7th International Conference on Recent Innovations in Science, Engineering and Management
The Institutions of Engineers, Delhi State Center (India) RISEM - 16
16th September 2016, www.conferenceworld.in ISBN : 978-93-86171-07-8

MULTIPLIER DESIGN USING SQUARER IN REVERSIBLE LOGIC

Arindam Banerjee[1], Debesh Kumar Das[2]

[1]*Department of ECE, JIS College of Engineering, Nadia, West Bengal, India*

[2]*Department of CSE, Jadavpur University, Kolkata, West Bengal, India*

ABSTRACT

In this paper, two efficient multiplier design schemes using squaring in reversible logic has been reported. Here our objective is to reduce the quantum cost of the circuit at the cost of ancillary input and garbage output count. To achieve the goal we have used an efficient squaring scheme which has a recursive structure and the design provides significant reduction in quantum cost and also optimization of Ancillary input and garbage outputs. Here two design schemes for squaring have been proposed and the comparison of the design parameters like ancillary input and garbage output count etc. has been shown. Moreover, the quantum cost of the circuit has been optimized using NCV-|1> and NCV-|v_1> and double gate libraries. The most appreciable thing of the design is that the architecture has a generic structure.

Keywords : *Multiplier, Squarer, Recursion, Reversible Circuits*

I. INTRODUCTION

Multiplier is one of the core components of the arithmetic and logic unit of different processors. Binary and BCD multipliers have wide applications in DSP processors, image processing and artificial neural network [1-4]. But the power consumption in designing VLSI circuits is a great matter of concern. There is always a trade off between the propagation delay and power consumption in designing any digital circuit using MOS devices. Using MOSFET scaling, the propagation delay can be reduced drastically but power consumption is increased to a large extent. The researchers have been working on the techniques to reduce the power consumption for so many years. Finally it has been found that reversible logic may be a good alternative to reduce power consumption. Bennett [5] described the logical reversibility and showed mathematically that theoretically zero power consumption occurs in reversible circuits.

Reversible implementation of multiplier circuits has already been performed and reported in [6-8]. In [9], a multiplier technique has been reported using squaring. We are using that idea in this paper and show that this technique is very useful in reversible logic because binary squaring algorithm offers the elimination of redundant literals which in turn reduces the delay and power. The squaring technique, used here, has been adopted from our previous work [10].

II. BASIC REVERSIBLE GATES AND CIRCUITS

Reversible gates are all ($n \times n$) gates which have 1-to-1 mapping between input and output lines. The characteristic matrix of any reversible gate must be a unitary matrix. The unitary matrix can be defined as a matrix which inherits the following property, $A^* = A^{-1}$, where, A^* is the conjugate transpose of the matrix A.

Figs. 1(a) to 1(e) show some basic reversible gates used to design the reversible circuits which are as follows.

(i) NOT gate or inverter - 1×1 circuit (in Fig. 1(a)) having one input and one output line.

(ii) CNOT (Controlled NOT) gate - 2×2 circuit (in Fig. 1(b) having two input and two output lines.

(iii) Toffoli gate - $n \times n$ circuit (in Fig. 1(c) 3×3 circuit) which performs the logical AND operation based on the target input.

(iv) Peres gate - 3×3 circuit (in Fig. 1(d)) consisting of double gate structure used for arithmetic addition based on the target line.

(v) Double Peres gate - 4×4 circuit (in Fig. 1(e)) consisting of double gate structure used for arithmetic operation based on the target line.

In Fig. 2, the CNOT gate decomposition technique has been shown. CNOT gate can be decomposed into two V or V^+ gates as shown in Fig. 2. The number of quantum operations needed is known to be the quantum cost of the gate.

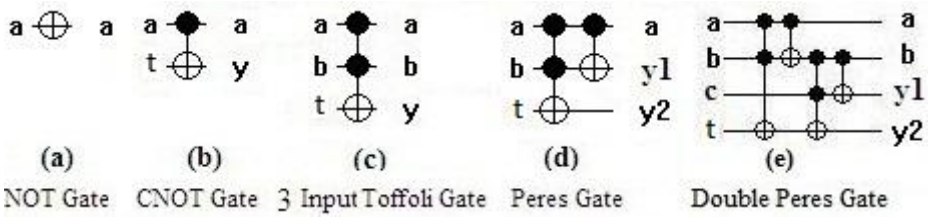

Fig. 1 Basic Reversible gates

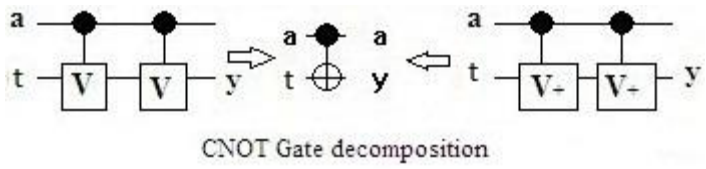

Fig. 2 Decomposition of CNOT gate into V and V^+ gate

III. PROPOSED MULTIPLIER ARCHITECTURE USING SQUARING TECHNIQUE

3.1 Mathematical Modelling of Multiplication using Squaring

Let us consider two binary numbers A and B whose multiplication is be determined. The product of A and B can be expressed as,

$$P = AB = \frac{4AB}{4} = \frac{(A+B)^2-(A-B)^2}{4} \quad (1)$$

This scheme was developed by Chen [9]. We assume that $(A \geq B)$. Then both (A+B) and (A-B) are positive numbers. If both A and B are n bit numbers then (A+B) is (n+1) bit number and (A-B) is n bit number. Therefore, $(A+B)^2$ and $(A-B)^2$ are respectively of 2(n+1) bits and 2n bits which are fed to the subtractor to achieve the final result. It is obvious that the subtracted output is 2(n+1) bit number and its 0^{th} and the 1^{st} output positions are occupied by 0. Therefore from the 2(n+1) bits result, the results of the 0^{th} and the 1^{st} bit positions are to be neglected and the remaining 2n bits are the final result.

