Performance Analysis of Reversible Fast Decimal Adders

Rekha K. James, Shahana T. K., K. Poulose Jacob, and Sreela Sasi

Abstract - **This paper presents a performance analysis of reversible, fault tolerant VLSI implementations of carry select and hybrid decimal adders suitable for multi-digit BCD addition. The designs enable partial parallel processing of all digits that perform high-speed addition in decimal domain. When the number of digits is more than 25 the hybrid decimal adder can operate 5 times faster than conventional decimal adder using classical logic gates. The speed up factor of hybrid adder increases above 10 when the number of decimal digits is more than 25 for reversible logic implementation. Such high-speed decimal adders find applications in real time processors and internet-based applications. The implementations use only reversible conservative Fredkin gates, which make it suitable for VLSI circuits.**

Index Terms - **decimal arithmetic, delay reduction, reversible logic, VLSI implementation**

I. INTRODUCTION

Currently, fast decimal arithmetic is gaining popularity in the computing community due to the growing importance of commercial, financial, and internet-based applications, which normally process decimal data. Low power designs with high performance are given prime importance by researchers, as power has become a first-order design consideration. While efforts are being made to reduce power dissipation due to leakage currents, alternate circuit design considerations are also gaining importance. In recent years, reversible logic has emerged as one of the most important approaches for power optimization. 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]. 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. 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.

Manuscript received July 4, 2007.

Rekha K. James is with Division of Electronics, Cochin University of Science and Technology, Kochi, Kerala-682022, India (phone: 0091-484-2320212; e-mail: rekhajames@ cusat.ac.in).

Shahana T. K., is with Division of Electronics, Cochin University of Science and Technology, Kochi, Kerala, India. (e-mail: shahanatk@cusat.ac.in).

Dr. K. Poulose Jacob is with Department of Computer Science, Cochin University of Science and Technology, Kochi, Kerala, India (e-mail: kpj@cusat.ac.in).

Dr. Sreela Sasi is with the Computer Science Department, Gannon University, Erie, PA, USA. (e-mail: sasi001@gannon.edu).

A reversible conventional Binary Coded Decimal (BCD) adder is proposed in [3] using NG (New Gate) and NTG (New Toffoli Gate) reversible gates. Even though the implementation is modified in [4] using TSG reversible gates, this approach is not taking care of the fanout restriction of reversible circuits, and hence it is only a near-reversible implementation. An improved reversible implementation of decimal adder with reduced number of garbage outputs is proposed in [5]. These implementations are for the conventional BCD adders, which are relatively slow.

Parity checking is one of the oldest and the most widely used methods for error detection in digital systems. Parity preservation proves to be useful for ensuring the robustness of reversible logic circuits. Parity-preserving reversible logic gates can be used for fault detection. B. Parhami demonstrated the feasibility of parity-preserving approach in the design of reversible logic circuits with examples of adder circuits [6].

In this research, carry select and hybrid decimal adders suitable for multi-digit BCD addition are 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, fast decimal adders such as carry select and hybrid adders are described. A comparison of carry select and hybrid BCD adders with conventional decimal adder in terms of speed and area is done for classical gate implementation. Next section describes implementations of carry select and hybrid adders using only parity preserving reversible Fredkin gates. An optimum block size for a hybrid adder is also derived. Finally, a graphical delay analysis of different fault tolerant implementations normalized to a Fredkin gate is presented.

II. CARRY SELECT BCD ADDER

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

The 4-bit binary adder adds the BCD inputs and generates a binary sum, S (S_{3-0}) that is checked by the '6-correction circuit'.

