CONSTRUCTING ONLINE TESTABLE CIRCUITS USING REVERSIBLE LOGIC

Sk Noor Mahammad, Siva Kumar Sastry Hari, Shyam Shroff and V Kamakoti[1]

Abstract

Testable fault tolerant system design has become vital for many safety critical applications. On the other hand, reversible logic is gaining interest in the recent past due to its less heat dissipating characteristics. Any Boolean logic function can be implemented using reversible gates. This paper proposes a technique to convert any reversible logic gate to a testable gate that is also reversible. The resultant reversible testable gate can detect online any single bit errors that include Single Stuck Faults and Single Event Upsets S.Karp et.al (1993). The proposed technique is illustrated using an example that converts a reversible decoder circuit to an online testable reversible decoder circuit.

Keywords: *Reversible Logic, Online Testing, Power Dissipation, Digital Circuits.*

1. INTRODUCTION

Reversible Logic has gained importance in the recent past. The rapid decrease in the size of the chips has lead to the exponential increase in the transistor count per unit area. As a result, the energy dissipation is becoming a major barrier in the evolving nano-computing era. Reversible logic ensures low energy dissipation R.W. Keyes et.al (1970), C.H. Bennet (1988). An operation is said to be physically reversible if there is no energy to heat conversion and no change in entropy. In reversible logic, the state of the computational device just prior to an operation is uniquely determined by its state just after the operation. In other words, no information about the computational state can ever be lost and hence the reversible logic can be viewed as a deterministic state machine. R.Landauer (1961) has shown that for every bit of information that is erased during an irreversible logic computation kTln2 joules of heat energy is generated, where k is the Boltzmann constant and T is the temperature in Kelvin at which the system is operating. C.H.Bennett (1973) showed that the kTln2 amount of energy dissipation would not occur if a computation is carried out in a reversible way.

Computations performed by the current computers are commonly irreversible, even though the physical devices that execute them are fundamentally reversible. At the basic level, however, matter is governed by classical mechanics and quantum mechanics, which are reversible. With computational device technology rapidly approaching the elementary particle level, it has been argued many times that this effect gains in significance to the extent that efficient operation of future computers requires them to be reversible

[1] Reconfigurable and Intelligent Systems Engineering Group, Dept. of Computer Science & Engg. IIT Madras, Chennai.

C.H.Bennett (1988), R.Landauer (1961). Hence, reversible logic is gaining grounds.

A reversible gate is a logical cell that has the same number of inputs and outputs. Also, the input and output vectors have a one-to-one mapping. Direct fan-outs from the reversible gate are not permitted. Feedbacks from gate outputs to inputs are not allowed. A reversible gate with n-inputs and n-outputs is called a n x n reversible gate. Several reversible gates have been designed till date that is discussed below. The Feynman gate R.Feynman (1985) shown in Figure 1 is one of the popular example of a 2 x 2 reversible gate. In this gate the first input is passed to the output without any change and the second output is the XOR of the first and second inputs. Toffoli Gate T.Toffoli (1980) is a n x n universal reversible gate. The first n - 1 input are directly passed to the corresponding outputs and the n^{th} output is the logical XOR of the n^{th} input with the logical AND of all first n – 1 inputs. The 3 x 3 Toffoli gate is shown in Figure 2. E.Fredkin (1982) proposed a 3 x 3 universal reversible gate. The Fredkin gate functions as a multiplexer with the select signal as the first input. Functionality of the Fredkin gate is shown in Figure 3. Many other gates have been proposed that include Kerntopf gate by P.Kerntopf (2002) and Margolus gate by N.Margolus (1988). An elaborate list of reversible gates studied in the literature is presented in P.Kerntopf (2002).

Many implementations techniques for the reversible circuits have been proposed till now. A technique called Charge Recovery Logic (CRL) was proposed by S.G.Younis et.al (1993). The technique relied on constructing and explicit reversible pipelined logic gates, where the information necessary to recover energy used to compute a value is provided by computing its functional inverse. Later Split-Level charge recovery logic (SCRL) was proposed S.G.Younis et.al (1994) which uses split-level voltages. Another technique called Reversible energy recovery logic (RERL) circuits for ultra-low-energy consumption was proposed J.Lim et.al (1998) and (1999). nMOS reversible energy recovery logic (nRERL) for ultra-low-energy having better performance has also been proposed J.Lim et.al (2000). The optoelectronic and nano-electronic implementations of reversible gates were also found in the literature P.Picton (1991), S.Bandyopadhyay (1998). However, there is relatively little progress in the synthesis of reversible gates

One of the major issues in designing a reversible circuit is garbage minimization. Garbage is defined as the number of outputs added to make an n-input, k-output Boolean function ((n, k) function) reversible D.Maslov et.al (2003).

