An Analysis of Several Proposals for Reversible Latches

J. E. Rice
Department of Mathematics & Computer Science
University of Lethbridge
4401 University Drive
Lethbridge, AB, Canada
T1K 3M4

Abstract–**Recent work has begun to investigate the advantages of using reversible logic for the design of circuits. The majority of work, however, has limited itself to combinational logic. Researchers are just now beginning to suggest possibilities for sequential implementations. This paper performs a closer analysis of three latch designs proposed in previous work and suggests advantages and disadvantages of each**.

I. Introduction

Reversible computing has recently been re-introduced as a potential solution to the problem of the ever-growing demand for lower power devices. As stated by Frank [1]

> ...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...

However, reversible logic is suffering from two problems. Firstly, there is a lack of technologies with which to build reversible gates. Work is certainly continuing in this area. Secondly, while there is much research into how to design combinational circuits using reversible logic, there is little in the area of sequential reversible logic implementations. There is no limitation inherent to reversible logic preventing the design of sequential circuits; in fact when Tommaso Toffoli first characterized reversible logic in his 1980 work Reversible Computing [2] he stated that "*Using invertible logic gates, it is ideally possible to build a sequential computer with zero internal power dissipation*."

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

Inputs *xyz*	outputs *x'y'z'*
000	000
001	001
010	011
011	010
100	100
101	101
110	111
111	110

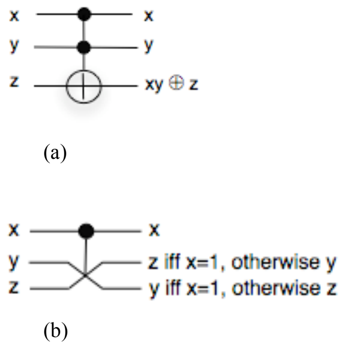
Researchers such as Rice [3] and Thapliyal, Srinivas and Zwolinski [4] have begun work in presenting memory elements such as reversible latches and reversible flip-flops. This paper presents an analysis of two reversible SR-latch implementations, and provides a comparison to the traditional SR-latch.

II. Background

A. Reversible Logic

Before discussing sequential reversible logic we first present the basic concepts underlying reversible logic. According to Shende *et al*. [5] a gate is reversible if the (Boolean) function it computes is bijective. This means that a function is reversible if there is a one-to-one and on-to mapping from the inputs to the outputs (and vice versa) of the function. At the very least, a reversible function must have the same number of inputs as it does outputs. For instance, the traditional NOT gate is reversible, but that the traditional AND gate is not.

Table 1 shows the truth table for a 3x3 reversible function. In such a function each output can be thought of as a transformed version of one of the inputs. The symbols and behaviours of the reversible gates used in this paper are shown in Figure 1.



(a)

(b)

Figure 1. (a) The Toffoli gate and (b) the Fredkin gate.

Table 2. The next state values for the SR-latch.

inputs	next state
S R	$Q+\ \overline{Q}+$
0 0	$Q\ \overline{Q}$ (same as previous state)
0 1	0 1
1 0	1 0
1 1	not permitted

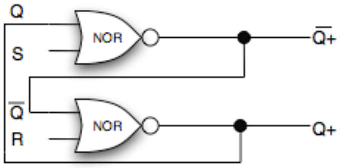


Figure 2.A traditional NOR-based SR=latch.

B. *The SR-Latch*

The primary focus of this paper is the SR-latch. This latch allows the outputs $Q+$ and $\overline{Q}+$ to be "set" to the values 1 and 0, respectively, or "reset" to 0 and 1. The primary inputs to the SR-latch are *S* (set) and *R* (reset). The behaviour of the SR-latch is characterized by the truth table given in Table 2. Figure 2 shows a traditional NOR-based structure for this latch.

III. Reversible SR-Latches

Given the need for reversible memory elements in order to build reversible sequential circuits, it seems reasonable to try to mimic the behaviour of the SR-latch using reversible logic gates. At first glance the SR-latch appears to exhibit most of the desirable reversible characteristics, except that there are two output possibilities when the inputs *SR* are set to 00. Closer examination shows that the SR-latch actually has four inputs: *S, R*, *Q* and $\overline{Q}$, where *Q* and $\overline{Q}$ represent the current state of the latch.