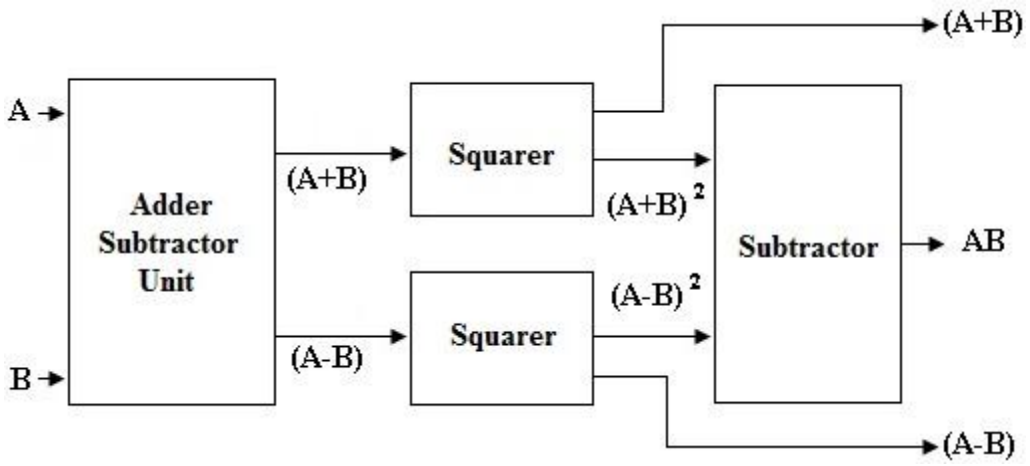

Fig.3 General architecture for reversible multiplier using squarer

The multiplier architecture is divided into three sub-modules:- (i) Adder, (ii) subtractor and (iii) Squarer as shown in Fig. 3. The modules are described as follows. Here two different design schemes have been proposed.

3.2 Adder

3.2.1 First Technique (T_1)

Fig. 4 shows the adder circuit for 4 bits using Peres gates. Here double Peres gates have been used as full adder. In this architecture for n bit input, the number of ancillary inputs and the garbage outputs are (n+1) and 2n respectively. The quantum cost is equal to 6n.

3.2.2 Second Technique (T_2)

Fig. 5 shows the adder circuit for 4 bit using the circuit shown by Cuccaro [11]. In this architecture for n bit input, the number of ancillary inputs and the garbage outputs are 2 and (n+1) respectively. The quantum cost is equal to (12n+1).

3.3 Subtractor

3.3.1 First Technique (T_1)

Fig. 6 shows the subtractor circuit for 4 bit using Peres gates. As discussed in the adder module, here also double Peres gates have been used as full adder and NOT gates have been used for subtraction purpose. In this architecture for n bit input, the number of ancillary inputs and the garbage outputs are (n+1) and 2n respectively. The quantum cost is equal to 8n.

3.3.2 Second Technique (T_2)

Fig. 7 shows the subtractor circuit for 4 bits using the reversible adder circuit shown by Cuccaro [11]. In this architecture for n bit input, the number of ancillary inputs and the garbage outputs are 2 and (n+1) respectively. The quantum cost is equal to (14n+1).