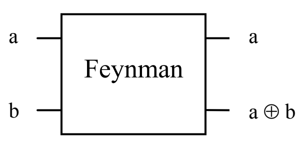


Fig.1. Feynman Gate

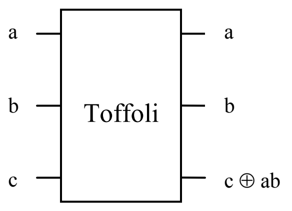


Fig.2. 3x3 Toffoli Gate

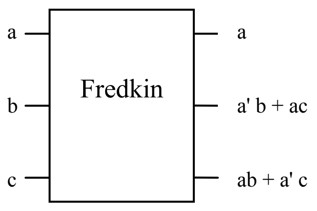


Fig.3. Fredkin Gate

2. TESTING REVERSIBLE CIRCUITS

Very little previous research has been done on testable reversible circuits. Conditions for a complete test set construction were discussed and the problem of finding a minimum test set was formulated as an integer linear program with binary variables in K.N.Patel et.al (2003). A new fault model called *missing gate fault model*, was proposed to represent physical failure modes of quantum technologies J.P.Hayes et.al (2004).

An online testing technique for reversible logic circuits was proposed in D.P.Vasudevan et.al (2004). The technique involved two testable reversible logic gates R1 and R2, which can be used to implement various reversible digital circuits. Another online testing technique for reversible logic circuits was given in D.P.Vasudevan et.al (2004). This technique proposed three reversible logic gates (R1, R2, and R3). The gates R1 and R2 proposed in D.P.Vasudevan et.al (2004) [20] and [21] are the same. A combination of these R1 and R2 gates were used to construct testable logic blocks. The R3 gate was used to construct a two pair rail checker, for detecting errors in any pair of blocks.

3. CONTRIBUTION OF THIS PAPER

The most recently reported work D.P.Vasudevan et.al (2004) focuses on the implementations using particular reversible gates. The technique proposed in this paper can be employed to convert any reversible circuit with arbitrary number of gates to an online testable reversible one and is independent of the type of reversible gate used. The constructed circuit can detect any single bit errors that include single bit stuck-at-fault and single event upset S.Karp et.al (1993). An important advantage of the technique is that the logic design of a reversible circuit remains the same and the reversible circuit need not be redesigned for adding the testability feature to it. Another advantage is that the technique ensures that the garbage generated during the process of conversion to testable reversible circuit is minimized. The proposed technique is illustrated using an example that converts a decoder circuit that is designed by reversible gates to an online testable reversible decoder circuit.

4. DEDUDED REVERSIBLE GATE

In this section we introduce the notion of a Deduced Reversible Gate (DRG), of any given reversible gate, which shall be used further to design testable blocks.
Let the n x n reversible gate R be as shown in Figure 4 with the input vector $I = [I_1, I_2, I_3 \ldots, I_n]$ and the output vector $O = [O_1, O_2, O_3 \ldots O_n]$. As the gate is reversible, we have a one-to-one mapping between vector I and vector O.
A Deduced Reversible Gate of R, DRG(R), is constructed by adding an extra input bit P_i and the corresponding output bit P_o to the gate R, as shown in Figure 5 in a way that, $P_o = F \oplus P_i$ where $F = O_1 \oplus O_2 \oplus O_3 \ldots \oplus O_n$, realized in terms of $I_1, I_2, \ldots I_n$.

