A Novel Approach for Recoding Canonical Signed Digit

Jugal Kishore Bhandari[1], B.Mamatha[2]
[1]Asst.Prof, [2]Asst.Prof
ECE Dept Gcet, Cheeryal

Abstract - In this study new approaches for conversion of regular unsigned or signed number into canonical signed digit are presented. We evaluate the proposed circuit and compare it with a circuit based on the conventional adder structure. We show that the proposed architectures perform faster by 8% or more than the circuit based on the conventional structure. In this work different adder logics to generate fast carry are used in canonical signed conversion method. As a result, the proposed circuits are efficient in terms of speed, area and power consumption in comparison with other conventional architectures. Simulations of different configurations are performed using Xilinx and synthesized using Cadence tools. These logics are applied to Wallace multiplier by applying CSD conversion to multiplier and /or multiplicand which gives far better results than counterpart previous architectures.

Keywords - canonical-signed-digit (CSD), Booth's Recoding, adders

I. INTRODUCTION

Asked for the total of thirty four and sixty two, some folks could even be ready to work this out while not pen and paper. However, most folks would struggle to search out the merchandise (unless we all know our thirty four times tables). Another, maybe additional vital reason for the distinction in issue is that the variety of operations that should be performed. To do the addition, I will break the matter down into 2 little additions, 4+2 = six and 3+6 = nine. In fact it is also potential that there's a carry I actually have to require care of, however we'll ignore these in our analysis. Currently contemplate the multiplication. I actually have to try and do 2×4 = eight, 30×2 = 60, 60×4 = 240 and 30×60 = 1800. Currently i would like to feature of these along. So, in total four easier multiplications we've got to try to and 3 additions. This method is typical for multiplication using school text algorithms.

Digital FIR filter is one in every of the essential elements in Digital Signal method (DSP) and communication system. With Associate in Nursing large growth in mobile computing and multimedia system applications, would like for low power and high speed DSP system has seen a tremendous growth [1]. Digital filters square measure accustomed modify the attributes of signal by removing noise from the initial signal and kind the spectral characteristics of the following signal [2]. Digital filters are superior in level of performance as they're extremely stable, correct and versatile as compared to analog filter [3]. Due to this reason, the necessity of a digital filter with optimized space, power and delay may be a difficult task. DSP applications need an oversized order FIR filter. And therefore the complexity will increase with increasing filter order owing to needs of larger mathematical computations [5]. Therefore, real time implementation of this filter with precise value is sitting as a heavy challenge. so as to attain efficient digital filter style, order of filter should be as little as doable. This paper focuses primarily on the FIR filter owing to its absolute stability and linear part response [6].

This paper focuses mainly on the FIR filter due to its absolute stability and linear phase response [6]. On the premise of hardware, digital filter may be classified into 2 major categories: multipliers based mostly and memory based [7]. The most components of digital filter embrace registers to save lots of the samples of signals, adders to carry out total operations and variety for multiplication of the filter coefficients with signal samples [8]. Even with the particular undeniable fact that designing of digital filter seems straightforward, but the planning bottleneck is its number block for speed, house and power consumption [9]. Complexness is primarily dominated by constant multiplication operation [10,11]. therefore on cut back quality, the filter coefficients square measure depicted in CSD illustration that wants the littlest quantity form of Computations [12].The multiplication of 2 numbers x*y is enforced by accumulating the shifted partial product xiy, for every digit xi of the multiplier x.

So, the amount of necessary addition operations needed to total the partial product is one less than the amount of nonzero digits within the illustration of the corresponding constant number x. the event of quick CSD and MSD conversion algorithms has been the main target of a lot of effort. Booth's coding was given [13] in 1951, to expeditiously multiply 2 numbers using coding multipliers. In 1960, Reitwiesner developed an rule to convert two's complement numbers to CSD [14].

The remaining of the paper is organized as follows: an outline of CSD is given in section II. Section III consists of various adder logics for quick carry generation. Section IV describes the planned work for CSD improvement. In section V simulation results are mentioned. Finally, section VI concludes the paper by summarizing the foremost contributes.