The output of the 6-correction circuit 'L' is given as

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

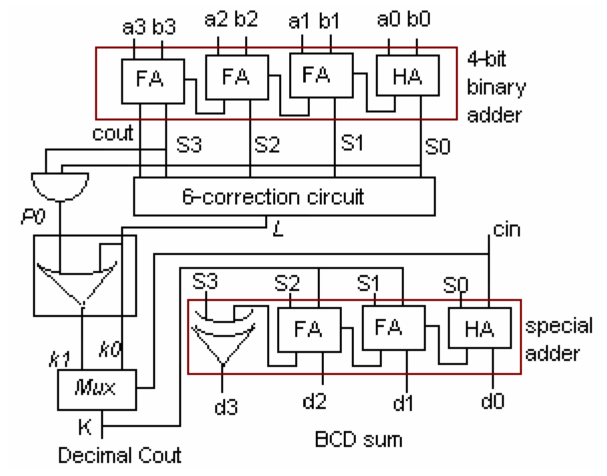

Fig 1: Carry Select BCD adder

On receiving C_{in}, a K-bit can be generated using equation (2).

$$K = S_3S_0C_{in} + L = P_0C_{in} + L \tag{2}$$

where P_0 is the carry propagate signal (S_3S_0)

If the carry select technique is adopted for K-bit generation then k_1 denotes the K-bit with $C_{in} = 1$ and k_0 with $C_{in} = 0$ and is given by $k_1=P_0+L$ and $k_0=L$. After computing both bits (k_1 and k_0) a selection is done using a 2:1 multiplexer.

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 a 4-bit binary adder consisting of a half adder, 2 full adders and an XOR gate as in Fig. 1.

An N-digit carry select adder will have a total (worst case) delay ($T_{dsum\ (carry\text{-}select)}$) equal to the sum of the 'carry delay' through the first digit ($T_{dcout(carry\text{-}select)}$), the carry select delays through the next (N-1) digits, and the 'sum delay' through the last digit ($T_{sum\text{-}digit(carry\text{-}select)}$). This is given in equation (3).

$$T_{dsum(carry\text{-}select)} = T_{d\text{-}cout(carry\text{-}select)}+(N-1)T_{mux}+T_{sum\text{-}digit(carry\text{-}select)} \tag{3}$$

where $T_{dcout(carry\text{-}select)}$ is the delay to generate K-bit from the BCD inputs for the first digit
T_{mux} is the delay of a 2:1 multiplexer
$T_{sum\text{-}digit(carry\text{-}salact)}$ is the delay of special adder for last digit

III. Hybrid BCD Adder

Hybrid logic for N-digit BCD addition can be used for delay reduction and is shown in Fig.2. The N-digit BCD input is divided into m-digit fixed blocks. Each m-digit adder consists of 'm' single digit carry select adders. To speed up addition, carry lookahead logic is included in m-digit blocks. For an m-digit adder, Decimal C_{out} at m^{th} digit (K_{m-1}) can be computed as given in equation (4) using equation (2).

$$K_{m-1} = C_{in} \prod_{k=0}^{m-1} P_k + \sum_{i=0}^{m-1} L_i \left[\prod_{j=i+1}^{m-1} P_j \right] \tag{4}$$

where L_i is the L-bit of i^{th} digit
P_i is the propagate bit for i^{th} digit

This can be written as
$$K_{m-1} = k_{0(m-1)}\, C_{in}' + k_{1(m-1)}\, C_{in} \tag{5}$$

where $k_{0(m-1)} = K_{m-1}$ with $C_{in} = 0$
$k_{1(m-1)} = K_{m-1}$ with $C_{in} =1$

The computations up to the generation of L_i and P_i bits at each digit are carried out in parallel for all digits. The delay for L-bit generation is given as

$$T_L = T_{adder} + T_{6\text{-}correction} \tag{6}$$

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

$k_{0(m-1)}$ and $k_{1(m-1)}$ for an m-digit block are computed using L_i and P_i as given in equation (4) after a delay of $T_{k1(m-1)}$ which is the delay of an m-input AND gate and (m+1) input OR gate.

