Volume 2, Issue 8, August 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Low Area and Power Efficient Architectures for Digital FIR Filters

A.Renuka Narasimha*, K.Rajasekhar,A.Sujana Rani
ECE, ASR, JNTUK
Andhra Pradesh, India

***Abstract*— Digital signal processing (DSP) is used in wide range of applications such as telephone, radio, video etc. Most of DSP computations involve the use of multiply accumulate operations and therefore the design of fast and power efficient multiplier is imperative. More over, the demand for portable applications of DSP architectures has dictated the need for low power & area designs. Digital Finite Impulse Response (FIR) filter has a lot of arithmetic operations. In general, arithmetic operation modules such as adder and multiplier modules, consume much power, energy, and circuit area. In some applications, the FIR filter circuit must be able to operate at high sample rates, while in other applications, the FIR filter circuit must be a low-power circuit operating at moderate sample rates.**

This paper presents the methods for implementing digital Finite Impulse Response (FIR) filter that requires optimized area and less power consumption .The methods include Modified Booth Encoding Algorithm combined with Spurious Power Suppression Technique, folding transformation in linear phase architecture, Low Power Digit Serial Multiplier along with carry look ahead adder, shift/add multipliers. These techniques are applied to fir filters to minimize the area and power consumption. The proposed designs for FIR filters have been designed using Verilog HDL and synthesized, implemented using Xilinx ISE Spartan FPGA

***Keywords*— DSP, FIR, booth encoding, folding transformation, Xilinx ISE Spartan.**

I. INTRODUCTION

Finite impulse response (FIR) filters are widely used in various DSP applications. In some applications, the FIR filter circuit must be able to operate at high sample rates, while in other applications, the FIR filter circuit must be a low-power circuit operating at moderate sample rates. The low-power or low-area techniques developed specifically for digital filters can be found in. Parallel (or block) processing can be applied to digital FIR filters to either increase the effective throughput or reduce the power consumption of the original filter. While sequential FIR filter implementation has been given extensive consideration, very little work has been done that deals directly with reducing the hardware complexity or power consumption of parallel FIR filters [1]. Traditionally, the application of parallel processing to an FIR filter involves the replication of the hardware units that exist in the original filter. The topology of the multiplier circuit also affects the resultant power consumption. Choosing multipliers with more hardware breadth rather than depth would not only reduce the delay, but also the total power consumption [2]. A lot of design methods of low power digital FIR filter are proposed, for example, in [3] they present a method implementing fir filters using just registered address and hardwired shifts. They extensively use a modified common sub expression elimination algorithm to reduce the number of adders. In [4] they have proposed a novel approach for a design method of a low power digital base band processing. Their approach is to optimize the bitwidth of each filter coefficient. They define the problem to find optimized bitwidth of each filter coefficient. In [5] presents the method reduce dynamic switching power of a fir filter using data transition power diminution technique (DPDT). This technique is used on adders, booth multipliers. In [6] this research proposes a pipelined variable precision gating scheme to improve the power awareness of the system. This research illustrates this technique is to clock gating to registers in both data flow direction and vertical to data flow direction within the individual pipeline stage based on the input data precision. The rest of the paper is structured as follow.

Section2 gives a brief summary of fir filter theory and in Section3 presents the architecture adopted in our implementation. Comparison of our implementation with those done is given at section4. Finally section5 provides the conclusion of this paper.

I. FIR FILTER THEORY

Digital filters are typically used to modify or alter the attributes of a signal in the time or frequency domain. The most common digital filter is the linear time-invariant (LTI) filter. An LTI interacts with its input signal through a process called linear convolution, denoted by y = f * x where f is the filter's impulse response, x is the input signal, and y is the convolved output. The linear convolution process is formally defined by:

$$Y[n] = x[n] * f[n] = \sum_{k=0} x[n]f\,[n-k] = \sum_{k=0} f[k]x\,[n-k]. \qquad (1)$$

