

Design of Reversible and Quantum Circuits

Edited by

Kenichi Morita¹ and Robert Wille²

1 Hiroshima University, JP, morita@iec.hiroshima-u.ac.jp

2 Universität Bremen, DE, rwille@informatik.uni-bremen.de

Abstract

It is a widely supported prediction that conventional computer hardware technologies are going to reach their limits in the near future. Consequently, researchers are working on alternatives. Reversible circuits and quantum circuits are one promising direction which allows to overcome fundamental barriers. However, no real design flow for this new kind of circuits exists so far. Physical implementations are in its infancy. Within this seminar, recent research questions of this emerging technology have been discussed.

Seminar 11.–14. December, 2011 – www.dagstuhl.de/11502

1998 ACM Subject Classification B.6 Logic Design, D.1 Programming Techniques, F.2 Analysis of Algorithms and Problem Complexity, I. Computing Methodologies


Keywords and phrases reversible computation, quantum computation, computer aided design, hardware and software design, physical implementation, applications

Digital Object Identifier 10.4230/DagRep.1.12.47

1 Executive Summary

Kenichi Morita

Robert Wille

License  Creative Commons BY-NC-ND 3.0 Unported license
© Kenichi Morita and Robert Wille

The development of computing machines found great success in the last decades. But the ongoing miniaturization of integrated circuits will reach its limits in the near future. Shrinking transistor sizes and power dissipation are the major barriers in the development of smaller and more powerful circuits. To further satisfy the needs for more computational power and further miniturization, alternatives are needed that go beyond the scope of conventional technologies like CMOS. Reversible logic and quantum logic provide a promising alternative that may enhance or even replace conventional circuits in the future. More precisely:

■ Low Power Computation

While conventional circuits dissipate energy for each lost bit of information, reversible circuits are information lossless, i.e. theoretically they are not affected by this. Considering the increasing miniaturization, this makes reversible logic interesting for domains like low-power design. Besides this general paradigm, reversible circuits are particularly suited for complementary low-power solutions like adiabatic circuits or on-chip interconnect encoders.

■ Quantum Computation

Quantum Computation offers the promise of more efficient computing for problems that are of exponential difficulty for conventional computing paradigms. Considering that many of the established quantum algorithms include a significant Boolean component



Except where otherwise noted, content of this report is licensed under a Creative Commons BY-NC-ND 3.0 Unported license

Design of Reversible and Quantum Circuits, *Dagstuhl Reports*, Vol. 1, Issue 12, pp. 47–61

Editors: Kenichi Morita and Robert Wille



Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

(e.g. the oracle transformation in the Deutsch-Jozsa algorithm, the database in Grover's search algorithm, and the modulo exponentiation in Shor's algorithm), it is crucial to have efficient methods to synthesize quantum gate realisations of Boolean functions. Since any quantum operation inherently is reversible, reversible circuits can be exploited for this purpose.

However, no real design flow for these new kinds of circuits exists so far. Proposed approaches for synthesis, verification, and test are only applicable for very small circuits and systems. This is crucial since the design for reversible and quantum systems significantly differs from their conventional counterparts. Nearly all concepts and methods developed for conventional hardware design in the last decades have to be redeveloped in order to support the new technologies. Additionally, considering that today researchers are still faced with serious challenges for conventional technologies, it is worth working towards design solutions for reversible and quantum technologies already today.

The goal of the seminar was to bring together experts in order to present and to develop new ideas and concepts for the design of complex reversible and quantum circuits. In total, 17 presentations together with 1 tool demonstration (of the open source toolkit *RevKit*) and one panel sessions (on the different opinions on how reversible circuits can help to reduce power consumption during computation) have been conducted within the seminar. This has been accompanied by several working group and proposal preparation meetings. The most important topics which have been discussed were:

- **Design Methods**

How to (automatically) synthesize reversible and quantum circuits as well as check them for correctness?

Most of today's synthesis approaches for reversible and quantum circuits still rely on Boolean function descriptions like e.g. permutations, truth tables, or binary decision diagrams. In order to design complex circuits, higher levels of abstractions have to be considered. While for this purpose hardware description languages like VHDL, SystemC, or SystemVerilog have been established in conventional hardware design, high level synthesis of reversible and quantum circuits is just at the beginning. In order to advance this area, ideas about concepts, languages, and synthesis approaches for high level design have been presented and collected at the seminar.

- **Theoretical Consideration**

How can theoretical studies show us the way for realizing reversible/quantum computers?