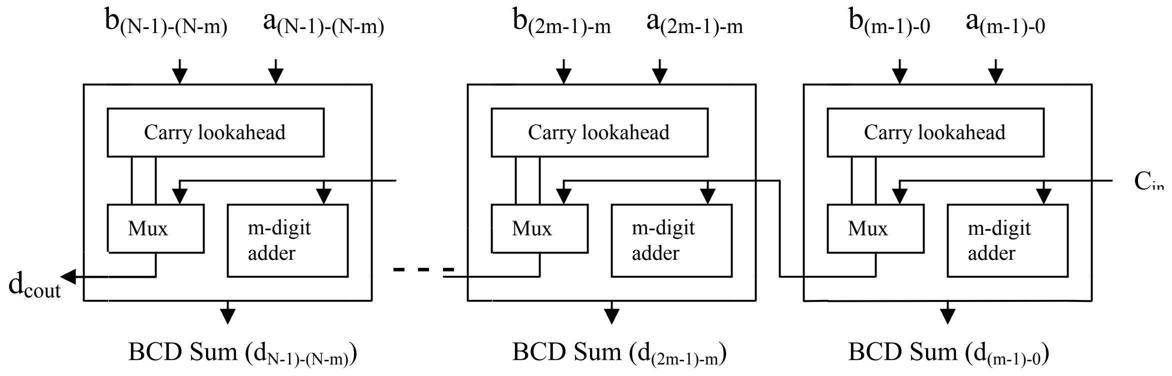

Fig 2. Hybrid N-digit Decimal Adder

On receiving C_{in}, the Decimal C_{out} at m^{th} digit (K_{m-1}) is generated after an additional delay of a 2:1 multiplexer (T_{mux}) and is given as

$$T_{m\text{-}dcout} = T_L + T_{k1(m-1)} + T_{mux} \quad (7)$$

The total (worst case) delay of an N-digit hybrid BCD adder ($T_{dsum\ (hybrid)}$) with fixed size carry look ahead block is the sum of the 'carry delay' through the first m-digit lookahead adder block ($T_{m\text{-}dcout}$), the carry select delays through the intermediate blocks, and the 'sum delay' through the last m-digit block ($T_{sum\text{-}m\text{-}digit}$). This is given in equation (8).

$$T_{dsum\ (hybrid)} = T_{m\text{-}dcout} + [(N/m)-2]\ T_{mux} + T_{sum\text{-}m\text{-}digit} \quad (8)$$
where $T_{sum\text{-}m\text{-}digit} = mT_{mux} + T_{sum\text{-}digit}$ (9)

The total (worst case) delay of an N-digit conventional BCD adder ($T_{dsum\ (conventional)}$) given in equation (10) is the sum of 'N' times the 'carry delay' through one digit and the 'sum delay' through the last digit ($T_{sum\text{-}digit(conventional)}$).

$$T_{dsum\ (conventional)} = N\ T_{dcout(conventional)} + T_{sum\text{-}digit(conventional)} \quad (10)$$

A comparison of conventional, carry select and hybrid (with m=4) BCD adders in terms of area and critical path delay is 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. Fig. 3 shows the graphical analysis of delay, and Fig. 4 shows the area-delay product normalized to that of a full adder. The area overhead of carry select and hybrid adders is compensated by the speed advantage compared to the conventional adder.