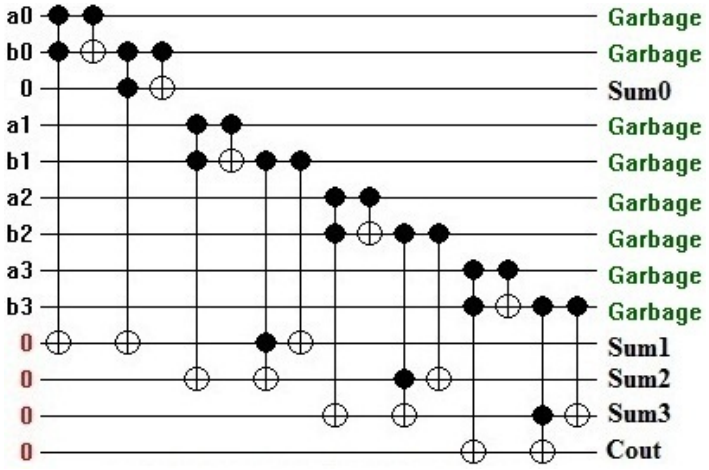

Fig. 4 Adder using Peres gate

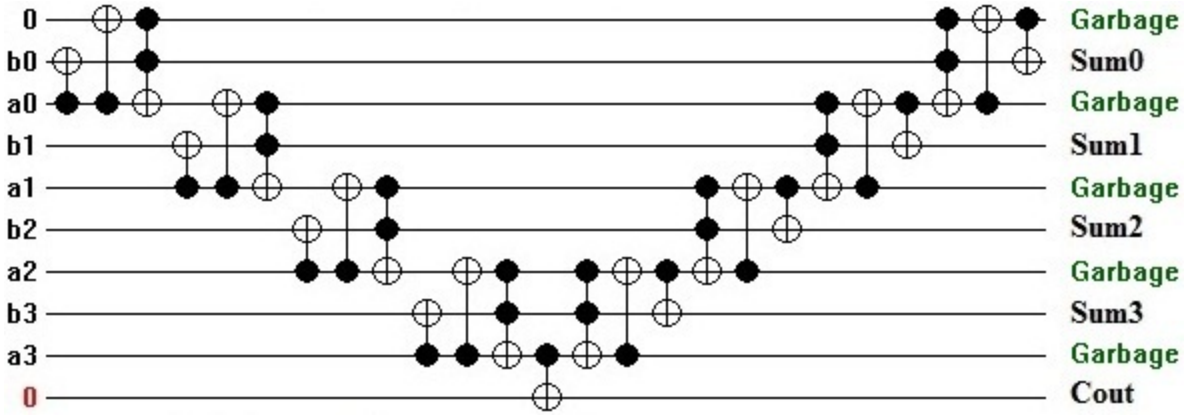

Fig. 5 Adder using the structure shown by Cuccaro [11]

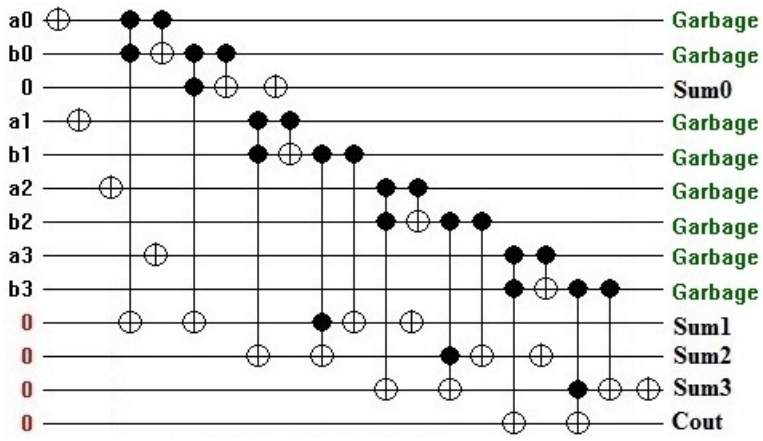

Fig. 6 Subtractor using the structure shown by Peres

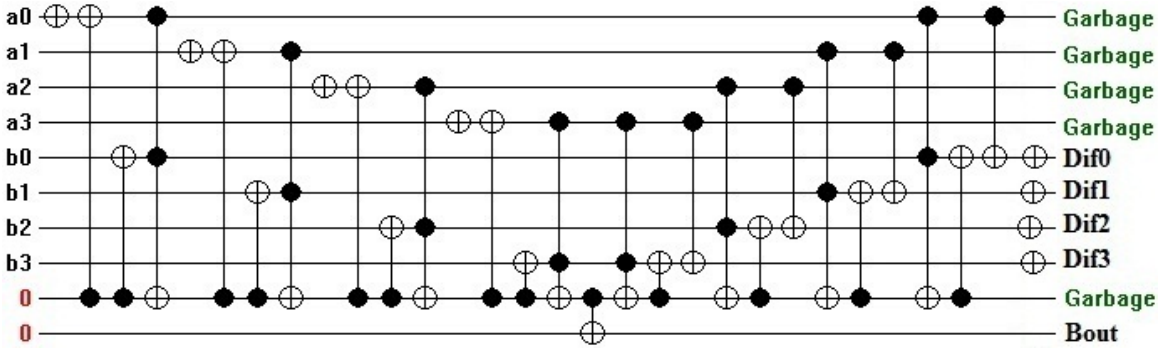

Fig. 7 Subtractor using the structure shown by Cuccaro [11]

For two n bit number multiplication, we first require addition and subtraction. After addition, (n+1) bits are generated as outputs. Similarly, after subtraction, n bits are generated as outputs. The squaring of (n+1) bits number produces 2(n+1) bits output. Similarly, the squaring of n bits number produces 2n bits outputs. Thus finally we need a 2(n+1) bit subtractor. Therefore for (n+1) bit subtractor design using Peres gates, (2(n+1) +1) ancillary inputs are required and 4(n+1) garbage outputs are generated. Total quantum cost is equal to 16(n+1).

If the addition technique described by Cuccaro [11] is used for subtraction, then number of ancillary inputs and garbage outputs are 2 and (2(n+1) +1) respectively. The quantum cost is (28(n+1)+1).

3.4 Squarer

Mathematical Modelling of squaring technique

The squaring technique has already been reported in [10]. The technique has been implemented using optimized garbage outputs and ancillary inputs. The mathematical modelling and the architecture of the proposed squaring technique is described below.

Let A_n is an n bit binary number whose square is to be determined. A_n can be expressed as,

$$A_n = \sum_{i=0}^{n-1} a_i 2^i = a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i = a_{n-1}2^{n-1} + A_{n-1} \quad (2)$$

Therefore, A_n^2 can be expressed as,

$$A_n^2 = (a_{n-1}2^{n-1} + A_{n-1})^2 \quad (3)$$

$$= a_{n-1}2^{2(n-1)} + 2^n a_{n-1}A_{n-1} + A_{n-1}^2 \quad (4)$$

where, $A_{n-1} = a_{n-2}2^{n-2} + a_{n-3}2^{n-3} + \cdots + a_2 2^2 + a_1 2^1 + a_0 2^0$.

$$= (a_{n-1} + a_{n-1}a_{n-2})2^{2n-2} + a_{n-1}\sum_{i=0}^{n-3} a_i 2^{n+i} + A_{n-1}^2 \quad (5)$$

Lemma 1 [10]: *For squaring of an n bit number, the result can't exceed 2n bits*

The second lemma has the similar interpretation as of the first lemma but it is applicable for addition architecture.

