International Journal of Engineering In Advanced Research Science and Technology ISSN: 2278-256

November 2016
VOLUME -2
ISSUE-4
Page:7926-32

POWER OPTIMIZED FAULT TOLERANT FIR FILTERS USING MODIFIED HAMMING AND BOOTH ENCODING

1. N.SUJEETH RAMA SHARMA, 2.R.SATEESH KUMAR, 3. K.SRINIVAS
1. PG Scholar, Dept of ECE, Bonam Venkata Chalamayya institute of technology & science, amalapuram
2. Assistant Professor, Dept of ECE, Bonam Venkata Chalamayya institute of technology & science, amalapuram
3. Associate Professor, Dept of ECE, Bonam Venkata Chalamayya institute of technology & science, amalapuram

ABSTRACT:
The main objective of this project is to design an efficient and error tolerant FIR filter using VHSCIHDL. In many cases, some of those elements operate in parallel, performing the same processing on different signals. A typical example of those elements is digital filters. A scheme based on error correction coding has been recently proposed to protect parallel filters. In that scheme, each filter is treated as a bit, and redundant filters that act as parity check bits are introduced to detect and correct errors. Further, this project is enhanced with modification in multiplier design. Here, in this context Radix8 modified booth encoding algorithm is used to reduce further power.
KEYWORDS: Modified Hamming, Fault tolerant, Soft ware defined radio, Finite impulse response, on canonical signed digit, Modified booth encoding.

INTRODUCTION:
FIR DIGITAL filters find extensive applications in mobile communication systems for applications such as channelization, channel equalization, matched filtering, and pulse shaping, due to their absolute stability and linear phase properties. The filters employed in mobile systems must be realized to consume less power and operate at high speed. Recently, with the advent of software defined radio (SDR) technology, finite impulse response (FIR) filter research has been focused on reconfigurable realizations. The fundamental idea of an SDR is to replace most of the analog signal processing in the transceivers with digital signal processing in order to provide the advantage of flexibility through reconfiguration. This will enable different air-interfaces to be implemented on a single generic hardware platform to support multi standard wireless communications [1]. Wideband receivers in SDR must be realized to meet the stringent specifications of low power consumption and high speed. Reconfigurability of the receiver to work with different wireless communication standards is another key requirement in an SDR. The most computationally intensive part of an SDR receiver is the channelizer since it operates at the highest sampling rate [2]. It extracts multiple narrowband channels from a wideband signal using a bank of FIR filters, called channel filters. Using polyphase filter structure, decimation can be done prior to channel filtering so that the channel filters need to operate only at relatively low sampling rates. This can relax the speed of operation of the filters to a good extent [22]. However due to the stringent adjacent channel attenuation specifications of wireless communication standards, higher order filters are required for channelization and consequently the complexity and power consumption of the receiver will be high. As the ultimate aim of the future multi-standard wireless communication receiver is to realize its functionalities in mobile handsets, where its full utilization is possible, low power and low area implementation of FIR channel filters is inevitable. In [37], the filter multiplications are done via state machines in an iterative shift and add component and as a result of this there is huge savings in area. For lower order filters, the approach in [37] offers good trade-off between speed and area. But in general, the channel filters in wireless communication receivers need to be of high order to achieve sharp transition band and low adjacent channel attenuation requirements. For such applications, the approach in [37] results in low speed of operation. The complexity of FIR filters is dominated by the complexity of coefficient multipliers. It is well known that the common subexpression elimination (CSE) methods based on canonical signed digit (CSD) coefficients produce low complexity FIR filter coefficient multipliers [3]. The goal of CSE is to identify multiple occurrences of identical bit patterns that are present in the CSD representation of coefficients, and eliminate these redundant

International Journal of Engineering In Advanced Research Science and Technology ISSN: 2278-256