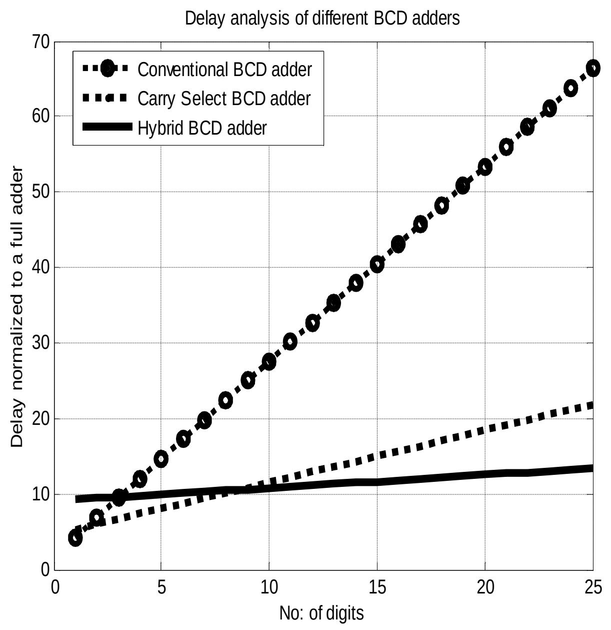

Fig. 3: Delay analysis of Conventional, Carry Select and Hybrid BCD Adders

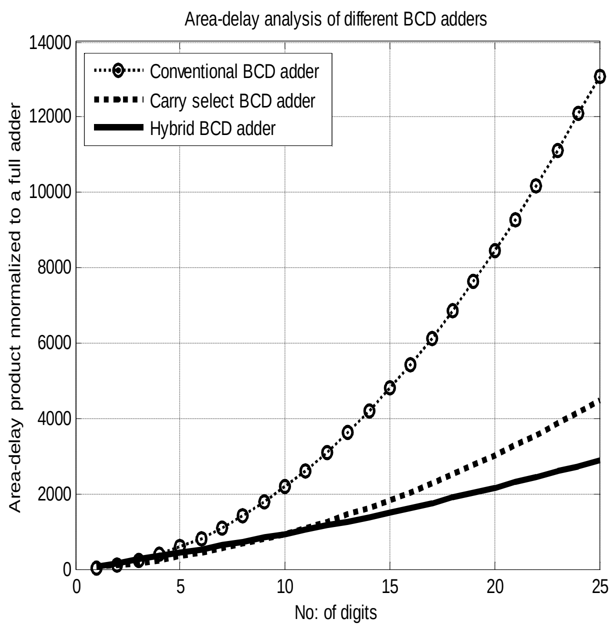

Fig. 4: Analysis of area-delay product of Conventional, Carry Select and Hybrid BCD Adders

Fig. 5 demonstrates the speed up factor of carry select and hybrid BCD adders compared to conventional BCD adder as the number of digits increases. Hybrid decimal adder is three times faster than the conventional BCD adder as the number of digits increases above 12 while the carry select BCD adder attains a speed up factor of 2.5 at this level. It is noted that the hybrid adder attains speed up over carry select BCD adder only when the number of input digits increases above 8. The delay comparison graphs show that hybrid adder is 5 times faster than that for the conventional BCD adder, when the input word length is above 25.

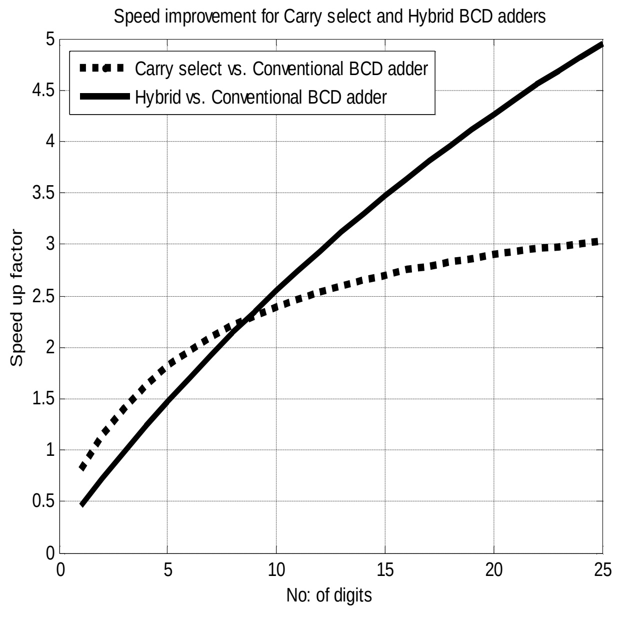

Fig. 5. Speed up factor for Carry select and Hybrid BCD adders vs. conventional BCD adder

