Eight Bit Serial Triangular Compressor Based Multiplier

Aqib Perwaiz, Shoab A Khan

Abstract- **This paper proposes a novel and area efficient bit serial multiplier architecture in which both the multiplier and multiplicand are processed in real time. The major advantage of proposed multiplier is the bit serial data which results in reduced area and simple circuitry, the use of compressor enables us to get bit serial out put every clock cycle. The proposed architecture is best suited for bit serial communication system. The proposed bit serial multiplier is an integral part of bit serial digital down converter. The design uses a compressor algorithm for partial product addition which removes the dependency of each data bit from its previous one by using a triangular compressor. The complexity of our algorithm is 2n+1.** [1]

Key Words: digital down converter, digital receiver, direct digital frequency synthesizer, multirate signal processing.

I. INTRODUCTION

Bit serial multiplication techniques are most often used in designing of different systems for reduction in wiring issues to a reasonable level. There are two choices for adopting a multiplication scheme, one is serial-parallel scheme in which one factor is fixed and other enters serially, the other is serial-serial multiplication scheme in which both the factors enter serially, in this paper our focus will be on the serial-serial multiplication scheme that uses a triangular compressor to achieve this multiplication in an efficient manner.

A high throughput two's compliment pipelined but truncated output serial multiplier was presented by R. F. Lyon[1] which was utilizing more resources, another full-precision modular serial multipliers for unsigned numbers were introduced by H. J. Sips [2] and by N. R. Strader and V. T. Rhyne [3]. In a similar paper, R. Gnanasekaran[4] presented the first multiplication scheme for two's complement numbers, It directly takes into account the negative weight of the most significant bit in the two's complement representation. It results in an overcomplicated design which requires the knowledge of the cycle when the sign bit is presented. A very complicated booth recoded multiplication scheme was presented by Rhyne and Strader[5] in which 2k-bit product was produced using k identical cells, in the same design Dadda[6] pointed unnecessary complexity and presented a multiplier based on sign extension. Gnansekaran proposed a hybrid serial/parallel implementation [7]. Denyer and Renshaw[8] considered the design of an nMOS serial multiplier based on a modified Booth's algorithm. The design utilizes multiplier cells described in [9] but does not discuss design procedure and trade-offs in any depth. Kanopoulos has also considered nMOS implementations of serial multipliers as part of the design of a bit serial 3 x 3 matrix/vector multiplier [10] but with a little discussion regarding the derivation of the multiplier and the associated design trade-offs. This paper presents a systematic understanding of bit serial compressor based algorithm multipliers in which both the multiplier and multiplicand are both serial input to the system. Detailed data movement diagrams are presented to provide a thorough understanding of the associated algorithm. Finally, FPGA implementations and cost/performance trade-offs in VLSI are provided so that the designs described here can be used with ease in design projects requiring bit serial multipliers.

II. BIT SERIAL MULTIPLICATION

The multiplication of two eight-bit fixed point [12] numbers is illustrated in Fig. 1. Eight partial products (pp0 to pp7) are generated as a result Of this multiplication as

[1] Manuscript received December 10, 2007. This work was supported Higher Education Commission.
Aqib Perwaiz did his B.S.C Electrical Engineering from National University of Sciences and Technology, Rawalpindi and is doing his PHD at the moment.
(Phone:+923009548042, e-mail: *Aqib2003@hotmail*).
Dr. S. A. Khan did his PhD in Electrical and Computer Engineering from Georgia Institute of Technology; Atlanta, GA. Dr. Khan's areas of specialization are Digital Signal Processing, Digital Design and Communication System.
(Phone:+923315868714, e-mail: *shoab@carepvtltd.com*).