Lemma 1: DRG(R) is reversible.
Proof: *Let* $[I_1, I_2, I_3 \ldots I_n]$ *map to* $[O_1, O_2, O_3 \ldots O_n]$ *in* R*, then, we can easily say that* $[I_1, I_2, I_3 \ldots I_n, 0]$ *map to* $[O_1, O_2, O_3 \ldots O_n, F]$*, since* $P_o = F$ *when* $P_i = 0$*, and* $[I_1, I_2, I_3 \ldots I_n, 1]$ *map to* $[O_1, O_2, O_3 \ldots O_n, \sim F]$*, since* $P_o = \sim F$ *when* $P_i = 1$*. Where* $F = O_1 \oplus O_2 \oplus O_3 \ldots \oplus O_n$*. From the above fact, and given that* R *is reversible, we easily see that* DRG(R) *is reversible.* □
The above lemma implies a procedure to construct DRG(R) given any reversible gate R.

5. TESTABLE REVERSIBLE CELL

This section proposes a method to construct a Testable Reversible Cell, TRC(R), for a given reversible gate R. Consider a n x n gate X such that all the inputs to the gate X are mapped to the outputs without any change. It is obvious that the gate X is reversible. Let R be an n x n reversible gate. Let DR_a = DRG(R) and DR_b = DRG(X). DR_a and DR_b are (n+1) x (n+1) gates. By Lemma 1, DR_a and DR_b are reversible gates.

Cascade the gates DR_a and DR_b as shown in Figure 6 by connecting the first n outputs of DR_a to the first n inputs of DR_b in order. The resultant gate of Figure 6 can be viewed as a (n + 2) x (n + 2) gate as shown in Figure 7 which we denote as the Testable Reversible Cell of R, TRC(R). The input vector of TRC(R) is defined as [I, P_{ia}, P_{ib}], where I is the input vector of gate R and P_{ia}, P_{ib} are the added one bit inputs to the gates DR_a and DR_b respectively. Similarly, the output vector is defined as [O, P_{oa}, P_{ob}] where O is the output vector of gate R and P_{oa}, P_{ob} are the added output bits to gate DR_a and gate DR_b respectively.

Lemma 2: TRC(R) is reversible.
Proof: *The fact that* DR_a *and* DR_b *are reversible and the construction of TRC(R) imply that TRC(R) is reversible.* □

Lemma 3: TRC(R) has the following fault detection properties:

Case 1: Given $P_{ia} = P_{ib}$, then, the output of TRC(R) is erroneous if the parity bits, P_{oa} and P_{ob} are complementary.

Case 2: If $P_{ia} = \sim P_{ib}$, then, the output of TRC(R) is erroneous if the parity bits, P_{oa} and P_{ob} are same.

Proof: *From construction of* TRC(R), *note that* $P_{oa} = P_{ia} \oplus O_1 \ldots \oplus O_n$ *and* $P_{ob} = P_{ib} \oplus O_1 \ldots \oplus O_n$ *and that* P_{oa} *and* P_{ob} *are realized directly from the inputs of the respective gates.*

Case 1: $P_{ia} = P_{ib}$, *then if* $P_{oa} \neq P_{ob}$ *then there is an error in* DR_a *or* DR_b.
The proof for **Case 2** *follows in similar lines.* □

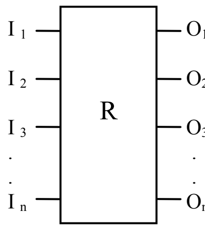


Fig.4. n x n Reversible Gate

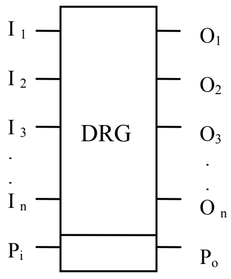


Fig.5. Deduced Reversible gate

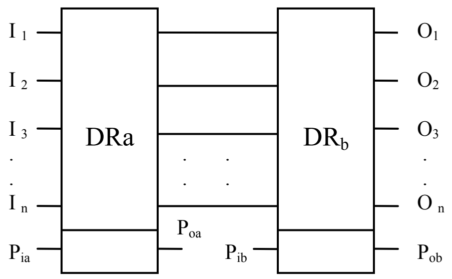


Fig.6. Cascade of Two Deduced Reversible Gates

6. CONSTRUCTION OF ONLINE TESTABLE CIRCUIT

This section describes an algorithmic approach to convert any reversible circuit to an online testable reversible circuit. Given a reversible circuit consisting of reversible gates, the following algorithm converts it into an online testable reversible circuit.

