AREA EFFICIENT HIGH BIT RATE SERIAL-SERIAL MULTIPLIER WITH 1'S ASYNCHRONOUS COUNTERS

Ch.Nirosha[#], P.Sunitha[*]

ECE Department Pragati Engineering College, Surampalem, India

Abstract

In this Paper, a technique for Serial Multiplication completes the partial product Formation in n cycles is presented. The Proposed technique effectively forms partial product matrix in just 'n ' cycles for an nxn multiplication instead of at least '2n' cycles in traditional Serial-Serial Multiplier. Here Serial- Serial algorithm is used. The newly developed design is capable of processing input data (GBs) without buffering and with reduced number of computational cycles .This multiplication of partial products by considering two series inputs among which one is starting from LSB and other from MSB. The architecture consists of a series of asynchronous 1's counters instead of 5:3 Counters, then critical path is limited to AND gate & D flip- flop. The proposed multiplier consists of a Serial-Serial data accumulator and CSA, is designed to carry out both signed and unsigned multiplication. We can able to extend area reduction of proposed multiplier by using CLA instead of RCA in CSA.

Keywords: Serial-Serial multiplier; Serial-Serial Algorithm; Carry Save Adder; Carry Look Ahead Adder; Ripple Carry Adder; Asynchronous Counters.

I. Introduction

Serial-serial multiplication techniques have been in use for many years . The proposed a structure based on 1's counters, which computed the N bits of the product of an NxN multiplication in N bits clock cycles, using N processing cells. Multipliers are the fundamental and essential building blocks of VLSI systems. The design and implementation approaches of multipliers contribute substantially to the area, speed and power consumption of computational intensive VLSI systems.Hardware implementation of a multiplication operation consists of three stages, specifically the generation of partial products (PPs), the reduction of partial products (PPs), and the final carry propagation addition. The partial products can be generated either in parallel or serially, depending on the target application and the availability of input data. The partial products are reduced by carry-save adders (CSA) using an array or tree structure. Carry propagation addition is inevitable when the number of partial products is reduced to two rows. This final adder can be a simple ripple carry adder (RCA) for low power or a carry look-ahead adder (CLA) for high speed. As the height of PP tree increases linearly with the word length of the multiplier, it aggravates the area, delay and power dissipation of the two subsequent stages.

Therefore, it is highly desirable to reduce the number of partial products before the CSA Stage. In the proposed method the partial product formation is revamped using an algorithm named as serial-serial algorithm which is explained in the following sections. The generated partial products are passed to a group of asynchronous 1's counters for accumulation. The counters will count the number of ones in the partial products which is used for addition. In the following sections an approach to the design of serial multiplier that is capable of processing input data without input buffering and with reduced total number of computational cycles is proposed.

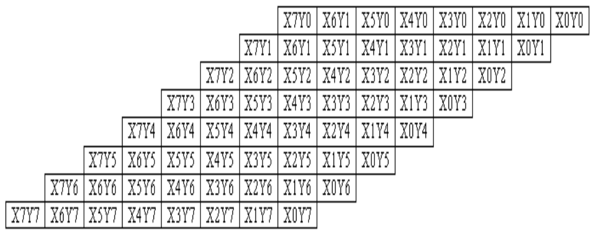
II. REVIEW OF SERIAL MULTIPLIERS

In a serial-serial multiplier both the operands are loaded in a bit-serial fashion, reducing the data input pads to two serial multipliers are popular for their low area and power. Bit-serial processing can result in efficient communications, both within and between VLSI chips, because of the reduced number of interconnections required. Serial multiplier designs which are particularly suitable for applications where input data are sequentially presented .The operating speeds are determined mainly by the Propagation delays along the critical path within the processing elements. A major advantage offered by bit serial processors is when the operands are available only one bit at a time, the processing speed can be improved using bit-serial arithmetic elements. The structures that use this approach can achieve moderate speeds with comparatively small area. The existing method of multiplication is the CSAS (carry save add shift) multiplier .CSAS architecture consists of a FA, D FF and an AND gate in the critical path for unsigned multiplication and an additional EXOR gate for signed multiplication. This architecture takes 2n cycles for the partial product formation.