II. CANONICAL SIGNED DIGIT

Canonical signed digit (CSD) selection illustration is one version of signed-digit (SD) selection illustration. SD selection illustration is also a radix-2 illustration using a digit set. For a given constant, the corresponding CSD illustration is exclusive and has 2 principal properties: (1) the amount of nonzero digits is lowest and (2) no 2 consecutive digits are nonzero, that is, 2 nonzero digits aren't adjacent. Property (1) implies a lowest performing weight that ends up in a discount within the variety of additives in arithmetic operations. CSD illustration has verified to be useful for the planning and implementation of digital filters, like those with the finite impulse response (FIR) digital filter style. CSD code has been used largely to implement economical multipliers It permits reduction within the variety of partial products that has got to be calculated quick, and it's low power consumption and a low are structure for multipliers in digital signal process (DSP) applications.

It's well-known that several researchers have addressed the CSD writing to convert two's complement into CSD code. Already in 1960, Reitwiesner projected algorithmic program for changing two's complement numbers to a minimum weight radix-2 (binary) signed digit illustration. From the sensible purpose of read, the standard approach to come up with the CSD illustration uses look-up table. Some algorithms to convert two's complement into CSD numbers attempt to scale back the machine complexness, however don't seem to be appropriate for hard ware implementation. Other hardware approaches propose the by-pass technique, quick carry look-ahead circuits or parallel prefix schemes to cut back hardware however they solely concentrate on carry optimization while not considering the CSD secret writing. All of those algorithms generate the CSD code recursively from the smallest amount vital bit (LSB) to the foremost vital bit (MSB).

However, in some applications, like the computation of mathematical process, the conversion from MSB to LSB brings some benefits. The CSD illustration of a number could be a signed and distinctive digit illustration that contains no adjacent nonzero digits. Given n-digit binary un-signed number. X={x0, x1,….xn-1} expressed as

$$X = \sum_{i=0}^{n-1} xi.2i, \qquad x \in \{0,1\}$$

then the (n+1)-digit CSD representation Y= {y0, y1,…..yn} of X is given by

$$X = \sum_{i=0}^{n-1} x_i.\ 2^i = \sum_{i=0}^{n-1} y_i.\ 2^i \quad y \in \{1,0,-1\}$$

The condition that all nonzero digits in a CSD number are separated by zeros implies that yi+1.yi = 0, 0≤ i ≤ n-1. The adoption of a ternary number system adds flexibility to the CSD representation. The example below shows how it works for both signed and unsigned numbers.

$$0\ \ 1\,0\,1\,0\,\underbrace{1\,1\,1\,1}\ (175)$$

$$0\ \ 1\,0\,\underbrace{1\,1}\ \ 0\,0\,0\ \ \bar{1}$$

$$0\ \ \underbrace{1\,1}\,0\ \ ^{1}\ 0\,0\,0\ \ \bar{1}$$

$$1\ \ 0\ ^{1}\ 0\ ^{1}\ 0\,0\,0\ ^{1}$$

$$2^8\text{-}2^6\text{-}2^4\text{-}2^0 = 175$$

$$\begin{array}{cccccccc} & 1 & 1 & 1 & 1 & 1 & 1 & 1\ (+127) \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & \bar{1} \end{array}$$

$$2^7\text{-}2^0 = 127$$

It requires that each digit yi must be encoded over two bits {ysi, ydi}.Table 1 shows the two most frequently used encodings.

Yi	y_s^i, y_d^i
0	0 0
1	0 1
$\bar{1}$	1 0

Table 1: encoding used in the binary representation of a CSD digit

Conventional Approach using full adder logic - Basically in order to get Ys and Yd , the following is the procedure, where first step is to get the carry propagated in each stage by increasing index of input. For example if X is input of 8 bits the index based positions of bits is X[7],X[6]…….X[1],X[0]. Considering a full adder, give inputs to the adder as 1'b0,x[0]& x[1], and we will get one carry (sum is ignored) now that carry will be forwarded to the next full adder with other inputs as X[1] & X[2]. Now for computing Ys and Yd we use the following expressions.

Y_0^s = X[0]&X[1]

Y_0^d = X[0]&(~X[1])

Y_1^S=(X[1]^C_{Prev})&X[2]

Y_1d=(X[1]^C_{Prev})&(~X[2]) and so on

So gereral expression is Y_i^s=(X[i]^C_{Prev})&X[i+1]

Y_i^d=(X[i]^C_{Prev})&(~X[i+1])

From the above expressions, it is clear that one method to get CSD is to get two representations Ys and Yd from which we can get the final canonical signed digit. Figure 1 shows the architecture of CSD using full adder carry generation logic.