Algorithm 1
Input: Reversible Circuit C
Output: An online testable reversible circuit C^T

1) Construct C' by replacing every reversible gate R in C by TRC(R). The parity input bits of TRC(R) are set such that $P_{ia} = P_{ib}$ in the construction of TRC(R). By Lemma 2, C' is reversible.
2) Let n be the number of reversible gates in C. Construct a (2n+1) x (2n+1) Test Cell (TC) as shown in Figure 8 as follows:
    - First 2n inputs are the output parity bits from each of the n testable reversible cell TRC of C' gate.
    - The last bit of the input, called e, is either set to logic 0 or logic 1.
    - First 2n inputs are transferred to the output without any change.
    - The last output bit (T) of the Test cell (TC) is:
      $T = [((P_{oa1} \oplus P_{ob1}) + (P_{oa2} \oplus P_{ob2}) + \ldots + (P_{oan} \oplus P_{obn})) \oplus e]$
      where P_{oak} and P_{obk} are the output parity bits of the kth TRC of C'. □
3) Cascade C' and TC as stated in step 2 to obtain C^T.

As errors are detected dynamically at-speed during normal working of the circuit and without affecting the functionality of the reversible circuit, the proposed C^T qualifies as an online testable reversible circuit.

Theorem 1: The cell TC constructed in *Algorithm* 1 has the following properties:

1) It is reversible.
2) If there is a single bit error in any TRC in C^T, then, T = 1 provided e = 0.
3) Function T is implemented with minimum possible garbage.

Proof:

1) *We can easily see that* $[P_{oa1}, P_{ob1}, \ldots P_{oan}, P_{obn}, 0]$ *maps to* $[P_{oa1}, P_{ob1}, \ldots P_{oan}, P_{obn}, T]$ *and* $[P_{oa1}, P_{ob1}, \ldots P_{oan}, P_{obn}, 1]$ *maps to* $[P_{oa1}, P_{ob1}, \ldots P_{oan}, P_{obn}, \sim T]$, *where* $T = [((P_{oa1} \oplus P_{ob1}) + (P_{oa2} \oplus P_{ob2}) \ldots + (P_{oan} \oplus P_{obn})) \oplus e]$ *Hence,* TC *is reversible.*
2) $P_{ia} = P_{ib}$ *in* TRC, *from the step 1 of* Algorithm 1. *If there is a single bit error in any* TRC *then by Lemma 3,* P_{oa} *is complementary to* P_{ob}. *Therefore,* $P_{oa} \oplus P_{ob} = 1$. *Hence* T = 1.
3) *For an n-input k-output function* f, *the minimum number of garbage bits required to make it reversible is* ceil(log M)*, where* M *is the maximum number of times an output pattern is repeated in the truth table of* f D.Maslov (2004). *For the function* T, $M = 2^{2n} - 2^n$, *where* n *is the number of* TRCs *in* C^T. *Therefore,* $\log M = \log (2^{2n} - 2^n) > \log (2^{2n}/2) = (2n - 1)$. *Therefore,* ceil (log M) = 2n. *Hence, the minimum garbage that must be produced in* TC *is* 2n. □

The garbage bits that are generated for any circuit that is implemented using the cascaded block of R1 and R2 in D.P.Vasudevan et.al (2004), will be certainly greater than or equal to the circuit implemented using TRC(R). For example a two input AND gate implemented using R1 gate generates extra 2 garbage bits compared to TRC (Toffoli gate). The two pair rail checker used in D.P.Vasudevan et.al (2004) to detect errors is constructed from 6 R3 gates and produces garbage of 8 bits. For a reversible circuit with 2^n testable reversible gates the technique proposed in D.P.Vasudevan et.al (2004) generates garbage of 8n bits where as the Test Cell, TC, generates garbage of 2n bits

This paper proposes a hierarchical construction for online testable reversible circuits. Consider a reversible circuit C that is the integration of the individual reversible modules C_i $\forall_i = 1, 2, . . k$. C_i is made online testable by applying *Algorithm*1. A Multi Modular Testable Cell MMTC is used to detect errors in the integrated circuit C.
MMTC is defined as follows:
Inputs: $T_1, T_2, . . . T_n$ and e, where e can be either 0 or 1.
Outputs: $T_1, T_2, . . . T_n$ and $MMT = (T_1 + T_2 + . . + T_n) \oplus e$.