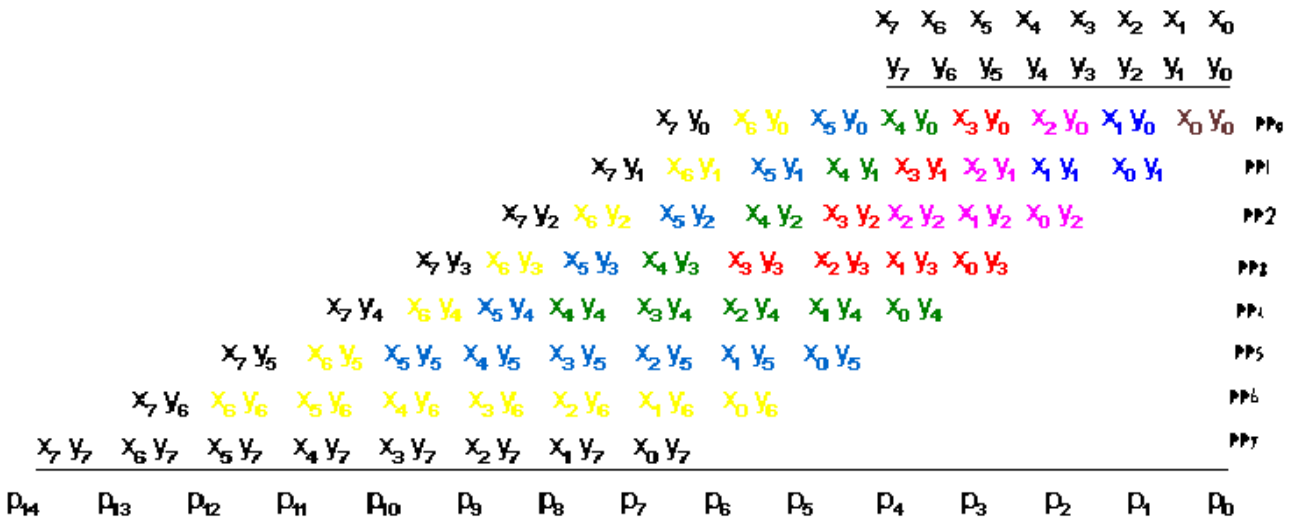

Fig. 1 Multiplication Of Two Eight-Bit Fixed Point Numbers

Cycle 1: maroon Cycle 5: green
Cycle 2: blue Cycle 6: grayish blue
Cycle 3: magenta Cycle 7: yellow
Cycle 4: red Cycle 8: black
Color Code for Fig.1.

shown in Fig.1, the final product is obtained by summing a set of partial products (pp0 through pp7). In most parallel multipliers, all partial products are generated and summed concurrently; however, this mode of operation requires substantial hardware resources. The purpose of using a bit serial multiplier is to perform the multiplication using far fewer resources. Fig. 1 shows the bit serial multiplication, with the arrival of x_0 (LSB of x) and y_0 (LSB of y) a multiplication which is the dot product of both the terms takes place and as a result of dot product the term $x_0\ y_0$ which is the LSB p_0 of final product p is generated along with a carry out, we term it as first stage(cycle 1). In the next clock cycle x_1 and y_1 become available for multiplication, three more terms i.e $x_1\ y_0$, $x_0\ y_1$ and $x_1\ y_1$ are generated as a result of bit by bit dot product.

These three terms along with the carry out of first stage are input to triangular compressor which results in bit p_1 along with carry out. This process goes on and on till the final stage i.e cycle 8 is achieved. In each stage the number of terms grow following the general formula 2n+1, where n is the cycle/ stage number. With the arrival of x_0 and y_0 serially an ***And*** operation takes place and we get the term p_0 which is the LSB. With the arrival of x_1 and y_1 we get three more terms $x_1\ y_0$, $x_0\ y_1$ and $x_1\ y_1$ which are send to the triangular compressor whose functioning is shown in dot notation illustrated in Fig. 2. We see that as soon as the cycle two terms enter the triangular compressor we get p_1 in the very next cycle and rest of the two terms in Fig.2 are transferred to the cycle 3. Now with the arrival of x_2 and y_2 we get five additional terms $x_2\ y_0$, $x_2\ y_1$, $x_0\ y_2$, $x_1\ y_2$ and $x_2\ y_2$.

