International Journal of Computer Sciences and Engineering Open Access

Research Paper Volume-1, Issue-4 E-ISSN: 2347-2693

Implementation of Low Power Digital FIR Filter Design Based on low power multipliers and adders

U.Venkata Sivaiah[1*], P.Prasanna murali krishnna[2], Y.Devaraju[3]

[1*,2,3]*Department of ECE, Dr.S.G.I.E.T, Markapur, Andhra Pradesh, India*

Available online at www.ijcseonline.org

Received: 16 Dec 2013 Revised: 20 Dec 2013 Accepted: 20Dec 2013 Published: 30 Dec 2013

Abstract-This paper presents the methods to reduce dynamic power consumption of a digital Finite Impulse Response (FIR) filter these methods include low power serial multiplier and serial adder, combinational booth multiplier, shift/add multipliers, folding transformation in linear phase architecture and applied to fir filters to reduce power consumption due to this glitching is also reduced. The minimum power achieved is 110mw in fir filter based on shift/add multiplier in 100MHZ to 8taps and 8bits inputs and 8bits coefficients. The proposed FIR filters were synthesized implemented using Xilinx ISE Spartan 3E FPGA and power is analyzed using Xilinx XPower analyzer.

Keywords: Low Power, Booth Multiplier, Folding Transformation

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 baseband processing. Their approach is to optimize the bit width of each filter coefficient. They define the problem to find optimized bit width 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.

Corresponding Author: U.Venkata Sivaiah

II. 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 inputsignal,and y is the convolved output. The linear convolution process is formally defined by:

$$y[n] = x[n] * f[n] = \sum_{k=0} x[n]f[n-k] = \sum_{k=0} f[k]x[n-k]. \quad (1)$$

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:

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

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

$$Y(z)=F(z)X(z), \quad (3)$$

where F(z) is the FIR‟s transfer function defined in the z domain by L-1

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

The Lth-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 "transversalfilter," suggesting its "tapped delay line"structure[7].

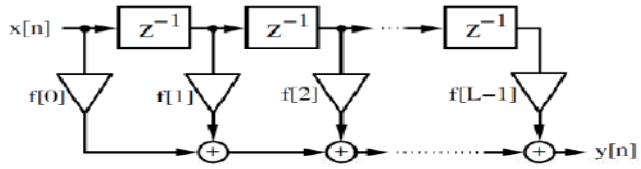


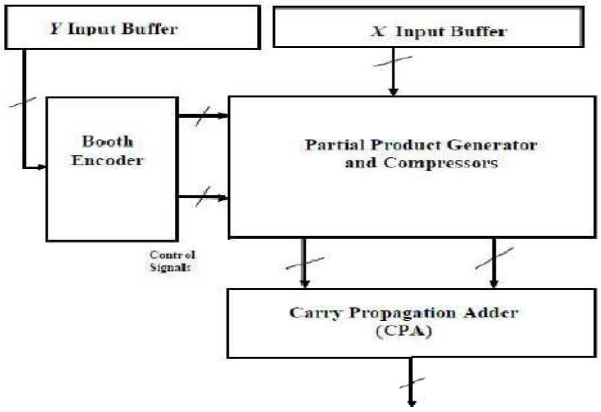
Fig. 1: FIR filter in the transposed structure.

III. IMPLEMENTATION FIR FILTER

In this paper, the linear-phase architecture and MAC architecture are considered. There are four design to reduce the power consumption.

Mac Fir Filter Based Booth Multiplier :

A multiplier has two stage. In the first stage, the partial products are generated by the booth encoder and the partial product generator(PPG), and are summed by compressors. In the second stage, the two final products are added to form the final product through a final adder.



$$Y=\sum_{i=0}^{n/2-1} vi4i=\sum_{i=0}^{n/2-1} (-2y2i+1 + y2i + y2i-1) 4i \quad (5)$$

module and compressor form the major part of the multiplier. Carry propagation adder (CPA) is the final adder used to merge the sum and carry vector from the compressor module. For radix-4 recoding, the popular algorithm is parallel receding or Modified Booth recoding . In parallel radix-4 recoding, Y becomes:

That truth table it shown in tableI.

