ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

# A NOVEL APPROACH FOR FIR FILTER DESIGN USING DUAL-PORT MEMORY BASED MULTIPLIER

**P.Arudra[1], N.V.Ratna Kishor.Gade[2],K.Rameshchandra[3], A.M.V.Pathi[4]**

M.Tech, Swarnandhra College of Engg. and Technology, Seetharampuram, Andhra Pradesh, India[1]

Assistant Professor, Swarnandhra College of Engg. and Technology, Seetharampuram, Andhra Pradesh, India[2]

Assistant Professor, Vishnu Institute of Technology, Bhimavaram, Andhra Pradesh, India[3]

Assistant Professor, Vishnu Institute of Technology, Bhimavaram, Andhra Pradesh, India Andhra Pradesh, India[4]

**Abstract** — The objective of this project is to develop Finite Impulse Response (FIR) filter using new multiplier.The multiplier is based on dual port memory which stores all possible pre-computed product values corresponding to all possible values of input multiplicand. The configuration is expected to reduce memory size and logic gate count compared to Distributed Architecture processor. The processor finds application in digital communication, speech processing and other signal processing applications.The processor comprises of modules such as 4bit to 3bit address encoder, 3 to 8 line address decoder, control circuit, memory and barrel shift register. The design will be implemented using HDL.Simulation will be done to verify the functionality and synthesis using Xilinx Tools.

**Index Terms**— Finite Impulse Response (FIR) filter, LUT.

## I. INTRODUCTION

FINITE-IMPULSE response (FIR) digital filter is widely used as a basic tool in various signal processing and image processing applications. The order of an FIR filter primarily determines the width of the transition-band, such that the higher the filter order, the sharper is the transition between a pass-band and adjacent stop-band. Many applications in digital communication (channel equalization, frequency channelization), speech processing (adaptive noise cancelation), seismic signal processing (noise elimination), and several other areas of signal processing require filter output increases linearly with the filter order, real-time implementation of these filters of large orders is a challenging task. Several attempts have, therefore, been made and continued to develop low-complexity dedicated VLSI systems for these filters.

As the scaling in silicon devices has progressed over the last four decades, semiconductor memory has become cheaper, faster and more power-efficient. According to the projections of the international technology roadmap for semiconductors (ITRS), embedded memories will continue to have dominating presence in the system-on-chip (SoC), which may exceed 90% of total SoC content. It has also been found that the transistor packing density of SRAM is not only high, but also increasing much faster than the transistor density of logic devices According to the requirement of different application environments, memory technology has been advanced in a wide and diverse manner. Radiation hardened memories for space applications, wide temperature memories for automotive, high reliability memories for biomedical instrumentation, low power memories for consumer products, and high-speed memories for multimedia applications are under continued development process to take care of the special needs. There are two basic variants of memory-based techniques.

One of them is based on distributed arithmetic (DA) for inner product computation and the other is based on the computation of multiplication by look-up-table. In the LUT-multiplier-based approach, multiplications of input values with a fixed-coefficient are performed by an LUT consisting of all possible pre-computed product values corresponding to all possible values of input multiplicand, while in the DA-based approach, an LUT is used to store all possible values of inner-products of a fixed -point vector with any possible-point bit-vector. If the inner-products are implemented in a straight-forward way, the memory-size of LUT-multiplier based implementation increases exponentially with the word length of input values, while that of the DA-based approach increases exponentially with the inner-product-length. Attempt shave been made to reduce the memory-space in DA-based architectures using offset binary coding (OBC), and group distributed technique. A decomposition scheme is suggested in a recent paper for reducing the memory-size of DA-based implementation of FIR filter. But, it is observed that the reduction of memory-size achieved by such decompositions, is accompanied by increase in latency as

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

well as the number of adders and latches. Significant work has been done on efficient DA-based computation of sinusoidal transforms and filters. Various algorithm-architecture co-designs have also been reported for efficient LUT-multiplier-based implementation of sinusoidal transforms. In an early paper, as introduced a memory-based structure for the LUT-multiplier-based implementation of FIR filter. But, we do not find any further work to improve the efficiency of LUT-multiplier-based implementation of FIR filter. In this paper, we aim at presenting two new approaches for designing the LUT for LUT-multiplier-based implementation, where the memory-size is reduced to nearly half of the conventional approach. Besides, we find that instead of direct-form realization, transposed form realization of FIR filter is more efficient for the LUT-multiplier-based implementation. In the transposed form, a single segmented-memory core could be used instead of separate memory modules for individual multiplications in order to avoid the use of individual decoders for each of those separate modules.