November 2016
VOLUME -2
ISSUE-4
Page:7926-32

multiplications. A modification of the 2-bit CSE technique in [3] for identifying the proper patterns for elimination of redundant computations and to maximize the optimization impact was proposed in [4]. In [5], the technique in [3] was modified to minimize the logic depth (LD) (LD is defined as the number of adder-steps in a maximal path of decomposed multiplications [27]) and thus to improve the speed of operation. In [6], we have proposed the binary common subexpression elimination (BCSE) method which provided improved adder reductions and thus low complexity FIR filters compared to [3]–[5]. In [7], a method based on the pseudo floating point method was used to encode the filter coefficients and thus to reduce the complexity of the filter. But the method in [7] is limited to filter lengths less than 40. In general, the methods in [3]–[7] are only suitable for application specific filters where the coefficients are fixedand hence not suitable for reconfigurable filters. Several implementation approaches for reconfigurable FIR filters have been proposed in literature [8]–[15]. These designs include either a fully programmable multiply-accumulate (MAC) based filter processor or dedicated architectures where the filter coefficients can be stored in registers. The architecture of a filter processor consists of a datapath with a single MAC unit, data and program memories, and a control unit [8], [9]. The datapath includes a 16-bit adder/subtractor, a multiplier, and a 32-bit accumulator. The performance of the processor is mainly restricted by the delay of this datapath, more specifically that of the multiplier. The main disadvantage of the filter processors is that the area and power requirements are significantly large. In [10], a comparison was done for the performance of speech based algorithms on dedicated architectures and general-purpose processors. It was shown that the power consumption for a general-purpose processor can be a factor of four times more than dedicated architectures for a complex algorithm [10]. The works in [11]–[15] and [20], [21] present reconfigurable FIR filter architectures. In [11], a CSD based digit reconfigurable FIR filter architecture was proposed. This architecture was independent of the number of taps because the number of taps and non-zero digits in each tap were arbitrarily assigned. The intention of the authors was to reduce the precision of coefficients and thus the filter complexity without affecting the filter performance.

FINITE IMPULSE RESPONSE:

In signal processing, a finite impulse response (FIR) filter is a filter whose impulse response (or response to any finite length input) is of finite duration, because it settles to zero in finite time. This is in contrast to infinite impulse response (IIR) filters, which may have internal feedback and may continue to respond indefinitely (usually decaying).

The impulse response (that is, the output in response to a Kronecker delta input) of an Nth-order discrete-time FIR filter lasts exactly N + 1 samples (from first nonzero element through last nonzero element) before it then settles to zero.

FIR filters can be discrete-time or continuous-time, and digital or analog.

For a causal discrete-time FIR filter of order N, each value of the output sequence is a weighted sum of the most recent input values:

$$y[n] = b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N]$$

$$= \sum_{i=0}^{N} b_i \cdot x[n-i],$$

where:

$x[n]$ is the input signal,

$x[n]$ $y[n]$ is the output signal,

- N is the filter order; an Nth-order filter has $(N+1)$ terms on the right-hand side
- b_i is the value of the impulse response at the i'th instant for $0 \le i \le N$ of an Nth-order FIR filter. If the filter is a direct form FIR filter then b_i is also a coefficient of the filter .

This computation is also known as discrete convolution.

The $x[n-i]$ in these terms are commonly referred to as taps, based on the structure of a tapped delay line that in many implementations or block diagrams provides the delayed inputs to the multiplication operations. One may speak of a 5th order/6-tap filter, for instance.

