art on QCA synthesis [5]. It is clear that there is a gap between their technology mapping methods. Using the QCA library presented in [5], the ABC mapping has presented better results in most of cases. However, the Kong's method [5] presents pretty good results in other cases. Therefore, the regular technology mapping could be improved to achieve better results on QCA technology. Besides the mapping method itself, the QCA library could be improved by increasing the number of available Boolean functions. This would lead to area reduction.

The lack of a quickly prototyping environment for technology mapping has motivated us to start the development of a generic technology mapping flow. This flow should allow the specification of a technology mapping method in a high abstraction level and ease the integration processes of other logic synthesis algorithms. It would allow us to quickly prototype different mapping methods and eventually apply it on different technologies. As future works, we intend to extend the mapping experiments by using augmented libraries. Furthermore, by experimenting different mapping procedures, we could be able to propose a new algorithm for the QCA technology.

6. ACKNOWLEDGMENTS

Research partially funded by CNPq and FAPERGS Brazilian funding agencies under grant 11/2053-9 (Pronem) and 11/1925-8 (ARD).

7. REFERENCES

[1] Zhang, R.; Gupta, P.; Jha, N.K.. 2005. "Synthesis of majority and minority networks and its applications to QCA, TPL and SET based nanotechnologies," *VLSI Design, 2005. 18th International Conference* on, vol., no., pp. 229- 234, 3-7 (Jan. 2005).

[2] Zhang, R., Walus, K., Wang, W., and Graham, A. J.. 2004 "A method of majority logic reduction for quantum cellular automata". *IEEE Transactions on Nanotechnology*, 3(4), pp. 443-450, (Dec.2004).

[3] Momenzadeh, M., Huang, J., Tahoori, M. B., and Lombardi, F..2005. "Characterization, test, and logic synthesis of and-or-inverter (AOI) gate design for QCA implementation". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(12), pp. 1881-1893. (Dec. 2005).

[4] Cho, H. and Swartzlander Jr., E. E. "Adder designs and analyses for quantum-dot cellular automata". 2007. *IEEE Transactions on Nanotechnology*, 6(3), pp. 374-383. (May. 2007).

[5] Kong, K., Lu, R., and Shang, Y. "An optimized majority logic synthesis methodology for quantum-dot cellular automata". 2010. *IEEE Transactions on Nanotechnology*, 9(2), pp. 170-183. (Mar. 2010).

[6] Mishchenko, A.; Chatterjee, S.; Brayton, R.; Wang, X.; Kam, T. "Technology Mapping with Boolean Matching, Supergates and Choices". *ERL Technical Report*, [S.l.], http://www.eecs.berkeley.edu/ alanmi/abc/abc.htm, 2005

[7] Marques, F. d. S. "Technology Mapping for Virtual Libraries Based on Cells with Minimal Transistor Stacks". *PhD thesis*. University Federal of Rio Grande do Sul-UFRGS, Porto Alegre, Brazil, 2008.

[8] AIGER format. Available in http://fmv.jku.at/aiger/, 2012

[9] Manohararajah, V., D.Brown, S., and Vranesic, Z., "Heuristics for area minimization in LUT-based FPGA technology mapping," IEEE *Transactions On Computer-Aided Design Of Integrated Circuits And Systems* 25(11), pp. 2331-2340, Nov. 2006.

[10] Debnath, D.; Sasao, T. "Efficient Computation of Canonical form for Boolean matching in large libraries". *.2004. Design Automation Conference, 2004. Proceedings of the ASP-DAC 2004*. Asia and South Pacific, Page(s): 591- 596, 2004.

[11] Chatterjee, S.; Mishchenko, A.; Brayton, R.; Wang, X.; Kam, T. "Reducing Structural Bias in Technology Mapping".2006. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*. 25(6), pp. 2894 – 2903, Dec. 2006.

[12] Mishchenko, A., Chatterjee, S., Brayton, R., "Improvements to Technology Mapping for LUT-Based FPGAs". 2006. *Int'l Symp. on FPGA*, 2006.

[13] Cong J., Wu, C., Ding, Y., "Cut Ranking and Pruning: Enabling A General and Efficient FPGA Mapping Solution". 1999. *Int'l Symp. on FPGA*, 1999.

[14] Sentovich, E. et al. "SIS: A system for sequential circuit synthesis."1992. *Berkeley: EECS Department, University of California*, 1992. (TR UCB/ERL M92/41)