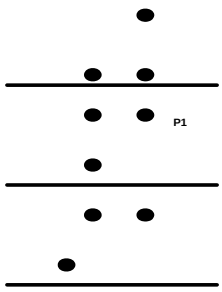

Fig. 2 Cycle 1 Dot Notation Compression

Fig. 4 shows the input to triangular compression where we see that the term p_2 is immediately available after the cycle 2, here the innovative design is that in any case we do not get more than three terms for addition and our sum and carry both are essentially one bit by using this triangular compression technique, this saves the problems in handing a two bit carry forward.

As the bits keeps on arriving serially we keep getting bit wise final product starting from LSB, the complexity of this algorithm is O(n) . Fig. 4 shows the detail working of serial multiplication algorithm as how the terms are generated in

each cycle, as the terms generation follow a triangular shape we have termed it as a **triangular compressor**.

III. BIT SERIAL MULTIPLICATION ALGORITHM

In this part a bit serial multiplication algorithm is proposed which will help us developing the complete architecture. The proposed algorithm is an innovation and can be deployed in any bit serial architecture requiring multiplication of two Serial bit streams.

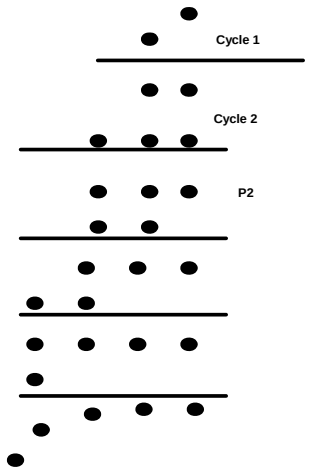

Fig. 3 Cycle 2 Dot Notation Compression

Consider the bit serial multiplication of two N bit numbers x and y to result in product p as described by the algorithm.

Algorithm

INPUT: x, y

OUTPUT: p

INITIALIZE: x_i and y_i =0 for i>W-1

$c_{i,j}$ and $s_{i,j}$ =0 for all i, j

Terms generation
begin
for i=0 to W-1
begin
for j=0 to W-1
begin

$x_i \,\&\, y_i + c_{i,j-i} + s_{i-1,j+1} = 2c_{i,j} + s_{i,j}$;

