ISSN:0975-9646

An Analysis of Multipliers in a New Binary System

R.K. Dubey & Anamika Pathak

Department of Electronics and Communication Engineering ,
Swami Vivekanand University,
Sagar (M.P.) India 470228

Abstract:Bit-sequential multiplier is being studied in a (+1, -1) binary representation. The maximum length of multiplier for a fix point numbers consists of n module. Each module is a five bit adder which contains 5 inputs and 3 outputs. It also contains an additional pin at the input as well as at the output. Computation time for multiplication is of the order of O(n). Hardware realization of module is being discussed. In this design the input generated bit sequentially and the output is also generated bit sequentially. The model is being compared with the bit sequential multiplier in conventional binary system and the merits and demerits are described in detail.

Index Terms: Add-shift multiplier, (+1, -1) Representation, Carry-save addition, unified way.

1. INTRODUCTION:

In digital signal processing design of fast efficient multipliers is currently an interesting area of research. Multipliers have been extensively studied by various authors [1-5] for VLSI design. Many scientific and engineering problems, such as inversion of materials, solution of line or equation, computation of eigen value, Fast Fourier Transform and Discrete Fourier Transform and modular arithmetic etc. require large number of multiplications. For computer hardware designers, it is most important to know current research concerning the multiplier for LSI technology.

The conventional add-shift technique of multiplication take time $O(n^2)$ with n being the maximum lengths of multiplier and multiplicand. This time can be reduced to $O(n^{\log\text{-}n})$ using carry look ahead technique for addition but it complicates the hardware design.

Bit serial arithmetic is often used in parallel systems with high connectivity to reduce the wiring to a reasonable level. When multiplication is required, a serial-parallel multiplier is a typical choice but this scheme is not always possible. When both the inputs are required to enter serially at the same time then serial multiplier is required. Bit serial input and output multipliers referred to as on line multipliers [1] are potentially attractive in speeding up the execution of arithmetic expression, where multiprocessor arithmetic structure is needed. Bit serial input and output multiplier was presented by Lyon [2]. Full precision modular serial multipliers for unsigned numbers were introduced by Sips [3] and Strader and Ryne [6]. Gnanasekaran and Sunar [7, 18] presented a bit serial input and bit serial output multiplier for unsigned and two's complement numbers. It directly takes into account the negative weight most significant digit in two's complement representation. Chen & Willoner and Hasan [8, 17] presented O (n) parallel multiplier with bit-sequential input and output. This multiplier for n bit operands requires 2n clocks and 2n number of 5 input modules.

L. Dudda (9) discussed the serial multipliers for two's complement numbers. The multiplier includes a linear array of (5, 3) parallel counters and a set of three static registers for internal carries. He showed that equivalent multiplier can be built using two linear arranges of full adders and two carry registers. In a faster circuit an addition carry register is required. Bit-serial multipliers and squares is presented for signed and unsigned numbers by Inne and Viredaz [10]. It produces a full precision result which can be extended to an arbitrarily large number. It is fully modular and it has zero clock cycle latency. It has got one disadvantage that last sub product most be subtracted keeping only first k bits. Large number extra bits are required for sign. It results in over complicated design which requires the knowledge of the cycle when the sign bit is presented.

In digital signal processing, the number representation (+1, -1) is found suitable as discussed by author [11]. It is a unified representation of positive as well as negative numbers. Full addition, filter design, D/A and A/D conversion is being discussed there in detail. The flip-flops, adders and comparators in this system is described in detail by Tiwari and Varma [12, 13]. In this paper our main aim is to study bit serial multipliers in (+1, -1) system. The method of multiplication is based on add-shift-technique where multiplication bits and multiplier bits are entered sequentially. A module of 5 bit adders is employed. The positive and negative numbers are multiplied in a unified manner. Additional bits are required to modify the effect of -1 bit in place of 0 bit. For fast addition, carry save addition is employed.

II. BIT SEQUENTIAL MULTIPLIER

Following [7] serial multiplier operation for positive and negative members in (+1, -1) representation can be constructed in the following way.

