Technical Report: Projects & Background in Reversible Logic

TR-CSJR1-2005
J. E. Rice
University of Lethbridge
j.rice@uleth.ca
updated Jan. 2013

1 Introduction - What is Reversible Logic?

In order to discuss new trends and projects in the area of reversible logic one must first have an understanding of what this is. First of all, we'll restrict our discussion of logic functions to two-valued functions describing switching logic. Reversible multiple-valued functions are also possible, but are beyond the scope of this report. Circuits that implement logic functions are generally built using logic *gates*. According to Shende *et al.* [22],

Definition 1.1 *a gate is reversible if the (Boolean) function it computes is bijective.*

Bijective means one-to-one and onto; or, for those of us who forget our mathematics terminology, there must be the same number of inputs as outputs, and for each output value there is a unique input value that leads to it.

Table 1 shows the truth table for a 3x3 reversible function. Note that a reduced representation can be obtained from the truth table by simply listing the row numbers correspoinding to the binary expansions represented by the inputs and by the outputs. If we assume that the inputs are given in numerical order from 0 to 2^{n-1} then we can list simply the decimal numbering of the outputs, in this case $0, 1, 3, 2, 4, 5, 7, 6$.

Since, by our previous definition, reversible gates are required in order to build reversible logic, I next define some reversible gates and how they behave. Again according to Shende *et al.* [22],

Definition 1.2 *A k-CNOT is a $(k + 1) \times (k + 1)$ gate. It leaves the first k inputs unchanged, and inverts the last* iff *all others are 1.*

There are many types of k-CNOT gates, and they are referred to in the literature in a variety of ways. Here is a few of the names given to the varieties:

- A 0-CNOT gate is just an inverter, referred to as a NOT gate.

	xyz	x'y'z'	
0	000	000	0
1	001	001	1
2	010	011	3
3	011	010	2
4	100	100	4
5	101	101	5
6	110	111	7
7	111	110	6

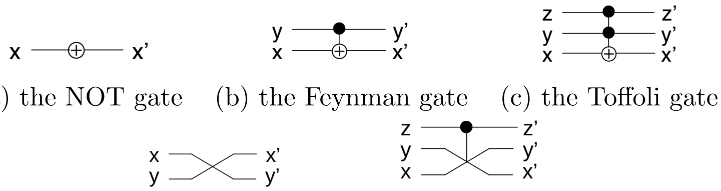
Table 1: The truth table of a 3x3 reversible function.

gate	behaviour
Not	$(x) \rightarrow (x \oplus 1)$
Feynman	$(y, x) \rightarrow (y, x \oplus y)$
Toffoli	$(z, y, z) \rightarrow (z, y, x \oplus yz)$
swap	$(x, y) \rightarrow (y, x)$
Fredkin	$(z, y, z) \rightarrow (z, x, y)\ iff\ z = 1$

Table 2: The behaviour of a selection of more commonly used reversible logic gates.

- A 1-CNOT gate is called a controlled-NOT, or CNOT (this is also known as a Feynman gate).
- A 2-CNOT gate is called a TOFFOLI gate.

Other gates include the SWAP gate and the FREDKIN gate. Table 2 lists the behaviour of each of the most commonly used reversible gates. Figure 1 illustrates the symbols usually used for each of the gates. Each of the Toffoli,



(a) the NOT gate (b) the Feynman gate (c) the Toffoli gate

(d) the swap gate (e) the Fredkin gate

Figure 1: Symbols for some of the more commonly-used reversible logic gates.

Fredkin and CNOT gate are universal gates [12]; that is, they each can be used to create any logic circuit without the addition of any other type of gate. In traditional logic the AND gate is a universal gate.

There are many different notations for these and other gates. The symbols in Figure 1 are from [6]. Dueck *et al.* [6] also use the following notation:

- TOF(C; T) denotes a Toffoli gate, where C is a set of 0 or more control inputs and T is the input to be inverted,
- FRE(C; T) denotes a Fredkin gate, where C is a set of 0 or more control inputs and T is the inputs to be swapped.

The NOT gate can then be written TOF(; T) and a SWAP can then be written FRE(; T).