Theorem 2: The cell MMTC has the following properties:
1) It is reversible.
2) MMTC detects any single bit error in $C_1, C_2, . . . C_k$, where C_i is a online testable reversible module $\forall_i$.
3) Function MMT is implemented with minimum possible garbage.

Proof:
1) *We can easily see that* $[T_1, T_2, . . T_k, 0]$ *maps to* $[T_1, T_2, . . . T_k, MMT]$ *and* $[T_1, T_2, . . . T_k, 1]$ *maps to* $[T_1, T_2, . . T_k, \sim MMT]$ *and* $MMT = (T_1 + T_2 + T_3 . . . + T_k) \oplus e$ *Hence,* MMTC *is reversible.*
2) T_i *bit is the test bit from the* TC *of module* C_i. *The multi modular test bit* MMT *is the logical* OR *of* T_i *bits for* i = 1, 2, . . k. *Hence, if there is any error in any of the modules* C_i, *then* MMT *will be logical* 1, *provided* e = 0. *Hence, the error is detected.*
3) *The minimum number of garbage bits required to make function* MMT *reversible is* ceil(log M), *where* M *is the maximum number of times an output pattern is repeated in the truth table of* MMT, D.Maslov et.al (2004). *For the function* MMT, $M = 2^{2k} - 1$, *where* k *is the number of online testable reversible modules. Therefore,* $\text{ceil}(\log M) = \log (2^{2k} - 1) = 2k$ *Hence, the minimum garbage needed for making function* MMT *reversible is* 2k. □

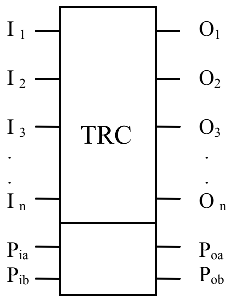


Fig.7. Testable Reversible Cell

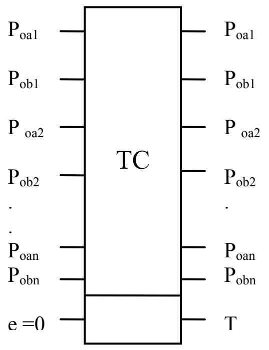
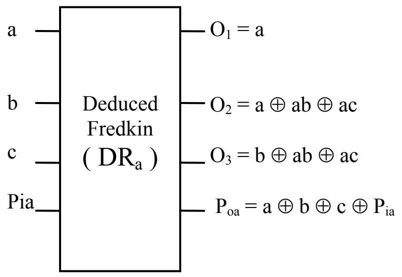


Fig.8. Test Cell



a
b
c
Pia
Deduced Fredkin (DR_a)
$O_1 = a$
$O_2 = a \oplus ab \oplus ac$
$O_3 = b \oplus ab \oplus ac$
$P_{oa} = a \oplus b \oplus c \oplus P_{ia}$

Fig.10. Deduced Fredkin Gate DR_a

The modules are made testable at different hierarchy levels. The main advantage of this hierarchical structure is that it can be extended to the construction of online testable reversible multi-processing systems.

The single stuck faults and single event upsets that occurs anywhere except the primary inputs to the reversible circuit are detected using *Algorithm* 1. Let us consider a case where the single stuck-at-fault or a single event upset at the input of a reversible gate can flip multiple output bits. The proposed technique checks for any single-fault in the reversible gate which means that if the inputs are correct the technique detects the errors at any output bit. The input of some gate is the output of some other gate. Hence we can detect all those errors which are internal to the circuit. Only the primary inputs to the circuits are unchecked.

Therefore, we conclude that the proposed technique can detect all the single event upsets or single stuck-at-faults in the reversible circuit with the assumption that the primary inputs to the complete circuit are error-free.

Note that the deduced reversible DR_b gate is added to the DR_a gate for online testing purpose. The DR_b gate is a parity generator circuit and the complexity is comparable to the Toffoli gate as shown in Figure 2. Therefore, in most cases the logic complexity of the DR_b gate would be significantly less when compared to that of the DR_a gate. Hence, based on the above observations the addition of the DR_b gate does not amount to 100% redundancy in terms of logic complexity.