Lemma 2 [10]: *For an n bit addition the size of the output can't exceed (n+1) bit*

Suppose squaring of A is denoted as a 2n bit number $(S_{2n-1}^n, S_{2n-2}^n, S_{2n-3}^n, \ldots, S_2^n, S_1^n, S_0^n)$. Let A_{n-1}^2 is known that is $(S_{2n-3}^{n-1}, \ldots, S_n^{n-1}, S_{n-1}^{n-1}, \ldots, S_2^{n-1}, S_1^{n-1}, S_0^n)$ are all known. Equation (5) can be rewritten as,

$$A_n^2 = (a_{n-1} + a_{n-1}a_{n-2})2^{2n-2} + Sum_n \quad (6)$$

where, $Sum_n = \sum_{i=3}^{n}(a_{n-1}a_{n-i} + S_{2n-i}^{n-1})2^{2n-i} + A_{n-1}^2$.

From equation (6) we can now compute A_n^2 from A_{n-1}^2. Obviously, $S_i^n = S_i^{n-1}$ for $0 \leq i \leq (n-1)$. Then to compute A_n^2, we have to perform arithmetic addition of two (n-1) bit numbers as shown in Fig. 8.

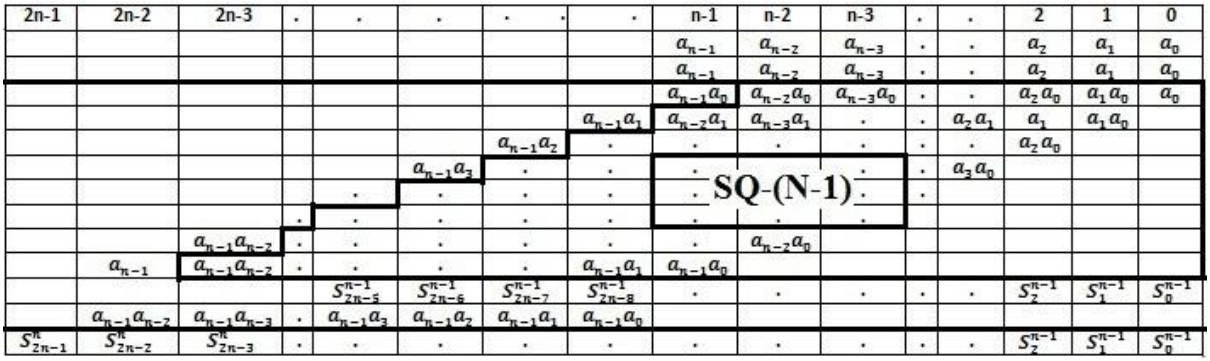

Fig. 8 n bit squaring using (n-1) bit squaring architecture

Consider the number $(a_{n-1}, a_{n-2}, a_{n-3}, \ldots, a_1, a_0)$. Following the schematic of Fig. 8, suppose we have the result of square of (n-1) lowest significant bits given by $(S_{2n-3}^{n-1}, S_{2n-4}^{n-1}, \ldots, S_0^{n-1})$. We can see that the square of n bit number can be obtained by adding two (n-1) bit numbers $(a_{n-1}, S_{2n-3}^{n-1}, \ldots, S_n^{n-1})$ and $(a_{n-1}a_{n-2}, a_{n-1}a_{n-3}, \ldots, a_{n-1}a_1, a_{n-1}a_0)$. From the above mentioned two lemmas it is obvious that $(n+1)^{th}$ stage of addition does not exist for (n-1) bit addition therefore only one carry from the preceding stage must be propagated at the $(n-1)^{th}$ stage to generate the result at the n^{th} position.

In every bit position a carry will be generated which will be added to next bit. Suppose these carries are $(C_{n-1}, C_n, \ldots, C_{2n-2})$ as shown in Fig. 9. It is obvious that $C_{n-1} = 0$ since there is no initial carry.

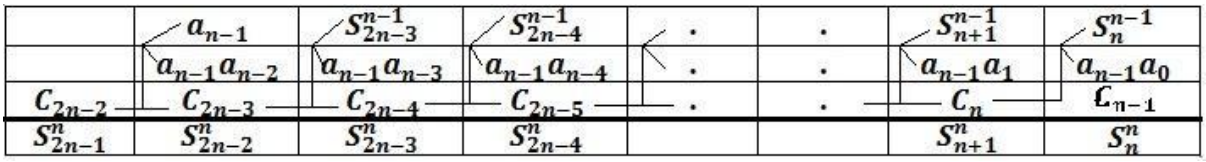

Fig. 9 Schematic Diagram for addition of higher order n bit

Lemma 3 [10]: *The output terms of n bit squaring* S_{2n-j}^n *for* $1 \leq j \leq n$ *can be expressed as,*

$S_{2n-j}^n = S_{2n-j}^{n-1} \oplus a_{n-1}a_{n-j} \oplus C_{2n-j-1}$ *for* $3 \leq j \leq n$,

$= a_{n-1} \oplus a_{n-1}a_{n-j} \oplus C_{2n-j-1}$ *for* $j = 2$

$= C_{2n-2}$ *for* $j = 1$

Lemma 4 [10]: *For* $(2 \leq n \leq 4)$, $C_{2n-2} = a_{n-1}a_{n-2}$

Using Lemma 3, we can compute all the terms S_i^n for $0 \leq i \leq (2n-1)$. Lemma 4 suggests that for $n \leq 4$, we have a reduced design and for $n \geq 5$, the design has a general structure.

