Quick Addition of Decimals using Reversible Conservative Logic

Rekha K. James, Shahana T. K, K. Poulose Jacob
Cochin University of Science and Technology
Kochi, Kerala, India
E-mail:{rekhajames, shahanatk, kpj}@cusat.ac.in

Sreela Sasi
Gannon University
Erie, PA, USA
sasi001@gannon.edu

Abstract

In recent years, reversible logic has emerged as one of the most important approaches for power optimization with its application in low power CMOS, nanotechnology and quantum computing. This research proposes quick addition of decimals (QAD) suitable for multi-digit BCD addition, using reversible conservative logic. The design makes use of reversible fault tolerant Fredkin gates only. The implementation strategy is to reduce the number of levels of delay there by increasing the speed, which is the most important factor for high speed circuits.

Index Terms- decimal arithmetic, delay reduction, fault detection, reversible logic

1. Introduction

Energy loss during computation is an important consideration in low power digital design. Landauer's principle states that a heat equivalent to *kT*ln2* is generated for every bit of information lost, where *k* is the Boltzmann's constant and *T* is the temperature [1]. At room temperature though the amount of heat generated may be small it cannot be neglected for low power designs. The amount of energy dissipated in a system bears a direct relationship to the number of bits erased during computation. Bennett showed that energy dissipation would not occur if the computations were carried out using reversible circuits [2] since these circuits do not lose information. A reversible logic gate is an *n*-input, *n*-output (denoted as *n*n*) device that maps each possible input pattern to a unique output pattern. There is a significant difference in the synthesis of logic circuits using conventional gates and reversible gates [3]. While constructing reversible circuits with the help of reversible gates fan-out of each output must be 1 without feedback loops. As the number of inputs and outputs are made equal there may be a number of unutilized outputs in certain reversible implementations. The unutilized outputs from a reversible gate/circuit are called "garbage". This is the number of outputs added to make an n-input k-output function reversible. For example, a single output function of 'n' variables will require at least n-1 garbage outputs. Classical logic gates such as AND, OR, and XOR are not reversible. Hence, these gates dissipate heat and may reduce the life of the circuit. So, reversible logic is in demand in high-speed power aware circuits. In recent years, reversible logic has emerged as one of the most important approaches for power optimization with its application in low power CMOS, nanotechnology and quantum computing.

A reversible conventional Binary Coded Decimal (BCD) adder was proposed in [4] using NG (New Gate) and NTG (New Toffoli Gate) reversible gates. Even though the implementation was improved in [5] using TSG reversible gates, this approach is not taking care of the fan-out restriction of reversible circuits, and hence it is only a near-reversible implementation. These implementations were for the conventional BCD adder which is slow. Currently fast decimal arithmetic is rapidly gaining popularity in the computing community due to the growing importance of commercial, financial, and internet-based applications, which process decimal data.

Fault detection can be done by using parity-preserving reversible logic gates. The feasibility of the parity-preserving approach in design of reversible logic circuits was demonstrated by B. Parhami [6] with examples of adder circuits. Parity checking is one of the oldest, as well as one of the most widely used, methods for error detection in digital systems. The parity preservation proves useful for ensuring the robustness of reversible logic circuits.

In this research, an adder for quick addition of decimals (QAD) suitable for multi-digit BCD addition is implemented using parity preserving reversible Fredkin gates. Fredkin gates are conservative reversible gates. A gate is conservative if the Hamming weight (number of logical ones) of its input equals the Hamming weight of its output. If a gate is conservative and reversible then it is parity preserving.

The organization of this paper is as follows: Initially, the design of the proposed BCD adder for 'Quick Addition of Decimals' (QAD) is done. The necessary background on parity preserving reversible logic gates is given. The proposed QAD adder is then designed using parity preserving reversible gates only so that fault detection can be done. It is also demonstrated that the proposed design is highly optimized in terms of number of levels of reversible gate delays. A graphical delay analysis of different implementations is also presented.

2. Quick Decimal Adder

The proposed BCD adder shown in Figure 1 consists of a 4-bit binary adder, a 6-correction circuit, and a modified special adder along with a circuit (3-input AND, 2-input OR) to generate decimal carry out (d_{cout}).

