FIR FILTER ARCHITECTURE IMPLEMENTATION USING NEW DOUBLE BASE NUMBER SYSTEM FOR BETTER PERFORMANCE

T. GAYATHRI¹, G. RAMESH², V. USHA³.
(¹Department of ECE NOVA INSTITUTE OF TECHNOLOGY, ELURU, INDIA)
(²Department of ECE NOVA INSTITUTE OF TECHNOLOGY, ELURU, INDIA)
(³Department of ECE NOVA INSTITUTE OF TECHNOLOGY, ELURU, INDIA)

Abstract: High Speed Complex multipliers are the stumbling blocks in programmable finite impulse response (FIR) digital filters. As the filter coefficients change either dynamically or periodically, the search for common sub expressions for multiplier less implementation needs to be performed over the entire gamut of integers of the desired precision, and the amount of shifts associated with each identified common sub expression needs to be memorized. The complexity of a quality search is thus beyond the existing design algorithms based on conventional binary and signed digit representations. This paper presents a new design paradigm for the programmable FIR filters by exploiting the extended double base number system (EDBNS). Due to its sparsity and innate abstraction of the sum of binary shifted partial products, the sharing of HCSLA adders in the time-multiplexed multiple constant multiplication blocks of the programmable FIR filters can be maximized by a direct mapping from the quasi-minimum EDBNS. The multiplexing cost can be further reduced by merging double base terms. In our Proposed paper we are replacing shifters in the second level is replaced by Barrel shifter so the delay can be reduced Logic synthesis results on more than one hundred programmable filters with filter taps ranging from 10 to 100 and co-efficients word lengths of 8 have been verified.

Keywords - Barrel Shifter, Double based number system, FIR filter, High Speed Carry Select adder, Programmable filter.

1. INTRODUCTION

FINITE impulse response (FIR) filters offer many advantages, such as easily attainable linear phase response, computational efficiency in multi-rate applications and desirable numerical property for finite precision and fractional arithmetic [2]–[5]. Adaptive filters, in particular, are inevitable in many important applications in communications, image processing, computer vision, data acquisition and control [6]–[10]. With any adaptive filter, there is a requirement for a programmable filter, which is a primary reason behind the increasing dominance of digital instead of analog system implementations. In applications such as multi-rate decimation [7], discrete cosine transform [8], [9], channelization [10], [11], high efficiency video coding (HEVC) [12], wide bandwidth photonic filter [13] and high-rate communication [14], the filter coefficients need to be run-time reconfigurable by the error feedback signal or adapt-able to varying filtering specifications in real time. Owing to the huge search space over the complete gamut of integer values of a given precision, the intricacy of common sub-expression sharings within and across multiple sets of coefficients in a TM-MCM block is beyond the capability of existing CSE algorithms and fixed-coefficient filter design methodologies even for moderate coefficient word length and filter taps. As the shifters are no longer fixed, more succinct number representation than CSD and binary are explored. DBNS and multidimensional logarithmic number system (MDLNS) are considered for the design of DSP operators such as constant multiplier [11]–[13].

In [10], logarithmic and double base number representations are used to synthesize inexact fractional filter coefficients for time-invariant filters. To minimize the additions of each coefficient multiplier, multiple-radix DBNS representation is used by limiting the maximum power of the second base number in [12]. In this paper, the double base number representation is uniquely exploited to maximize the sub expression sharing for all filter coefficients of a given word length in a more general time-varying filter implementation problem that has no leverage of the fixed quantized coefficients at design time. This work is an extension of our antecedent work [14], which is the first ever attempted to harness the sparseness of the canonical DBNS representation for the complexity reduction of TM-MCM block. In this paper, we have extended the double base number system (DBNS) [12] to encapsulate the bi-nary shifts in tandem with several most frequently encountered common sub-expressions. A new formulation of the common sub-expression search problem as a quasi-minimized extended
DBNS (EDBNS) generation problem is proposed, which has led to considerable reduction in the number of distinct partial products for a given word length of programmable coefficients. With this number system, an efficient architecture for the implementation of TM-MCM is derived as shown in Fig. 1. It consists of a power-of-2 generator (POBG), N blocks of power-of-2 selector (POBS) and N blocks of double base coefficient generator (DBCG), where b is the second base number in EDBNS and N is the number of taps. In our design, those unique partial product terms can be generated incrementally with one adder each. The sizes of the multiplexers in the programmable units and the lookup tables for the selector logic are further minimized by exploiting the unique properties of EDBNS in the exponential equation.