## II. LUT DESIGN FOR MEMORY-BASED MULTIPLICATION

The basic principle of memory-based multiplication is depicted in fig 1. Let A be a fixed coefficient and X be an input word to be multiplied with A.

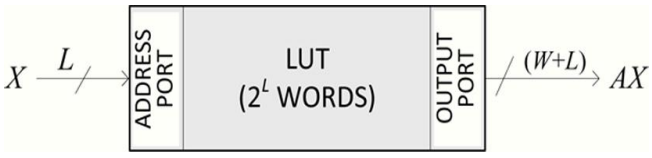



Fig.1. Conventional Memory-Based Multiplier

If we assume X to be an unsigned binary number of word-length L, there can be $2^{\wedge L}$ possible values of X , and accordingly, there can be possible values of product C=A.X . Therefore,for the conventional implementation of memory_based multiplication, a memory unit of $2^L$ words is required to be used as look-up-table consisting of pre-computed product values corresponding to all possible values of X. The product-word ($A.X_i$), for $0 \le X_i \le 2^L_{-1}$, is stored at the memory location whose address is the same as the binary value of $X_i$, such that if L -bit binary value of Xi is used as address for the memory-unit, then the corresponding product value is read-out from the memory. Although $2^{\wedge L}$ possible values of correspond to possible values of C=A.X, recently we have shown that only words corresponding to the odd multiples of may only be stored in the LUT.

LUT WORDS AND PRODUCT VALUES FOR INPUT LENGTH L=4 TABLE-1

| address $d_2d_1d_0$ | word symbol | stored value | input $x_3x_2x_1x_0$ | product value | # of shifts | control $s_1$ $s_0$ |
|---|---|---|---|---|---|---|
| 0 0 0 | $P0$ | $A$ | 0 0 0 1 | $A$ | 0 | 0 0 |
| | | | 0 0 1 0 | $2^1 \times A$ | 1 | 0 1 |
| | | | 0 1 0 0 | $2^2 \times A$ | 2 | 1 0 |
| | | | 1 0 0 0 | $2^3 \times A$ | 3 | 1 1 |
| 0 0 1 | $P1$ | $3A$ | 0 0 1 1 | $3A$ | 0 | 0 0 |
| | | | 0 1 1 0 | $2^1 \times 3A$ | 1 | 0 1 |
| | | | 1 1 0 0 | $2^2 \times 3A$ | 2 | 1 0 |
| 0 1 0 | $P2$ | $5A$ | 0 1 0 1 | $5A$ | 0 | 0 0 |
| | | | 1 0 1 0 | $2^1 \times 5A$ | 1 | 0 1 |
| 0 1 1 | $P3$ | $7A$ | 0 1 1 1 | $7A$ | 0 | 0 0 |
| | | | 1 1 1 0 | $2^1 \times 7A$ | 1 | 0 1 |
| 1 0 0 | $P4$ | $9A$ | 1 0 0 1 | $9A$ | 0 | 0 0 |
| 1 0 1 | $P5$ | $11A$ | 1 0 1 1 | $11A$ | 0 | 0 0 |
| 1 1 0 | $P6$ | $13A$ | 1 1 0 1 | $13A$ | 0 | 0 0 |
| 1 1 1 | $P7$ | $15A$ | 1 1 1 1 | $15A$ | 0 | 0 0 |

$s_0$ and $s_1$ are control bits of the logarithmic barrel-shifter.