end

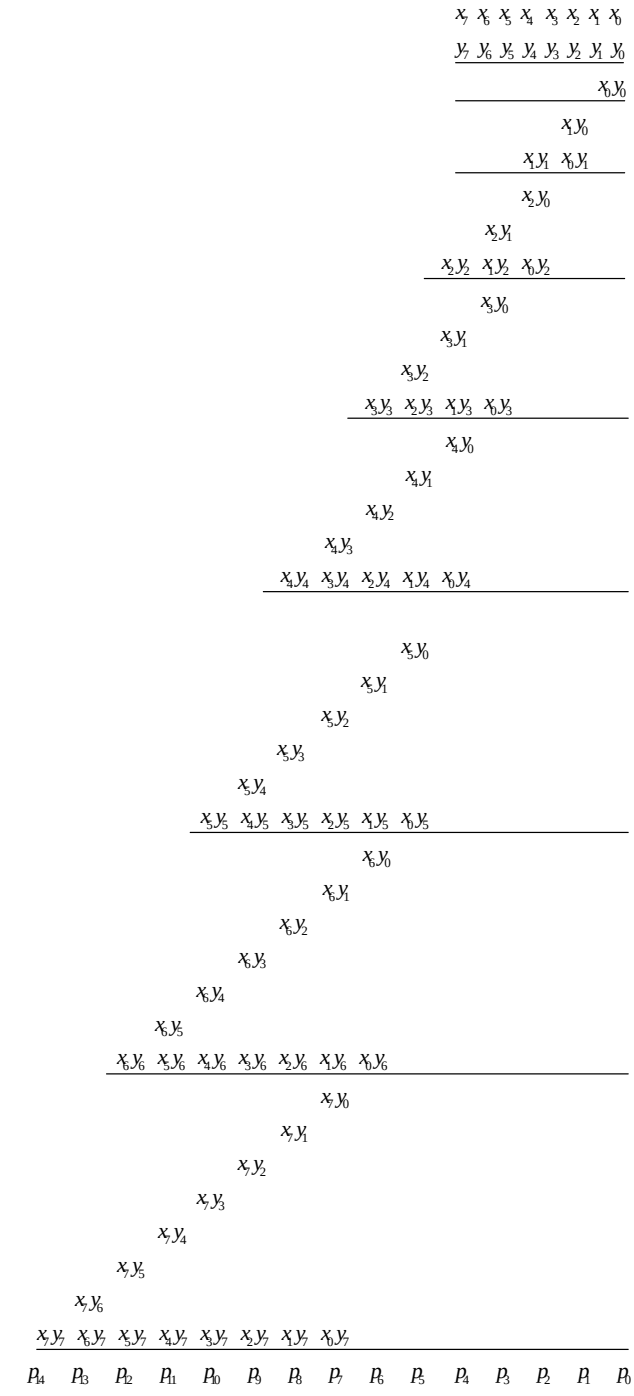

Fig. 4 Serial Multiplication Input To Triangular Compressor.

$p_{i=} s_{i,0}$

for i = W to 2W-1

$p_{i=} s_{W-1,i-W+1}$

For triangular compression
begin
for i=0 to W-1
begin
for j=0 to W-1
begin

{C[i+1],p[i-1]}<=cycle[i][j]+cycle[i+1][j]+p[i];

end.

The proposed methodology involves bit by bit processing resulting in one output of final product bit every cycle.

IV. DESIGN EXAMPLE OF BIT STREAM MULTIPLIER

An example of designed multiplier is discussed for easy understanding, here both the serial inputs x and y are four bits each for simplicity. Let

$x = 0101$

$y = 1111$

The multiplication is shown as per Fig.5a.

```
          0101 x
          1111 y
--------------------
          0101pp1
         0101  pp2
        0101   pp3
       0101    pp4
--------------------
       1001011 p
```

Fig. 5a Multiplication of Two numbers

Now we will see how this multiplication will be performed as per our Algorithm.

As soon as **'1'** LSB of x and **'1'** LSB of y as indicated in Fig.5b become available the triangular compressor after necessary compression returns **'1'** which is basically the LSB of final product p, now with the next cycle the second bit of x and y are available and triangular compressor returns **1** along with no carry forward as indicated in Fig.5c. With the third cycle the third bit of x and y are available and the triangular compressor ' **0'** which is the third of final product p along with a carry forward **100** as shown in Fig.5d.

```
  1
  1
_____
  1  P1
```

Fig.5b Bit Serial of First bit (LSB) of X and Y

```
    01
    11
_______
    0
    1
  0
_______
    1P2
  0
```

Fig.5c Bit Serial of Second bit of X and Y

```
      101
      111
_________
    1 1
  1 0 1
_________
    1 1 0 P3
  0 1
_________
    1 0
  0 1
_________
    0 0
1
```

Fig.5d Bit Serial of Third bit of X and Y

```
        0101
        1111
_____________
    1 0 0
_____________
    0 0 0
0 1 0 1
_____________
    0 0 1 P4
1 0 0
_____________
1 0 0
```

Fig.5e Bit Serial of Fourth bit (MSB) of X and Y

The carry forward from the last stage and the new generated partial products are available for the triangular compressor and after the triangular compression we have **1001** as shown in Fig.5e , these bits are available at the output in a serial manner thus completing our final product p **'1001011'.**

V. Architecture

Our bit serial triangular compressor based multiplier uses bit wise adder, triangular compressor and a tracker to keep track of cycles. The output is serial and is started soon after the input is received from LSB to MSB. Since out input is 8 bit so we truncate first 8 bits starting from our LSB to keep our multiplication result to 8 bits only how ever we loose some precision to a max of $2^7 = 128$ which is not an issue at all . As per Fig.4 it takes as many cycles as the no. of input serial bits for multiplication, in our case it take about 8 cycles for complete terms generation resulting in availability of LSB soon after the execution of cycle 1 and so on.