3.4.1 First Technique (T_1) for $n \leq 4$

Let us consider the squaring of a two bit number (a_1, a_0). The multiplication of this number by itself is shown in Fig. 10. Obviously the 0th bit of the squaring is $S_0^2 = a_0$, 1st bit is $S_1^2 = 0$, as $a_1a_0 \oplus a_1a_0 = 0$ and a_1a_0 is propagated to 2nd bit position which produces $S_2^2 = a_1 \oplus a_1a_0$. The carry generated will be propagated to the next bit. The carry generated by a_1 and a_1a_0 is simply a_1a_0, thus $S_3^2 = a_1a_0$.

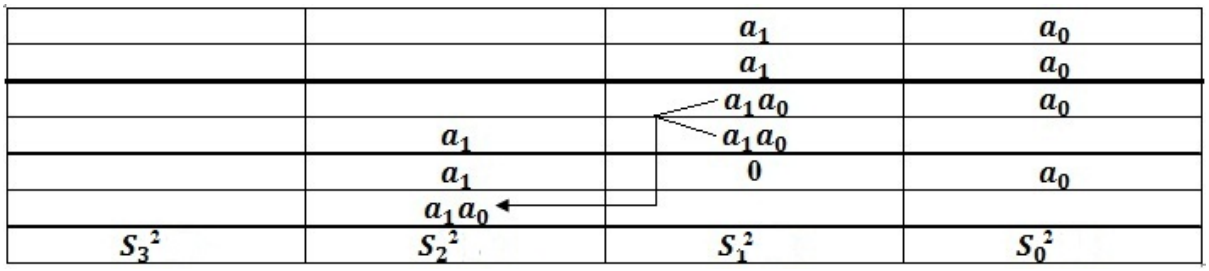

Fig. 10 Structural Methodology for 2 bit squaring

This scheme can be implemented as in Fig. 11. This has been implemented using a circuit described in [13]. The difference between the technique in [13] and our approach is that, in [13], the authors used a Peres gate to execute S_2 and S_3, as shown in Fig. 11 where we did it in a simpler way (Fig. 12).

From the squaring of two bits number we like to achieve the design for squaring of three bits number. Let us see the multiplication of the three bits number (a_2, a_1, a_0). It is obvious that this squaring procedure contains a portion of squaring of 2 bits (a_1, a_0) as shown in Fig. 12. Suppose the squaring of two bit number produces $(S_3^2, S_2^2, S_1^2, S_0^2)$. Obviously $S_0^3 = S_0^2, S_1^3 = S_1^2, S_2^3 = S_2^2, S_3^3 = S_3^2 \oplus a_2a_0$. The carry generated by S_3^2 and a_2a_0 is

nothing but performing logical AND operation on S_3^2 and a_2a_0. The value of S_4^3 and S_5^3 are derived from Lemma 3. The implementation is shown in Fig. 13. The architecture of a four bit squarer using three bit squarer is implemented in the similar manner as shown in Fig. 14.

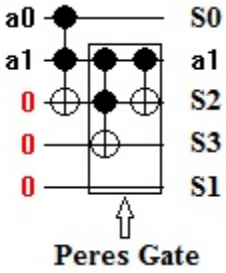

Fig.11 The two bit squaring as in [13]