A reversible version of this latch must have at least three inputs; in an ideal situation these would consist of *S*, *R* and the non-inverted *Q*. The truth table for such a latch is shown in Table 3. One can see that an additional output has been added, labeled *g* for garbage. This output would not be used although is required to maintain the reversibility of the device. The first two rows of the table are illustrative of the process used in constructing such a table. In row 0 in order to maintain the characteristic of the SR-latch $Q+$ must be 0 and $\overline{Q}+$ must be 1. In row 1 the *R* input is a 1, thus outputs $Q+ = 0$ and $\overline{Q}+ = 1$. However this is a combination we have already used, so to differentiate between rows 0 and 1 we'll arbitrarily assign $g = 0$ in row 0 and $g = 1$ in row 1.

We encounter the same problem in rows 2 and 4, and so again we arbitrarily assign values to the garbage output *g*. In row 5, however we encounter problems: the outputs $Q+\overline{Q}+ = 01$ are required according to the functionality of the SR-latch, but this combination has been used twice before and so no possible assignment to *g* can make the outputs unique. The same problem is encountered in row 6, making it clear that a three-input/three-output reversible SR-latch is not possible.

Table 3. A truth table for a latch with two primary inputs *S* and *R*, and *Q* representing the current state of the latch.

row number	*Q S R*	$Q+\ \overline{Q}+\ g$
0	0 0 0	0 1 0
1	0 0 1	0 1 0
2	0 1 0	1 0 0
3	0 1 1	not permitted
4	1 0 0	1 0 1
5	1 0 1	0 1 ?
6	1 1 0	1 0 ?
7	1 1 1	not permitted

To solve the problem encountered in Table 3 we can add an output. This of course requires that we add an input. The desired behaviour is characterized in Table 4. Note that we have fixed the value of the additional input at 1, as we only need this input to "balance" the added output. The rows containing "XXXX" for the outputs are those where $S = R = 1$, which is not permitted.

The following subsections describe two possible alternatives for reversible SR-latches. Each is based on a similar design, but we describe various differing characteristics.

A. *Fredkin-based SR-Latch*

One of the first researchers to characterize a reversible latch was Picton [6]. He suggested a reversible SR-latch built out of Fredkin gates, as illustrated in Figure 2 (a). The problem with Picton's latch is that it incorporates fan-out, which is not permitted in reversible designs. One solution to this problem is shown in Figure 2 (b).

In this analysis and in all following discussions we assume a delay of 1 for each level of gates and an additional delay of 1 for the long wires propagating the values for $Q+$ and $\overline{Q}+$ back to the inputs. For instance, if the current state of the latch is $Q\overline{Q}SR = 0000$ then Table 5 illustrates how each of the values change.

Table 4. The state table for a four-input/four-output reversible latch.

1 *Q S R*	$Q+\ \overline{Q}+\ g_1\ g_2$
1 0 0 0	0 1 0 0
1 0 0 1	0 1 0 1
1 0 1 0	1 0 0 0
1 0 1 1	X X X X
1 1 0 1	0 1 1 0
1 1 1 0	1 0 1 0
1 1 1 1	X X X X

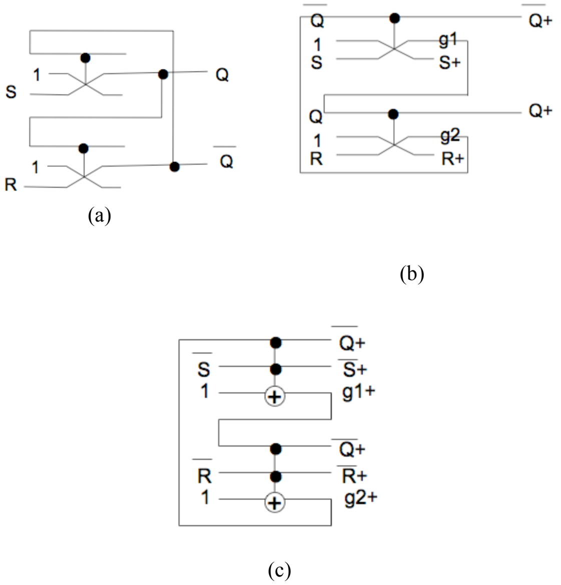


Figure 2. (a) The Fredkin-based reversible SR-latch as suggested by Picton, (b) a modified version of Picton's latch, and (c) A reversible SR-latch based on Toffoli gates.

Table 5 shows that with an initial state of $Q\bar{Q}$ = 00 and inputs SR = 00 the latch is unstable, oscillating (as does the traditional NOR-based SR-latch) between $Q{+}\bar{Q}{+}$ = 00 and 11. A similar table can be built starting values of $Q\bar{Q}SR$ = 0001. With these values the final state of the latch should be $Q{+}\bar{Q}{+}$ = 01, and our analysis shows that after 5 time periods the latch is stable at these values. Similar state tables can be derived for all 16 possible initial starting states of the latch, and these tables show that this latch is unstable when any one of the following four starting values for $\bar{Q}QSR$ are used: 0000, 1100, 0100 or 1000.