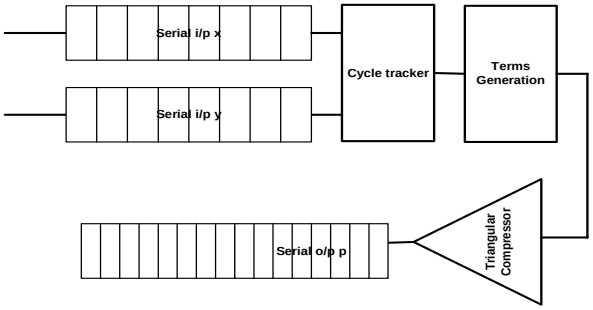

Fig. 6 Bit Serial Compressor Based Multiplication Architecture

VI. IMPLEMENTATION AND RESULTS

In order to verify the efficiency of this proposed bit serial multiplier based on triangular compressor a conventional bit stream multiplier based on conventional (4,2) adder[11] are implemented on FPGA's of Xilinx and Altera. Table 1 and 2 present the implementation result for conventional and proposed design using the above mentioned devices respectively. Comparing with the (4,2) adder based multiplier the proposed design showed very amazing results about 38% of LUT reduction, 30% flip flop reduction and 25 % more clock frequency which simply means our proposed design is more efficient.

Table 1
Implementation Results on Xilinx

	Conventional	Proposed
Vendor	Xilinx	Xilinx
Family	Virtex5	Virtex5
Device	XC5VLX30	XC5VLX30
Number of LUTs **Number of FFs**	13 12	9 8
Frequency(Mhz)	454MHz	565MHz

Table 2
Implementation Results on Altera

	Conventional	Proposed
Vendor	Altera	Altera
Family	Stratix III	Stratix III
Device	EP3SE50	EP3SE50
Number of ALUTs **Number of FFs**	11 12	8 8
Frequency(Mhz)	656 MHz	840MHz

VII. Conclusion

The simplicity efficiency and resource utilization of proposed bit serial compressor based multiplier design is apparent from its implementation results. The proposed architecture definitely has an edge over the existing bit serial multipliers using different addition techniques and can be used in the designing of efficient Adaptive filters or can result in an efficient system design in which serial multiplication is involved.

VIII. References

[I] R. F. Lyon, "Two's complement pipeline multipliers," *IEEE Trans.* commun., vol. COM-24, no. 4, pp. 418-425, Apr. 1976.
[2] H. J. Sips, "Comments on 'An *O(n)* parallel multiplier with bit sequential input and output,'" *IEEE Trans. Comput.*, vol. C-31, no. 4, pp. 325-327, Apr. 1982.
[3] N. R. Strader and V. T. Rhyne, "A canonical bit-sequential multiplier," *IEEE Trans. Comput.*, vol. C-31, no. 8, pp. 791-795, Aug. 1982.
[4] R. Gnanasekaran, "On a bit-serial input and bit-serial output multiplier," *IEEE Trans. Comput.*, vol. C-32, no. 9, pp. 878-880, Sept. 1983.
[5] T. Rhyne and N. R. Strader, 11, "A signed bit-sequential multiplier," *IEEE Trans. Comput.*, vol. C-35. no. 10, pp. 896901, Oct. 1986.
[6] L. Dadda, "On serial-input multipliers for two's complement numbers," *IEEE Trans. Comput.*, vol. 38. no. 9, pp. 1341-1345, Sept. 1989.
[7] "A fast serial-parallel binary multiplier," *IEEE Trans. Comput.*, vol. C-34, no. 8, pp. 741-744, 1985.
[8] P. Denyer and D. Renshaw, *VLSI Signal Processing: A Bit-Serial Approach*, Addison-Wesley, 1985.
[9] J. Newkirk and R. Mathews, *The VLSI Designer's Library*. Addison-Wesley, 1983
[10] N. Kanopoulos, "A bit-serial architecture for digital signal processing," *IEEE Trans. Circuits* Sysf., vol. CAS-32, no. 3, pp. 289-291, 1985.
[11] C.W.Ng, N.Wong and T.S Ng "Efficient FPGA implementation of bit stream multipliers" Electronics letter online no: 20070293, department of Electrical and Electronic Engineering, The University on Hong Kong 26 April 2007.
[12] Woon-Seng Gan, Sen M. Kuo ,"Teaching DSP Software Development: From Design to Fixed-Point Implementations" IEEE Transactions On Education, Vol. 49, No. 1, February 2006.