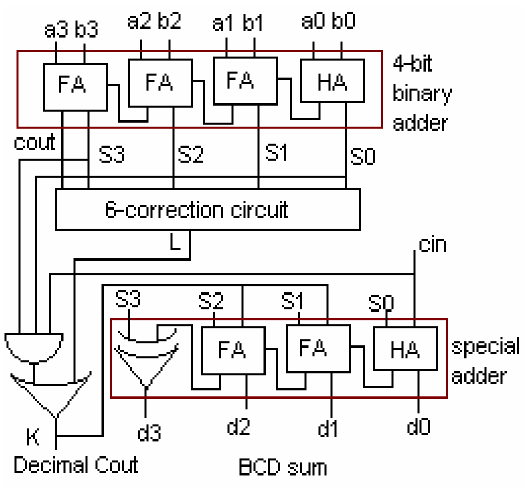

Figure 1: BCD adder for Quick Addition of Decimals (QAD)

The 4-bit binary adder, consisting of 3 full adders and 1 half-adder, adds the BCD inputs and generates a binary sum, S (S_{3-0}). This output is checked for a value greater than '9' or for a carry out, by the 6-correction circuit and generates a '6-correction' bit, 'L' using (1).

$$L= C_{out} + S_3 (S_1+S_2) \quad (1)$$

The circuit requires the carry from the previous stage (C_{in}) only after this level. On receiving C_{in} the circuit checks whether the sum, S is '9' and C_{in} is '1'. It then generates 'K' bit using (2).

$$K = S_3S_0C_{in} + L \quad (2)$$

The inputs to the second adder are S (S_{3-0}) and 4-bit number, N (N_{3-0}) whose value is depending on 'K' and 'C_{in}' as given below.

If K=1 then N=6 (0110_2) if Cin='0'
N=7 (0111_2) if Cin='1'
else N=0 (0000_2) if Cin='0'
N=1 (0001_2) if Cin='1'

So N_3 is always zero. N_2 and N_1 is K-bit and N_0 is 'C_{in}'. To reduce the hardware and to increase the speed of the circuit, the final adder stage (4-bit special adder) is a modified version of the 4-bit binary adder consisting of a half adder, 2 full adders and an XOR gate. The implementation of the special adder is shown in Fig. 1.

3. 4-Digit Quick Decimal Adder

This adder accepts two 4 digit decimal numbers as inputs in BCD (16 bits) and generates the BCD_{sum}. Figure 2 shows the 4-digit QAD implementation. The shaded parts indicate the critical path.

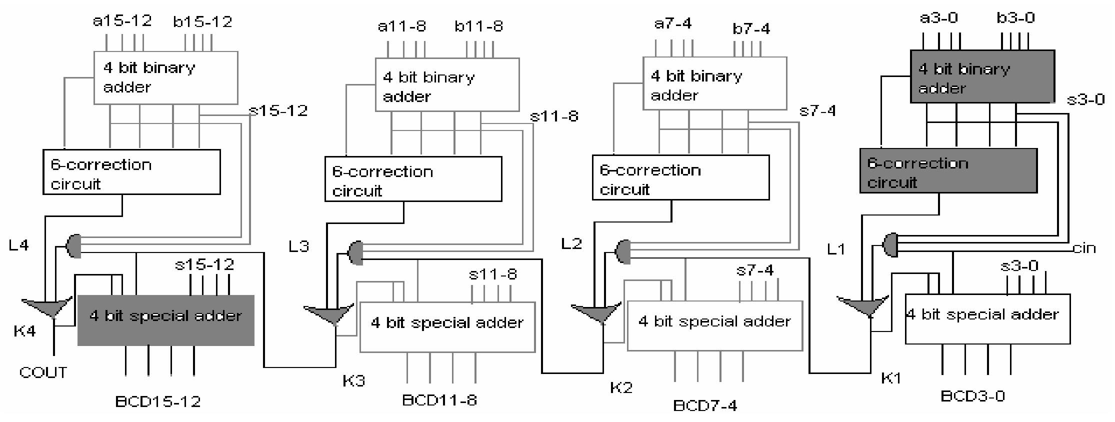

Figure 2: Proposed 4-Digit Quick Decimal Adder

The first stage addition is carried out in parallel for all digits and a '6-correction' bit (L_i) is generated simultaneously by all the stages. So, the delay up to this stage is given as

$$T_{delay} = T_{adder} + T_{6\text{-correction}} \quad (3)$$

where T_{adder} is the delay of the 4-bit binary adder and $T_{6\text{-correction}}$ is the delay of 6-correction circuit

On receiving the C_{in}, the Decimal C_{out} bit (K_i) is generated after the delay of a 3-input AND gate and a 2-input OR gate at each stage as given in (4).

$$T_{dcout} = T_{and} + T_{or} \quad (4)$$

So, for a 4-digit adder the delay in generating Decimal C_{out} (d_{cout}) after receiving C_{in} is $4T_{dcout}$. In general, for an 'm' digit BCD adder the delay for generating the Decimal C_{out} (d_{cout}) after receiving 'C_{in}' is mT_{dcout}. The total delay of the m-digit adder is given in (5).

$$T_{mdigit} = T_{adder} + T_{6\text{-correction}} + mT_{dcout} + T_{sp\text{-adder}} \quad (5)$$

where $T_{sp\text{-adder}}$ is the delay of the special adder.