LTI digital filters are generally classified as being finite impulse response (i.e., FIR), or infinite impulse response (i.e., IIR). As the name implies, an FIR filter consists of a finite number of sample values, reducing the above convolution sum to a finite sum per output sample instant. An FIR with constant coefficients is an LTI digital filter. The output of an FIR of order or length L, to an input time-series x[n], is given by a finite version of the convolution sum given in (1), namely:

$$y[n]=x[n]*f[n]=\sum_{k=0}^{L-1} f[k]x[n-k] \qquad \textbf{(2)}$$

where f [0] ≠ 0 through f [L-1] ≠ 0 are the filter's L coefficients. They also correspond to the FIR's impulse response. For LTI systems it is sometimes more convenient to express in the z-domain with

$$\mathbf{Y (z) = F (z) X (z),} \qquad \textbf{(3)}$$

where F (z) is the FIR's transfer function defined in the z-domain by

$$F(z)=\sum_{k=0} f[z]z^{-k} \qquad \textbf{(4)}$$

The L^{th}-order LTI FIR filter is graphically interpreted in Fig.1. It can be seen to consist of a collection of a "tapped delay line," adders, and multipliers. One of the operands presented to each multiplier is an FIR coefficient, often referred to as a "tap weight" for obvious reasons. Historically, the FIR filter is also known by the name "transversal filter," suggesting its "tapped delay line" structure [6].

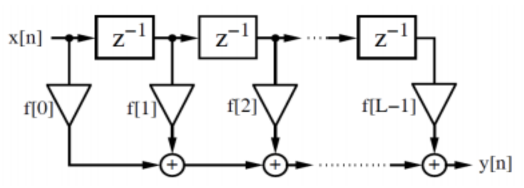

Fig. i: FIR filter in the transposed structure

II. FIR IMPLEMENTATION

In order to achieve high-speed multiplication, modified Booth algorithm has been presented in this section, Fig2 [7]. This type of multiplier operates much faster than an array multiplier for longer operands because its computation time is proportional to the logarithm of the word length of operands. Booth multiplication is a technique that allows for smaller, faster multiplication circuits, by recoding the numbers that are multiplied. It is possible to reduce the number of partial products by half, by using the technique of radix-4 Booth recoding. The basic idea is that, instead of shifting and adding for every column of the multiplier term and multiplying by 1 or 0, we only take every second column, and multiply by ±1, ±2, or 0, to obtain the same results. The advantage of this method is the halving of the number of partial products.

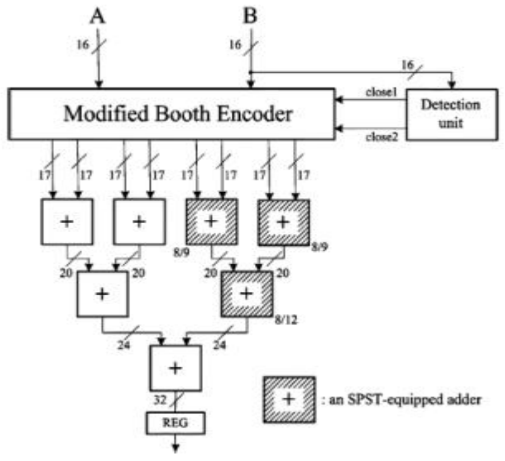

Fig ii: Proposed high performance low power multiplier

To Booth recode the multiplier term, we consider the bits in blocks of three, such that each block overlaps the previous block by one bit. Grouping starts from the LSB, and the first block only uses two bits of the multiplier. Fig 3 shows the grouping of bits from the multiplier term for use in modified booth encoding.

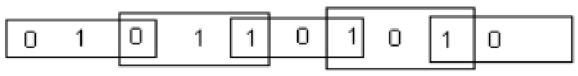

Fig iii: Grouping of bits from the multiplier term

Each block is decoded to generate the correct partial product. The encoding of the multiplier Y, using the modified booth algorithm, generates the following five signed digits, -2, -1, 0, +1, +2. Each encoded digit in the multiplier performs a certain operation on the multiplicand, X, as illustrated in Table 1

Table 1

Block	Re - coded digit	Operation on X
000	0	0 X
001	+1	+1 X
010	+1	+1 X
011	+2	+2 X
100	-2	-2 X
101	-1	-1 X
110	-1	-1 X
111	0	0 X

The PP generator generates five candidates of the partial products, i.e., {-2A,-A, 0, A, 2A}. These are then selected according to the Booth encoding results of the operand B. When the operand besides the Booth encoded one has a small absolute value, there are opportunities to reduce the spurious power dissipated in the compression tree. Fig 4. Shows the booth partial product generation circuit. It includes AND/OR/EX-OR logic.