Yi = Ys-Yd

$$\text{For an unsigned number } \boldsymbol{X}\begin{cases} y_{n-1}^{d}=d_{n-1} \\ y_{n-1}^{s}=0 \\ y_{n}^{d}=d_{n}=c_{n-1}=x_{n}c_{n-2} \\ y_{n}^{s}=0 \end{cases}$$

$$\text{For a signed number } \boldsymbol{X}\begin{cases} y_{n-1}^{d}=\overline{x}_{n}d_{i}=\overline{x}_{n-1}c_{n-2} \\ y_{n-1}^{s}=x_{n}d_{i}=x_{n-1}\overline{c}_{n-2} \end{cases}$$

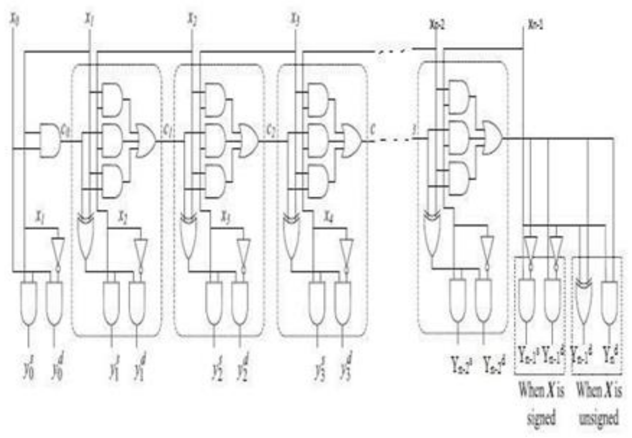


Figure1: Schematic circuit for CSD conversion of n-bit binary data

III. PROPOSED LOGICS

Two adjoining carries share the same input variable. This means that the same input Xi is used in the generation of both carries Ci-1 and Ci. This characteristic permits the algebraical expressions of carry generation to be simplified so as to get economical circuits. The carry look-ahead principle is one in every of the foremost wide used strategies to implement quick adders by introducing some reasonably correspondence so as to cut back the crucial path of the circuit. Carry generate and carry propagate area unit used that reduces the computation of logic for carry production. Depending on inputs it will directly decide either to propagate the previous carry or to generate the new value. Figure2 shows the Schematic of CLA.

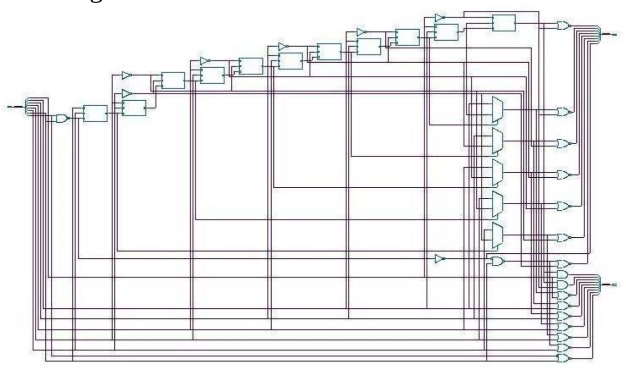


Figure2: Schematic circuit for carry generation using CLA

Next method is to use Kogge-stone Adder logic for generation of carry.Figure3 shows the Schematic of Kogge-stone adder. Two new ways are proposed here named as new_15 and new_15a. The logic used in new_15 is instead of Exclusive OR operation for carry computation, only OR operation is performed which gives somewhat better results in comparison to conventional designs. Figure4 shows the schematic of new_15. In next method instead of using Carry to be forwarded for next level carry generation, we perform Exclusive-OR operation between present and next input bit. In this case bit logic zero is appended to LSB of input. Figure 5 shows the RTL Schematic of new_15.