Parallel multipliers are popular as the size is less critical due to technology scaling. However due to the emerging development of the on-chip serial-link bus architectures serial-serial multipliers could find their potential roles in the new generation of SoCs and FPGAs. In the following sections, approach to the design of serial multiplier that is capable of processing input data at Gb/s without input buffering and with reduced total number of computational cycles is proposed.

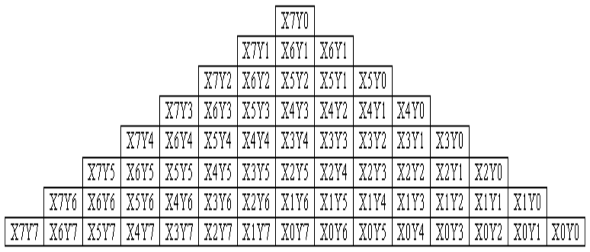
III. CONCEPT OF SERIAL ALGORITHM:

The paper addresses an algorithm named serial algorithm that reduces the computation time of the partial products such that they can be formed in just n cycles for an n x n multiplier. According to this algorithm the partial product row and column structure is revamped. The figures (1) and (2) shows the partial product formation of the conventional and the proposed multiplier.



							X7Y0	X6Y1	X5Y0	X4Y0	X3Y0	X2Y0	X1Y0	X0Y0
						X7Y1	X6Y1	X5Y1	X4Y1	X3Y1	X2Y1	X1Y1	X0Y1	
					X7Y2	X6Y2	X5Y2	X4Y2	X3Y2	X2Y2	X1Y2	X0Y2		
				X7Y3	X6Y3	X5Y3	X4Y3	X3Y3	X2Y3	X1Y3	X0Y3			
			X7Y4	X6Y4	X5Y4	X4Y4	X3Y4	X2Y4	X1Y4	X0Y4				
		X7Y5	X6Y5	X5Y5	X4Y5	X3Y5	X2Y5	X1Y5	X0Y5					
	X7Y6	X6Y6	X5Y6	X4Y6	X3Y6	X2Y6	X1Y6	X0Y6						
X7Y7	X6Y7	X5Y7	X4Y7	X3Y7	X2Y7	X1Y7	X0Y7							

Figure 1: Conventional PP Formation



							X7Y0							
						X7Y1	X6Y1	X6Y1						
					X7Y2	X6Y2	X5Y2	X5Y1	X5Y0					
				X7Y3	X6Y3	X5Y3	X4Y3	X4Y2	X4Y1	X4Y0				
			X7Y4	X6Y4	X5Y4	X4Y4	X3Y4	X3Y3	X3Y2	X3Y1	X3Y0			
		X7Y5	X6Y5	X5Y5	X4Y5	X3Y5	X2Y5	X2Y4	X2Y3	X2Y2	X2Y1	X2Y0		
	X7Y6	X6Y6	X5Y6	X4Y6	X3Y6	X2Y6	X1Y6	X1Y5	X1Y4	X1Y3	X1Y2	X1Y1	X1Y0	
X7Y7	X6Y7	X5Y7	X4Y7	X3Y7	X2Y7	X1Y7	X0Y7	X0Y6	X0Y5	X0Y4	X0Y3	X0Y2	X0Y1	X0Y0

Figure 2: Proposed PP Formation

The partial products so formed are counted column wise for the number of ones.

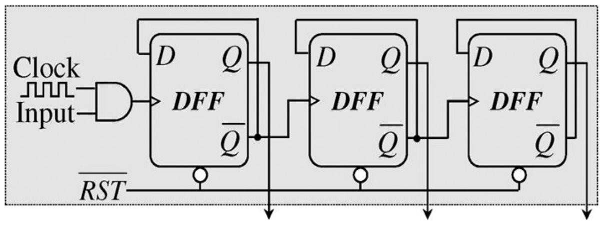


Figure 3: Hardware architecture of a 3-bit 1's counter.

The counters corresponding to the columns that have a 1 input are incremented. The counters can be clocked at high frequency and all the operands will be accumulated at the end of the clock. The final outputs of the counters need to be further reduced to only two rows of partial products by a CSA tree.

IV. CONCEPT OF SERIAL ACCUMULATOR:

Accumulation is an integral part of serial multiplier design. A typical accumulator is simply an adder that successively adds the current input with the value stored in its internal register.