### II DISTRIBUTED ARITHMETIC

Finite impulse response (FIR) filters are one of the most fundamental components in digital signal processing. Many simplifications in their hardware implementation can be made when the coefficients are constant. However, reconfigurable FIR filters for which the coefficients can be changed in run-time are required in many application scenarios like, e.g., Software defined radios (SDR). This motivates the work of many researchers to extend optimization methods that were developed in the context of multiple constant multiplications (MCM) to reconfigurable multiplier blocks [1]–[8]. Such multiplier blocks are usually realized using additions, subtractions and shifts only. In a reconfigurable multiplier block, additional multiplexers are inserted in the data path to configure the multiplication with a lot of resources of a finite set of coefficients.

#### 2.1 EXPLANATION ON DISTRIBUTED ARITHMETIC:

The fundamental operation of a digital filter with N taps is the inner product of two vectors which can be represented as a sum-of-products of its components

\[ y = c \cdot x = \sum_{n=0}^{N-1} c_n x_n \]

where \( c_n \) are usually constants and \( x_n \) are the time-shifted input samples. If each \( x_n \) is represented as a binary \( B_x \) bit 2'th complement number, where \( x_{n,b} \) denotes the \( b \)th bit of \( x_n \), (1) can be rewritten

\[ y = \sum_{n=0}^{N-1} c_n x_{n,b} \]

where \( x^{-N}_{n,b} = (x_{0,b}; \ldots; x_{N1,b})^T \) is a bit vector of length N containing the \( b \)th bit of each element of \( x \).

Fig. 1. Sequential realization of a distributed arithmetic FIR filter.

The function can be pre-computed and stored in a single LUT with \( N \) inputs. The storage requirement of the LUT is \( B_{fN}^N \) bit, where \( B_{fN}^N \) denotes the output word size of the \( N \)-input LUT \( f(-x^{-N}_{N,b}) \). The inner product can now be obtained by accumulating the shifted outputs of the LUT according to (3). A sequential realization of (3) which computes a valid output every \( N \) samples is shown in Fig. 1. For higher throughput, a parallel implementation using \( B_x \) LUTs can be obtained by unfolding. So far, this \( N \)-input LUT cannot be directly mapped to the reconfigurable 4/5-input CFGLUTs described above. Therefore, a method to reduce the LUT input size [11] was used to break the \( N \)-input LUT into several 4/5-input LUTs which is described in the following.

**A. Dividing LUTs into Smaller Partial LUTs**

The input size of the LUT can be reduced by splitting the sum in (4) into several smaller sums can be realized by partial \( L \)-input LUTs. If \( N \) is not dividable by \( L \), one additional partial LUT of size \( L^{N-L} \) is necessary, which is represented with the last term in (5). By setting \( L = 4 \) or \( L = 5 \), the LUT \( f(-x^{-N}_{N,b}) \) can be directly mapped to CFGLUTs by using the decomposition of (5). Furthermore, this method reduces the LUT storage requirements for the \( N \)-input LUT \( f(-x^{-N}_{N,b}) \) from \( B_{fN}^N \) to \( B_{fN}^L \) \( 2^4 \) bits to \( bN=Lc \) \( B_{fN}^L \) \( 2^4 \) \( 2^{10} \) bits. Note that for parallel DA, the \( N \)-input LUT is used \( B_x \) times. For a fixed \( L \), this realization style grows linear with the number of filter taps \( N \) in contrast to (4) which grows exponentially. This memory reduction is paid by \( bN=Lc \) additional adders.
2.2 RECONFIGURABLE ARCHITECTURES:

2.2.1. Resource Optimized Architecture:

Modern DSP systems are well suited for VLSI implementation. Indeed, they are often technically feasible or economically viable only if implemented using VLSI technologies. Many DSP systems are produced in very large numbers and require high performance circuits with respect to throughput and power consumption. The combined advances in system design capability and VLSI technology have made it possible to economically design unique integrated circuits or use in dedicated applications, so called Application Specific Integrated Circuits (ASIC). The possibility of incorporating a whole signal processing system into an chip has multitude of effects. It will dramatically increase the processing capacity and simultaneously reduce the size of the system, the power consumption.

Finite-impulse response (FIR) Filter is extensively used in wireless sensor networks as a signal pre-processing step. Because sensor nodes require a long working periods and ultra-low cost, traditional FIR structures are inapplicable as multipliers occupy too much die size for such node’s chips. This paper proposes novel FIR filter structures used in the design of application specific integrated circuits (ASICs) for sensor nodes, which can reduce the hardware cost to a minimum. The experiments show that the proposed FIR structure can lead to significant hardware savings from the traditional FIR filter. It’s a better choice for sensor node ASICs.

Memory Based structures

The phrase we use “memory-based structures” or “memory-based systems” for those systems where memory elements like RAM or ROM is used either as a part or whole of an arithmetic unit. Memory-based structures are more regular compared with the multiply-accumulate structures; and have many other advantages, e.g., greater potential for high-throughput and reduced-latency implementation, (since the memory-access-time is much shorter than the usual multiplication-time) and are expected to have less dynamic power consumption due to less switching activities for memory-read operations compared to the conventional multipliers. Memory-based structures are well-suited for many digital signal processing (DSP) algorithms, which involve multiplication with a fixed set of coefficients.

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 (LUT). 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 -N-point vector with any possible N-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. Attempts have 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 well as the number of adders and latches.

III PROPOSED FIR FILTER

Proposed FIR filter consists of 3 blocks those are

1. POBG (Power-of- generator) 2.POBS (Power-of – b selector) 3. DBCG (double base coefficient generator)
The shifters in DBCG block shift the T outputs from POBS to generate the double base terms 
$2^{\alpha_t}b_t^{\beta_t}$ for $t=1,2,\ldots,T$, and the subsequent adder tree sums up the T terms to produce the coefficient multiplier output $h[n]x[n]$ of each filter tap. Given $T$ and $F_{\text{min}}$, the EDBNS representations of all-bit coefficients that result in the minimum number of different $\alpha_t$ for each of the POBS outputs are selected. Thus, each shifter needs only to realize different amounts of shift, where is determined by the minimum number of distinct exponents $\alpha_t$ that can appear in the $i$-th double base term. Each POBS output is hardwire-shifted by different numbers of bits and fed into the data inputs of a multiplexer. One of these shifted versions of the POBS output will be selected from each multiplexer to compose the EDBNS representation of the programmed coefficient. The control signals to the multiplexers and the programmable Barrel shifters are generated by an external look up table (LUT). Because the EDBNS representation of any even integers can be obtained by a double base scalar multiplication of a factor and an odd integer, i.e. $c=2^\alpha x c_|$, where is an even number and $c_|$ is known as a fundamental, the control information of the even integers are stored together with their fundamentals in the LUT, $2^\alpha$ plus a factor stored for each even number. This will reduce the LUT size by almost half.

The complete design procedure is summarized as follows:

1: Compute $T_{\text{min}}$ and $F_{\text{min}}$ for all $i$-bit coefficients. Generate the EDBNS array for all the $i$-bit coefficients using the search algorithm presented.