TableI. Truth Table for Booth encoding

Y_{2i+1} Y_{2i} Y_{2i-1}	Booth op.	Dir.	Sht.	Add.
0 0 0	0x	0	0	0
0 0 1	1x	0	-	1
0 1 0	1x	0	-	1
0 1 1	2x	0	1	0
1 0 0	-2x	1	1	0
1 0 1	-1x	1	-	1
1 1 0	-1x	1	-	1
1 1 1	-0x	1	0	0

In our design we described Booth function as three basic operations, which they called „direction", „shift", and „addition" operation. Direction determined whether the multiplicand was positive or negative, shift explained whether the multiplication operation involved shifting or not and addition meant whether the multiplicand was added to partial products. The expressions for Booth encoding were stated below as :

$$Direction, D_{2i}=Y_{2i+1}; \quad (6)$$

$$Shift, S_{2i} = Y_{2i-1} \cdot (Y_{2i+1} \oplus Y_{2i}) + Y_{2i-1}' \cdot (Y_{2i+1} \oplus Y_{2i}) = Y_{2i+1} \oplus Y_{2i}; \quad (7)$$

$$Addition, A_{2i}=Y_{2i-1} \oplus Y_{2i}; \quad (8)$$

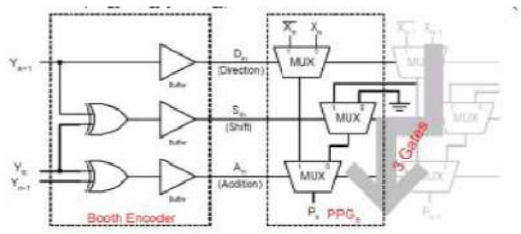


Fig.3: Booth encoder and PP(m=2i)

The Booth encoder was implemented using two XOR gates and the selector using 3MUXes and an inverter Careful optimization of the partial-product generation can lead to some substantial delay and hardware reduction.[8] In the normal 8*8 multiplication 8 partial products need to be generated and accumulated. For accumulation seven adders to reduce power are requiuerd but in the case of booth multiplier only 4 partial products are required to be generated and for accumulation three

adders, reduced delay required to compute partial sum and reduces the power consumption.[5]

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.4, 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.

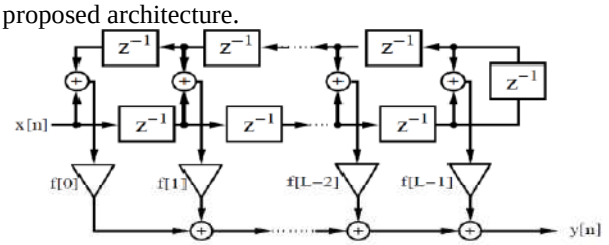


Fig.4: Linear-phase filter with reduced number of multipliers

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.5 show linear phase folding architecture fir filter.[2]

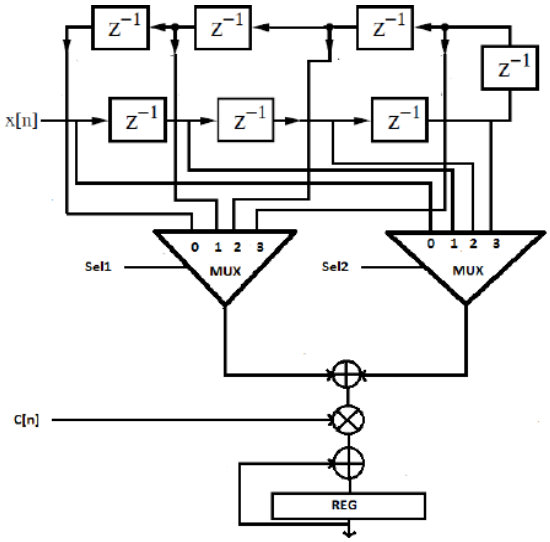


Fig.5: linear phase folding architecture fir filter.

Mac Fir Filter Based Low Power Serial Multiplier And Serial Adder:

Digit-serial implementation styles are best suited for implementation of digital signal processing systems which require reduce dynamic power consumption for desing MAC fir filter we use of low power serial multiplier and low power serial adder consider the bit-serial multiplier:shown in Fig.6 where the coefficient word length is four bits. this architecture contains four full adders, four multipliers, and some delay elements. in this multiplier the carry-out signal of every adder is feed back after a delay to the carry-in signal of the same adder. the critical path of this architecture is w full-adder delays. the traditional approach for designing the digit-serial structure involves unfolding this structure by a factor equal to the digit-size n . however, the resulting critical path would be w +n full-adder delays; which can be further reduced to n full-adder delays after pipelining reduction in the critical path below n full-adder delays is not possible because of the presence of feedback loops. the multiplier (which is just an and gate in the bitserial case) fig.6 show low power serial multiplier.[10]