In order to implement efficient reversible/quantum circuits and computers, we still need very basic and theoretical studies on them. This is because the paradigm of reversible/quantum computing has very different natures from that of conventional computing. Therefore, we shall still be able to find many novel and useful ideas for them through theoretical studies. In this seminar, theoretical consideration on various models in several levels have been presented and discussed. These models range from the element level to the software level, which include reversible logic elements and circuits, quantum automata, reversible Turing machines, and reversible programming languages.

- **Physical Realizations and Accuracy of Models**

How to physically implement the respective circuits?

How to close the gap between the theoretical models and the physical implementation?

In order to design reversible and quantum circuits, abstractions of the precise physical realizations are applied. These include the used gate library and the respective cost metrics, but also fault models for testing or abstractions for technology mapping. Due

to the progress in the development of physical realizations, these models constantly are subject to modifications which needed to be considered in the design phase. During the seminar, recent achievements in the development of physical realizations have been presented. This built the basis for discussions about updating and refining the applied models and abstractions.

■ **Applications**

How can reversible and quantum circuits be exploited in practically relevant application? How to measure and proof the benefits of these emerging technologies (e.g. how to substantiate improvements in the power consumption)?

So far, design methods have mostly been applied to “academic” examples only. However, the design of reversible circuits for precise applications is the next logical step. Possible directions (e.g. in the low-power domain) have been discussed at the seminar. This also included discussions of the requirements for such applications and how the benefits can be measured.

Results of the seminar are currently used in the preparation of upcoming scientific papers. As one example, a special issue on the results triggered by the ideas of this seminar is planned for 2013. Furthermore, the discussions encouraged the preparation of proposals for national and international research projects.

2 Table of Contents

Executive Summary

<i>Kenichi Morita and Robert Wille</i>	47
--	----





Overview of Talks

What do reversible Turing machines compute? <i>Holger Bock Axelsen</i>	51
Reversible CMOS computers <i>Alexis De Vos</i>	51
Reversible Logic Synthesis via Template Matching <i>Gerhard W. Dueck</i>	52
Quantum Finite automata on infinite words <i>Rusins Martins Freivalds</i>	52
Formal Verification of Quantum Systems <i>Simon Gay</i>	52
Principles of Reversible Computing <i>Robert Glück</i>	53
Quantum Automata Theory – A Review <i>Mika Hirvensalo</i>	54
Describing and Optimizing Reversible Logic using a Functional Language <i>Michael Kirkedal Thomsen</i>	54
Decision Diagram Techniques for Reversible and Quantum Circuit Equivalence Checking <i>D. Michael Miller</i>	56
Permutation Decision Diagrams (π DDs) and Analysis of Primitive Sorting Networks <i>Shin-ichi Minato</i>	56
Reversible logic elements with memory <i>Kenichi Morita</i>	57
Realization of Reversible Gates with New Quantum Gate Libraries <i>Zahra Sasanian</i>	57
RevKit: A Toolkit for Reversible Circuit Design <i>Mathias Soeken</i>	58
Testing and fault tolerance of reversible logic <i>Mehdi B. Tahoori</i>	58
Challenges in the Synthesis of Reversible Circuits: Today and Tomorrow <i>Robert Wille</i>	59
Logic level circuit optimization for topological quantum computation <i>Shigeru Yamashita</i>	60
A High-Level Reversible Programming Language <i>Tetsuo Yokoyama</i>	60
Participants	61

3 Overview of Talks

3.1 What do reversible Turing machines compute?

Holger Bock Axelsen (University of Copenhagen, DK)

License     Creative Commons BY-NC-ND 3.0 Unported license
© Holger Bock Axelsen





We gave an overview of recent computability and complexity results for reversible Turing machines. We showed that garbage-free reversible Turing machines (without an extraneous input copy in its output) can compute injective functions only, but that all injective, computable functions are in range [1]. We gave a definition of universality for reversible programs, and briefly outlined a universal reversible Turing machine [2]. Moving on to computational complexity, we discussed some consequences of tape reduction [3], and raised the question of whether analogous results are obtainable for reversible circuits.