The total delay for an m-digit 'Quick Decimal Adder' is 'm' times the delay of the additional logic required (which is a 3-input AND and a 2-input OR) along with the complete delay of single stage BCD adder which is

$$T_{1stage} = (T_{adder} + T_{6\text{-correction}} + T_{sp\text{-adder}}). \quad (6)$$

Table 1 shows a comparison of conventional BCD adder and the quick decimal adder in terms of area and critical path delay done with the logic synthesis tool Leonardo Spectrum from Mentor Graphics Corporation using ASIC Library. The critical path delay and area are normalized with respect to a full adder critical path delay of 1.98 ns and area of 38μm^2.

TABLE 1: Simulation Results of BCD Adders

BCD Adder	Critical path Delay (ns)	Area (μm^2)
Conventional BCD Adder (1 digit)	5.83	179
Conventional BCD Adder (4 digit)	14.59	690
QAD (1 digit)	5.95	192
QAD (4 digit)	12.06	742

Figure 3 shows the graphical analysis of area-delay product normalized to that of a full adder. It can be seen from the Figure 4 that there is a significant reduction in area-delay product for the proposed BCD adder compared to conventional implementation as number of bits increases above 8.

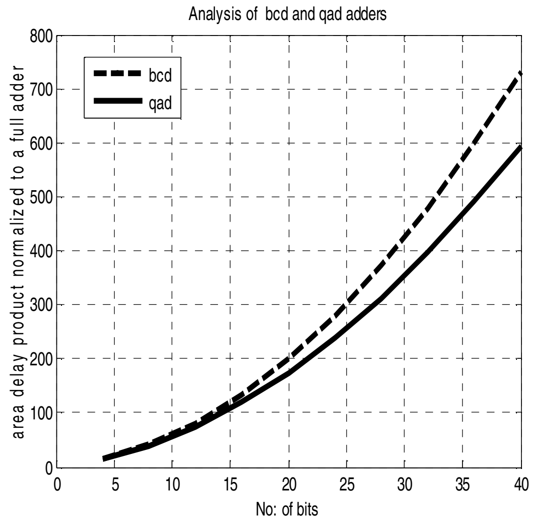

Figure 3: Analysis of area-delay product of conventional BCD and QAD adders

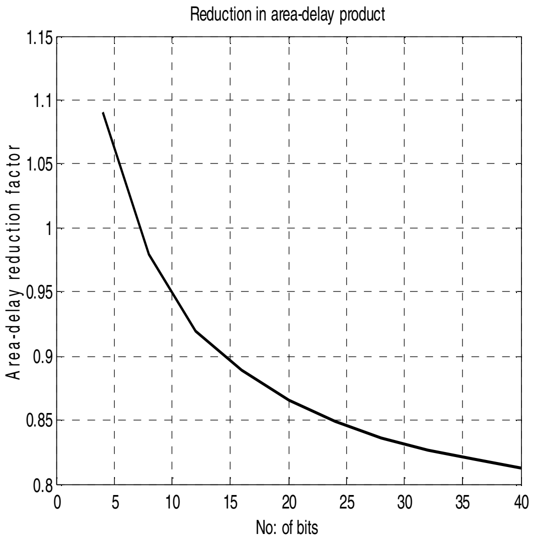

Figure 4: Reduction in area-delay product of QAD over BCD adders

4. Reversible Gates

This section describes parity preserving reversible logic gates. Figure 5 shows a Feynman Double Gate (F2G) [6, 7] which can be used as an XOR gate or as a copying gate. Figure 6 shows a Fredkin Gate (FRG) [8]. These two parity preserving reversible gates satisfy the condition $A \oplus B \oplus C = P \oplus Q \oplus R$. In general, a parity preserving reversible gate is a gate in which the following condition is valid.

$$\oplus \sum X_i = \oplus \sum Y_i \quad (7)$$

where X indicates an input, Y an output and 'i' indicates the number of inputs or outputs of the reversible gate.