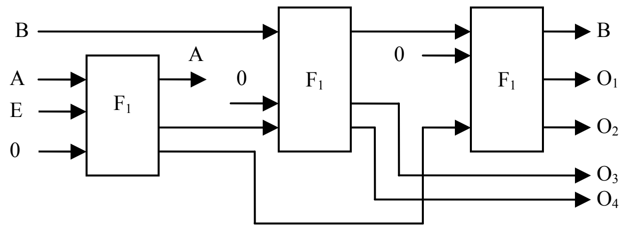


Fig.9. 2-to-4 Reversible Decoder (C)

7. ILLUSTRATION OF THE PROPOSED TECHNIQUE

To illustrate the proposed technique, a reversible decoder circuit is converted to an online testable reversible decoder. A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines. Let us consider a 2-to-4 decoder with the enable bit as shown in Figure 9. The truth table of the decoder is shown in Table I. The construction of the reversible decoder circuit uses three Fredkin gates (F_1, F_2, and F_3).

E	A	B	O_1	O_2	O_3	O_4
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Table I. Truth table for Decoder Circuit

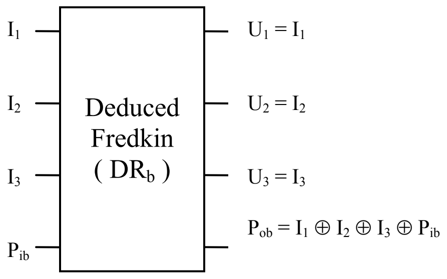

I_1
I_2
I_3
P_{ib}
Deduced Fredkin (DR_b)
$U_1 = I_1$
$U_2 = I_2$
$U_3 = I_3$
$P_{ob} = I_1 \oplus I_2 \oplus I_3 \oplus P_{ib}$

Fig.11. Deduced Gate DR_b

The Fredkin gate is shown in Figure 3. A and B are the one-bit inputs to the decoder, E is the one-bit enable and O_1, O_2, O_3 and O_4 are the output bits of the decoder. *Algorithm* 1 is applied to convert the decoder circuit into an online testable one. We illustrate this technique in a detailed step by step manner.

Input: Reversible Decoder Circuit C

Step 1: Replace Fredkin gate F_k with its Testable Reversible TRC (F_k) for k = 1, 2, 3. Let the input vector of F_k be [a, b, c] and the output vector be [O_1, O_2, O_3]. The deduced Fredkin gate, DR_a can be obtained as shown in Figure 10, with the following as the inputs and outputs:

Inputs: a, b, c and P_{ia}

Outputs: $O_1 = a$; $O_2 = a \oplus ab \oplus ac$

$O_3 = b \oplus ab \oplus ac$; $P_{oa} = O_1 \oplus O_2 \oplus O_3 \oplus P_{ia}$

$P_{oa} = (a) \oplus (c \oplus ab \oplus ac) \oplus (b \oplus ab \oplus ac) \oplus (P_{ia}) = a \oplus b \oplus c \oplus (P_{ia})$

The truth table of the deduced Fredkin gate DR_a is shown in Table II.

To construct DR_b, take X to be a 3 x 3 gate that has inputs as [I_1, I_2, I_3] and outputs as [U_1, U_2, U_3], where U_i and I_i are related as $U_i = I_i$ for i = 1, 2, 3. The deduced gate DR_b is as shown in Figure 11, with the following as the inputs and outputs: **Inputs**: I_1, I_2, I_3 and P_{ib}

Outputs: $U_i = I_i$ where i = 1, 2, 3.

$P_{ob} = P_{ib} \oplus U_1 \oplus U_2 \oplus U_3 = P_{ib} \oplus I_1 \oplus I_2 \oplus I_3$

Truth table of DR_b is as shown in Table III. We cascade the above two deduced gates, DR_a and DR_b to get a Testable Reversible Fredkin Cell (TRC). Note that by *Algorithm* 1, $P_{ia} = P_{ib}$, let us set them to logic 0. This completes the construction of TRC(R) for the Fredkin gate. Now we replace each Fredkin gate F_k for k = 1, 2 and 3 in the input decoder circuit with TRC(R) constructed above. Now the construction consists of three testable Fredkin cells, TRC_1, TRC_2 and TRC_3 and it has six output parity bits P_{oak} and P_{obk} for k = 1, 2 and 3.

a	b	c	P_{ia}	O_1	O_2	O_3	O_4
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	1	0
0	1	1	1	0	1	1	1
1	0	0	0	1	0	0	1
1	0	0	1	1	0	0	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	0

Table II: Truth table for DRG of a Fredkin gate.