References

- 1 Axelsen, H.B., Glück, R.: What do reversible programs compute? In: Hofmann, M. (ed.) FOSSACS 2011. LNCS, vol. 6604, pp. 42–56. Springer-Verlag (2011)
- 2 Axelsen, H.B., Glück, R.: A simple and efficient universal reversible Turing machine. In: Dediu, A.H., Inenaga, S., Martn-Vide, C. (eds.) LATA 2011. LNCS, vol. 6638, pp. 117–128. Springer (2011)
- 3 Axelsen, H.B.: Time complexity of tape reduction for reversible Turing machines. In: De Vos, A., Wille, R. (eds.) RC 2011. LNCS, vol. 7165, pp. 1–13. Springer (2012)

3.2 Reversible CMOS computers

Alexis De Vos (Gent University, BE)

License     Creative Commons BY-NC-ND 3.0 Unported license
© Alexis De Vos

Main reference A. De Vos, “Reversible computing,” Wiley-VCH, Weinheim (2010).
URL <http://users.elis.ugent.be/~aldevos/projects/computer.html>

CMOS (complementary metal-oxide-semiconductor) technology is the standard silicon technology for fabricating every-day electronic chips.

Applying the same transistor technology, but replacing AND, NAND, OR, NOR, and XOR gates by reversible gates, such as NOT gates, Feynman gates, Toffoli gates, and Fredkin gates, allows to build (classical) reversible circuits, i.e. chips that can compute either forwards or backwards, according to the applied driving force (i.e. applied voltages).

By using so-called adiabatic input signals (i.e. smooth voltage ramps), combined with dual-logic pass-transistor architecture, in principle, energy consumption per elementary computational step can be made arbitrarily small, and thus smaller than the Landauer limit. However, there is a fundamental trade-off with speed: the Landauer barrier can only be crossed if computing happens sufficiently slowly. A more practical obstacle is the threshold voltage of standard transistors. Only zero-threshold transistors can guarantee asymptotically zero energy consumption.


Standard threshold voltages lead to a more modest result: a reduction of the energy consumption by about a factor 10, compared to conventional digital circuit design.

References

- 1 A. De Vos: “Reversible computing”, Wiley-VCH, Weinheim (2010).

3.3 Reversible Logic Synthesis via Template Matching

Gerhard W. Dueck (University of New Brunswick at Fredericton, CA)


License  Creative Commons BY-NC-ND 3.0 Unported license
© Gerhard W. Dueck

Joint work of Dueck, Gerhard W.; Soeken, Mathias; Rahman, Mazder; Wille, Robert;

A review of reversible logic synthesis is presented. Heuristic synthesis of reversible circuits can be divided in two parts. First, find a reversible circuit (the circuit may be far from optimal). Second, optimize the circuit. One way to optimize the circuit is by applying rewriting rules, also known as templates. In this talk we prove two new results regarding template matching. First, the traditional definition of template does not guarantee minimality, since some templates are discarded. Secondly, we conjecture that given all templates with m lines and an exact template matching method an optimal circuit can be found for any circuit with m lines. A new solution for exact template matching is discussed. It should be feasible to find all templates with three lines. It remains to be seen how much reduction is possible for circuits with more than three lines. Some directions for further work are given.

3.4 Quantum Finite automata on infinite words


Rusins Martins Freivalds (University of Latvia, LV)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Rusins Martins Freivalds

The definition of language recognition by quantum finite automata in the case when the language consists of infinite words is somewhat non-trivial because there is no easy way how to consider a measurement after processing an infinite word. However a natural definition is possible based on measure theory developed by E.Borel and the standard definition of language recognition by deterministic and nondeterministic finite automata in the case when the language consists of infinite words developed by R.Büchi. We prove that Measure-Many finite quantum automata can recognize with probability 1 languages not recognizable by deterministic and nondeterministic finite automata. On the other hand, Measure-Once finite quantum automata can recognize with a bounded error only languages recognizable by deterministic finite automata.

3.5 Formal Verification of Quantum Systems

Simon Gay (University of Glasgow, GB)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Simon Gay

Joint work of Gay, Simon; Nagarajan, Rajagopal; Papanikolaou, Nikolaos;
Main reference S. J. Gay, N. Papanikolaou, R. Nagarajan, “QMC: a model checker for quantum systems,” in Proc. of the 20th Int’l Conf. on Computer Aided Verification (CAV). Springer LNCS 5123:543–547, 2008.
URL http://dx.doi.org/10.1007/978-3-540-70545-1_51

The field of formal methods is well established in classical computing. It defines formal languages in which to model systems and specify their desired properties, theories in which to express satisfaction of specifications by systems, and automated tools for verification. A range of formal methods techniques and tools have been applied to the analysis of classical


computing systems, especially concurrent and distributed systems, in diverse areas including networking, security, and safety-critical systems.