Generally, the adder can be a simple RCA but the speed of accumulation is limited by the carry propagation chain. The accumulation can be speed up by using a CSA with two registers to store the intermediate sum and carry vectors, but a more complex fast vector merged adder is needed to add the final outputs of these registers. In either case, the basic functional unit is an FA cell. A new approach to serial accumulation of data by using asynchronous counters is suggested here which essentially count the number of 1's in respective input sequences (columns).

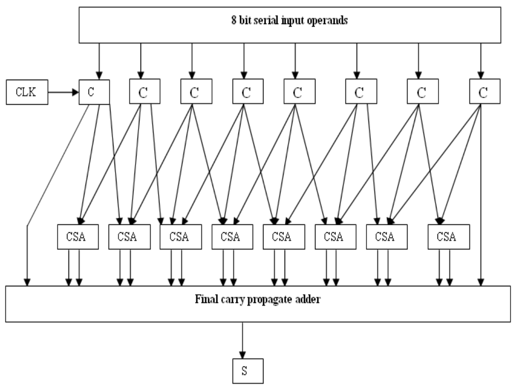


Figure 4: Architecture of Accumulator

For an 8 bit operands multiplication the counter counts the number of ones in the columns and these count values are positioned and arranged for addition.

V. PROPOSED SERIAL-SERIAL MULTIPLIER:

This section describes an unsigned multiplier in which the operands are fed serially one starting from LSB and the other from MSB Using this feeding sequence and the proposed counter based accumulation, it takes only n cycles to complete the entire partial product generation for nxn multiplication.

The product of two unsigned numbers X and Y can be written as

$$P = X \cdot Y = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} x_i y_j 2^{i+j} \qquad (1)$$

Where xi and yj are the ith and jth bits of X and Y with bit 0 being the LSB.

Reversing the sequence of index I and rearranging the above equation can be written as

$$P = \sum_{r=0}^{n-1} PP_r \qquad (2)$$

Where

$$PP_r = PP_r^L + PP_r^C + PP_r^R$$

$$PP_r^L = \begin{cases} 0, & r=0 \\ \sum_{k=0}^{r-1} x_{n-k-1} \cdot y_r \cdot 2^{n+r-k-1}, & r=1,2,\ldots,n-1 \end{cases}$$

$$PP_r^C = x_{n-r-1} \cdot y_r \cdot 2^{n-1}, \quad r=0,1,\ldots,n-1$$

$$PP_r^R = \begin{cases} 0, & r=0 \\ \sum_{k=0}^{r-1} x_{n-r-1} \cdot y_{r-k-1} \cdot 2^{n-k-2}, & r=1,2,\ldots,n-1. \end{cases}$$

The partial product row PPr can be generated in rth cycle If X is fed from MSB (bit n-1) first and Y is fed from LSB first (bit 0), then in the rth cycle ,PPrC is a partial product bit generated by the current input bits xn-r-1 and yr,PPrL are partial product bits of the current input bit, yr and each of the preceding input bits of X, ie xn-k-1, for k=0,1,….r-1.and PPrR are the partial product bits of the input bit xn-r-1,and each of the preceding input bits of Y, i.e Yr-k-1 for k = 0,1,2,….r-1. By appropriately sequencing the input bits of X and Y in to a shift register, one PP (PPr) in each cycle can be generated. As a result P can be obtained in n cycles.

The figure (5) illustrates the PP generation of an 8X8 multiplier for unsigned numbers using the serial algorithm.

According to the algorithm, the partial products formation sequence is as shown in the figure (2) like a pyramid. From the figure it is clear that in the first cycle of operation the pp X7Y0 is formed. In the next cycle x7 is shifted one position to the left and y0 is shifted one position to the right. At the same time the x6 and y1 is also generated. In this way at the next cycle x7 and x6 will be shifted again one position to the left and y0 and y1 one position to the right. Similarly the process continues till the last row of partial products is being formed. Once the partial products are formed they are accumulated using the asynchronous one's counter explained in the last section.