One of the possible product words is zero, while all the rest $(2^L/2)$ -1 are even multiples of A which could be derived by left-shift operations of one of the odd multiples of A. We illustrate this in Table I for L=4. At eight memory locations, eight odd multiples A×(2i+1) are stored as $P_i$ for i=0,1,2,….7.The even multiplies 2A,4A and 8A are derived by left-shift operations of A. Similarly,6A and 12Aare derived by left-shifting 3A, while 10A and 14A are derived by left-shifting 5A and 7A, respectively. The addressX=(0000) corresponds to (A.X)=0,which can be obtained by resetting the LUT output. For an input multiplicand of word-size L similarly, only $(2^L/2)$odd multiple values need to be stored in the memory-core of the LUT, while the other $(2^L/2-1)$ non-zero values could be derived by left-shift operations of the stored values. Based on the above, an LUT for the multiplication of an L -bit input with W-bit coefficient is designed by the following strategy:

- A memory-unit of $2^L/2$ words of (W+L)-bit width is used to store all the odd multiples of A.
- A barrel-shifter for producing a maximum of (L-1)left shifts is used to derive all the even multiples of A.
- The L-bit input word is mapped to (L-1)-bit LUT-address by an encoder.
- The control-bits for the barrel-shifter are derived by a control-circuit to perform the necessary shifts of the LUT output. Besides,a RESET signal is generated by the same control circuit to reset the LUT output when X=0.

### *Proposed LUT-Based Multiplier for 4-Bit Input*

The proposed LUT-based multiplier for input word-size L=4 is shown in Fig.2. It consists of a memory-array of eight words(w+4) of -bit width and a 3-to-8 line address decoder, along with a NOR-cell, a

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

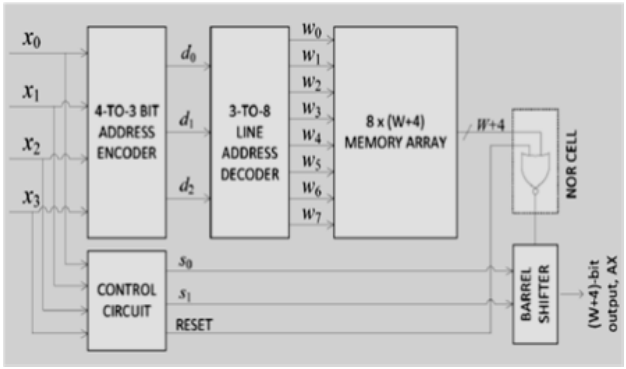
barrel-shifter, a 4-to-3 bit encoder to map the 4-bit input operand to 3-bit LUT-address, and a control circuit for generating the control-word(S0,S1) for the barrel-shifter, and the RESET signal for the NOR-cell.

The 4-to-3 bit input encoder is shown in Fig. 2(b). It receives a four_bit input word (S3,S2,S1) and maps that onto the three-bit address word , according to the logical relations (d2,d1,d0).

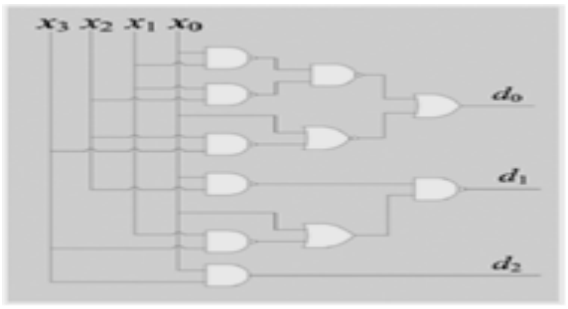
$$d_0 = \overline{\overline{(x_0 \cdot x_1)} \cdot \overline{(x_1 \cdot x_2)} \cdot (x_0 + \overline{(x_2 \cdot x_3)}}$$
$$d_1 = \overline{\overline{(x_0 \cdot x_2)} \cdot (x_0 + \overline{(x_1 \cdot x_3)})}$$
$$d_2 = x_0 \cdot x_3$$

The pre-computed values of A×(2i+1) are stored as pi for i=0,1,2..7 at 8 consecutive locations of the memory-array as specified in Table I in bit-inverted form. The decoder takes the 3-bit address from the input encoder, and generates 8 word-select signals, to select the referenced-word from the memory-array. The output of the memory-array is either or its sub-multiple in bit-inverted form depending on the value of . From Table I, we find that the LUT output is required to be shifted through 1 location to left when the input operand is one of the values Two left-shifts are required if is either (0 1 0 0) or (1 1 0 0). Only when the input word, three shifts are required For all other possible input operands, no shifts are required. Since the maximum number of left-shifts required on the stored-word is three, a two-stage logarithmic barrel-shifter is adequate to perform the necessary left-shift operations. The number of shifts required to be performed on the output of the LUT and the control-bits and for different values of are shown Table I.The control circuit shown in fig 2(c) accordingly generates the control-bits given by