$[P_n \ldots P_1] = [a_k \ldots a_1] [b_x \ldots b_1]$

$$= \quad (2^{k-1} a_k + [a_{k-1} \ldots a_1]) [b_x \ldots b_1]$$

$$= \quad 2^{k-1} a_k [b_x \ldots\ldots b_1] + [a_{x-1} \ldots\ldots a_1]$$

$$[(2^{k-l} b_k + [b_{k-l} \ldots\ldots b_1])$$

$$= \quad [P_{2k-2} \ldots\ldots P_1] + a_k 2^{k-1} [b_k \ldots\ldots b_1] +$$

$$b_k 2^{k-l} [a_{k-1} \ldots\ldots a_1) \quad \text{for } k \geq 1 \qquad (1)$$

where $[P_1] = a_l b_1$.

From equation (1) it is clear that the addition of three terms can be accomplished by carry save adders in which the carry generated as each stage is saved and propagated only to the adjacent stage in the next addition. At each stage, the second and third terms shifted one bit left then added to 1st term. The right most bit of 1st term is not involved in addition process and is already available as output. Thus during the whole multiplication process maximum number of adders required is equivalent to the total number of multiplier bits. In equation (1), a_k and b_k can take the values as $a_k = +1$, $b_k = +1$, $a_k = -1$, $b_k = -1$, $a_k = -1$, $b_k = +1$ and $a_k = +1$, $b_k = -1$. Thus the product bit $a_k b_k$ is either positive (+1) or negative (-1).

As an illustration, the step by step multiplication process according as eqn. (1) for four bit positive as well as negative operand is shown in fig. 1. For a carry save addition P-digits in 1st term is in redundant binary form. In the 1st step, the multiplied bit is a_1 b_1 which is the 1st term of eqn. (1) and remaining two bits will be -1 and + 1 or + 1 and -1 depending on product from A carry bit + 1 or -1 is generated which will be saved and added to the next step. The sum bit is represented as P_1. During second cycle or step, the product bits will be a_2 b_1, $a_2 b_2$ and $b_2 a_1$ and they are added to the sum bit P_1 after shifting one bit left. The sum bit P_2 contains the product bits $a_2 b_1$ and $b_2 a_1$ and bit –1 or $\overline{1}$ since $\overline{1}$ produces an error during addition, so the error is removed by adding an extra bit +1. It also contains the carry bit from the previous stage. Thus, there will be total 5 bits for addition. The addition of 5 bits produces a sum bit P_2 and two bits carry for the next step. The sum bit P_3 contains the product bit a_2 b_2 and bits $\overline{1}$ and $\overline{1}$. The bits $\overline{1}$ and $\overline{1}$ are error bits and can be eliminated by adding two extra bits +1 and +1. Thus, the sum of 5 bits ($\overline{1}$, $\overline{1}$ a_2 b_2, 1 1), will produce P_3 and two bits for carry to the next step.

For step-3, we have the product bits $a_3 b_3$, $a_3 b_2$, $a_3 b_1$, $b_3 a_2$ and $b_3 a_1$. These bits are shifted one bit left and added to previous sum with saved carry bits. During addition the previous sum P_2, P_1 become free but P_3 is added to a_3 b_1 and $b_3 a_1$ two bits of previous carry. Thus there will be total 5 bits. The sum of these bits produce the sum bit P_3 and two bits for carry for the next step. The sun bit P_4 contains $a_3 b_2$ and $b_3 a_2$ -1 and two carry bits from the 2nd step. The bit, $\overline{1}$ produce an error which is over come by taking into consideration an extra bit of opposite nature i.e. + 1. Thus, there will be total six digit for sum. The addition of two bits is not possible in this system because zero does not exit in our system. The extra bit directly goes to the sum bit and the sum bit P_4 now contains two bits having the same weight. Thus, sum bit P_4 is treated as a combination of two bits. Thus the sum bit P_4 contains two bits as sum and two bits as carry for the next step. The sum bit P_5 contains the product bit $a_3 b_3$ and two bits -1 and –1 respectively. To overcome the effect of two bits, two extra bits of opposite nature +1 and +1 are added. Thus there will be 5 bits in all. The sum will produce P_5 with the two bits as carry for the next step.