If one creates a reversible function by listing the 2^n possible rows for any n-input n-output function and creates the outputs by simply permuting the input rows, one might call the resulting logic function a "gate". However there are some characteristics that are desirable before we can label an arbitrary function as a gate. Firstly, it is desirable for the gate to be universal, if possible. This means that it should be possible to create any possible function by using a cascade of the gate in question. For instance, the NAND gate is universal in traditional logic, in that any logic function can be built simply by cascading NAND gates together. The Toffoli gate is also universal. Secondly, it is desirable to have a simple, or even elemental implementation available for each gate. This is so that a conversion from a gate layout to a physical implementation can be performed in a straight-forward way by mapping each gate to its basic underlying implementation. For instance, in traditional logic a NOT gate requires 2 transistors, while a two-input NAND gate requires 4 transistors. Thus in traditional logic the elementary unit could be considered to be the transistor. Similarly, in quantum and reversible computing, the elementary unit is the quantum gate. There are a variety of suggestions for how such a gate could be implemented; the reader is directed to works such as [17] for recent developments and to [18] for a comprehensive discussion of quantum computing. In general, an elementary quantum gate acts on a single qubit and is described as some mathematical transform that represents the change of state of that qubit to some other state. For instance, the Hadamard gate represents a rotation of π about the x and z axes. Many of the researchers in the reversible logic literature refer to Barenco *et al.* [3] for the numbers of elementary quantum gates required for implementing gates such as the Toffoli gate. More recently Miller, Wille and Sasanian have proposed updates to these quantum costs, as they are now referred to [16]. Reversible circuits are generally compared based on their quantum cost and number of qubits required, thus the computation of quantum cost is very important in this area.

2 Motivation - Why Reversible Logic?

In addition to understanding the background of a topic it is equally important to understand why a particular topic is of interest. According to Frank [8]

> ...computers based mainly on reversible logic operations can reuse a fraction of the signal energy that theoretically can approach arbitrarily near to 100% as the quality of the hardware is improved...

Many researchers believe that Moore's law is at an end. We can't keep increasing performance as we have previously done, in order to meet consumer demands, because we simply can't keep up with the power requirements. For an extremely convincing explanation of why this is so the reader is directed to section 1.1 of [8]. If this doesn't convince you then possibly Bennet's statement that "loss of information implies energy loss" [4] and Perkowski *et al.'s* convictions that "every future technology will have to use reversible gates in order to reduce power" and "[our reversible techniques are] useful for arbitrary reversible technology, *e.g.* quantum, CMOS, DNA, optical, *etc.*" [19] may convince us of the usefulness of pursuing research in the area of reversible logic.

3 Issues

This is a relatively new area of work. One prominent researcher has identified four big problems in the area. Frank [8] states that

1. we need to develop fast and cheap switching devices with adiabatic energy coefficients well below those of transistors;

2. we also need clocking systems that are themselves of very high reversible quality;

3. it is also essential that we pursue research into the design of highly-optimized reversible logic circuits and algorithms.

4. Finally, the area faces an uphill social battle in overcoming the enormous inertia of the established semiconductor industry.

Below I describe three research directions that fit into these areas.

4 Research Directions

Much of my prior work has been in the area of traditional logic synthesis. Thus it has been logical for me to extend this knowledge to problem 3 in Frank's list; the problem of developing highly-optimized reversible logic circuits and algorithms. This is, of course, logic synthesis.

Definition 4.1 *Reversible logic synthesis: given a truth table or other specification of a reversible Boolean logic function, how do we generate (what is usually) a cascade of reversible gates to implement the function(s)?*

4.1 Sequential Logic Synthesis for Reversible Logic Circuits

Much of the early *e.g. prior to 2008* literature on reversible logic synthesis addressed primarily combinational logic. Some work in sequential logic synthesis for reversible logic is listed below:

- M. Frank, in “Approaching the Physical Limits of Computing” [7],
- Kwon *et al.* in “A three-port nRERL register file for ultra-low-energy applications” [11],
- Picton, in “Multi-Valued Sequential Logic Design Using Fredkin Gates” [20], and
- Thapliyal *et al.* in “A Beginning in the Reversible Logic Synthesis of Sequential Circuits” [23],
- Rice, “An Introduction to Reversible Latches” [21].