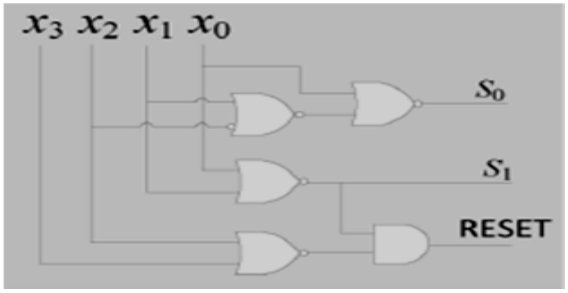
$$s_0 = \overline{x_0 + \overline{(x_1 + \overline{x_2})}}$$
$$s_1 = \overline{(x_0 + x_1)}$$



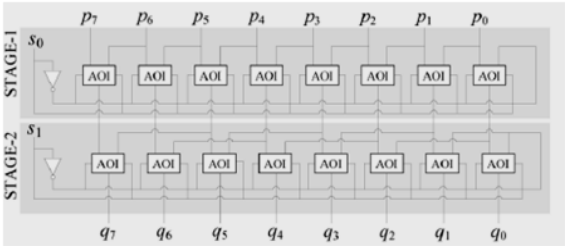

**a**



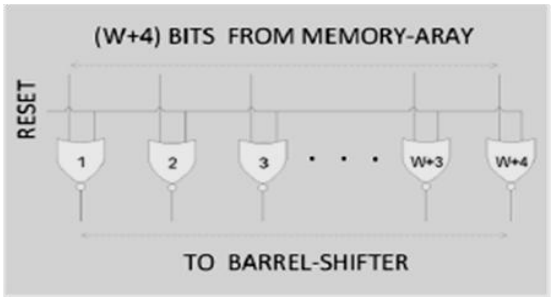

**b**




**c**




**d**




**e**

Fig. 2. Proposed LUT design for multiplication of w-bit fixed
coefficient, (a) A and 4-bit input operand, The proposed LUT-based multiplier.(b)The 4-to-3 bits input encoder. (c)Control circuit. (d) Two-stage logarithmic barrel-shifter for w=4 (e) Structure of the NOR-cell.

A logarithmic barrel-shifter for W=L=4 is shown in Fig. 2(d). It consists of two stages of 2-to-1 line bit-level multiplexors with inverted output, where each of the two stages involves (W+4) number of 2-input AND-OR-INVERT (AOI)gates. The control -bits (s0,$\overline{s0}$)and(s1,$\overline{s1}$) are fed to the AOI gates of stage-1 and

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

stage-2 of the barrel-shifter, respectively. Since each stage of the AOI gates perform inverted multiplexing,after two stages of inverted multiplexing, outputs with desired number of shifts are produced by the barrel-shifter in(the usual) un-inverted form.

The input X=(0000) corresponds to multiplication by X=0 which results in the product value A.X=0 . Therefore ,when the input operand word X=(0000) , the output of the LUT is required to be reset. The reset function is implemented by a NOR-cell consisting (W+4) of NORgates as shown in Fig. 2(e) using an active-high RESET. The RESET bit is fed as one of the inputs of all those NOR gates, and the other input lines of (W+4) NOR gates of NOR cell are fed with (W+4) bits of LUT output in parallel. When X=(0000),the control circuit in Fig.2(c), generates an active-high RESET according to the logic expression:

$$\text{RESET} = \overline{(x_0 + x_1)} \cdot \overline{(x_2 + x_3)}$$

When RESET=1, the outputs of all the NOR gates become 0, so that the barrel-shifter is fed with (W+4) number of zeros .When RESET=0, the outputs of all the NOR gates become the complement of the LUT output-bits. Note that, keeping this in view, the product values are stored in the LUT in bit-inverted form. Reset function can be implemented by an array of 2-inputAND gates in a straight-forward way, but the implementation of reset by the NOR-cell is preferable since the NOR gates have simpler CMOS implementation compared with the AND gates .Moreover, instead of using a separate NOR-cell, the NOR gates could be integrated with memory-array if the LUT is implemented by a ROM. The NOR cells, therefore, could be eliminated by using a ROM of 9 words, where the 9th word is zero and RESET is used as its word-select signal.