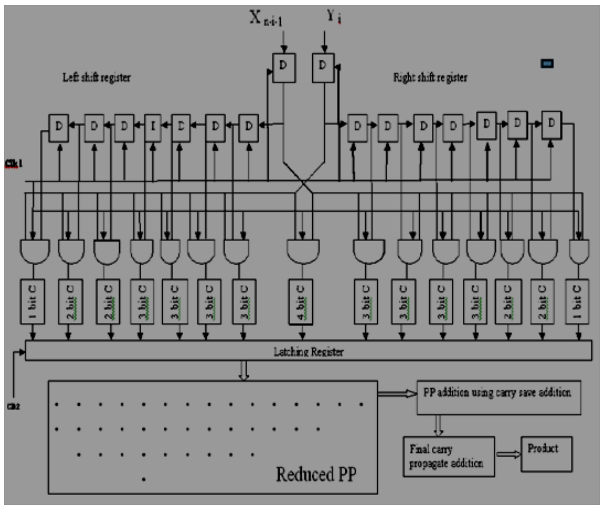


Figure 5: Proposed Architecture for 8x8 Serial Serial Unsigned Multiplications

VI. EXTENDING THE LOGIC TO SIGNED NUMBERS

The above said architecture can be modified such that the same could be used for the multiplication of signed numbers. The signed multiplication is carried out with Baugh Wooley algorithm. By combining the Baugh Wooley and the serial algorithm, the architecture could be modified to perform signed multiplication. Using Baugh Wooley algorithm the equation for the multiplication of two signed numbers in two's complement form can be written as

$$\begin{aligned} P' = X'Y' &= \sum_{i=0}^{n-2} \sum_{j=0}^{n-2} x_i y_i 2^{i+j} \\ &+ 2^{n-1} \left(\sum_{i=0}^{n-2} \overline{x_i y_{n-1}} 2^i + \sum_{j=0}^{n-2} \overline{x_{n-1} y_i} 2^j \right) \\ &+ x_{n-1} y_{n-1} 2^{2n-2} + 2^n - 2^{2n-1}. \end{aligned} \qquad (3)$$

Using the proposed architecture the multiplication of the signed numbers can be written as

$$P = XY = \sum_{i=0}^{n-2}\sum_{j=0}^{n-2} x_i y_i 2^{i+j} + 2^{n-1}\left(\sum_{i=0}^{n-2} x_i y_{n-1} 2^i + \sum_{j=0}^{n-2} x_{n-1} y_i 2^j\right) + x_{n-1} y_{n-1} 2^{2n-2}. \quad (4)$$

The difference between 3 and 4 can be written as

$$\begin{aligned}\psi &= P' - P \\ &= 2^{n-1}\left(\sum_{i=0}^{n-2} \overline{x_i y_{n-1}} 2^i + \sum_{j=0}^{n-2} \overline{x_{n-1} y_i} 2^j\right) \\ &- 2^{n-1}\left(\sum_{i=0}^{n-2} x_i y_{n-1} 2^i + \sum_{j=0}^{n-2} x_{n-1} y_i 2^j\right) \\ &+ 2^n - 2^{2n-1}\end{aligned} \quad (5)$$

The above expression could be simplified as

$$\psi = 2^{n-1}\left[\left(\sum_{i=0}^{n-2}(1 - 2x_i y_{n-1})2^i\right) + \left(\sum_{j=0}^{n-2}(1 - 2x_{n-1} y_j)2^j\right)\right] + 2^n - 2^{2n-1} \quad (6)$$

To extend the unsigned multiplier architecture for signed multiplication without introducing a high over-head, the difference expressed in 6must be simplified. Since $\sum_i^{n-2} 2^i = \sum_j^{n-2} 2^j$,the following summation terms embedded in (6) can be simplified by the closed form expression of a geometric progression

$$2^0 + 2^1 + 2^2 + \ldots + 2^{2n-2} = 2^{2n-1} - 1$$

Therefore

$$2^{n-1}\left(\sum_{i=0}^{n-2} 2^i + \sum_{j=0}^{n-2} 2^j\right) = 2^{2n-1} - 2^n$$

hence Ψ can be written as

$$\psi = -2^n\left(\sum_{i=0}^{n-2} x_i y_{n-1} 2^i + \sum_{j=0}^{n-2} x_{n-1} y_j 2^j\right); \quad (7)$$

The difference Ψ is added to the proposed architecture such that the architecture can be used for signed multiplication. Thus we can write

$$P' = P + \Psi \quad (8)$$

In the proposed PP generation method, Yn-1 arrives only in Cycle n-1. The generation of $\sum_{i=0}^{n-2} x_i y_{n-1} 2^i$ has to be delayed until cycle n-1. The remaining terms $\sum_{j=0}^{n-2} x_{n-1} y_j 2^j$ can be computed during the initial n-1cycles. Hence, the difference can be corrected in the CSA tree. It is trivial that a n- 1-bit shift register, a NAND gate and several FAs are required for adding Ψ.