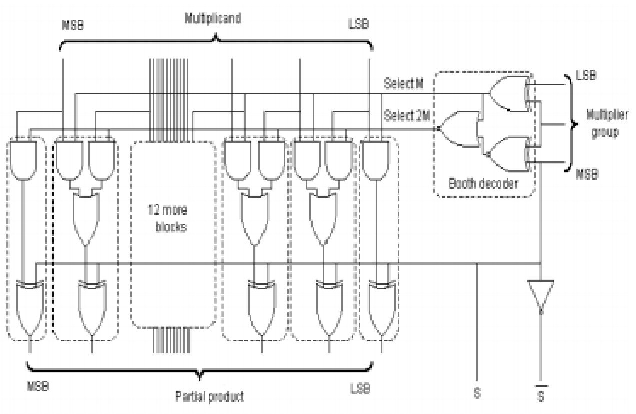

Fig iv: Booth partial product selector logic

A. *Proposed spurious power suppression technique*

Figure 5, shows a 16-bit adder/subtractor design example based on the proposed SPST [8]. In this example, the 16-bit adder/subtractor is divided into MSP and LSP at the place between the 8th bit and the 9th bit. Latches implemented by simple AND gates are used to control the input data of the MSP. When the MSP is necessary, the input data of MSP remain the same as usual, while the MSP is negligible, the input data of the MSP become zeros to avoid switching power consumption

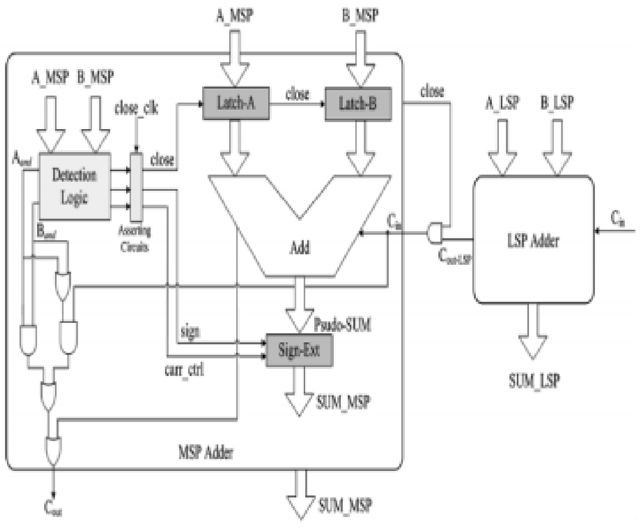

Fig v: Low power adder/subtractor implementing the SPST

B. *Linear-Phase-Folding Architecture FIR Filter Based Booth Multiplier*

If the phase of the filter is linear, the symmetrical architecture can be used to reduce the multiplier operation. Comparing Fig.1 and Fig.6, the number of multipliers can be reduced half after adopting the symmetrical architecture. But number of adders remains constant and it is the basic model to develop the proposed architecture.

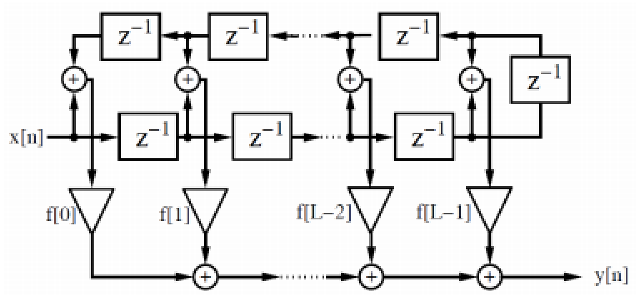

Fig vi: Linear-phase filter with reduced number of multipliers

Many algorithm transformation techniques are available for optimum implementation of the digital signal processing algorithms. Reducing the implementation area is important for complex algorithms, such as the receiver equalizer in the metal link digital communications. For example folded architectures provide a trade-off between the hardware speed and the area complexity. The folding transformation can be used to design time-multiplexed architectures using less silicon area. Power consumption can be even reduced with the folding transformation. Thus folding is a technique to reduce the silicon area by time multiplexing many operations (e.g. multiply & add) into single function units Folding introduces registers Computation time increased. Fig.7 show linear phase folding architecture fir filter. [2]

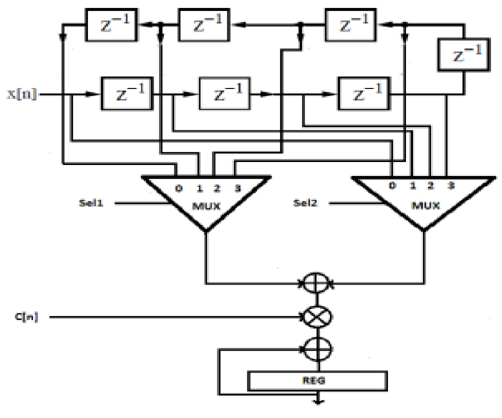

Fig vii: linear phase folding architecture fir filter

C. *Low Power Digit-Serial Multiplier*

In this section, a systematic transformation approach is proposed which enables the direct design of digit-serial architecture from bit serial architectures. Consider the multiply-add structure shown in Fig.8 (a) which forms the basic building block of a bit-serial multiplier. The basic idea behind the transformation approach involves treating the bits in this multiplier as digits. Therefore, the inputs bits a and b to the multiplier in Fig.8 (a) are replaced by digits $A=a_{N-1}\ldots a_1a_0$ and $B=b_{N-1}\ldots b_1b_{0,\ \text{respectively}}$, where N represents the digit-size.