To compare the area of the proposed LUT-multiplier and the existing LUT-multiplier, we have synthesized the multipliers for L=4for different coefficient width by Synopsys Design Compiler using TSMC 90 nm library Both the designs have nearly the same data arrival time, but the proposed LUT design is found to offer a saving of nearly 23% of area over the conventional design. The saving of area in the proposed LUT design resulting from lower storage and less decoder complexity is reduced mainly due to the overhead of barrel-shifter and NOR cells.

Multiplication of an 8-bit input with a W -bit fixed coefficient can be performed through a pair of multiplications using a dual-port memory of 8 words (or two single-port memory units) along with a pair of decoders, encoders, NOR cells and barrel shifters as shown in Fig.3. The shift-adder performs left-shift operation of the output of the barrel-shifter corresponding to more significant half of input by four bit-locations, and adds that to the output of the other barrel-shifter. In the next sub-section, we present two other optimization schemes which has been proposed recently for reduction of storage size of LUT-multipliers

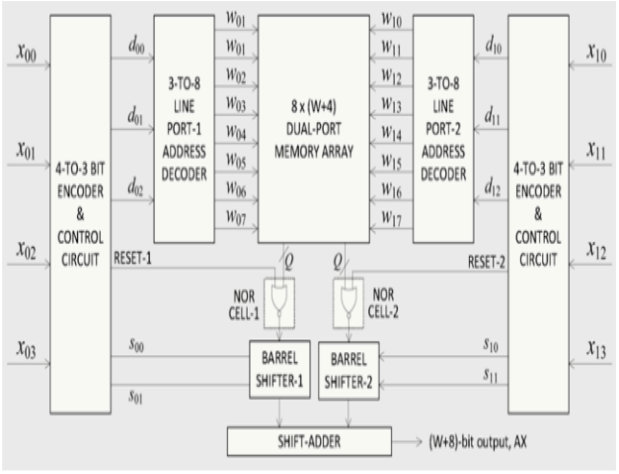



Fig.3 Memory-based multiplier using dual-port memory-array.Q=(W+4)

## III .MEMORY-BASED STRUCTURES FOR FIR FILTER USING LOOKUP-TABLE-MULTIPLIERS

We derive here the proposed structure for memory-based realization of an -tap FIR filter, and discuss the design of memory cell to be used as LUT in the structure. The input-output relationship of an -tap FIR filter in time-domain is given by

$$Y(n)=h(0)x(n)+h(1)x(n-1)+h(2)x(n-2)\ldots+h(n-1)x(n-N+1).$$

Where h(n) , for n= 0,1,2,…N-1 , represent the filter coefficients, while x(n-i) , for i= 0,1,2..N-1. Represent N recent input samples, and y(n) represents the current output sample. Memory-based multipliers can be implemented for signed as well as unsigned operands. In case of signed operands, the input words and the stored product values need to be in two's complement representation. Since the stored product values require sign-extension in case of two's complement representation during shift-add operations, the LUT-based multiplication could have a simpler implementation when the multiplicands are unsigned numbers. Besides, without loss of generality, we can assume the input samples to be unsigned numbers and the filter coefficients to be signed numbers, in general. Keeping this in view, we write equation alternatively as

$$Y(n)=\text{sign}(0).h(0).x(0)+\text{sign}(1).h(1)x(n-1)+\ldots..\text{sign}(N-1).h(N-1).x(n-N+1).$$

where h(n)=sign(n).h(n), for n= 0,1,2,….N-1, h(n) denotes the absolute value of h(n) and sign(n)=+1 is the sign-factor, which could be absorbed with the additions of the corresponding term.