IV. REVERSIBLE GATES

This section describes parity preserving reversible logic gates such as Feynman Double Gate (F2G) and Fredkin Gate (FRG). A 3*3 Feynman Double Gate (F2G) [6] has 3 inputs A, B, C and 3 outputs P=A, Q=A⊕B, R=A⊕C. A 3*3 Fredkin Gate (FRG) [7] has 3 inputs A, B, C and 3 outputs P=A, Q= A'B⊕AC, R= AB⊕A'C. These two gates satisfy the condition A⊕B⊕C=P⊕Q⊕R. In general, a parity preserving reversible gate is a gate in which the following condition is valid.

$$\oplus\sum X_i = \oplus\sum Y_i \qquad (11)$$

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

V. PARITY PRESERVING REVERSIBLE CARRY SELECT BCD ADDER

Recently, Hafiz [3], Thapliyal [4] and James [5] proposed reversible implementations of conventional BCD adders. But these implementations make use of reversible gates other than parity preserving gates, and hence they are not fault tolerant implementations. This research proposes a reversible implementation of fast BCD adders using the parity preserving reversible 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, 8]. Fig. 6 and Fig. 7 show the implementation of a half adder and a full adder using parity preserving Fredkin gates. The full adder implementation 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 [8] while observing the fanout restrictions.

The 4-bit binary adder realized using a half adder and 3 full adders will achieve delay reduction by using implementations in Fig. 6 and Fig. 7. The least significant bit (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], the delay is of 3 levels for each bit. So, an advantage of 1 delay level/bit is achieved in this implementation. The delay to generate C_{out} or 'Sum' in the n-bit binary adder is

$$T_{c\text{-ripple}} = T_{sum\text{-ripple}} = 2+2(n-1) \qquad (12)$$

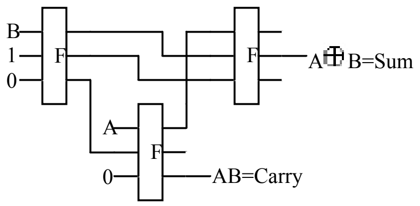

Fig. 6: Half adder using Fredkin Gates

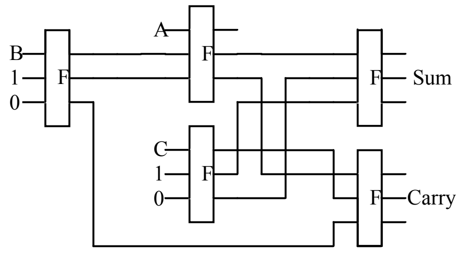

Fig. 7: Full adder using Fredkin Gates

For a conventional n-bit adder with 'n' full adders, it is

$$T_{c\text{-ripple (conventional)}} = T_{sum\text{-ripple(conventional)}} = 3+2(n-1) \qquad (13)$$

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

The parity preserving reversible implementation of a 6-correction circuit is shown in Fig. 8. The implementation requires 3 FRGs to generate the 'L' output. This circuit takes only 2 more delays after generating the 'Sum' to generate the L-bit. The delays to generate L-bit from the BCD inputs for carry select and conventional BCD adders are given in (14) and (15).

$$T_{L(carry\text{-}select)} = 4+2(n-1) \qquad (14)$$
$$T_{L\ (conventional)} = 5+2(n-1) \qquad (15)$$

Fig. 9 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 Fig. 9. 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. So, the additional delay in each digit to generate K-bit after receiving C_{in} is only due to one FRG.