During the last several years, my collaborators and I have been developing formal methods for quantum systems and, more generally, systems that combine classical and quantum computation and communication. Our results include development of the theory of quantum process calculus, and a prototype model-checking tool for automatic verification of quantum systems (for example, communication protocols).

In my talk I motivated this research programme, argued that formal specification and verification should form part of the design process and tool chain for quantum systems, and presented a selection of results. The talk included a demonstration of the quantum model-checking tool QMC.

3.6 Principles of Reversible Computing

Robert Glück (University of Copenhagen, DK)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Robert Glück


We presented the main principles of reversible programming languages and discussed their relation to the design of reversible computer hardware. This includes a clear distinction between the mathematical specification and the operational properties of reversible programs which manifests itself in two main design steps: injectivization of the functional specification and the reversiblization of a program. The simple reversible programming language Janus was introduced as well as design principles for "good" reversible programming languages. We also outlined the main direction for what one may call 'reversible computing science'.

References

- 1 Abramov S. M., Glück R., Principles of inverse computation and the universal resolving algorithm. In: Mogensen T. A., Schmidt D. A., Sudborough I. H. (eds.), *The Essence of Computation: Complexity, Analysis, Transformation*. Lecture Notes in Computer Science, Vol. 2566, 269–295, Springer-Verlag 2002.
- 2 Axelsen H. B., Glück R., Yokoyama T., Reversible machine code and its abstract processor architecture. In: Diekert V., Volkov M.V., Voronkov A. (eds.), *Computer Science – Theory and Applications*. Proceedings. Lecture Notes in Computer Science, Vol. 4649, 56–69, Springer-Verlag 2007.
- 3 Axelsen H. B., Glück R., De Vos A., Thomsen M. K., MicroPower: towards low- power microprocessors with reversible computing. *ERCIM News*, Special Theme: Towards Green ICT, 79(1): 20–21, 2009.
- 4 Axelsen H. B., Glück R., What do reversible programs compute? In: Hofmann M. (ed.), *Foundations of Software Science and Computation Structures*. Proceedings. Lecture Notes in Computer Science, Vol. 6604, 42–56, Springer- Verlag 2011.
- 5 Glück R., Kawabe M., A method for automatic program inversion based on LR(0) parsing. *Fundamenta Informaticae*, 66(4): 367–395, 2005.
- 6 Yokoyama T., Glück R., A reversible programming language and its invertible self-interpreter. In: *Partial Evaluation and Program Manipulation*. Proceedings. 144–153, ACM Press 2007.
- 7 Yokoyama T., Axelsen H. B., Glück R., Principles of a reversible programming language. *Conference on Computing Frontiers*. Proceedings. 43–54, ACM 2008.

3.7 Quantum Automata Theory – A Review

Mika Hirvensalo (University of Turku, FI)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Mika Hirvensalo

Main reference M. Hirvensalo, “Quantum Automata with Open Time Evolution,” *International Journal of Natural Computing Research* 1, pp. 70–85 (2010).

URL <http://dx.doi.org/10.4018/jncr.2010010104>

The main models of quantum finite automata are presented and their main properties are reviewed.


Measure-once -model by Moore and Crutchfield [5] is presented as a natural variant of probabilistic automata [6]. A further variant, Measure-many automata [4] were introduced by Kondacs and Watrous to enhance the language recognition capability. Ambainis & al. introduced Latvian automata [1], a model with more elegant closure properties and algebraic characterization. Hirvensalo [2], [3] introduced quantum automata with open time evolution, and this model can be seen as a true generalization, not only variant, of classical finite automata.

References

- 1 Andris Ambainis, Martin Beaudry, Marats Golovkins, Arnolds Kikusts, Mark Mercer and Denis Therien: Algebraic Results on Quantum Automata. *Theory of Computing Systems* 39: 1, 165–188 (2006).
- 2 Mika Hirvensalo: Various Aspects of Finite Quantum Automata. LNCS 5257 (Proceedings of DLT 2008), pp. 21–33 (2008)
- 3 Mika Hirvensalo: Quantum Automata with Open Time Evolution. *International Journal of Natural Computing Research* 1, pp. 70–85 (2010).
- 4 A. Kondacs and J. Watrous: On the power of quantum finite state automata. *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pp. 66-75 (1997).
- 5 C. Moore, J. Crutchfield: Quantum automata and quantum grammars. *Theoretical Computer Science* 237, pp. 275–306 (2000).
- 6 M.O. Rabin: Probabilistic Automata. *Information and Control* 6, pp. 230–245 (1963).