MULTIPLICATION ALGORITHM:

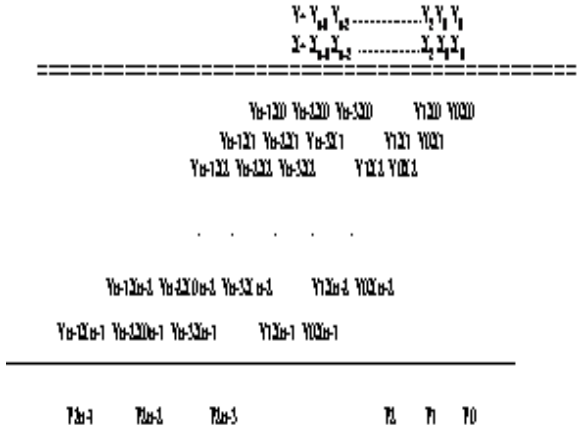
The multiplication algorithm for an N bit multiplicand by N bit multiplier is shown below:

International Journal of Engineering In Advanced Research Science and Technology ISSN: 2278-256

November 2016
VOLUME -2
ISSUE-4
Page:7926-32

Y= Y_{n-1} Y_{n-2} .....Y_2 Y_1 Y_0 Multiplicand
X= X_{n-1} X_{n-2} ..... X_2 X_1 X_0 Multiplier

Generally



AND gates are used to generate the Partial Products, PP, If the multiplicand is N-bits and the Multiplier is M-bits then there is N* M partial product. The way that the partial products are generated or summed up is the difference between the different architectures of various multipliers.

Example **1101** **4-bits**

1101 **4-bits**

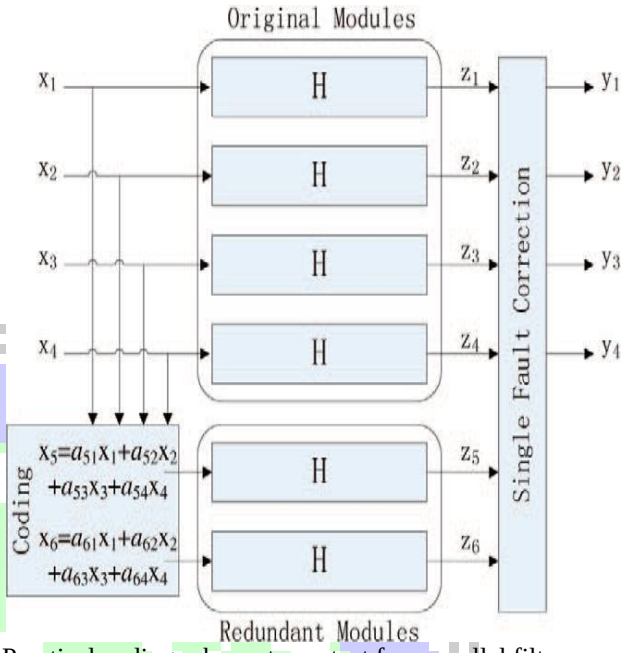
1101

0000

1101

1101

10101001



Practical coding scheme to protect four parallel filters

The corresponding *A* matrix
is the identity matrix on the first four rows, and only the last
two rows have generic coefficients. The matrix is

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{pmatrix}.$$

Error check matrix is given as

$$\bar{e}_{\text{sim}} = \begin{bmatrix} p_1 - p_2 \\ 2p_1 - p_2 \\ 3p_1 - p_2 \\ 4p_1 - p_2 \end{bmatrix}$$

Thus, in total, the scheme requires only six multiplications. This shows that the error location logic can be efficiently implemented. Finally, when an error is detected, it can be corrected by re-computing the affected filter output using $z5$ and the remaining original filter outputs. For example, for an error in filter 1, correction is implemented as
$z\text{corrected}1 = z5 - (z2 + z3 + z4)$.

International Journal of Engineering In Advanced Research Science and Technology ISSN: 2278-256

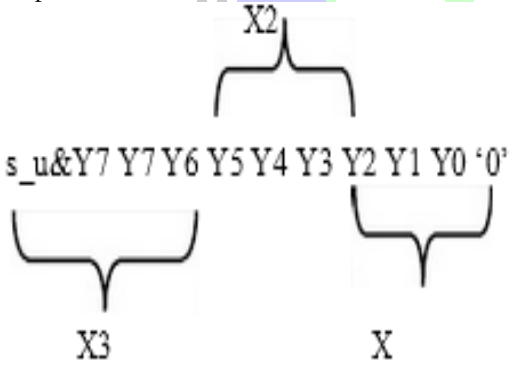
November 2016
VOLUME -2
ISSUE-4
Page:7926-32

RADIX-8 MODIFIED BOOTH ALGORITHM:

Booth's algorithm involves repeatedly adding one of two predetermined values to a product P, and then performing a rightward arithmetic shift on P.