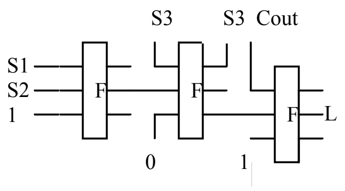

Fig. 8: Generation of L-bit using Fredkin Gates

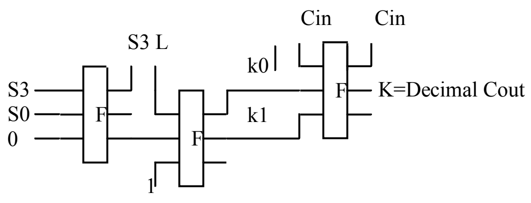

Fig. 9: Generation of K-bit using k_0 and k_1

The delay in generation of K-bit (Decimal C_{out}) for one digit for carry select BCD adder is given in equation (16), where n=4.

$$T_{d\text{-}cout(carry\text{-}select)} = 6+ 2(n-1) \tag{16}$$

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 ($d_{3\text{-}0}$). The Decimal C_{out} or the K-bit is the last input to be received for the special adder. The 'K' input passes through a maximum of 5 Fredkin gates to generate the BCD sum ($d_{3\text{-}0}$). But C_{in} 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. The 2 full adders and one XOR gate adds 5 more Fredkin gate delays. So the delay of special adder ($T_{sum\text{-}digit(carry\text{-}select)}$) is the delay of 6 Fredkin gates.

For an N-digit BCD adder, Decimal C_{out} at Nth digit ($K_{(N-1)}$) is generated after a delay equal to the sum of delay of K-bit generation for the first digit ($T_{dcout(carry\text{-}select)}$)and the multiplexer delays through the next (N-1)digits. It is given in equation (17).

$$\begin{aligned} T_{N\text{-}d\text{-}cout(carry\text{-}select)} &= T_{dcout(carry\text{-}select)}+(N-1)T_{mux} \\ &= 6+2(n-1) + (N-1) \end{aligned} \tag{17}$$

Substituting the delays in equation (3), the total worst case delay ($T_{dsum\ (carry\text{-}select)}$) in terms of Fredkin gate delay is

$$T_{d\text{-}sum\ (carry\text{-}select)} = 6+2(n-1) + (N-1) + 6 \tag{18}$$

For a conventional BCD adder the final adder is a 4-bit binary adder. The 'K' input passes through a maximum of 6 Fredkin gates (3 full adders) of the final adder to generate the BCD sum. So total (worst case) delay of an N-digit conventional BCD adder in terms of Fredkin gate delay is

$$\begin{aligned} T_{d\text{-}sum\ (conventional)} &= N\ T_{dcout(conventional)} + T_{sum\text{-}digit\ (conventional)} \\ &= N\ T_{L(conventional)} + T_{sum\text{-}digit(conventional)} \\ &= (5+2(n-1))\ N + 6 \end{aligned} \tag{19}$$

VI. Hybrid Reversible BCD Adder

The total (worst case) delay of an N-digit hybrid BCD adder with fixed size carry look ahead block is given in equation (8). The first term in equation (8) requires a delay as given in equation (7). In reversible implementation using Fredkin gates the delay to generate all L_i bits is T_L (given in equation (14)) with n=4. All P_i will be available when the generation of L_i gets over. $T_{k1(m-1)}$ is the delay of an m-input AND gate and (m+1) input OR gate. A 2 input AND or a 2 input OR can be implemented by a single Fredkin gate. Higher order AND and OR gates can be constructed using Fredkin gates arranged in a binary tree. An m-input AND gate or an m-input OR gate requires (m-1) Fredkin gates. In a binary tree implementation an input passes through a maximum of $\lceil \log_2 m \rceil$ Fredkin gates [8]. On receiving C_{in}, the selection of $k_{1(m-1)}$ or $k_{0(m-1)}$ requires one more Fredkin delay for each m-digit block. Hence the 'carry delay' through the first m-digit lookahead adder block is

$$T_{m\text{-}dcout} = 10 + \lceil \log_2 m \rceil + \lceil \log_2 (m+1) \rceil + 1 \tag{20}$$

The delay for carry select for intermediate blocks is $\frac{N}{m}$ - 2.