3.8 Describing and Optimizing Reversible Logic using a Functional Language

Michael Kirkedal Thomsen (University of Copenhagen, DK)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Michael Kirkedal Thomsen

The presentation showed the design of a language for the description and optimization of reversible logic circuits. The language is functional and based on combinators, where the most recognizable of these combinators is inversion, βf , that defines the inverse function of f using an efficient semantics.

It is important to ensure that all language constructs are reversible and for this language we will, furthermore, require that this is done by static analysis only.

For most reversible languages a run-time analysis is needed, but for circuit design a run-time check of the description is undesirable.


Future work will show that the combination of the functional language and the restricted reversible model will allow more possibilities in the term rewriting and, thus, we expect to do a better optimization than other optimization techniques for reversible circuits.

References

- 1 T. Altenkirch and J. Grattage. A functional quantum programming language. In 20th Annual IEEE Symposium on Logic in Computer Science, 2005. LICS 2005. Proceedings, pages 249–258. IEEE, June 2005.
- 2 H. B. Axelsen and R. Glück. What do reversible programs compute? In M. Hofmann, editor, FOSSACS, volume 6604 of LNCS, pages 42–56. Springer, 2011.
- 3 H. B. Axelsen, R. Glück, and T. Yokoyama. Reversible machine code and its abstract processor architecture. In V. Diekert, M. V. Volkov, and A. Voronkov, editors, CSR, volume 4649 of LNCS, pages 56–69. Springer, 2007.
- 4 J. Backus. Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. In Communications of the ACM, volume 21, pages 613–641. ACM, 1978.
- 5 A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. Physical Review A, 52(5):3457–3467, 1995.
- 6 C. H. Bennett. Logical reversibility of computation. IBM Journal of Research and Development, 17:525–532, 1973.
- 7 E. Fredkin and T. Toffoli. Conservative logic. International Journal of Theoretical Physics, 21(3-4):219–253, 1982.
- 8 C. Lutz. Janus: A time-reversible language. A letter to R. Landauer. <http://tetsuo.jp/ref/janus.pdf>, 1986.
- 9 M. Sheeran. muFP, a language for VLSI design. In Proceedings of the 1984 ACM Symposium on LISP and functional programming, LFP 84, pages 104–112. ACM, 1984.
- 10 M. Sheeran. Hardware design and functional programming: a perfect match. Journal of Universal Computer Science, 11(7):1135–1158, jul 2005.
- 11 M. K. Thomsen, R. Glück, and H. B. Axelsen. Reversible arithmetic logic unit for quantum arithmetic. Journal of Physics A: Mathematical and Theoretical, 43(38):382002, 2010.
- 12 T. Toffoli. Reversible computing. In J. W. de Bakker and J. van Leeuwen, editors, ICALP, volume 85 of LNCS, pages 632–644. Springer, 1980.
- 13 V. Vedral, A. Barenco, and A. Ekert. Quantum networks for elementary arithmetic operations. Physical Review A, 54(1):147–153, July 1996.
- 14 R. Wille, S. Offermann, and R. Drechsler. SyReC: A programming language for synthesis of reversible circuits. In Proceedings of the Forum on Specification & Design Languages, pages 1–6, Southampton, UK, September 2010. IET.
- 15 T. Yokoyama, H. B. Axelsen, and R. Glück. Towards a reversible functional language. In A. De Vos and R. Wille, editors, RC 2011. Revised Selected Papers, LNCS. Springer, 2012, *to appear*.
- 16 T. Yokoyama and R. Glück. A reversible programming language and its invertible self-interpreter. In Partial Evaluation and Program Manipulation. Proceedings, pages 144–153. ACM Press, 2007.

3.9 Decision Diagram Techniques for Reversible and Quantum Circuit Equivalence Checking

D. Michael Miller (University of Victoria, CA)

License  Creative Commons BY-NC-ND 3.0 Unported license
© D. Michael Miller

Joint work of Miller, D. Michael; Wille, Robert; Grosse, Daniel; Drechsler, Rolf

Main reference R. Wille, D. Grosse, D.M. Miller, R. Drechsler, “Equivalence Checking of Reversible Circuits,” in Proc. ISMVL-2009, pp. 324–330, May, 2009.