Multiplier architecture comprise of two architectures, i.e., Modified Booth and Wallace tree. Based on the study of various multiplier architectures, we find that Modified Booth increases the speed because it reduces partial products to half. Further, the delay in multiplier can be reduced by using Wallace tree. Power consumption of Wallace tree multiplier is also less as compared to booth and array. Features of both multipliers can be combined to produce high speed and low power multiplier. Modified Booth multiplier consists of Modified Booth Recorder (MBR). MBR have two parts, i.e., Booth Encoder (BE) and Booth Selector (BS). The basic operation of BE is to decode the multiplier signal and output will be used by BS to generate the partial product. The partial products are then, added with the Wallace tree adders, similar to the carry save adder approach. The last row of carry and sum output is added together by carry look-ahead adder with the carry skewed to the left by position.

Radix-8 Booth encoding is most often used to avoid variable size partial product arrays. Before designing Radix-8 BE, the multiplier has to be converted into a Radix-8 number by dividing them into four digits respectively according to Booth Encoder Table given after wards. Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) of the multiplier.



Radix 8 Booth recoding applies the same algorithm as that of Radix 4, but now we take quartets of bits instead of triplets. Each quartet is codified as a signed digit using below Table. Radix 8 algorithm reduces the number of partial products to n/3, where n is the number of multiplier bit s. Thus it allows a ti me gain in the partial products summation Radix-8 recoding applies the same algorithm as radix-4, but now we take quartets of bits instead of triplets. Each quartet is codified as a signed-digit using the table

Quartet value	*Signed-digit value*
0000	0
0001	+1
0010	+1
0011	+2
0100	+2
0101	+3
0110	+3
0111	+4
1000	-4
1001	-3
1010	-3
1011	-2
1100	-2
1101	-1
1110	-1
1111	0

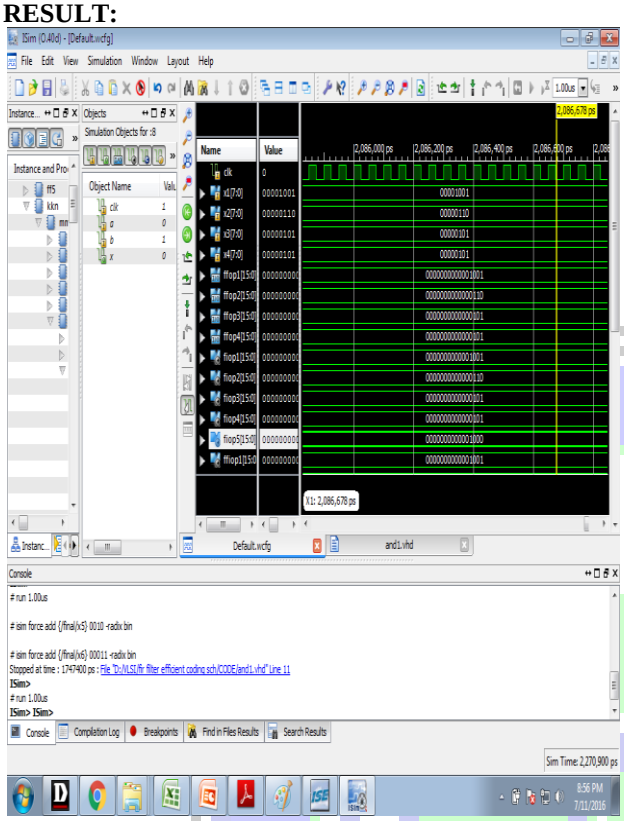
Here we have an odd multiple of the multiplicand, 3Y, which is not immediately available. To: generate it we need to perform this previous add:2Y+Y=3Y. But we are designing a multiplier for specific purpose and thereby the multiplicand belongs to a previously known set of numbers which are stored in a memory chip. We have tried to take advantage of this fact, to ease the bottleneck of the radix-8 architecture, that is, the generation of 3Y. In this manner we try to attain a better overall multiplication time, or at least comparable to the time we could obtain using a radix-4 architecture (with the additional advantage of using a less number of transistors). To generate 3Y with 21-bit words we only have to add 2Y+Y, that is, to add the number withthe same number shifted one position to the left.