Part of the problem is that the traditional description of reversible logic gates seems to contradict the requirements for building sequential circuits. However, in order to make reversible logic a feasible tool there must be some type of revrsible memory gate or object, and tools for sequential logic design. The first mention of sequential design for reversible logic was by Picton. He describes the use of Fredkin gates to build clocked D-type latches, which can then be combined to form more complex memory elements required in sequential logic design. Since then Frank (in various presentations, including [7]) and Thapliyal *et al.* have considered this issue. I include Kwon *et al.* in the above list since in order to design a register, by nature sequential logic must be used. However synthesis techniques are not mentioned at all in this work; indeed it appears that the design is a one-off hand-tooled design, not suitable for general logic synthesis or what we today know as CAD.

My own more recent work has investigated the design of latches on a reversible basis, and further work since then has also been presented. However the area of general synthesis, incorporating sequential elements for reversible logic, is still lacking at the time that this report was updated.

4.2 Decision Diagram-based Synthesis for Reversible Logic Circuits

Existing research into reversible logic synthesis seems to fall into two main categories:

1. transform-based techniques, such as those introduced by Dueck *et al.* [6, 15, 14], and
2. decomposition techniques, such as those introduced by Perkowksi *et al.* [19], De Vos *et al.* [26] and Miller [13].

Other researchers in the area include Kerntopf [10, 9], Agrawal *et al.* [2], and Shende *et al.* [22]. These latter works rely more on heuristics for deciding how to build reversible circuits.

Any of us familiar with traditional logic synthesis will likely be aware of the impact Bryant’s [5] binary decision diagrams have had on the area. Perkowski does certainly mention the use of PKDDs [19] but gives no information on how the DD structure is then converted into a reversible gate layout. Thornton, Miller and Goodman [24] have introduced a DD package for quantum (and reversible) circuit simulation, but have had difficulties in converting this to a synthesis tool, and [25] and [1] each have reported on DD-based tools for simulation and representation, but not synthesis. More recently, Wille and Drechsler [27] have publised a DD technique for *exact* synthesis of reversible circuits; this is the first work to report on a DD technique for synthesis in this area. Work should continue on comparing this approach to others and to determining varying DD-types and their applicability in this area.

4.3 Transforming a Traditional Gate-level Layout to a Reversible Logic Layout

This section cannot begin with a list of existing work because in this area, there is none. The motivation of this particular topic is to provide a bridge from existing traditional logic designs to the novel reversible logic designs. Let's face it; academia and industry has put a lot of time and effort into the existing designs we have. If we want to transition to reversible logic we must plan for it.

It would seem logical to begin with traditional gate layouts such as (N)AND-(N)OR implementations and devise algorithms and/or heuristics for generating reversible equivalents. This, of course, is the simplest of traditional logic designs; muxes, registers, and other more complex devices must be incorporated as the work progresses. Comparisons between reversible circuits developed with this approach, and those based on the synthesis approaches described above may provide useful insights as to how each technique can be improved, and how combinations of different techniques might be used in conjunction to achieve the best results.

5 Conclusion

This report provides the reader with a brief overview of and introduction to the area of reversible logic. Some motivation is given, followed by which some areas of work are suggested, accompanied by some references from the literature in each of these areas. This report was originally written in 2005, with minor updates in 2012. In this time work has progressed in the first of the two area of research suggested, but it is interesting to note that there is still no work in the final area of research presented in this report.

References

[1] A. Abdollahi and M. Pedram. Analysis and Synthesis of Quantum Circuits by Using Quantum Decision Diagrams. In *Volume 1 of the Proceedings of Design, Automation and Test in Europe (DATE)*, pages 1–6, 2006.

[2] A. Agrawal and N. K. Jha. Synthesis of Reversible Logic. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 1384–1385, 2004. 16–20 February, Paris, France, IEEE Computer Society.

[3] 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 computations. *Physical Review A*, 52(5):3457–3467, Nov 1995.

[4] C. H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 6:525–532, 1973.

[5] R. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Trans. on Comp.*, C–35(8):677–691, Aug. 1986.

[6] G. W. Dueck, D. Maslov, and D. M. Miller. Transformation-based synthesis of networks of Toffoli/Fredkin gates. In *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*, pages 211–214, vol.1, 2003.

[7] M. P. Frank. Approaching the physical limits of computing. In *Proceedings of the International Symposium on Multiple-Valued Logic (ISMVL)*, pages 168–187, 2005. May 18–21, Calgary, Alberta, IEEE Computer Society.

[8] M. P. Frank. Introduction to reversible computing: Motivation, progress, and challenges. In *Proceedings of the 2nd Conference on Computing Frontiers*, pages 385–390, 2005. May 4–6, Ischia, Italy, ACM Press.