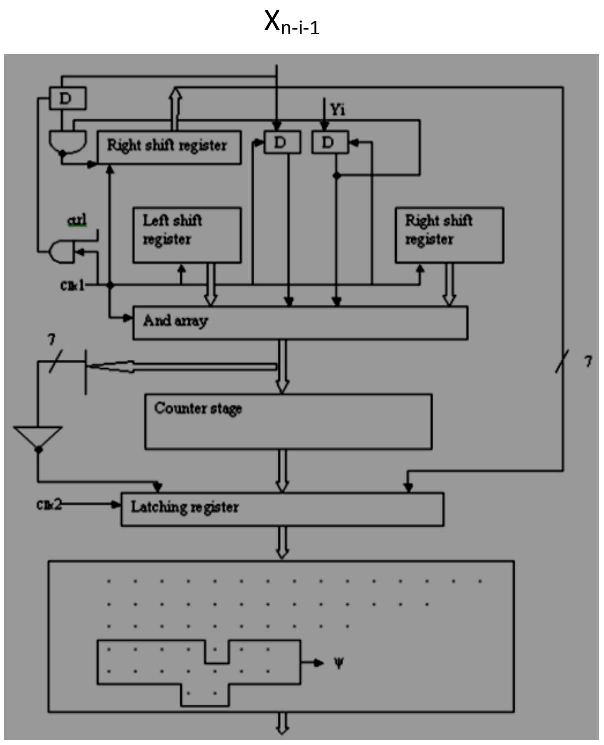


Figure 6: Proposed Architecture for 8x8 Serial Serial Signed Multiplication

The architecture of the proposed 2’s complement serial-serial multiplier is depicted in the figure (6). A control input is required to latch Xn-1 in the first clock cycle to generate $\sum_{j=0}^{n-2} x_{n-1} y_j 2^j$ Serially in n-1 cycles. The bits of Ψ to be added in the CSA tree is shown in the figure.

VII. CARRY PROPAGATION ADDITION USING CLA IN CSA

Look ahead carry algorithm speed up the operation to perform addition, because in this algorithm carry for the next stages is calculated in advance based on the input signals. By using this CLA, the carry propagation time is

reduced by using a tree like circuit to compute the carry rapidly. The CLA exploits the fact that the carry generator by a bit position depends on the 3 inputs to that position.
If X & Y are two inputs then

if X=Y=1 → a carry generated independently of the carry from the previous bit position

X=Y=0 → no carry generated.

X ≠ Y → a carry generated if and only if the previous bit-position generates a carry.

The multiplier architecture's area can be reduced with CLA than RCA used in the carry propagation addition stage of CSA.

VIII. PERFORMANCE COMPARISON AND RESULTS

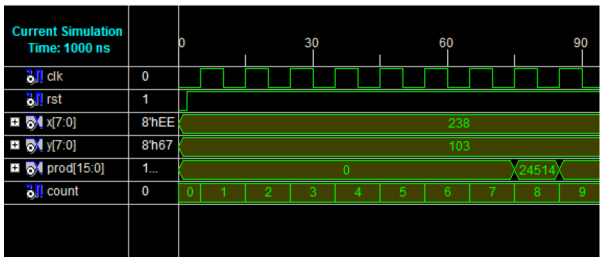


Fig: Simulation results of 8 x 8 unsigned multiplication using RCA.