A product formed by multiplying the multiplicand by one digit of the multiplier when the multiplier has more than one digit. Partial products are used as intermediate steps in calculating larger products.

International Journal of Engineering In Advanced Research Science and Technology ISSN: 2278-256

November 2016
VOLUME -2
ISSUE-4
Page:7926-32

RESULT:



PARAMETER	POWER
EXISTING	495 mw
PROPOSED	318 mw

CONCLUSION:

A scheme based on error correction coding has been recently proposed to protect parallel filters. In that scheme, each filter is treated as a bit, and redundant filters that act as parity check bits are introduced to detect and correct errors. In this brief, the idea of applying coding techniques to protect parallel filters is addressed in a more general way. In particular, it is shown that the fact that filter inputs and outputs are not bits but numbers enables a more efficient protection. This reduces the protection overhead and makes the number of redundant filters independent of the number of parallel filters.

REFERENCES:

[1] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Englewood Cliffs, N.J., USA: Prentice Hall, 1993.

[2] A. Sibille, C. Oestges and A. Zanella, *MIMO: From Theory to Implementation*, New York, NY, USA: Academic, 2010.

[3] N. Kanekawa, E. H. Ibe, T. Suga and Y. Uematsu, *Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and Electro-Magnetic Disturbances*, New York, NY, USA: Springer Verlag, 2010.

[4] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005.

[5] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124–134, Mar. 1984.

[6] A. Reddy and P. Banarjee "Algorithm-based fault detection for signal processing applications," *IEEE Trans. Comput.*, vol. 39, no. 10, pp. 1304–1308, Oct. 1990.

[7] S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, "Totally fault tolerant RNS based FIR filters," in *Proc. IEEE IOLTS*, 2008, pp. 192–194.

[8] Z. Gao, W. Yang, X. Chen, M. Zhao and J. Wang, "Fault missing rate analysis of the arithmetic residue codes based fault-tolerant FIR filter design," in *Proc. IEEE IOLTS*, 2012, pp. 130–133.

[9] B. Shim and N. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 14,no. 4, pp. 336–348, Apr. 2006.

[10] Y.-H. Huang, "High-efficiency soft-error-tolerant digital signal processing using fine-grain subword-detection processing," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 18, no 2, pp. 291–304, Feb. 2010.

[11] P. Reviriego, C. J. Bleakley, and J. A. Maestro, "Structural DMR: A technique for implementation of soft-error-tolerant FIR filters," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 58, no. 8, pp. 512–516, Aug. 2011.

[12] P. Reviriego, S. Pontarelli, C. Bleakley and J. A. Maestro, "Area efficient concurrent error detection and correction for parallel filters," *IET Electron. Lett.*, vol. 48, no 20, pp. 1258–1260, Sep. 2012.

[13] Z. Gao *et al.*, "Fault tolerant parallel filters based on error correction codes," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 23, no. 2, pp. 384–387, Feb. 2015.

[14] R. W. Hamming, "Error correcting and error detecting codes," *Bell Sys. Tech. J.*, vol. 29, pp. 147–160, Apr. 1950.