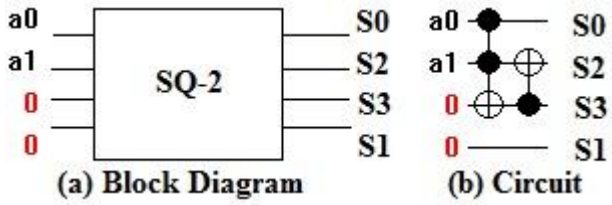

Fig. 12 Schematic Diagram and the circuit for two bit squaring

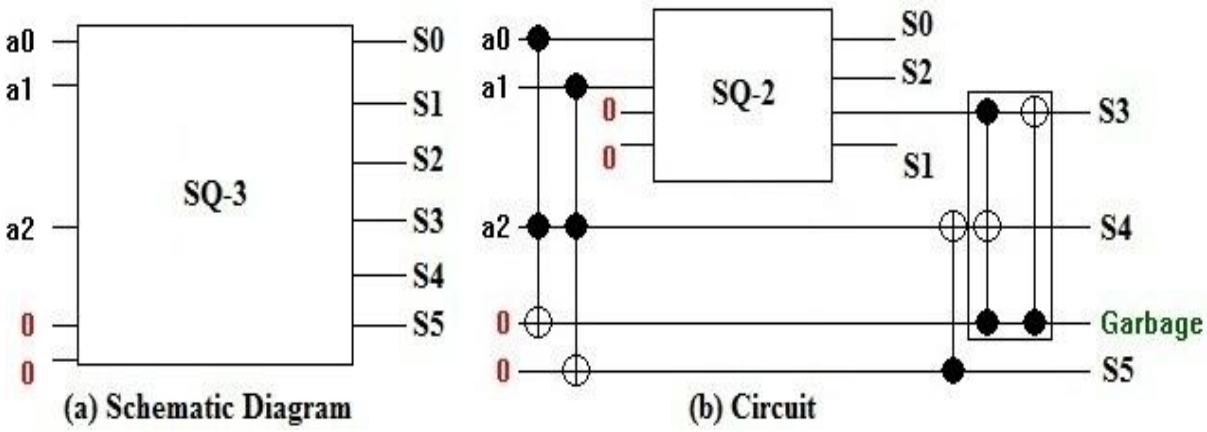

Fig. 13 Schematic Diagram and the circuit for three bit squaring

3.4.2 First Technique (T_1) for $n \geq 5$

Following the architectural description of Fig. 13 and Fig. 14, if we like to design the squarer for n bits using the squarer for (n-1) bits then an adder of (n-1) bits is needed. From the architecture it seems that we need (n-2) full adders and 1 half adder. But by derivation of Boolean algebra, we can show that (n-4) full adders and 4 half adders and one CNOT gate are required. Suppose in the column of S_{2n-2}^n, there are three elements to be added (a_{n-1}, $a_{n-1}a_{n-2}$ and C_{2n-3}).

The carry generated by a_{n-1} and $a_{n-1}a_{n-2}$ is simply $a_{n-1}a_{n-2}$ thus it can be simply passed to the next level where it should have modulo-2 addition with $a_{n-1} \oplus a_{n-1}a_{n-2}C_{2n-3}$. Fig. 15 shows the general architecture for n bit squarer using (n-1) bit squarer. The structure for n=5 is shown in Fig. 16.

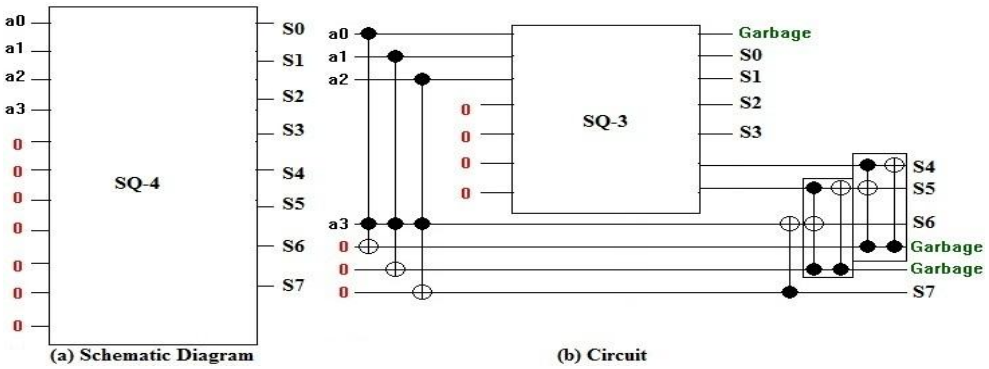

Fig. 14 Scheme for four bit squaring

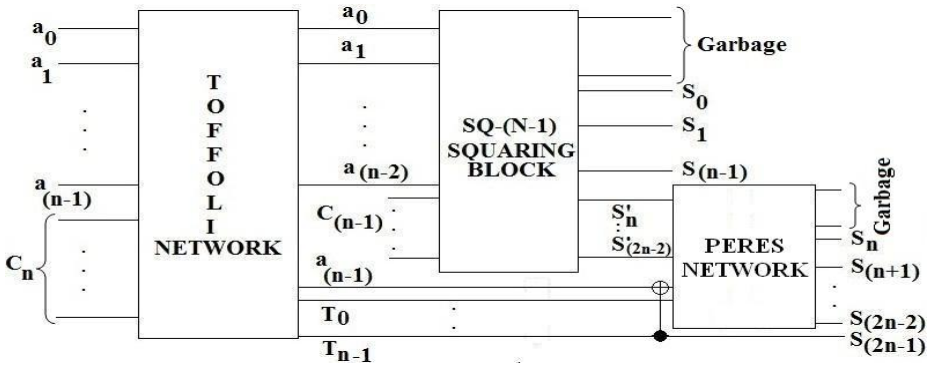

Fig. 15 Schematic Diagram for n bit squaring

The design consists of some full adders using double Peres gate. Fig. 17 shows the full adder design using double Peres gate which has been used in Figs. 4 and 6 and its decomposition into elementary quantum gates. The middle box of Fig. 17 (b) consists of V and V^+ gates which results in Identity gate of cost 0. Thus the full Adder has quantum cost 6. The ancillary inputs, garbage outputs and quantum cost for the squarer using the (T_1) have been calculated in [10].