Step-4 consists of the product bits $a_4 b_4$, $a_4 b_3$, $a_4 b_l$, $b_3 a_3$, $b_3 a_2$ and $b_3 a_1$. Addition of bits take place in similar fashion as discussed in step-3. Finally, we find that P_1, P_2, P_3 are free bits and they do not take part in addition. Four 5 bit adders are required for whole multiplication process. Fig. (2) points out the specific example for four bit numbers A = $\overline{1}$ $\overline{1}$ 1 $\overline{1}$ = - 11 and B = 1 $\overline{1}$ $\overline{1}$ 1 = 3 multiplication in four steps.

Hardware realization of five bit adders is shown in figure (3). It consists two product bits A and B for a particular step one previous sum bit and two bits for carry from previous step as inputs; two bits for carry and one bit for sum as outputs. 4 bit serial multiplier for 4th clock cycle connection of module is shown in Fig. (4). The third module of adder produces a "two bit" sum where one bit of the sum is extra bit to make the result error free because an error bit + 1 exists in the input. This extra bits is +1. Thus the sum bit contains an additional bit +1. Fourth adders contain fixed bits $\overline{1}$, $\overline{1}$, 1, 1 and one product bit $a_4 b_4$. The sum bit is always the product bit. Hardware realisation of a four bit serial multiplier during 4th clock interval is shown in Fig. (5).

Multiplication of two bits is realised by XNOR gate. The module of 5 bit adder is discussed earlier. 1 bit latch contains D flip-flop. D-flip flop is discussed by Tiwari and Varma. 2 bit latch for carry or sum contain two D-flip-flops working synchronously. Two bits are latched together. It produces one clock delay because two bits are entered simultaneously and taken out together. Bits are entered in queue fashion. Queue register implementation is shown in Fig. (6) It consists of shift registers. Transistor switches and tristate inverter are connected in parallel and connected to shifts registers. Switch and tristate inverters are controlled by clock. When clock is high (+1), the switch becomes closed and on the other hand if it is low (-1), tristate inverters are energized. The two bits are multiplied by XNOR gate and the output bit energizes the D-flip-flops. Initially the LSB is kept +1 and other bits are at -1 in the register. First D-flop-flop is activated because clk is +1 and other flip-flops are inactive because clock is low (-1) and so, data is entered in 1st flip-flop. On the other hand, if the clock is low (-1) all the bits of registers get inverted and again 1st flip-flop is at +1 and others at (-1). Thus the state of flip-flops remains unchanged. When the 2nd clock enters, the 2nd bit of register becomes + 1 and others becomes -1 and the 2nd D-flip-flop becomes active and remaining flip-flops remain inactive. Thus, Data + 1 or -1 is entered sequentially in queue fashion in the flip-flops.