The first two cases are not entirely surprising, since they reflect the behaviour of the NOR-based latch on which this design was modeled. However, the second two cases are dismaying, since this behaviour is in violation of the required characteristics of the SR-latch. One possible fix for this behaviour is to intead add Toffoli gates to Picton's design in Figure 2 (a) in order to provide the fan-out signals. However, this modification was not initially suggested due to the additional delay caused by having two levels of gates internal to the latch and because of the additional requirement of another input.

B. *Toffoli-based SR-Latch*

A second reversible design for the SR-latch was also proposed by Rice [3]. This design is shown in Figure 3 (c). Again, state tables showing the changes in values for the latch can be derived for all 16 input combinations of $\bar{Q}QSR$. This latch is unstable for only two input cases: $\bar{Q}QSR$ = 0000 and 1100. Additionally, this latch responds to inputs such as $\bar{Q}QSR$ = 0100 very quickly, taking only two timesteps to stabilize.

We should note that a similar alternative to the latch shown in Figure 2 (c), again based on Toffoli gates, was also proposed by Rice [3]. Our analysis for this paper showed that this alternative does not have the required behaviour of a SR-latch, and so we have not included this latch in these comparisons.

C. *Comparisons*

During the course of this work we found that the traditional SR-latch became unstable under the input conditions $SRQ\bar{Q}$ = 0000 or $SRQ\bar{Q}$ = 0011. In both these cases the next state would oscillate between $Q{+}\bar{Q}{+}$ = 00 and 11. This behaviour is also reflected in the Toffoli and Fredkin-based SR-latches.

Table 5. An illustration of how the internal state of the Fredkin-based SR-latch changes, assuming a delay of 1 for processing at each level of gates and a delay of 1 for each propagation of values from outputs back to inputs.

Time period	$\bar{Q}\,Q\,S\,R$	$\bar{Q}{+}\,g_1\,S{+}$	$Q{+}\,g_2\,R{+}$
0	0 0 0 0		
1 (after gates process inputs)	0 0 0 0	0 1 0	0 1 0
2 (after propagating values back to inputs)	1 1 0 0	0 1 0	0 1 0
3 (after gates process inputs)	1 1 0 0	1 0 1	1 0 1
4 (after propagating values back to inputs)	0 0 0 0	1 0 1	1 0 1
5 (after gates process inputs)	0 0 0 0	0 1 0	0 1 0

Table 6. The average number of timesteps required for each latch to reach a stable state.

Latch type	Average delay
Traditional SR-latch	3.36
Toffoli-based SR-latch	3.71
Modified Picton SR-latch (no Toffoli gates)	3.83
Picton SR-latch with Toffoli gates	4.57

Table 6 compares the average number of timesteps required for each latch to reach a stable state. We identified a stable state by detecting a repetition of values on the inputs or the outputs. Input combinations resulting in instability were not included in our computations. The values were determined by assuming a delay of one for each level of gates to process their inputs, and a delay of one for signals to be propagated back to the inputs.

IV. Conclusion & Future Work

The purpose of this paper is both to illustrate the feasibility of reversible logic in sequential logic design, and to examine more closely the behaviour of a basic memory element, the SR-latch. We found that one of the proposed latch designs has some flaws, and while correction of these flaws was possible, it resulted in additional delay and the need of an additional input. We would conclude that the better reversible design for an SR-latch is the Toffoli-based design, as shown in Figure 2 (c).

There are many areas of work that may lead from this paper, most notably similar types of analysis for other latches and for flip-flops designed from these latches. Additionally, during the course of this work the authors noted the lack of simulation tools that support reversible gates, and this is most definitely an area worthy of attention. Finally an ongoing goal is to develop a synthesis process that will support reversible logic, and incorporate the sequential elements we are proposing in this work.

References

[1] M. P. Frank, “Introduction to Reversible Computing: Motivation, Progress, and Challenges,” in Proceedings of the 2nd Conference on Computing Frontiers, 2005, pp. 385–390.

[2] T. Toffoli, Automata, Languages and Programming. Springer Verlag, 1980, chapter: Reversible Computing, pp. 632–644.

[3] J. E. Rice, “A New Look at Reversible Memory Elements,” in Proceedings of the International Symposium on Circuits and Systems (ISCAS), 2006, to appear.

[4] 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, 2005, submission 1012.

[5] 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), 2002, pp. 353–360.

[6] P. Picton, “Multi-Valued Sequential Logic Design using Fredkin Gates,” Multiple-Valued Logic, pp. 241–251, 1996.