The sum delay through the last m-digit block is (m+6).
Total delay in generating N-digit BCD sum is given as

$$T_{d\text{-}sum\ (hybrid)} = 11+\lceil \log_2 m \rceil+\lceil \log_2 (m+1) \rceil+\frac{N}{m} - 2 + m+6 \tag{21}$$

However, the assumption $\lceil \log_2 m \rceil = \frac{m}{2}$ is valid for the small block sizes applicable to carry look ahead adder designs. Thus, (21) can be written as

$$T_{d\text{-}sum\ (hybrid)} = 15+2m+ \frac{N}{m} \tag{22}$$

Minimizing $T_{d\text{-}sum\ (hybrid)}$ with respect to block size m

$$m_{opt} = \sqrt{0.5N} \tag{23}$$

Substituting (23) into (22) gives the shortest delay for a fixed block size hybrid BCD reversible adder.

$$T_{d\text{-}sum\ (hybrid)} = 15 + \sqrt{8N} \tag{24}$$

Fig. 10 graphically demonstrates the computation of optimum block size (corresponding to shortest delay) of hybrid reversible BCD adders for different input lengths. Fig. 11 shows a comparative delay analysis of conventional, carry select and hybrid BCD adder reversible implementations normalized to that of a Fredkin gate.

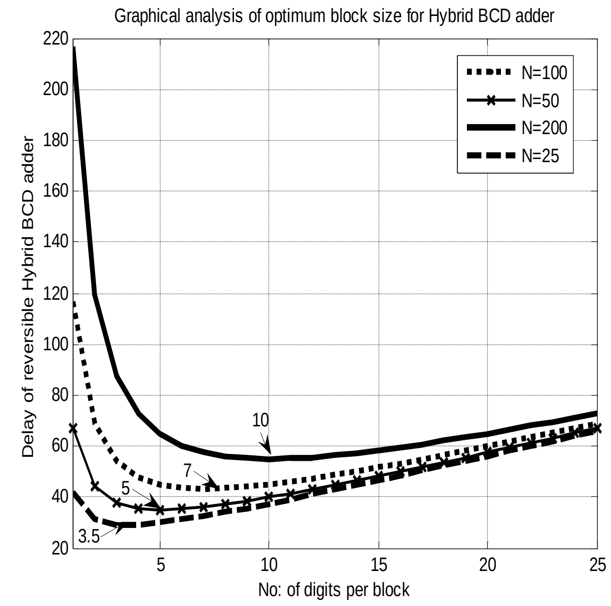

Fig. 10: Graphical analysis of optimum block size of Hybrid reversible BCD adder for different input lengths