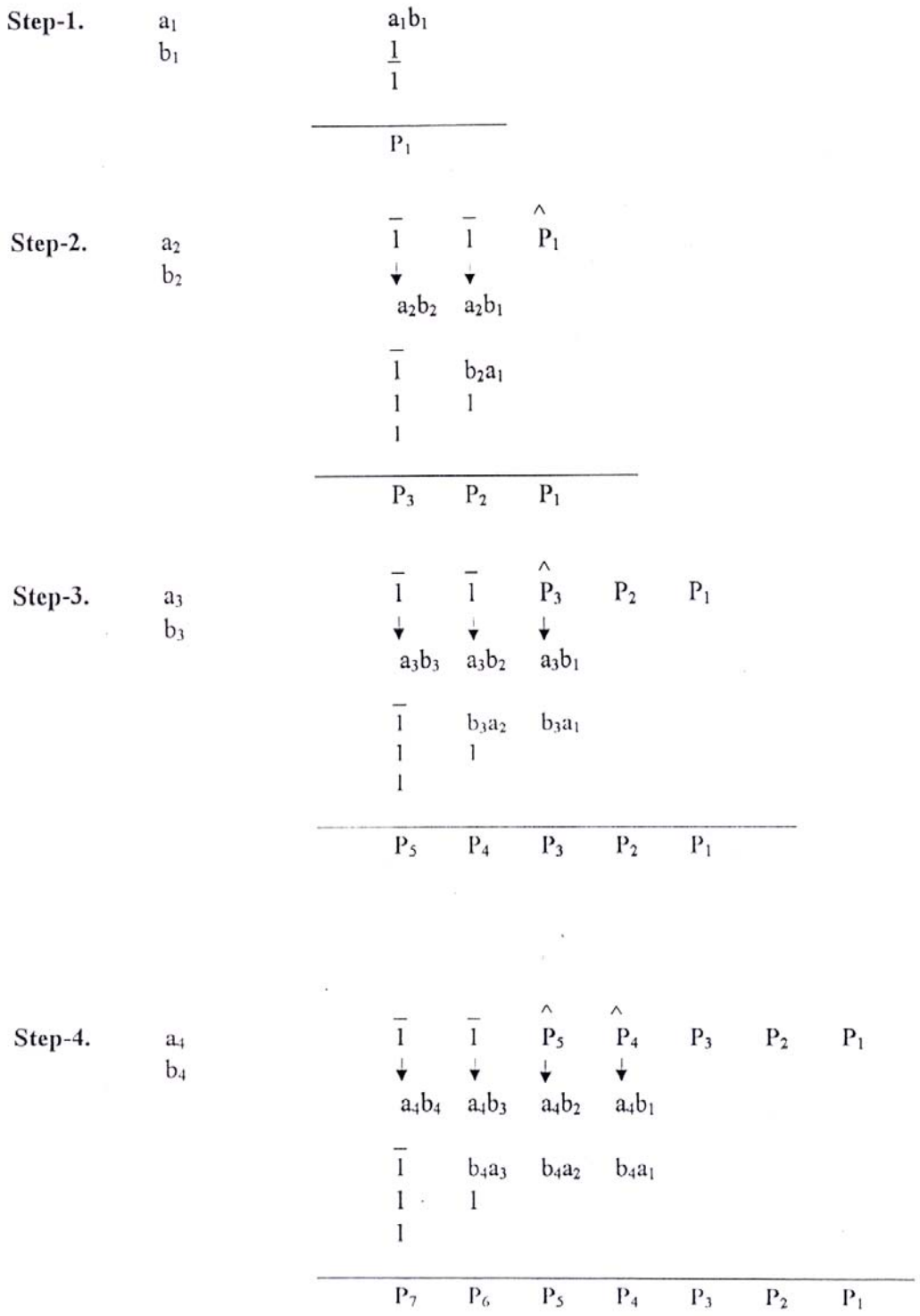