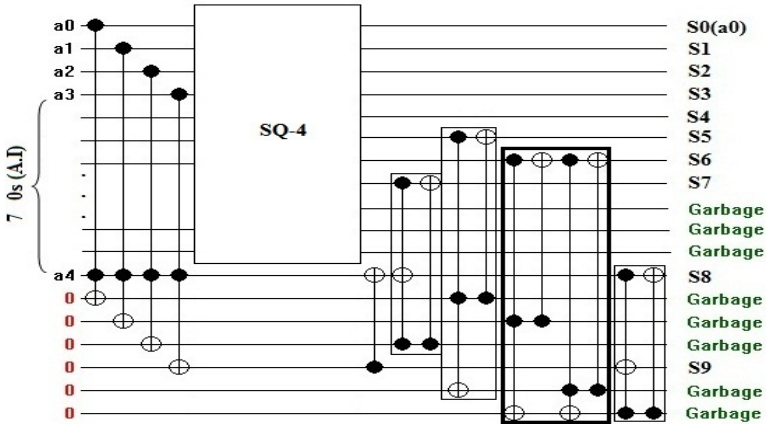

Fig. 16 Circuit for five bit squaring using four bit squaring architecture

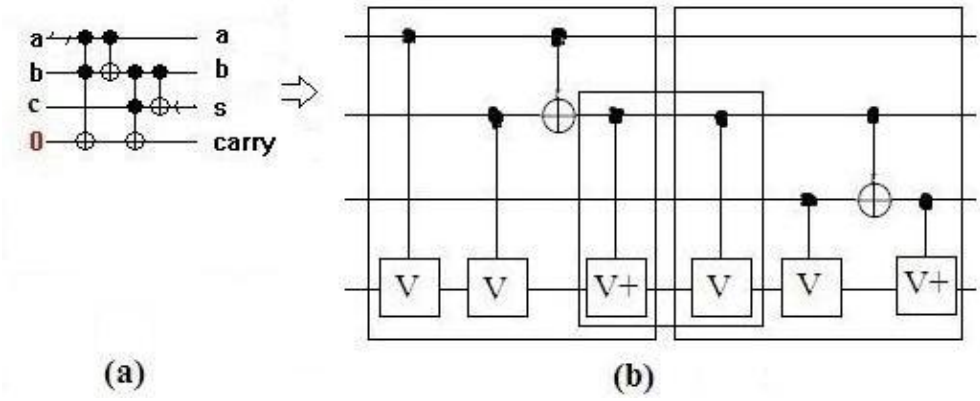

Fig. 17 Decomposition of single bit full adder using double Peres gate

3.4.3 Second Technique (T_2)

A full adder has an alternative design as depicted in Fig. 5 which has been described in [11]. Using the structure of Fig. 5, we can also design a squarer in a recursive manner. The decomposition of the circuit has been shown in Fig. 18. Following it, we need only 1 ancillary input. For $n \leq 4$, the circuit is same as in the first technique. For $n \geq 5$, we provide the calculations in [10].

3.4.4 Multiplier Parameters Calculation

3.4.5.1 Ancillary Input Count

The overall design needs one adder, two subtractors and two squarers. Here the calculations are shown for two different techniques described above.

3.4.5.2 First Technique (T_1)

As described earlier, the adder for n bit input takes (n+1) and 2n ancillary inputs and the garbage outputs respectively. The quantum cost is equal to 6n. Similarly, the subtractor for n bit input, the number of ancillary inputs and the garbage outputs are (n+1) and 2n respectively and the quantum cost is equal to 8n. The first squarer for (n+1) bit which is coming from the adder has the ancillary inputs $\frac{n^2-n+2}{2}$ for $(n+1) \leq 4$ and $(n-1)^2-2$ for $(n+1) > 4$. The second squarer which has the output coming from the subtractor has the ancillary inputs $\frac{n^2-n+2}{2}$ for $n \leq 4$ and $(n-1)^2-2$ for $n > 4$. The last subtractor used in the design requires (2(n+1) + 1) ancillary inputs. In the similar manner, for n=4 and $n \geq 5$, the ancillary inputs can be calculated.

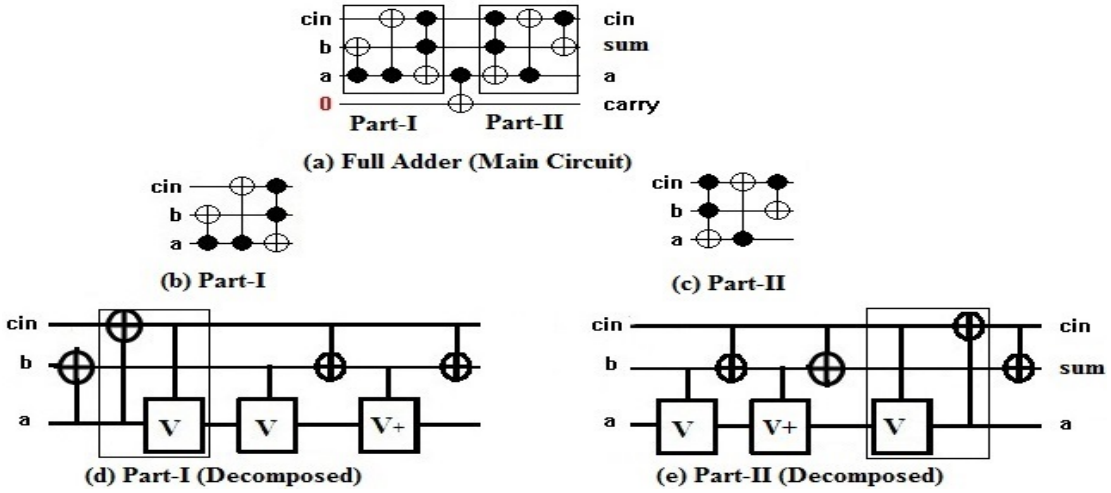

Fig. 18 Decomposition of single bit full adder described in [11]

So, for n $\leq$ 3, $A.I(n) = n^2 + 4n + 6$. For n=4, $A.I(n) = \frac{3n^2+7n+6}{2}$. For n>4, $A.I(n) = 2n^2 + 2n + 1$.