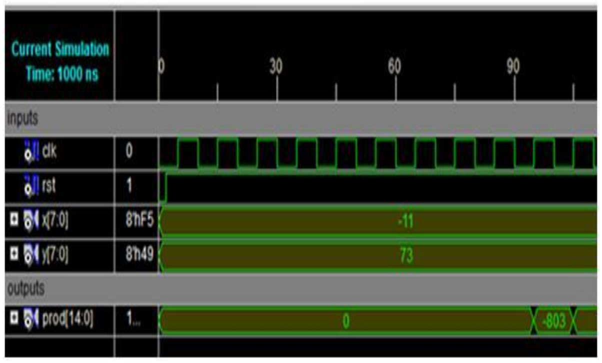


Fig: Simulation results of 8 x 8 signed multiplication using RCA.

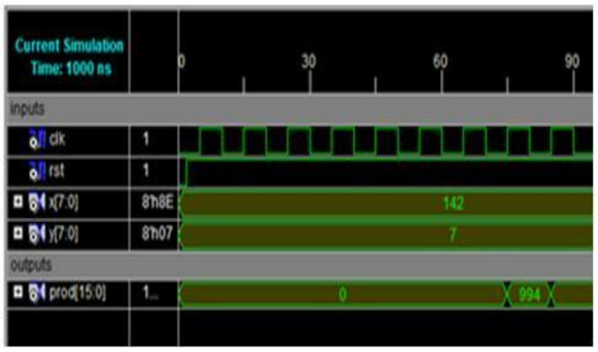


Fig: Simulation results of 8 x 8 unsigned multiplication using CLA.

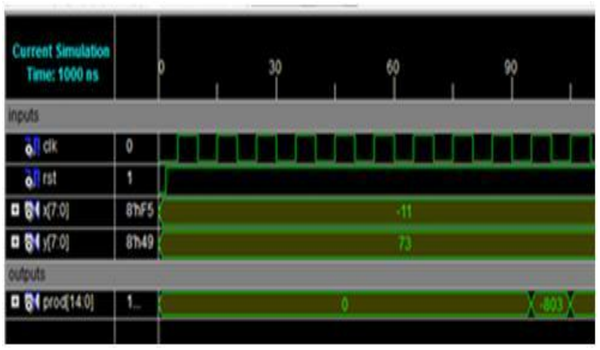


Fig: Simulation results of 8 x 8 signed multiplication using CLA.

In this section the results of the proposed work is compared with the existing carry save adder multiplier using RCA and CLA .The architecture is implemented in VHDL and simulated using XYLINX.

Table 1: Proposed serial-serial multiplication Area comparison

Method	Bit Rate	Operating Mode	Number of Slices	
			Using RCA	Using CLA
Proposed	8 x 8	Un Signed	91	86
Proposed	8 x 8	Signed	166	165

The proposed work can be made area efficient by using CLA for PP addition than RCA. From the synthesis report which is shown in the above table, the no. of slices are reduced by using CLA than RCA.

IX.CONCLUSION:

In this paper a new method of computing serial-serial multiplication is introduced by using low complexity asynchronous counters. By exploiting the relationship among the bits of a partial product matrix it is possible to generate all the rows serially in just n cycles for an n x n multiplication. Employing counters to count no of ones in each column allows the partial products bits to be generated on-the-fly and partially accumulated in place with a critical path delay of only an AND gate and a DFF . The counter-based accumulation reduces the partial product height logarithmically and makes it possible to achieve an effective reduction rate. The proposed method outperforms many serial-serial and serial-parallel multipliers in speed. This approach has clear advantage of low I/O requirement and hence is most suitable for complex SOCs, advanced FPGAs and high speed bit serial applications.

REFERENCES:

1) P. Ienne and M. A Viredaz, “Bit-serial multipliers and squarers,” IEEE Trans. Comput., vol. 43, no. 12, pp. 1445–1450, Dec. 1994.

2) Aggoun, A. Ashur, and M. K. Ibrahim, “Area-time efficient serial-serial multipliers,” in Proc. IEEE Conf. Circuits Syst. (ISCAS), Geneva, Switzerland, 2000, pp. 585–588.

3) A high bit rate serial-serial multiplier [online].

4) R. Gnanasekaran, “On a bit-serial input and bit-serial output multiplier,” IEEE Trans. Comput., vol. C-32, no. 9, pp. 878–880, 1983.

5) O. Nibouche, A. Bouridarie, and M. Nibouche, “New architectures for serial-serial multiplication,” in Proc. IEEE Conf. Circuits Syst. (ISCAS), Sydney, Australia, 2001, vol. 2, pp. 705–708