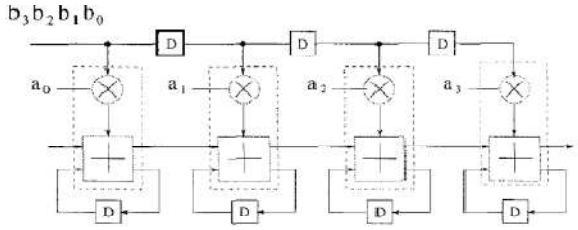


Fig.6: low power serial multiplier

and show in next page VHDL code low power serial adder is at the behavioural level. The interface description contains data and control input and outputs. A single process statement in the function description of low power serial adder encloses several sequential if-then-else statements.

```
library ieee; use ieee.std_logic_1164.all;
        use ieee.std_logic_signed.all;
        entity serial_adder is
        port(a,b,start,clock:in std_logic;
        ready:out std_logic;
        out_3:buffer std_logic_vector(15 downto 0));
        end serial_adder;
architecture ar_serial_adder of serial_adder is begin
process(clock)
        variable count:integer:=0;
        variable sum,carry:std_logic:='0';
        begin
        if(clock 'event and clock='1') then
```



```
        if start='1' then
        count:=0;
        carry:='0';
else
        if count<16 then
        count:=count+1;
        sum:=a xor b xor carry;
        carry:=(a and b)or (a and carry)or (b and carry);
        out_3<=sum&out_3(15 downto 1);
        end if;
        end if;
        if count=16 then
        ready<='1';
else
        ready<='0';
        end if;
        end if;
        end process;
        end ar_serial_adder;
```

Fir Filter Based Shift/Add Multiplier:

Fir filter is implemented using the shift and add method. We perform all our optimization in the multiplier block. The constant multiplications are decomposed in to additions and shifts and the multiplication complexity is reduced. Its possible to implement the design in the two form described below:

The coefficients are changed to integer getting multiplied to a multiple power of 10, then we arrange these coefficients the positive power of 3. This procedure is shown below in graph form. Graph branches stand for left shift and notches stand for sum.

First we arrange decimal coefficients according to negative and positive power of 2 (no need to them into integer). So the filter hardware and power consumption will reduce. Matlab code for implementation of conversion any number based on power of 2:

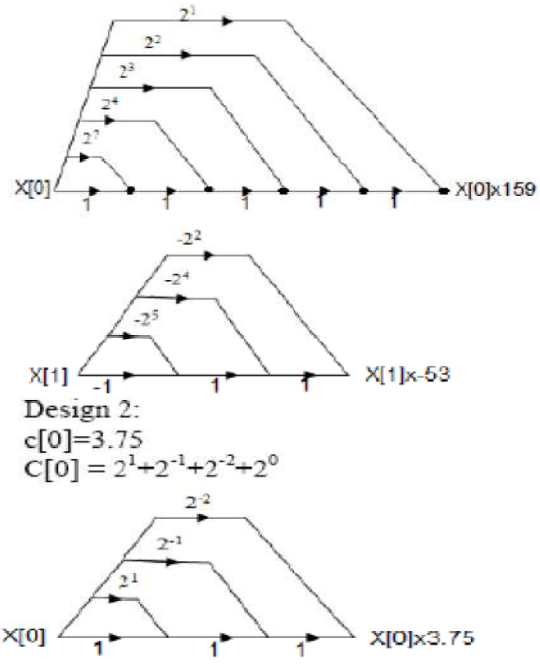
```
x=input('enter a number:');
        c=1;k=0;
        if(x==0)
        k(c)=0;
        else
        [i]=func(x);
        While (x>=1)
        k(c)=i;
        x=x-2^i;
        [i]=func(x);
        c=c+1;
        end
        end
        k
        function [i]=func(x)
        i=0;
        if (x==1)
        i=0;
        else
        while (x>=2)
        i=i+1;
        x=x/2;
        end
        end
        end
```

Design 1:

$c[0]= 0.159 , c[1]= -.053 \xrightarrow{x1000} c[0]= 159, c[1]= -53$

$c[0]= 128+16+8+2+1 = 2^7+2^4+2^3+2^2+2^1+2^0$

$c[1]= -32-16-4-1 = -2^5-2^4-2^2-2^0$



IV. COMPARISIONS

Designs equipped to 8bit and 16bits adders, 8bit multiplier and are accomplished via VHDL hardware description language by using Xilinx ISE software synthesized and implemented on FPGA in spartan3E

family. Also power is analized using Xilinx XPower analyzer. Tables [II,III,IV,V] shows the comparison between power consumption , numbers of LUTs, numbers of Slices and FFs and the type of device that has been used in different articles and the characteristic of our filters designed has been shown in tables VI,VII.
Table II. slices, LUTs and FFs Comparisons[3]
Table II.slices,LUTsandFFs Comparisons[3]

Table II.slices,LUTsandFFs Comparisons[3]

Filter(#taps) Virtex II [3]	Slices	LUTs	FFs
6	264	213	509
10	474	406	916
13	386	334	749

Table III. slices Comparisons[3]

Filter(#taps) Virtex IV [3]	Slices(add shift method)	Slices(MAC)
6	264	219
10	475	418
13	387	462

TableIV. Power Consumption [4]

Digital filter[4]	Power consumption(mw)
16-bits coefficient	1248
8-bits coefficient	502
Optimized bitwidth	450

TableV. Power Consumption8 taps, 16bits coeffician, 8bits input[5]

Freq	Signed array mult (mw)	Booth without DPDT (mw)	Booth with DRD	Booth with DPDT using REG (mw)	Booth with DPDT using and gate (mw)
25	612	469	423	326	868
50	1170	879	851	596	669
75	1773	1297	1256	881	1018
100	2293	1703	1658	1137	1283

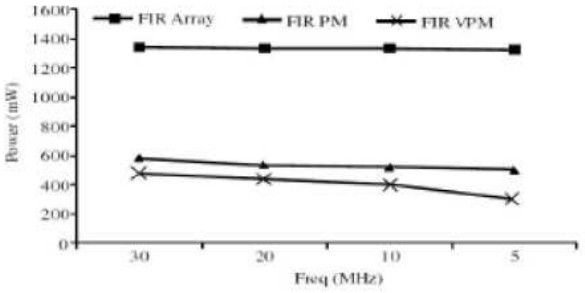


Fig.7: Power consumption of 8bits and 8 tap FIR filter[6]

TableVI. Power Consumption proposed filter

Proposed filter	25 MHZ	50 MHZ	75 MHZ	100MHZ
MAC_booth(mw)	100	140	180	210
Linear pheas_folding_booth(mw)	110	150	190	230
Shift/add_form1(mw)	110	150	190	230
Shift/add_form2(mw)	90	120	140	110
Serial multiplier and serial adder (mw)	112	126	141	155
Fir_base(8bits,8taps) (mw)	130	190	250	300

TableVII. characteristic of our filters designed

Proposed filter	Slices	FFs	LUTs	Dsp	Latch	Frequency (MHZ)
MAC_booth	Reg 19	14	31	0	5	461.681
Linear pheas_folding_booth	Reg 20	68	97	0	6	145.896
Shift/add_form1	101	48	119	0	0	79.171
Shift/add_form2	119	72	208	0	0	54.669
Serial multiplier and serial adder	103	97	162	0	0	225
Fir_base(8bits,8taps)	92	64	112	8	0	1040.583

V. CONCLUSION

In This paper we presented a low power and low area FIR filter. For reduce power consumption and area we are using of combination booth multiplier, low power serial multiplier and serial adder folding transformation in linear phase architecture. These filters were compared for area and power with other common implementations 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 synthesized and implemented using Xilinx ISE Spartan-3E FPGA and power is analyzed using Xilinx Power analyzer.

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 _lter indigital wireless communication systems,” 2004.

[5]. "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]. A. Senthilkumar, 2A.M. Natarajan, "FPGA Implementation of Power Aware FIR Filter Using Reduced Transition Pipelined Variable Precision Gating," Journal of Computer Science , pp. 87-94, 2008.

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

[8]. Shibi Thankachan, "64 x 64 Bit Multiplier Using Pass Logic",2006.

[9]. Ronak Bajaj, Saransh Chhabra, Sreehari Veeramachaneni, M B Srinivas, "A Novel, Low-Power Array Multiplier Architecture

[10]. Yun-NanChang, Janardhan H. Satyana:rayanaKeshabK.Parhi" LOW-POWER DIGIT-SERIAL MULTIPLIERS", 1997 IEEE. International Symposium on Circuits and Systems, June i3-12,1997