I_1	I_2	I_3	P_{ib}	U_1	U_2	U_3	P_{ob}
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	1	0
0	1	1	1	0	1	1	1
1	0	0	0	1	0	0	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	1	0
1	0	1	1	1	0	1	1
1	1	0	0	1	1	0	0
1	1	0	1	1	1	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	0

Table III: Truth table of DRG (X).

Step 2: Add a Test Cell (TC) with 2n + 1 input line. As n = 3 for the given decoder circuit, TC has 2n+1 = 7 input lines. Connect its first six input lines to the parity bits P_{oak} and P_{obk} of the Fredkin gates F_k for k = 1, 2 and 3 as shown in Figure 12. The last input bit e is set to 0. First six input lines are passed to the output lines without any change. Output bit T is the error detecting bit. The value of T will determine if there is an error in the circuit.
Output: Circuit thus obtained is shown in Figure 12 and is the required online testable reversible decoder circuit C^T. Let us consider the case when the input vectors [A, B, E] = [1, 0, 1]. From the Truth Table I, if the circuit is error free, output vector O should be [0010]. In this case, the parity vector [P_{oa1}, P_{ob1}, P_{oa2}, P_{ob2}, P_{oa3}, and P_{ob3}] is equal to [0, 0, 1, 1, 0 and 0] and T = 0 which shows that the circuit is error free.
Suppose there is some error in the circuit, say in F_1. For the given input vector [A, B, E] = [1, 0, 1], output of F_1 is [1, 1, 1] instead of [1, 0, 1]. In this case, parity vector [P_{ia1}, P_{ib1}, P_{ia2}, P_{ib2}, P_{ia3}, P_{ib3}] will be equal to [0, 1, 1, 1, 0, 0] and hence T = 1. So, the conclusion is that the circuit is erroneous.
It is straight forward to infer that if there is any single bit error in the circuit, it will be indicated by the error bit T. If the reversible circuit is made of more than one modules, then using the hierarchical structure these can be integrated by using MMTC making it online testable.

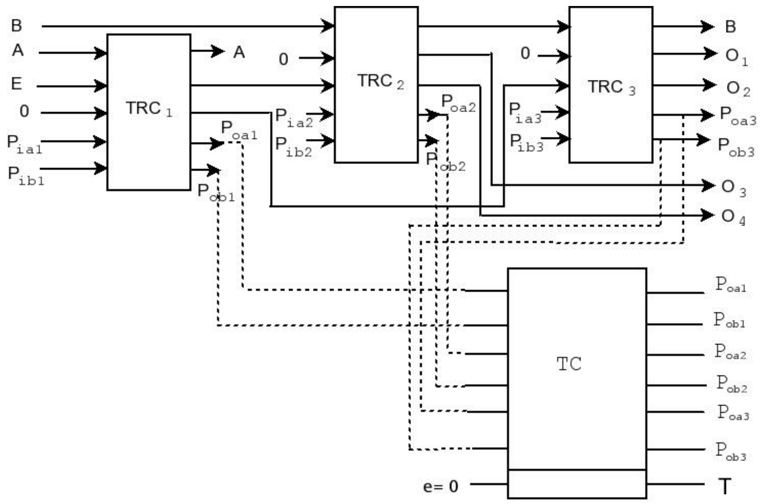


Fig.12. Testable Reversible Decoder Circuit (C^T)

8. CONCLUSIONS

In this paper, we proposed a methodology that converts any reversible gate into a testable reversible gate. Using the same, any reversible circuit made of reversible gates can be converted to an online testable one with minimum garbage. The resultant testable circuit can detect online any single bit errors that include Single Stuck Faults and Single Event Upsets. An important advantage of the technique is that the design of a reversible circuit need not be changed for the purpose of adding testability feature to it. This paper proposes the construction of multi-modular online testable reversible circuits hierarchically.

The proposed technique is illustrated using an example that converts a Fredkin based decoder circuit to an online testable reversible decoder circuit.