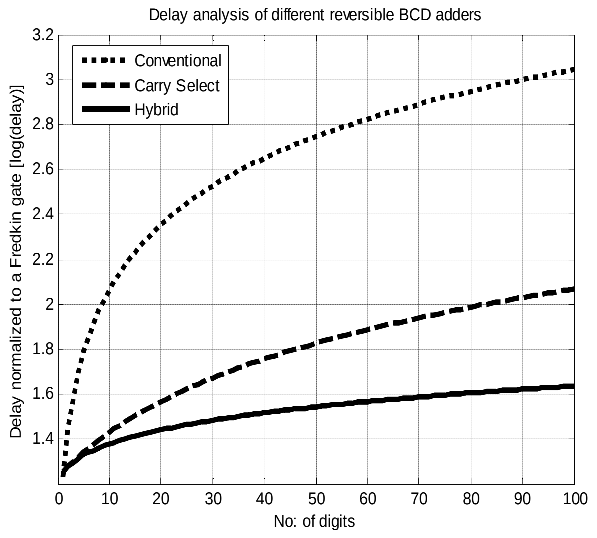

Fig. 11: Delay analysis of VLSI implementations of BCD adders

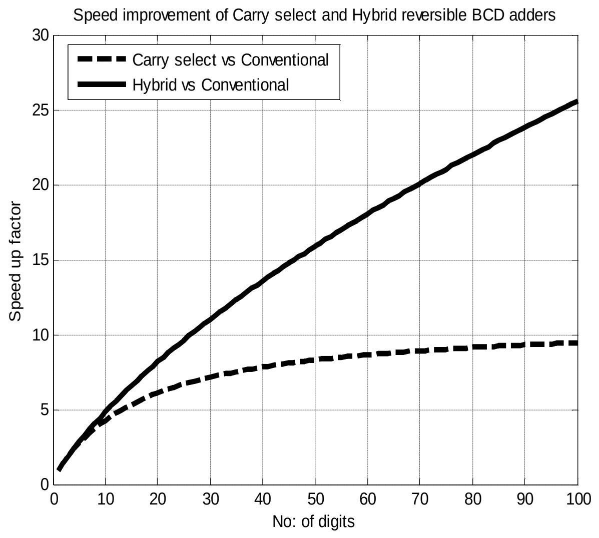

Fig. 12: Speed up factor for VLSI reversible implementations of fast BCD adders vs. Conventional BCD adder

Figure 12 demonstrates the speed up factor of reversible Fredkin gate implementations of carry select and hybrid BCD adders compared to conventional BCD adder. It can be noted that the hybrid adder attains speed over carry select BCD adder for all values of N in reversible implementation. Speed up factor of hybrid adder increases above 10 when the number of decimal digits is more than 25 for fault tolerant reversible logic implementation.

VII. Conclusion And Future Work

This research forms the basis of a fast Decimal ALU for a reversible CPU. Faster decimal adder circuits have been explored for several decades. This paper continues that practice by describing several reversible BCD adders using only Fredkin gate (FRG), a conservative reversible logic gate. VLSI implementations using only one type of modular building blocks can decrease system design and manufacturing cost.

The performance comparison of carry select and hybrid BCD adders with conventional BCD adder are presented. It is noted that the hybrid BCD adder attains speed up over carry select and conventional BCD adders, for any input length in a reversible implementation.

Varying the size of the carry lookahead blocks can reduce the total worst case delay, since carries generated or absorbed in the adder center have shorter data paths [9]. Investigations into determining alternate implementations can be done using logic synthesis methods [10, 11, 12]. Characterization of new families of 'n-input' – 'n-output' reversible gates that can be used for regular structures is an area which can be investigated further. Additionally, it is noted that there is a lack of simulation tools that support reversible gates, and this is most definitely an area worthy of attention.

References

[1] R. Landauer, "Irreversibility and Heat Generation in Computational Process", IBM Journal of Research and Dev., 5, pp.183-191, 1961.
[2] Bennett, C., "Logical Reversibility of Computation," IBM Journal of Research and Development, 17, pp.525-532, 1973.
[3] 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.
[4] H. Thapliyal, S. Kotiyal and M.B Srinivas, "Novel BCD Adders and their Reversible Logic Implementation for IEEE 754r Format", 19th VLSI Design 2006, pp. 387-392, Jan 2006.
[5] R. James, T. K. Shahana, K. P. Jacob and S. Sasi, "Improved Reversible Logic Implementation of Decimal Adder", To be published in Proceedings of IEEE 11th VDAT Symposium Aug, 2007.
[6] B. Parhami; "Fault Tolerant Reversible Circuits" Proc. 40th Asilomar Conf. Signals, Systems, and Computers, CA, Oct 2006.
[7] E. Fredkin and T. Toffoli, "Conservative logic", Intl. J. Theoretical Physics, Vol 21, 1982, pp. 219-253.
[8] 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.
[9] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, Oxford University Press, 2000.
[10] Dmitri Maslov, "Reversible Logic Synthesis", PhD Dissertation, Computer Science Department, University of New Brunswick, Canada, October 2003.
[11] 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.
[12] 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.