### *A. Memory-Based FIR Filter Using Conventional LUT*

The recursive computation of FIR filter output according to

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

(9) is represented by a transposed form data-flow graph (DFG) shown in Fig.4. It consists of multiplication nodes (M) and (N-1) add-subtract (AS) nodes. The function of these nodes are depicted in Fig.4(b). Each multiplication node performs the multiplication of an input sample value with the absolute value of a filter coefficient. The AS node adds or subtracts its input from top with or from that of its input from the left when the corresponding filter coefficient is positive or negative, respectively.It may be noted here that each of the multiplication nodes of the DFG performs multiplications of input samples with a fixed positive number. This feature can be utilized to implement the multiplications by an LUT that stores the results of multiplications of all possible input values with the multiplying coefficient of a node as unsigned numbers. The multiplication of an –L bit unsigned input with W-bit magnitude part of fixed filter-weight,to be performed by each of the multiplication-nodes of the DFG,can be implemented conventionally by a dual-port memory unit consisting of $2^L/2$words of (W+L)bit width. Each of the(N-1) AS nodes of the DFG along with a neighboring delay element can be mapped to an add-subtract (AS) cell.

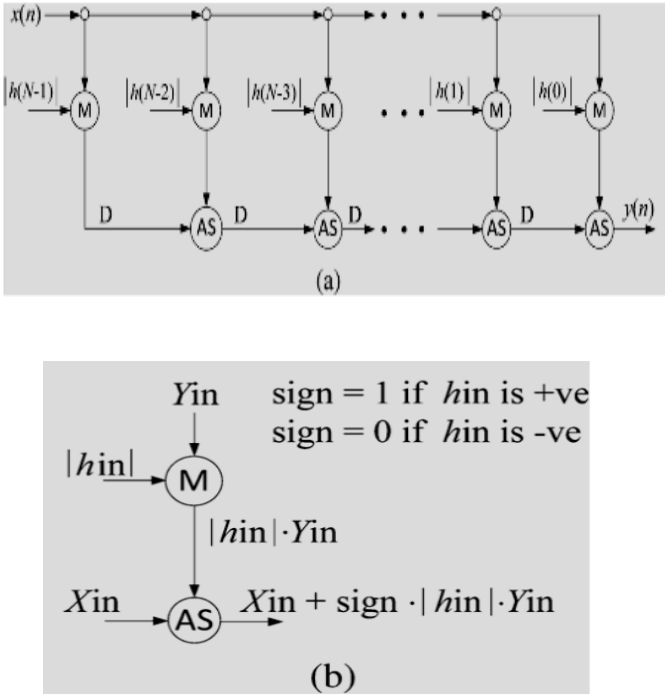



Fig. 4. Modified transposed form data-flow graph (DFG) of an N-tap FIR filter for LUT-multiplier-based implementation. (a) The DFG. (b) Function of each multiplication node (M) and add-subtract node (AS) of the DFG.

A fully pipelined structure for -tap FIR filter for input word length L=8, as shown in Fig.5,is derived accordingly from the DFG of Fig.4.It consists of N memory-units for conventional LUT-based multiplication, along with AS cells
and a delay register. During each cycle, all the 8 bits of current input sample x(n) are fed to all the LUT-multipliers in parallel as a pair of 4-bit addresses and . The structure of the LUT-multiplier is shown in Fig.5(b). It consists of a dual-port memory unit of size [16×(W+4)] (consisting of 16 words of (W+4)-bit width) and a shift-add (SA) cell. The SA cell shifts its right-input to left by four bit-locations and adds the shifted value with its other input to produce a(W+8) -bit output. The shift operation in the shift-add cells is hardwired with the adders,so that no additional shifters are required. The output of the multipliers are fed to the pipeline of AS cells in parallel. Each AS cell performs exactly the same function as that of the AS node of the DFG. It consists of either an adder or a subtractor depending on whether the corresponding filter weight h(n) is positive or negative, respectively. Besides, each of the SA cells consists of a pipeline latch corresponding to the delays in the DFG of Fig.4.The FIR filter structure of Fig.5 takes one input sample in each clock cycle, and produces one filter output in each cycle. The first filter output is obtained after a latency of three cycles (1 cycle each for memory output, the SA cell and the last AS cell).But, the first (N-1)outputs are not correct because they do not contain the contributions of all the filter coefficients. The correct output of this structure would thus be available after a latency of (N+2) cycles