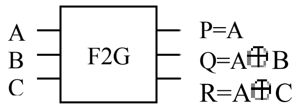

Figure 5: Feynman Double Gate (F2G)

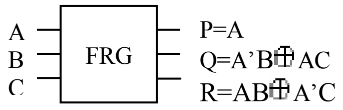

Figure 6: Fredkin Gate (FRG)

5. Parity Preserving Reversible Quick Decimal Adder

Recently, the reversible implementations of the conventional BCD adders were proposed by Hafiz [4] and Thapliyal [5]. But these implementations make use of reversible gates other than parity preserving gates also and hence are not fault tolerant implementations. The proposed reversible implementation of the quick decimal adder is done using Fredkin gates. The basic component of any adder is a full adder. A number of parity preserving reversible full adders are available in literature [6, 9]. Figure 7 and Figure 8 show the implementation of a half adder and a full adder using parity preserving Fredkin gates.

The full adder implementation in Figure 8 requires only 5 Fredkin gates at 3 levels, compared to 3 level 6 gate (5 Fredkin gates and 1 Feynman gate) implementation in [6], and 5 level 5 Fredkin gate implementation in [9] while observing the fanout restrictions.

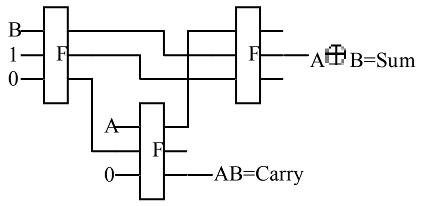

Figure 7: Half adder using Fredkin Gates

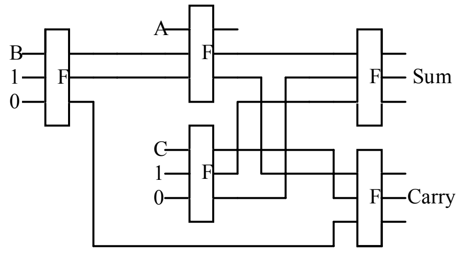

Figure 8: Full adder using Fredkin Gates

The 4-bit binary adder realized using one half adder and 3 full adders will achieve a reduced delay by using these implementations. The least significant half adder requires a path delay of two FRGs to generate C_0 from the addends. Then the carry ripples through the subsequent full adders with a path delay of two FRGs per bit. This is because the first Fredkin gates of all full adders work in parallel with the first Fredkin gate of half adder in an n-bit binary adder. But in the implementation in [6] delay is of 3 levels. So an advantage of 1delay level/bit is achieved in the proposed implementation. The delay to generate C_{out} in the 'n' bit binary adder is

$$d_{c\text{-ripple}} = 2+2(n-1) \quad (8)$$

For a conventional n-bit adder which makes use of full adder in [6] it is

$$d_{c\text{-ripple (conventional)}} = 3n \quad (9)$$

For a BCD adder this delay is the delay with n=4 for each digit. In QAD adder, since all digits are added in parallel the delay remains the same as the delay of a single digit for 'm' digit addition.

The parity preserving reversible implementation of the 6-correction circuit is shown in Figure 9. The implementation requires 3 FRGs to generate the L output where $L=C_{out}+S_3\,(S_1+S_2)$. This circuit takes only 2 more delays after generating the Sum to generate the L bit. The delay to generate L bit from the inputs for QAD adder is as given in (10)

$$d_L = 4+2(n-1) \quad (10)$$

The delay for 'L' bit generation for conventional BCD adders is given in (11).

$$d_{L\text{ (conventional)}} = 2+3n \quad (11)$$

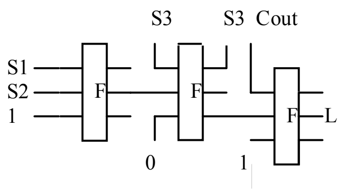

Figure 9: Generation of 'L' bit using Fredkin Gates

Figure 10 shows the reversible implementation for generating Decimal C_{out} (d_{cout}) or K-bit. The design makes use of three FRGs. It is seen that the first gate generate S_3S_0 as soon as the sum output 'S' is produced. When C_{in} is received the next two FRGs generate the Decimal C_{out} or K-bit. So, the additional delay in each stage is due to the two FRGs only. For an 'm' digit BCD addition the delay for the generation of K bit or Decimal C_{out} from the BCD inputs is

$d_{d\text{-}cout}$= 4+2(n-1) +2m (12)