2: Implement the POBG block by producing all power-of- integers in $F_{\text{min}}$ using the method described.

3: Design the POBS block with $T$ multiplexers. Each power-of- integer from the POBG block is first connected to an input of $K$ different multiplexers, where $k$ is the maximum number of times that power-of- value can
appear in the EDBNS representation of any coefficient. Then, minimize the number of input lines to the multiplexers of POBS block by the algorithm presented in Fig.

4: Design T programmable Barrel shifters for the DBCG block. Extract the amount of shifts $\alpha$ for each power-of-integer and store it in the LUT addressable by the fundamentals.

5: Sum the T double base terms in DBCG by a modified carry select adder (MCSLA) tree to reduce the delay.

The immediate structure is by and large favored in light of its higher execution and force proficiency. The issue of planning Poly-Phase decimator has gotten an extraordinary consideration because of expansive number of duplications. This execution must fulfill the authorized examining rate limitations of the constant DSP applications and must require less space and power utilization. Present works have concentrated on outline of Multirate Poly-Phase decimator by channels, information generator locks and viper. As the coefficients of an application particular channel are steady, the decay is more proficient than utilizing multipliers. The multifaceted nature of FIR channels for this situation is ruled by the quantity of increases and duplications. The multiplier square of the advanced FIR channel in its immediate structure is actualized in the configuration so that critical effect on the many-sided quality and execution of the outline will be made strides. Additionally, Poly-Phase channel is outlined utilizing MCM and digit serial adders which overcome issue of multifaceted nature, plan execution and creating low region.

A barrel shifter is a digital circuit that can shift a data word by a specified number of bits without the use of any sequential logic, only pure combinatorial logic. One way to implement it is as a sequence of multiplexers where the output of one multiplexer is connected to the input of the next multiplexer in a way that depends on the shift distance. A barrel shifter is often used to shift and rotate n-bits in modern microprocessors, typically within a single clock cycle. A common usage of a barrel shifter is in the hardware implementation of floating-point arithmetic. For a floating-point add or subtract operation, the significands of the two numbers must be aligned, which requires shifting the smaller number to the right, increasing its exponent, until it matches the exponent of the larger number. This is done by subtracting the exponents, and using the barrel shifter to shift the smaller number to the right by the difference, in one cycle. If a simple shifter were used, shifting by n bit positions would require n clock cycles.

Fig 4: Barrel Shifter of 4 bit diagram
Fig 5: Barrel Shifter of 8 bit diagram

IV. FIGURES OF RESULTS

Fig 6: POBG (Power-of-generator) Output

Fig 7: POBS (Power-of-b-selector) Output
Fig 8 : DBCG (double base coefficient generator) Output

Fig 9: Final 8 bit FIR FILTER Output

IV. CONCLUSION & FUTURE SCOPE

The method of sub expression has the capability of affecting noteworthy reserve funds in the quantities of augmentations utilized as a part of the usage of FIR channels. It has been watched that different procedures using normal sub expressions in coefficient representations other than CSD might on events lead to less adders, and as a rule can accomplish comparative investment funds. Preparatory studies have demonstrated that just a predetermined number of SPT terms are required to meet a respectable arrangement of particulars if a decent enhancement method exists. Henceforth, to speak to the coefficients of a channel thusly, the coefficient multipliers can be supplanted by a little number of include/subtract-shift operations. The equipment many-sided quality is in this manner to a great extent lessened. By utilizing MILP snake use is further minimized.

There are various techniques generated for designing of FIR filters. Every method has its own merits and demerit. For example design linear-phase FIR filters by a novel weighted BP neural networks algorithm has some limitations. It is not involved in operation of inverse matrix and the window method is also have some limitations like they are not very suitable for designing of filters with any given frequency response. The future work is develop a technique suitable for designing of filters with a given magnitude response and reduce the noise of signal.

REFERENCES