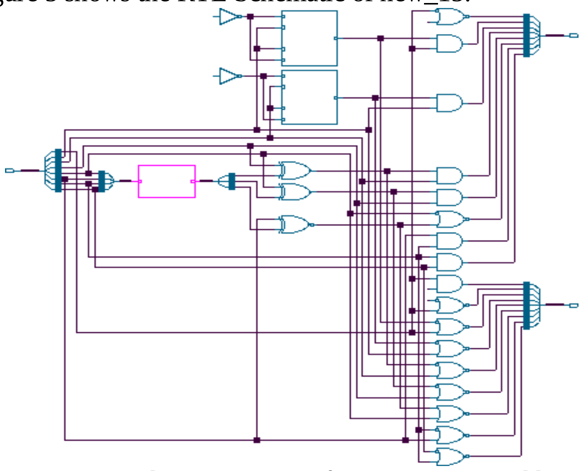


Figure 3: Schematic Circuit for Kogge-stone adder

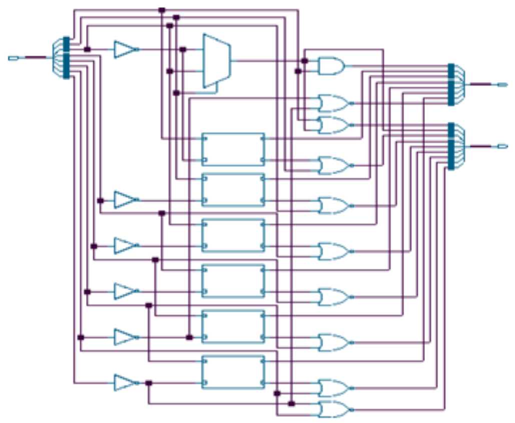


Figure 4: Schematic of new_15a

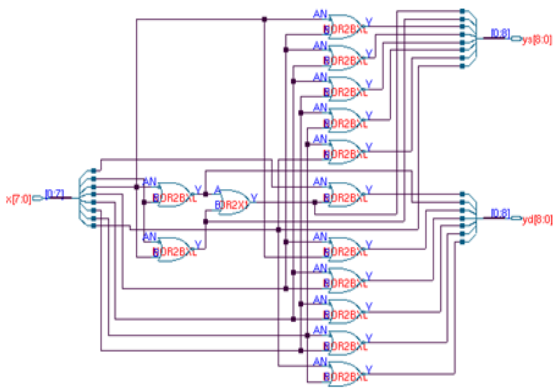


Figure 5: Schematic of new_15

The following are the expressions for new_15 circuit.

xorxor_forci(c[i], x[i],1'b0);

xor xor_forci+1(c[i+1],x[i],x[i+1]);

andand_forysi(ys[i],c[i],x[i]);

andand_forydi(yd[i],c[i],~x[i]);

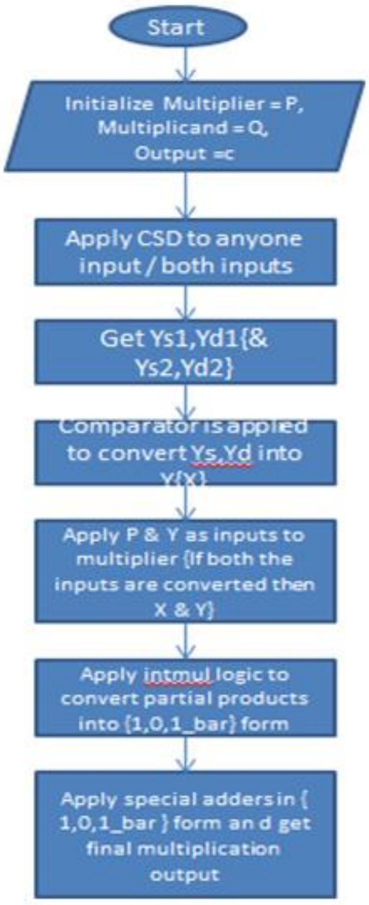
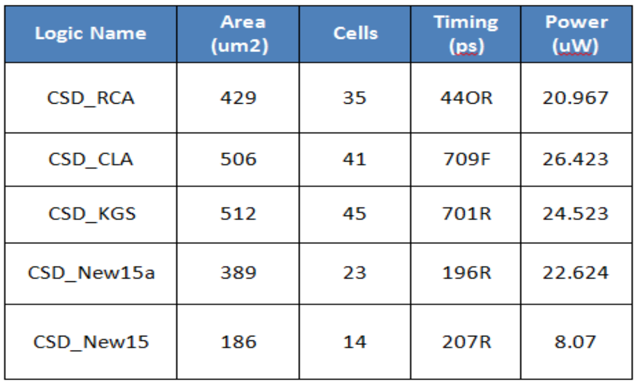