[9] P. Kerntopf. Synthesis of Multipurpose Reversible Logic Gates. In *Proceedings of the Euromicro Symposium on Digital System Design: Architectures, Methods and Tools (DSD)*, pages 269–267, 2002.

[10] P. Kerntopf. A New Heuristic Algorithm for Reversible Logic Synthesis. In *Proceedings of the Design Automation Conference (DAC)*, pages 834–837, 2004. June 7–11, San Diego, CA, USA, ACM.

[11] J.-H. Kwon, J. Lim, and S.-I. Chae. A three-port nRERL register file for ultra-low-energy applications. In *Proceedings of the 2000 International Symposium on Low Power Electronics and Design (ISLPED '00)*, pages 161–166, 2000.

[12] Jon Marshall. http://www.themilkyway.com/quantum/finalreport/quantumgates, 2000.

[13] D. M. Miller. Spectral and two-place decomposition techniques in reversible logic. In *Proceedings of the IEEE Midwest Symposium on Circuits and Systems (MWCAS)*, pages II493–II496, 2002.

[14] D. M. Miller, G. W. Dueck, and D. Maslov. A synthesis method for MVL reversible logic. In *Proceedings of the 34th International Symposium on Multiple-Valued Logic (ISMVL)*, pages 74–80, 2004. May 19-22, Toronto, Canada , IEEE.

[15] D. M. Miller, D. Maslov, and G. W. Dueck. A transformation based algorithm for reversible logic synthesis. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 318–323, New York, NY, USA, 2003. ACM.

[16] D.M. Miller, R. Wille, and Z. Sasanian. Elementary quantum gate realizations for multiple-control toffoli gates. In *Proceedings of the 41st IEEE International Symposium on Multiple-Valued Logic (ISMVL)*, pages 288–293, may 2011.

[17] T. Monz, K. Kim, W. Hänsel, M. Riebe, A. S. Villar, P. Schindler, M. Chwalla, M. Hennrich, and R. Blatt. Realization of the quantum toffoli gate with trapped ions. *Phys. Rev. Lett.*, 102:040501, Jan 2009.

[18] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[19] M. Perkowski, L. Jozwiak, A. Mishchenko, A. Al-Rabadi, A. Coppola, A. Buller, X. Song, M. Khan, S. N. Yanushkevich, V. P. Shmerko, and M. Chrzanowska-Jeske. A general decomposition for reversible logic. In *Proceedings of the International Workshop on Methods and Representations (RM)*, pages 119–138, 2001. August 10–11, Starkville, Mississippi, USA, RM Workshop.

[20] P. Picton. Multi-Valued Sequential Logic Design using Fredkin Gates. *Multiple-Valued Logic*, 1:241–251, 1996.

[21] J. E. Rice. An introduction to reversible latches. *The Computer Journal*, 51(6):700–709, 2007.

[22] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes. Reversible logic circuit synthesis. In *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pages 353–360, 2002. November 10–14, San Jose, CA, USA, ACM.

[23] H. Thapliyal, M. B. Srinivas, and M. Zwolinski. A Beginning in the Reversible Logic Synthesis of Sequential Circuits. In *Proceedings of Military and Aerospace Programmable Logic Devices (MAPLD) International Conference*, page number 1012 (online proceedings), 2005. September 7–9, Washington, DC, USA, NASA Office of Logic Design.

[24] M. Thornton, D. M. Miller, and D. Goodman. A decision diagram package for reversible and quantum circuit simulation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 8597–8604, July 2006. held at the IEEE World Congress on Computational Intelligence (WCCI).

[25] G. F. Viamontes, I. L. Markov, and J. P. Hayes. QuIDDPro: High-performance quantum circuit simulation, 2006. http://vlsicad.eecs.umich.edu/Quantum/qp/.

[26] A. De Vos and Y. Van Rentergem. Reversible computing: From mathematical group theory to electronical circuit experiment. In *Proceedings of the 2nd Conference on Computing Frontiers*, pages 35–44, 2005. May 4–6, Ischia, Italy, ACM Press.

[27] Robert Wille and Rolf Drechsler. Bdd-based synthesis of reversible logic for large functions. In *Proceedings of the 46th Annual Design Automation Conference*, DAC '09, pages 270–275, New York, NY, USA, 2009. ACM.