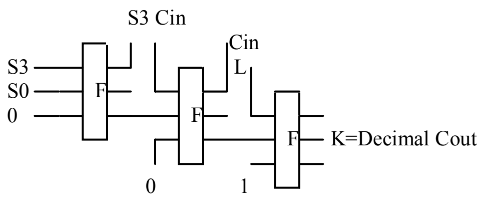

Figure 10: Generation of 'K' bit using Fredkin Gates

Special adder implemented using one half adder, two full adders and one XOR gate requires 15 Fredkin gates (3 for half adder, 5 for each full adder, 2 for XOR gate) to generate the BCD sum $d_{3\text{-}0}$. If the XOR gate is implemented by one F2G gate then the number of gates reduces to 14. For achieving a VLSI implementation, a design using only one type of standard reversible gate as the basic building block is to be adopted. So the use of Fredkin gates only makes this design suitable for a VLSI implementation. The Decimal C_{out} or the 'K' bit is the input to be received last for the special adder. The 'K' input passes through a maximum of 5 Fredkin gates to generate the BCD sum $d_{3\text{-}0}$. So the special adder gives an additional delay of 5 Fredkin gates. The total delay in generating the BCD sum ($d_{3\text{-}0}$) from the inputs in terms of Fredkin gate delay is

$d_{d\text{-}sum}$= 9+2(n-1) +2m (13)

For a conventional BCD adder the final adder is a 4-bit binary adder with delay as given in equation (9) with 'n'=4. So the total delay given by an 'm' digit conventional BCD adder is

$d_{d\text{-}sum\ (conventional)}$ = (2+3n + 3n)m = (2+6n)m (14)

The delay can be further reduced by adopting carry select technique for the 'K' bit generation. 'K' bit is computed in advance for C_{in}=1 and C_{in}=0 before receiving the actual C_{in}. Let K_1 denote the 'K' bit with C_{in} = 1 and K_0 with C_{in} = 0. K_1 and K_0 are generated using (15) and (16).

$K_1=S_3.S_0+L$ (15)

$K_0=L$ (16)

Figure 11 shows the generation of 'K' bit or the Decimal C_{out}. The generation of K_1 and K_0 takes the delay of only one Fredkin gate after receiving 'L' bit as seen in Figure 11. After computing both values (K_1 and K_0) a selection is done by a single FRG. Since an FRG works as a 2:1 multiplexer with 'A' input as control input and 'B' and 'C' inputs as data inputs, the selection of K_0 or K_1 can be done with one FRG. So, the additional delay in each stage to generate 'K' bit after receiving C_{in} is due to one FRG only, where as for the implementation in Figure 10 it is of two FRGs.

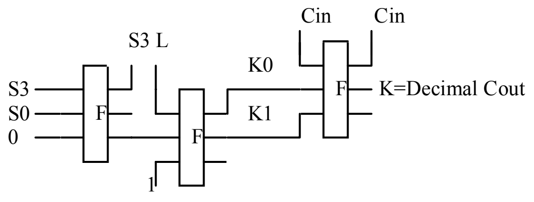

Figure 11: Generation of 'K' bit using K_0 and K_1

Now the delay gets modified as

$d_{d\text{-}cout\text{-}carry\ select}$= 5+2(n-1) +m (17)

But in this implementation Cin is received by the special adder along with the 'K' bit only. On receiving C_{in} the half adder of the special adder generates the carry bit after one Fredkin gate delay. So the special adder gives an additional delay of 6 Fredkin gates.

$d_{d\text{-}sum\text{-}carry\ select}$= 11+2(n-1) +m (18)

It is seen that no additional gates are required for carry select design and gives an implementation with reduced delay also.

Figure 12 shows the delay analysis of conventional BCD adder; quick decimal adder and the QAD carry select reversible implementations normalized to that of a Fredkin gate.

6. Conclusion and Future Work

A reversible fault tolerant logic for quick addition of decimals (QAD) suitable for multi-digit BCD addition is presented. This work forms an initial step in the building of complex reversible systems, which can execute more complicated operations. The reversible circuit proposed here forms the basis of a Decimal ALU for a reversible CPU. The proposed Decimal adder has 4 advantages:

(i) Reduced delay by using the technique of quick addition of decimals (QAD).

(ii) The use of reversible gates makes it a low power implementation.

(iii) The approach provides a path of incorporating fault detection by using parity preserving reversible Fredkin gates.

(iv) The use of only one type of modular building block (Fredkin gates) makes this suitable for a VLSI design.