Start
Initialize Multiplier = P, Multiplicand = Q, Output =c
Apply CSD to anyone input / both inputs
Get Ys1,Yd1{& Ys2,Yd2}
Comparator is applied to convert Ys,Yd into Y{X}
Apply P & Y as inputs to multiplier {If both the inputs are converted then X & Y}
Apply intmul logic to convert partial products into {1,0,1_bar} form
Apply special adders in { 1,0,1_bar } form an d get final multiplication output

Figure 6: Flowchart for Applying CSD to either one or both inputs of a multiplier

IV. SIMULATION AND SYNTHESIS RESULTS

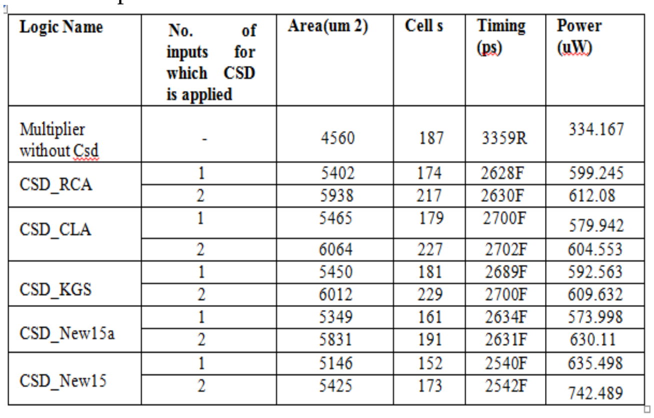
A Verilog HDL code for Wallace tree is written and simulated in Xilinx ISE suite Design 14.2 Version. This code is for a regular multiplier without applying the CSD. Table 3 shows the results of Wallace multiplier. Now the CSD is applied to Wallace tree multiplier with combinations of different adder logics for CSD conversion.(These adders are applied only to convert multiplicand/ multiplier, not for adding partial products). Following is a flowchart which shows how the canonical signed digit conversion is applied to the multiplier logic.



Logic Name	Area (um2)	Cells	Timing (ps)	Power (uW)
CSD_RCA	429	35	44OR	20.967
CSD_CLA	506	41	709F	26.423
CSD_KGS	512	45	701R	24.523
CSD_New15a	389	23	196R	22.624
CSD_New15	186	14	207R	8.07

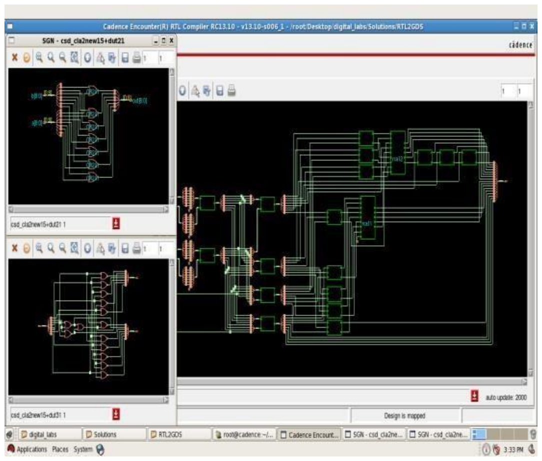
Figure 7: Comparison results of CSD using different adders

Table 3 shows comparison of multipliers with different combinations of adders. We have separately showed the area in um2, power consumption in uW, delay in & multiplier. These results are taken from Cadence tool- RTL Compiler and Nc- simulator. The comparison shows that new_15 logic occupies less area than other Ps and number of cells for CSD applied for only multiplicand and for both multiplicand logics with less number of cells and it gives less delay even in comparison to regular Wallace multiplier. With respect to power consumption CSD with CLA combination is best. But for Area and Timing new_15 gives better comparative results.



Logic Name	No. of inputs for which CSD is applied	Area(um 2)	Cell s	Timing (ps)	Power (uW)
Multiplier without Csd	-	4560	187	3359R	334.167
CSD_RCA	1	5402	174	2628F	599.245
	2	5938	217	2630F	612.08
CSD_CLA	1	5465	179	2700F	579.942
	2	6064	227	2702F	604.553
CSD_KGS	1	5450	181	2689F	592.563
	2	6012	229	2700F	609.632
CSD_New15a	1	5349	161	2634F	573.998
	2	5831	191	2631F	630.11
CSD_New15	1	5146	152	2540F	635.498
	2	5425	173	2542F	742.489