URL <http://dx.doi.org/10.1109/ISMVL.2009.19>

The first part of this presentation is tutorial in nature. It reviews a selection of decision diagram techniques and how they have been applied to matrix representation for reversible and quantum circuits. The second part of the presentation shows how decision diagrams can be used to check the equivalence of two circuits regardless of the number of ancillary inputs and garbage outputs using novel techniques employing simple matrix operations.

The presentation addresses reversible circuits with binary inputs and outputs, but it is readily extended to the multiple-valued case, and to general quantum circuits. While this method has been developed using one particular type of decision diagram, QMDD, it should be straightforward to implement it using other decision diagram structures developed for quantum gates and circuits.

3.10 Permutation Decision Diagrams (π DDs) and Analysis of Primitive Sorting Networks

Shin-ichi Minato (Hokkaido University, JP)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Shin-ichi Minato

Main reference S. Minato, “ π DD: A New Decision Diagram for Efficient Problem Solving in Permutation Space,” in Proc. of 14th Int’l Conf. on Theory and Applications of Satisfiability Testing (SAT’11), pp. 90–104, 2011.

URL http://dx.doi.org/10.1007/978-3-642-21581-0_9

Recently, we proposed a new type of decision diagram named “ π DD,” for compact and canonical representation of a set of permutations. Similarly to an ordinary BDD or ZDD, π DD has efficient algebraic set operations such as union, intersection, etc. In addition, π DDs have a special Cartesian product operation which generates all possible composite permutations for two given sets of permutations. This is a beautiful and powerful property of π DDs.

In this talk, we present “permutation family algebra” based on π DDs, and how to describe and solve permutational problems using the algebra. We also show an application of π DDs for analyzing primitive sorting networks.

We succeeded in calculating the number of ways to construct minimum primitive sorting networks as 2,752,596,959,306,389,652 for $n = 13$, which has not been known in the past, where n is the width of the primitive sorting network.

3.11 Reversible logic elements with memory

Kenichi Morita (Hiroshima University, JP)

License © © ⊖ Creative Commons BY-NC-ND 3.0 Unported license
© Kenichi Morita

Main reference K. Morita, “Constructing a reversible Turing machine by a rotary element, a reversible logic element with memory,” Hiroshima University Institutional Repository, 2010.

URL <http://ir.lib.hiroshima-u.ac.jp/00029224>

We investigate a possibility of using reversible logic elements with memory (RLEM) as primitives for constructing reversible computing systems, from a theoretical standpoint. One of the characteristics of RLEMs is that there is no need to synchronize two or more input signals as in the case of reversible logic gates, because, in an RLEM, an incoming signal interacts only with its internal state, a stationary information. Another feature of an RLEM is that it can be modeled by an idealized physically reversible mechanical system easily, though its practical implementation in a reversible physical system is still very difficult as in the case of reversible gates. An important property of RLEMs is that some models of computing systems such as reversible Turing machines can be constructed by using a suitable RLEM very simply. We also discuss universality/non-universality of RLEMs. In the case of 2-state RLEMs, all but only four among the infinite kinds of RLEMs are universal in the sense that any reversible Turing machine can be built by each of them.

3.12 Realization of Reversible Gates with New Quantum Gate Libraries

Zahra Sasanian (University of Victoria, CA)


License © © ⊖ Creative Commons BY-NC-ND 3.0 Unported license
© Zahra Sasanian

Joint work of Sasanian, Zahra; Miller, D. Michael

The synthesized reversible circuits are usually evaluated based on quantum cost models. The most common quantum library for realizing a class of reversible gates called Multiple-Control Toffoli (MCT) gates is the NCV (NOT, CNOT, V, and V+) library. We presented two new quantum gate libraries as the target for technology mapping MCT cascades and showed their advantages over the well-known NCV library. The first new quantum library, NCVW, contains all the gates in the NCV library plus a gate implementing a fourth root of the NOT gate called the W gate and its inverse gate denoted by W+. The second new library includes NCV gates that are controlled by the quantum value, V1, instead of zero or one. A linear decomposition structure has also been proposed that utilizes this library to realize MCT gates with low linear quantum cost. The experimental results show that using these target libraries lead to less expensive circuits in terms of the number of gates compared to NCV circuits.

3.13 RevKit: A Toolkit for Reversible Circuit Design

Mathias Soeken (Universität Bremen, DE)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Mathias Soeken

Joint work of Soeken, Mathias; Frehse, Stefan; Wille, Robert; Drechsler, Rolf

Main reference M. Soeken, S. Frehse, R. Wille, R. Drechsler, “RevKit: An Open Source Toolkit for the Design of Reversible Circuits,” in *Reversible Computation 2011*, ser. Lecture Notes in Computer Science, vol. 7165, 2012, pp. 64–76, RevKit is available at www.revkit.org.