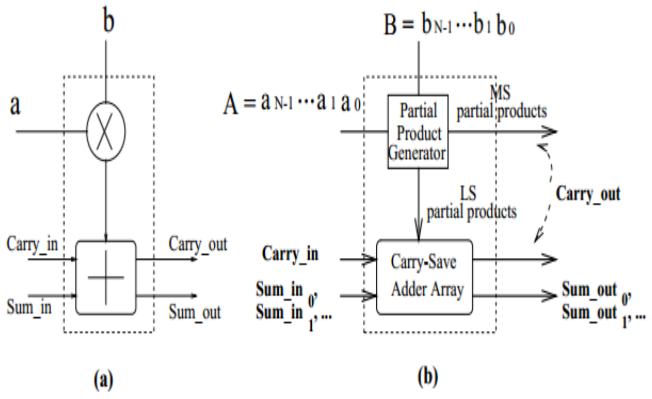

Fig viii. (a)Bit-serial multiply-add structure
(b) Digit-serial multiply-add structure obtained after applying the proposed transformation

The multiplier (which is just an AND gate in the bit-serial case) is now replaced by a partial product generator which generates both least significant (LS) and most significant (MS) partial products. The main advantage of separating the partial products is that the addition of the MS partial products can be carried out in the next clock cycle. Since the bits have been replaced by digits, the binaryadderinFig.8 (a) is now replaced by carry-save adder. The proposed transformation approach is illustrated below.

Consider the bit-serial multiplier [9] shown in Fig9, Where the coefficient word length is four bits. This architecture contains four full adders, four multipliers, and some delay elements. The idea behind the transformation approach involves treating the coefficient bits a0, a1, a2, a3 and b0, b1, b2, b3 as digits. In this manner each cell inside the dashed boxes in Fig.2 is now replaced by the corresponding structure shown in Fig.10.

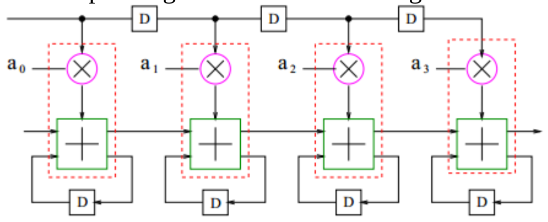

Fig ix : Bit serial multiplier with word length of 4 bits

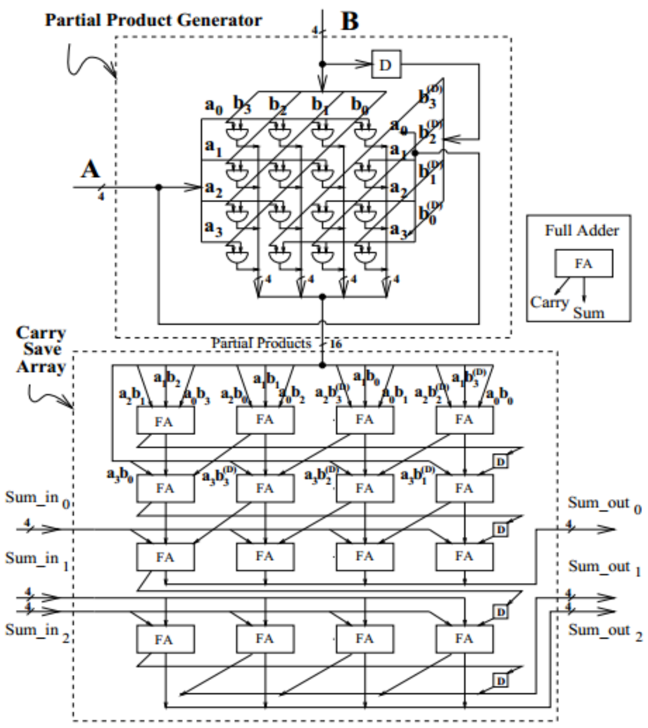

Fig x: Digit-cell for multiplier for N=4 bits

This structure consists of consists of a partial product generator module and a carry save adder (CSA) module. The partial product generator computes the16partial products. It should be noted that both the current value and a delayed value of signal are used to compute the partial products .For example, a_3b_1 $^{(D)}$ is used to denote the fact that a_3 is multiplied by a delayed version of b_1. The CSA module produces three sum out-put digits sum_out_0, sum_out_1and sum_out_2 in order to enable bit-level pipelining. Therefore, in the final stage a digit- serial adder is required to sum all these outputs. A simple digit-serial 3:2 compressor adder can be first used to reduce these three output digits to two digits. A digit-serial carry look-ahead adder or any other fast carry-propagate adder is then used to add these two digits to generate the final result.