Figure 8: Comparison results of Multiplier using CSD with different adders



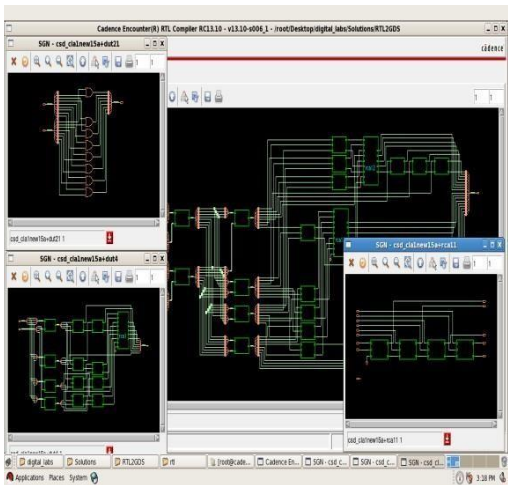


Figure 9 & 10 schematics of Multipliers using New_15 and new_15a schemes

V. CONCLUSION AND FUTURE SCOPE

The conversion of a binary number into its canonical signed digit (CSD) illustration may be expeditiously enforced exploitation new_15 logic. Simulation and Synthesis performed for projected circuit's shows potency in terms of space, delay compared to the standard approach and different adder circuits. The new technique given during this work is advantageous to supply high speed with low space value for DSP applications. We will additionally extend the project by exploitation state of state space transformation techniques that might have outer performance for high-speed.

VI. REFERENCES

[1]. D. L. Maskell, " variety of economical Multiplierless FIR filters," IET Circuits Device System, vol.1, no. 2, pp. 175-180, May 2007.

[2]. M. D. Cilletti, Advance Digital style with Verilog high-density lipoprotein, letter of the alphabet learning, initial edition, 2003.

[3]. A. Nandi, A. K. Saxena and S. Dasgupta, "Design and Analysis of Analog Performance of Dual-k Spacer based Underlap N/P- FinFET at 12nm Gate Length," IEEE Trans. on lepton Devices, vol. 60, no. 5, pp. 1529-1535, May 2013.

[4]. R. Mahesh and A. P. Vinod,"A New Common Subexpression Elimination formula for Realizing inferiority Higher Order Digital Filters," IEEE Trans. On computer motor-assisted sort of Integreted Circuits and Sytems, vol. 27, no. 2, pp. 217-229, Feb. 2008.

[5]. L. Wu, Y. Cui and J. Huang, "Design associate degreed Implemenation of an Optimized FIR Filter for IF GPS Signal machine," IEEE conf. on electronics and physical science, pp. 25-28, Sept. 2010.

[6]. J. Proakis and D. Manolakis, Digital Signal method, fourth edition, 2008.

[7]. S. F. Hsiao and J. H. Z. Jian, " Low worth FIR Filter designs supported reliably Rounded Truncated Multiple Constant Multiplications," IEEE Trans. on Circuits and Systems-II: Expression Briefs, vol. 60, no. 5, pp. 287-291, May 2013.

.

[8]. J. Um and T.Kim, "An optimum Allocation of Carry Save Adders in Arithmetic Circuits, "IIEEE Trans. on pc, vol. 50, no. 3, pp. 215- 230, Mar. 2001.

[9]. Mathworks, User Guide Filter style tool case, Version-2, 200seven.

[10]. L. Litwin, " FIR and IIR Digital Filters," IEEE Potentials, vol. 19, no. 4, pp. 28–31, Oct. i2000.

[11]. L. Aksoy, E. Costa, P. Flores and J. Monteiro, " precise and Approximate Algorithms for the development of house and Delay in Multiple Constant Multiplications", IEEE Trans. on Comp. motor- assisted form of Integreted Circuits and Sytems, vol. 27, no. 6, pp. 1013-1026, iJune 2008.

[12]. K. Priya and R. Mehra, " FPGA primarily based value economical FIR Filter exploitation Factored CSD technique ," International iJournal of Recent Technology and Engineering,vol. 6, issue 6, pp. 130-134, Jan. 2013

[13]. A.D. Booth, "A Signed Binary Multiplication Technique," Quarterly J.Mechanics and maths., vol. 4, pp. 236–240, 1951.

[14]. G.W. Reitwiesner, "Binary arithmetic," Advances in Computers, vol. 1,pp. 261–265,1960.