URL <http://www.revkit.org/>


RevKit is an open source toolkit for reversible circuit design. The motivation behind it is to make recent developments in the domain of reversible circuit design accessible to other researchers. Many approaches which are only available either independently or not at all can now be used under a common hood. This allows for the integration of different techniques in order to create new work flows. Furthermore, a C++ API and a Python interface make it possible to integrate own approaches. In this sense, RevKit addresses users who simply want to apply the framework and its tools as well as developers who actively want to develop further methods on top of the framework. RevKit is available at www.revkit.org.

References

- 1 M. Soeken, S. Frehse, R. Wille, and R. Drechsler, “RevKit: An Open Source Toolkit for the Design of Reversible Circuits,” in *Reversible Computation 2011*, ser. Lecture Notes in Computer Science, vol. 7165, 2012, pp. 64–76, RevKit is available at www.revkit.org.

3.14 Testing and fault tolerance of reversible logic

Mehdi B. Tahoori (KIT – Karlsruhe Institute of Technology, DE)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Mehdi B. Tahoori

Due to anticipated high failure rate of the emerging technologies to be used for realization of reversible logic, thorough testing and fault tolerance are crucial aspects in reversible circuits.

Fault masking techniques (to prevent error propagation) for reversible logic are presented and different implementations of reversible majority voters are shown. Using voter insertion techniques and taking advantage of available non-functional outputs, we are able to provide diagnosis information with adjustable resolution for reversible circuits. Such diagnosability has important applications in manufacturing testing as well as online repair. In contrast to previous work this voter is robust against single point of failure. We target missing and repeated gate fault models which are specific to reversible logic [1].

Also, an approach for online detection of faults in reversible circuits is presented. In this approach, reversible gates are modified in such a way that they can produce information on the number of cascaded gates. By using such information an appropriate reversible gate is added to detect missing and repeated gate faults [2].





By taking advantage of reversibility of these circuits, a new testing approach for reversible circuits is developed. In this method, the next test pattern is the response of the reversible circuit to the previous test pattern. This approach requires small amount of test information which makes it suitable for BIST implementation [3].

References

- 1 M. Zamani, and M.B. Tahoori. Online Missing/Repeated Gate Faults Detection in Reversible Circuits. In *Defect and Fault Tolerance of VLSI Systems*, pp. 435–442, oct. 2011.
- 2 M. Zamani, N. Farazmand, and M.B. Tahoori. Fault masking and diagnosis in reversible circuits. In *European Test Symposium*, pp. 69–74, may 2011.
- 3 M. Zamani, M. Tahoori, K. Chakrabarty, Ping-Pong Test: Compact Test Vector Generation for Reversible Circuits. In *VLSI Test Symposium*, 2012.

3.15 Challenges in the Synthesis of Reversible Circuits: Today and Tomorrow

Robert Wille (*Universität Bremen, DE*)

License     Creative Commons BY-NC-ND 3.0 Unported license
© Robert Wille

Joint work of Drechsler, Rolf; Wille, Robert;

Main reference R. Drechsler, R. Wille, “From Truth Tables to Programming Languages: Progress in the Design of Reversible Circuits,” *ISMVL’11Tutorial*.

URL http://www.informatik.uni-bremen.de/agra/doc/konf/11_ismvl_reversible_circuit_design_tutorial.pdf


In the last decade, significant progress in the development of design methods for reversible circuits has been made. First synthesis approaches relied on function representations like truth tables or permutations. They enabled synthesis of reversible circuits with a minimal number of circuit lines, but were applicable to quite small functions only. As a result, researchers strived for more scalable synthesis approaches leading e.g. to ESOP-based synthesis, BDD-based synthesis or synthesis based on the preliminary hardware description language SyReC. While they enabled synthesis of more complex functions, they usually lead to circuits with too many circuit lines. Hence, after solving the “scalability”-problem, how to keep the number of circuit lines small remains as important research question.