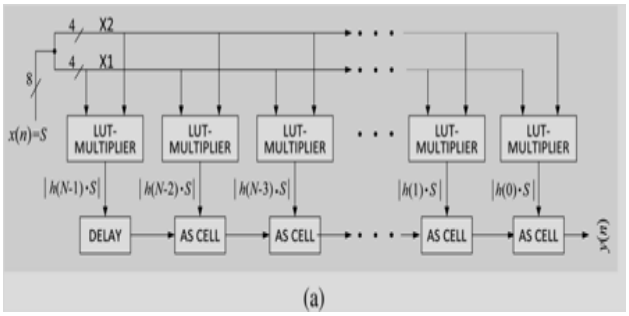



Fig. 5. (a) Conventional LUT-multiplier-based structure of an N-tap transposed form FIR filter for input-width L=8 (b) Structure of each LUT-multiplier

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 2, Issue 11, November 2013*

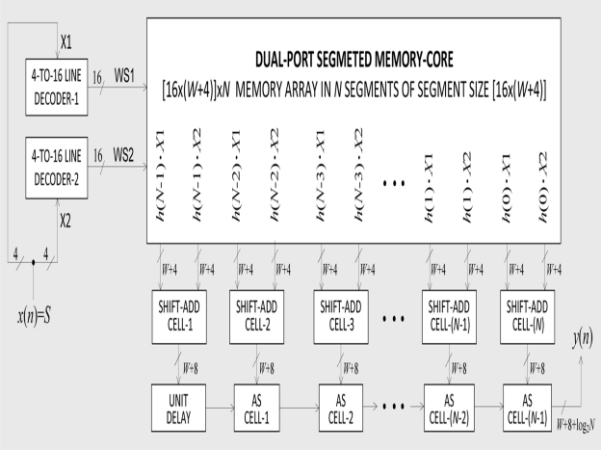



Fig.6. LUT-multiplier-based structure of an N-tap FIR filter by transposed form realization using segmented memory-core.

## IV. CONCLUSION

Develop Look-Up Table and memory based signal are suggested to reduce the LUT-size over that of conventional design. By odd-multiple-storage scheme, for address-length 4, the LUT size is reduced to half by using a two-stage logarithmic barrel-shifter and (w+ 4) number of NOR gates, where w is the word-length of the fixed multiplying coefficients. Three memory-based structures having unit throughput rate are designed further for the implementation of FIR filter. One of the structures is based on DA principle, and the other two are based on LUT-based multiplier using the conventional and the proposed LUT designs.

## REFERENCES

[1] J. G. Proakis and D. G. Manolakis, Digital Signal Processing: Principles, Algorithms and Applications. Upper Saddle River, NJ: Prentice-Hall, 1996.
[2] G. Mirchandani, R. L. Zinser Jr., and J. B. Evans, "A new adaptive noise cancellation scheme in the presence of crosstalk [speech signals]," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process.
[3] D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in Proc. IEEE Southeastcon'93, Apr. 1993, p. 6.
[4] K. K. Parhi, VLSI Digital Signal Procesing Systems: Design and Implementation. New York: Wiley, 1999.

## BIOGRAPHIES

**Mr. P.Arudra**, born in A.P, India. doing M.Tech from Swarnandhra College of Engineering And Technology, Seetharampuram. His Research work interests in VLSI design.

**Mr.G.RATNAKISHORE**, Completed M.Tech from Swarnandhra College of Engg And technology. Presently working as Asst.Prof.At Swarnandhra College of Engg And technology,.He has a Teaching Experience of Five years.His research work interests in Instrumentation.

**Mr.K.RAMESHCHANDRA**,CompletedM.Tech from IIT Madras. Presently working as Asst.Prof.At Vishnu Institute of Technology,Bhimavaram.He has a Teaching Experience of three years.His research work interests in Communications.

**Mr. A.M.V.PATHI**, born in A.P, India. Completed M.Tech& B.Tech from Swarnandhra College of Engineering And Technology, Seetharampuram. Presently working as Asst.Prof in Vishnu Institute of Technology, Bhimavaram. His Research work interests in VLSI design.