VLSI implementations using only one type of modular building blocks can decrease system design and manufacturing cost. Implementations using other standard reversible gates such as TSG [10] or Toffoli [3] gates can also be tried. But these are not parity preserving gates and hence will not give a fault tolerant implementation. Characterization of new families of 'n-input' – 'n-output' reversible gates that can be used for regular structures is an area, which can be explored further.

In this research, a known traditional logic implementation for BCD adder was modified to get a delay reduction for multi-digit addition, and then each of the internal elements was replaced with reversible equivalents. Further investigation into determining alternate implementations can be done using logic synthesis methods [11, 12, 13, 14]. Additionally, it was noted that there is lack of simulation tools that support reversible gates, and this is most definitely an area worthy of attention.

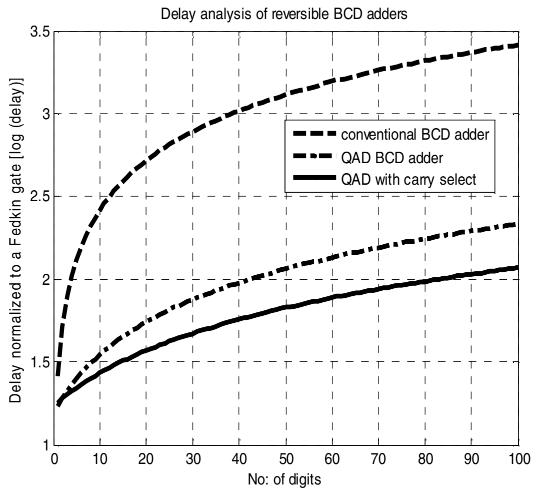

Figure. 12: Delay analysis of reversible BCD adders

7. References

[1] R. Landauer, "Irreversibility and Heat Generation in the Computational Process", IBM Journal of Research Development, 5, pp.183-191, 1961.

[2] Bennett, C., "Logical Reversibility of Computation," IBM Journal of Research and Development,17, pp.525-532, 1973.

[3] T. Toffoli., "Reversible Computing", Tech memo MIT/LCS/TM-151, MIT Lab for Computer Science, 1980.

[4] Md. Hafiz Hasan Babu and A. R. Chowdhury, "Design of a Reversible Binary Coded Decimal Adder by Using Reversible 4-bit Parallel Adder", VLSI Design '05, pp.255-260, Jan 2005.

[5] H. Thapliyal, S. Kotiyal and M.B Srinivas, "Novel BCD Adders and their Reversible Logic Implementation for IEEE 754r Format", 19th International Conference on VLSI Design (VLSI Design 2006), pp. 387-392, Jan 2006.

[6] B. Parhami; "Fault Tolerant Reversible Circuits" Proc. 40th Asilomar Conf. Signals, Systems, and Computers, Pacific Grove, CA, October 2006.

[7] R. Feynman, "Quantum Mechanical Computers", Optical News, 1985, pp.11-20.

[8] E. Fredkin and T. Toffoli, "Conservative logic", Intl. J. Theoretical Physics, Vol 21, 1982, pp. 219-253.

[9] J.W. Bruce, M.A. Thornton, L. Shivakumariah, P.S. Kokate, X.Li, "Efficient Adder Circuits Based on a Conservative Logic Gate", Proceedings of the IEEE Computer Society Annual Symposium on VLSI, April 2002, PA, USA, pp 83-88.

[10] H. Thapliyal and M.B Srinivas, "A Novel Reversible TSG Gate and Its Application for Designing Reversible Carry Look-Ahead and Other Adder Architectures", Tenth Asia-Pacific Computer Systems Architecture Conference, Singapore, Oct 24 -26, 2005.

[11] Dmitri Maslov, "Reversible Logic Synthesis", PhD Dissertation, Computer Science Department, University of New Brunswick, Canada, October 2003.

[12] A. Agrawal and N. K. Jha, "Synthesis of reversible logic," in Proc. Design Automation & Test in Europe Conf., Feb. 2004, pp. 21 384–21 385.

[13] P. Gupta, A. Agrawal, N. K. Jha, "An algorithm for synthesis of reversible logic circuits", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems", Nov. 2006, Volume: 25, Issue 11, pp. 2317-2330.

[14] Guowu Yang; Fei Xie; Xiaoyu Song; Hung, W.N.N.; Perkowski, M.A., "A constructive Algorithm for Reversible Logic synthesis" IEEE Congress on Evolutionary Computation, July 2006, pp. 2416- 2421.