References

- 1 R. Drechsler and R. Wille, “From Truth Tables to Programming Languages: Progress in the Design of Reversible Circuits,” in *Int’l Symp. on Multiple-Valued Logic*, 2011, pp. 78–85.
- 2 D. M. Miller, D. Maslov, and G. W. Dueck, “A transformation based algorithm for reversible logic synthesis,” in *Design Automation Conf.*, 2003, pp. 318–323.
- 3 D. Große, R. Wille, G. W. Dueck, and R. Drechsler, “Exact multiple control Toffoli network synthesis with SAT techniques,” *IEEE Trans. on CAD*, vol. 28, no. 5, pp. 703– 715, 2009.
- 4 K. Fazel, M. Thornton, and J. Rice, “ESOP-based Toffoli gate cascade generation,” in *Communications, Computers and Signal Processing, 2007. PacRim 2007. IEEE Pacific Rim Conference on*, 2007, pp. 206 –209.
- 5 R. Wille and R. Drechsler, “BDD-based synthesis of reversible logic for large functions,” in *Design Automation Conf.*, 2009, pp. 270–275.
- 6 R. Wille, S. Offermann, and R. Drechsler, “SyReC: A programming language for synthesis of reversible circuits,” in *Forum on Specification and Design Languages*, 2010, pp. 184–189.
- 7 R. Wille, O. Keszöcze, and R. Drechsler, “Determining the minimal number of lines for large reversible circuits,” in *Design, Automation and Test in Europe*, 2011, pp. 1204–1207.
- 8 R. Wille, M. Soeken. E. Schönborn, and R. Drechsler, “Circuit Line Minimization in the HDL-based Synthesis of Reversible Logic,” 2012.

3.16 Logic level circuit optimization for topological quantum computation

Shigeru Yamashita (Ritsumeikan University – Shiga, JP)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Shigeru Yamashita


Joint work of Yamashita, Shigeru; Devitt, Simon; Nemoto, Kae;

I formulated the logic level circuit optimization problem for topological quantum computation. Observing the properties of "braiding operations" in topological quantum computation, I formulated our problem as to find a good gate order and a good initial qubit order.

I then presented some heuristics for the problem.

3.17 A High-Level Reversible Programming Language

Tetsuo Yokoyama (Nanzan University, JP)

License  Creative Commons BY-NC-ND 3.0 Unported license
© Tetsuo Yokoyama

We presented a summarizing talk that covered previous work on programming methodology in reversible computing including developing a better language for writing reversible programming, better ways to write reversible programming, and the fundamental concepts on reversible languages. First, we attempted to share brief overview of a high-level reversible programming language Janus.

Next, apart from particular computational models, the fundamental problems of irreversibility and approaches to their solutions were identified. The source of irreversibility stems from irreversible data update and backward nondeterministic control flow. The solutions for avoiding those irreversibility are mainly divided into unclean and clean approaches. The superiority of clean approach was discussed by using Janus, which is a clean reversible language.

Then, we described the details on programming techniques in a high-level reversible programming language. As other programming paradigms do, reversible programming gives us opportunity to access to its own way of modularity. After that, properties of reversible programming languages were briefly discussed. We posed a few of our current research questions. Our recent result on the domain-specific optimization of reversible simulation, twice as fast as Bennett method, is represented in the form of reversible logic gates, to show the potential usefulness of the reversible simulation in the domain-specific optimization of reversible logic gates.

Participants

- Holger Bock Axelsen
University of Copenhagen, DK
- Stéphane Burignat
Ghent University, BE
- Alexis De Vos
Ghent University, BE
- Rolf Drechsler
Universität Bremen, DE
- Gerhard W. Dueck
University of New Brunswick at
Fredericton, CA
- Rusins Martins Freivalds
University of Latvia, LV
- Simon Gay
University of Glasgow, GB
- Robert Glück
University of Copenhagen, DK
- Mika Hirvensalo
University of Turku, FI
- Pawel Kerntopf
Warsaw Univ. of Technology, PL
- Michael Kirkedal Thomsen
University of Copenhagen, DK
- D. Michael Miller
University of Victoria, CA
- Shin-ichi Minato
Hokkaido University, JP
- Kenichi Morita
Hiroshima University, JP
- Zahra Sasanian
University of Victoria, CA
- Julia Seiter
Universität Bremen, DE
- Mathias Soeken
Universität Bremen, DE
- Marek Szymowski
Warsaw Univ. of Technology, PL
- Mehdi B. Tahoori
KIT – Karlsruhe Institute of
Technology, DE
- Robert Wille
Universität Bremen, DE)
- Shigeru Yamashita
Ritsumeikan Univ. – Shiga, JP
- Tetsuo Yokoyama
Nanzan University, JP