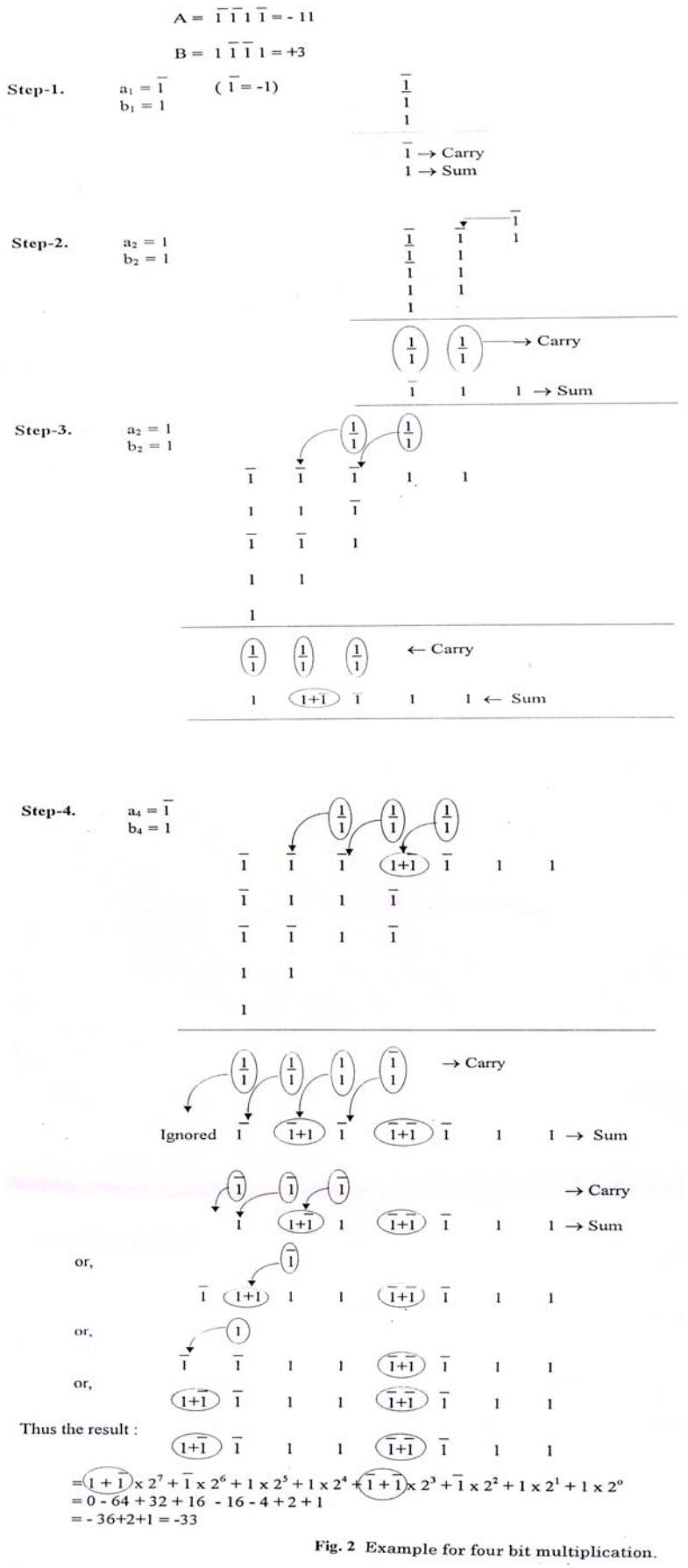