REFERENCES

[1]. R. W. Keyes and R. Landauer, “Minimal energy dissipation in logic,” *IBM J. Research and Development*, pp. 152–157, March 1970.

[2]. C. H. Bennett, “Notes on the history of reversible computation,” *IBM J. Research and Development*, vol. 32, pp. 16–23, January 1988.

[3]. R. Landauer, “Irreversibility and heat generation in the computing process,” *IBM J. Research and Development*, vol. 3, pp. 183–191, July 1961.

[4]. C. H. Bennett, “Logical reversibility of computation,” *IBM J. Research and Development*, pp. 525–532, November 1973.

[5]. R. Feynman, “Quantum mechanical computers,” *Optics News*, vol. 11, pp. 11–20, 1985.

[6]. T. Toffoli, “Reversible computing,” *Automata, Languages and Programming*, pp. 632–644, 1980.

[7]. E. Fredkin and T. Toffoli, “Conservative logic,” *Int’l Journal of Theoretical Physics*, vol. 21, pp. 219–253, 1982.

[8]. P. Kerntopf, “Synthesis of multipurpose reversible logic gates,” *Euromicro Symposium on Digital System Design (DSD’02)*, pp. 259–267, 2002.

[9]. N. Margolus, “Physics and computation,” *Ph. D. Thesis, Massachusetts Institute of Technology, Cambridge, MA*, 1988.

[10]. S. G. Younis and T. F. Knight, “Practical implementation of charge recovering asymptotically zero power cmos,” *Proceeding of the 1993 symposium on Research on integrated systems, MIT press*, pp. 234–250, 1993.

[11]. S. G. Younis and T. F. Knight, “Asymptotically zero energy split-level charge recovery logic,” *Proc. Workshop Low Power Design, Napa Valley California*, pp. 177– 182, 1994.

[12]. J. Lim, K. Kwon, and S.-I. Chae, “Reversible energy recovery logic circuit without non-adiabatic energy loss,” *Electronic Letters*, vol. 34, No.4, pp. 344–346, February 1998.

[13]. J. Lim, D.-G. Kim, and S.-I. Chae, “Reversible energy recovery logic circuits and its 8-phase clocked power generator for ultra-low-energy applications,” *IEICE Trans. Electron*, vol. E82-C, No.4, pp. 646–653, April 1999.

[14]. J. Lim, D.-G. Kim, and S.-I. Chae, “nmos reversible energy recovery logic for ultra-low-energy applications,” *IEEE Journal of Solid-State Circuits*, vol. 35, No.6, pp. 865–875, June 2000.

[15]. P.Picton, “Optoelectronic multi-valued conservative logic,” *Int. Journal of Optical Computing*, vol. 2, pp. 19–29, 1991.

[16]. S. Bandyopadhyay, “Nanoelectric implementations of recversible and quantum logic,” *Supperlattices and Microstructures*, vol. 23, pp. 445– 464, 1998.

[17]. D. Maslov and G. W. Dueck, “Garbage in reversible design of multiple output functions,” *In 6th International Symposium on Representations and Methodology of Future Computing Technologies*, pp. 162–170, March 2003.

[18]. K. N. Patel, J. P. Hayes, and I. L. Markov, “Fault testing for reversible circuits,” *In the Proceedings of the 21st IEEE VLSI Test Symposium*, pp. 410–416, April 2003.

[19]. J. P. Hayes, I. Polian, and B. Becker, “Testing for missing-gate faults in reversible circuits,” *In the Proceedings of the 13th Asian Test Symposium*, pp. 100–105, November 2004.

[20]. D. P. Vasudevan, P. K. Lala, and J. P. Parkerson, “A novel approach for on-line testable reversible logic circuit design,” *In the Proceedings of the 13th Asian Test Symposium*, pp. 325–330, October 2004.

[21]. D. P. Vasudevan, P. K. Lala, and J. P. Parkerson, “Online testable reversible logic circuit design using nand blocks,” *In the Proceedings of the 19th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 324–331, October 2004.

[22]. D. Maslov and G. W. Dueck, “Reversible cascades with minimal garbage,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, pp. 1497–1509, November 2004.

[23]. S. Karp, B.K. Gilbert, "Digital system design in the presence of single event upsets," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 29, no. 2, pp. 310-316, April 1993.