In the similar manner, garbage output and quantum cost can be calculated.

For n $\leq$ 3, $G.O(n) = n^2 + 6n + 6$. For n= 4, $G.O(n) = \frac{5n^2+9n+2}{2}$. For n>4, $G.O(n) = 2n^2 + 4n + 1$.

For n $\leq$ 3, 4 and $\geq$ 5, $Q.C(n) = 9n^2 + 15n + 2$, $9n^2 + 24n + 19$/ and $10n^2 + 16n + 2$ respectively.

3.4.5.3 Second Technique (T_2)

For the second technique, the parameters can be calculated as follows.

For $n \leq 3$, $A.I(n) = n^2 + 2n + 8$. For n=4, $A.I(n) = n^2 + 16$. For n>4, $A.I(n) = 2n^2 - 2n + 11$.

For n $\leq$ 3, $G.O(n) = n^2 + 4n + 8$. For n= 4, $G.O(n) = \frac{3n^2-n+28}{2}$. For n>4, $G.O(n) = 2n^2 + 3n + 1$.

For n $\leq$ 3, 4 and $\geq$ 5, $Q.C(n) = 9n^2 + 42n + 24, 9n^2 + 50n + 21$, and $10n^2 + 52n$ respectively.

IV. RESULT ANALYSIS

We have made the comparative study of our work with the similar work published in [8]. These comparative results are tabulated here in Table 1. It establishes better performance of our techniques in terms of the several circuit parameters. In the second approach, the ancillary inputs and the garbage outputs are reduced to a considerably good amount with the tolerable increase in quantum cost.

Bit length (n)	Ancillary Input			Garbage Output			Quantum Cost		
	[8]	T_1	T_2	[8]	T_1	T_2	[8]	T_1	T_2
2	8	18	16	6	22	20	74	88	144
3	12	27	23	9	33	29	245	148	231
4	16	41	32	12	59	36	518	259	365
8	54	145	123	46	161	153	2437	770	1056
16	176	545	491	160	577	561	9696	2818	3392

Table 1: Calculation of performance parameters of the proposed multiplier architectures

V. CONCLUSION

We proposed two techniques of dedicated multiplication in reversible logic. The first technique offered less quantum cost, the second one was better in terms of less ancillary inputs and garbage outputs. The comparison of our methods with a similar work was also shown. The designs had a systematic approach with modular structures and could be recursively built. It was observed that in both the techniques, quantum cost was reduced in comparison to [8].

VI. ACKNOWLEDGEMENT

The work has been supported by the fund granted by Technical Education Quality Improvement Programme Phase – II, Jadavpur University

VII. REFERENCE

[1] D. V. Poornaiah, “Sign extension bit minimisation algorithm for multibit coded multiplier structures for dsp applications,” IET Electronics Letters, vol. 32, pp. 1454–1456, August 1996.

[2] S. Hong, S. Kim, M. C. Papaefthymiou, and W. E. Stark, “Low power parallel multiplier design for dsp applications through co-efficient optimization,” in IEEE Int. Conf. On ASIC/SOC, 1999, pp. 286–290.

[3] M. Bhuyan, N. Amin, M. A. H. Madesa, and M. S. Islam, “Vlsi implementation of inverse discrete wavelet transform for jpeg 2000,” in IEEE Int. Conf. On Computer and Information Technology, 2007, pp. 1–5.

[4] S. L. Pinjare, K. Mudnal, and S. Kumar, “Distributed arithmetic multiplier based artificial neural network architecture for image compression,” in IEEE Int. Conf. On Parallel, Distributed and Grid Computing, 2012, pp. 135–140.

[5] C. H. Bennett, "Logical reversibility of computation," in IBM Journal of Research and Development, vol. 17, November 1973, pp. 525–532.

[6] S. Kotiyal, H. Thapliyal, and N. Ranganathan, "Circuit for reversible quantum multiplier based on binary tree optimizing ancilla and garbage bits," in Proc. of Int. Conf. on VLSI Design, January 2014, pp. 545–550.

[7] M. Haghparast, S. Jassbi, K. Navi, and M. Eshghi, "Optimized reversible multiplier circuits," in Journal of Circuits, Systems and Computers, vol. 18, 2009, pp. 311–323.

[8] S.Offermann, R.Wille, G.W.Dueck, and R.Drechsler, "Synthesizing multiplier in reversible logic," in IEEE Symposium on DDECS, April 2010, pp. 335–340.

[9] T. C. Chen, "A binary multiplication scheme based on squaring," in IEEE Trans. on Computers, vol. C-20, 1971, pp. 678–680.

[10] A. Banerjee and D. K. Das, "Squaring in reversible logic using iterative structure," in Proceedings of East West Design and Test Symposium, 2014, September 2014.

[11] S.A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, "A new quantum ripple-carry addition circuit," in arxiv:quant-ph/0410184v1, 2008.

[12] A. Banerjee and D. K. Das, in Technical Report, JU/CSE/02/14, 2014.

[13] H. V. Jayashree, H. Thapliyal, and V. K. Agrawal, "Design of dedicated reversible quantum circuitry for square computation," in Proc. of Int. Conf. on VLSI Design, January 2014, pp. 551–556.