Fig. 1 Step by Step multiplication process of four bit numbers.



$= (1 + \bar{1}) \times 2^7 + \bar{1} \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + (\bar{1} + \bar{1}) \times 2^3 + \bar{1} \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
$= 0 - 64 + 32 + 16 - 16 - 4 + 2 + 1$
$= -36+2+1 = -33$

Fig. 2 Example for four bit multiplication.

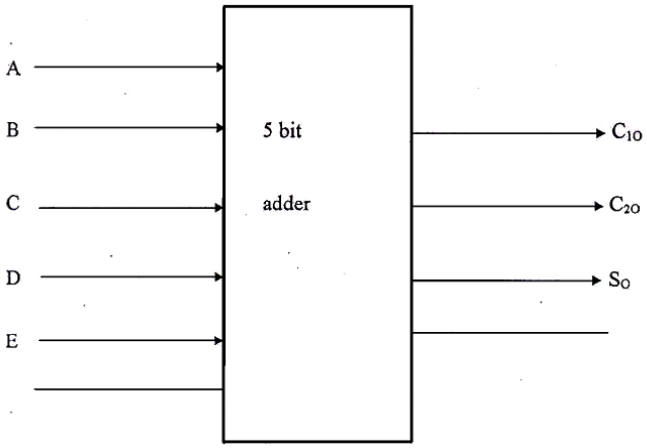


Fig. 3 Block Diagram of 5 bit Adder

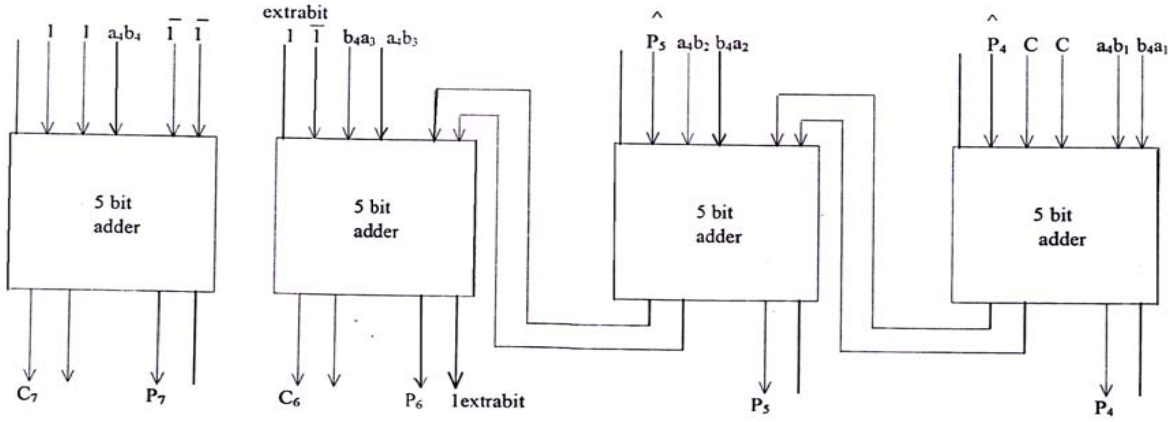


Fig. 4. Connection of 5-bit adders in module.

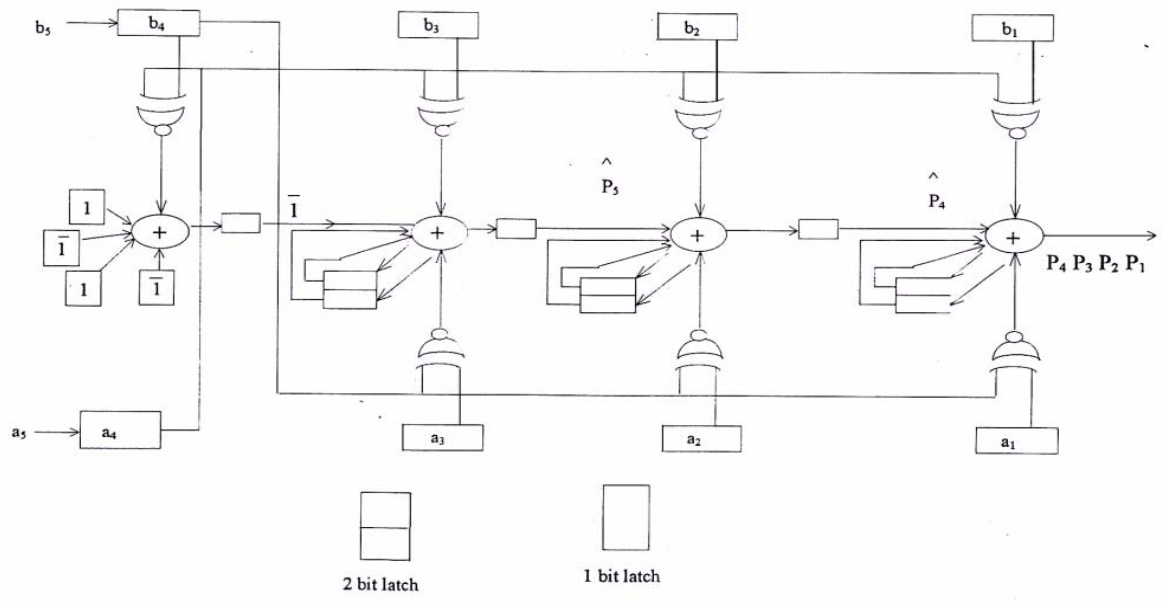


Fig. 5 Hardware Realization of a four bit serial multiplier during 4th cycle interval

CONCLUSION:

The system (+1, -1) is well suited for sequential multiplier. The system requires n number of five bit adders for n bit multiplication of positive as well as negative numbers. The algorithm used by author [7] is well applicable to this system with certain modifications. The hardware realization is different. It requires 5 bit adders in different way. It produces two bit carry and two bit sum having the same weight. This is one of extra feature of 5 bit adder. One requires two bit latch in addition to one bit. Multiplication of two bits is acquired by XNOR gate instead of AND gate. The queue register implementation requires additional hardware a switch and tristate inverter. Positive as well as negative numbers are multiplied with the same type hardware. In this way, the hardware realization of multiplier is straight and simple. The main advantage of this system is the unified implementation of positive as well as negative numbers. The two's complement multipliers are realized in different manners, as discussed by authors [8-10] compared to positive numbers. Our model is well suited for digital signal processing where parallel processing for positive and negative number are required simultaneously. A multiplier cell can be constructed and connected in modules, which helps in LSI design. The draw back of this system is that it is partially suitable for computation because even numbers can not be generated.

REFERENCES:

(1) K.S. Trivedi and M.D. Ercegovae, "On line algorithms for division and multiplication", IEEE Trans. Comput. Vol. C-26, pp 681-687. July 1977.

(2) R.F. Lyon, "Two's Complement pipe line multipliers", Vol. COM-24, no. 4 pp. 418-425, Apr. 1976.

(3) H.J. Sips, "Comments on 'An O(n) parallel multiplier with bit sequential input and output", IEEE Trans. Compo. Vol C-31, no. 4, pp. 325-327 Apr. 1982.

(4) Nhon T. Quach, Naojumi Takagi and Michal J. Flynn, "Systematic IEEE rounding method for high speed floating point multipliers. IEEE Trans. VLSI Syst. 12(5) (2004) 511.

(5) Raacemifar K. and Ahmadi M., "Fast 32 bit digital multiplier" IEEE International Conference on Electronic Circuits and System, 3 (2001) 1413.

(6) 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.

(7) 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.

(8) LN. Chen and R. Willoner, "An O(n) parallel multiplier with bit sequential input and output IEEE. Trans. Comput. Vol. C-23, no. 10.Oct. 1979.

(9) L. Dadda "On serial input multiplies for two's complement numbers" IEEE Trans. Comput. Vol. 38, No.9, pp 1341-1345, Sept. 1989.

(10) P. Ienne and M.A. Viredaz, "Bit Serial multipliers and squares", IEEE Trans. Comput. Vol. 43, no. 12 December, 1994.

(11) K.Z. Pekmestzi, "New number representation for digital signal processing", Int. J. Electronics, Vol. 66, No.5, pp. 709-723, 1989.

(12) M. Tiwari and H.V. Varma, "Logical design of adders and Comparators", Indian Journal of Technology; Vol. 31, pp. 126-130, March, 1993.

(13) M. Tiwari and R.V. Varma. "Flip-flops in a new binary system, Indian Journal of Engineering and Materials Sciences, Vol. 2, pp. 8-11 Feb. 1995.

(14) A. Reyhani- Masolrh, " Low complexity word level sequential normal basis multiplier" IEEE Transaction on Computers-2005

(15) Y. Chang, T. K. Truong, I. S. Reed, et. Al., "Algebraic decoding of(71,36,11),

(79, 40, 15), and (97, 49, 15) quadratic residue codes," *IEEE Trans. Comm.*, vol. 51, pp. 1463-1473, 2003.

[16] A. Reyhani-Masoleh and M. A. Hasan. "Efficient Digit-Serial Normal Basis Multipliers over GF (2^M)". In IEEE International Symposium on Circuits and Systems, ISCAS 2002, pages 781–784, May 2002.

[17] A. Reyhani-Masoleh and M. A. Hasan. "Efficient Multiplication beyond Optimal Normal Bases". to appear in IEEE Transactions on Computers, Special Issue on Cryptographic Hardware and Embedded Systems, April 2003.

[18] B. Sunar and C. K. Koc. "An Efficient Optimal Normal Basis Type II Multiplier". IEEE Transactions on Computers, 50(1):83–88, Jan. 2001.