D. *Shift-and-Add multiplier*

In this section we present a simple Shift-and-Add structure for multiplier used in Fir filters. It performs multiplication by generating partial products. It shifts the multiplicand left by one bit after every partial product calculation. The partial product of the current stage is set to the sum of the previous partial product and the shifted multiplicand of the current stage or 0, depending on whether the multiplier bit corresponding to the current stage is 1 or 0.

Reference Model: Shift-and-Add
Stage 1.
Rule x: product = product + mcand if(y [0])
Rule y: product = product + 0 if (: y [0])

Stage 2.
Rule x: product = product + mcand<<1 if(y [1])
Rule y: product = product + 0 if (: y [1])

Stage 3.
Rule x: product = product + mcand<<2 if(y [2])
Rule y: product = product + 0 if (: y [2])

IV. RESULTS

From the below table results shows that the techniques used are all having less area utilization ,these are all for 16 bit multiplier designed in verilog HDL ,and synthesized using Xilinx Spartan FPGA kit.

Table ii

Device utilization summary	*Booth multiplier*	*Linear folding*	*Digit cell multiplier*	*Shift/add multiplier*
Number of Slices	*585 out of 4656 12%*	*121 out of 4656 2%*	*93 out of 4656 1%*	*294 out of 4656 6%*
Number of 4 input LUTs	*1090 out of 9312 11%*	*114 out of 9312 1%*	*171 out of 9312 1%*	*522 out of 9312 5%*
Number of IOs	*100*	*72*	*39*	*64*
Number of bonded IOBs	*100 out of 232 43%*	*72 out of 232 31%*	*39 out of 232 16%*	*64 out of 232 27%*

V. CONCLUSIONS

In This paper we presented a low power and low area FIR filter. For reduce power consumption and area we using Modified Booth Encoding Algorithm combined with Spurious Power Suppression Technique, folding transformation in linear phase architecture, Low Power Digit Serial Multiplier along with carry look ahead adder, shift/add multipliers. These filters were compared for area and power and it demonstrated that our approach is most effective for implementations with the constraints of low cost and low power. The proposed FIR filters have been designed using Verilog HDL and synthesized, implemented using Xilinx ISE Spartan FPGA.

REFERENCES

[1] Jin-Gyun Chung, Keshab K. Parhi "Frequency Spectrum Based Low-Area Low-Power Parallel FIR Filter Design" EURASIP Journal on Applied Signal Processing 2002, vol. 31, pp. 944–953.

[2] AHMED F. SHALASH, KESHAB K. PARHI "Power Efficient Folding of Pipelined LMS Adaptive Filters with Applications" Journal of VLSI Signal Processing, pp. 199–213, 2000.

[3] Shahnam Mirzaei, Anup Hosangadi, Ryan Kastner, "FPGA Implementation of High Speed FIR Filters Using Add and Shift Method", IEEE, 2006.

[4] Kousuke TARUMI, Akihiko HYODO, Masanori MUROYAMA, Hiroto YASUURA, "A design method for a low power digital FIR filter in digital wireless communication systems," 2004.

[5] Senthilkumar, A.M.Natarajan S.Subha, "Design and Implementation of Low Power Digital FIR Filters relying on Data Transition Power Diminution Technique" DSP Journal, Volume 8, pp. 21-29, 2008.

[6] Uwe Meyer-Baese, "Digital Signal with Field Programmable Gate Arrays", Springer-Verlag Berlin Heidelberg 2007

[7] C. N.Marimuthu, P. Thangaraj, Low Power High Performance Multiplier ICGST-PDCS, Volume 8, Issue 1, December 2008

[8] Kuan-Hung Chen, Member, and Yuan-Sun Chu, Member, A Spurious-Power Suppression Technique for Multimedia/DSP Applications

[9] R.I.HartleyandK ,.K.Parhi, Digit-Serial Computation, Kluwer Academic, Boston, MA, 1995.