An Improved Synthesis Method for Low Power Hardwired FIR Filters

Vagner S. Rosa
Informatics Inst. UFRGS
PO Box. 15064
Porto Alegre, RS, Brazil
+55(51)3316-6165
vsrosa@inf.ufrgs.br

Eduardo Costa
Univ. Católica de Pelotas
Felix da Cunha, 412
Pelotas, RS, Brazil
+55(53)2848287
ecosta@atlas.ucpel.tche.br

José C. Monteiro
IST/INESC
Alves Redol, 9
Lisbon, Portugal
+351(21)3100283
jcm@inesc-id.pt

Sergio Bampi
Informatics Inst. - UFRGS
PO Box. 15064
Porto Alegre, RS, Brazil
+55(51)3316-6165
bampi@inf.ufrgs.br

ABSTRACT

This work presents a method to design parallel digital finite impulse response (FIR) filters for hardwired (fixed coefficients) implementation with reduced number of adders and logic depth in the multiplier block. The proposed method uses a combination of two approaches: first, the reduction of the coefficients to N-Power-of-Two (NPT) terms, where N is the maximum number of bits in '1' state allowed for each coefficient and Common Subexpression Elimination (CSE) among multipliers. An algorithm for selecting the best NPT coefficient set for a given filter specification is proposed. Initially, a floating point coefficient set is generated using classical methods for FIR filters and then several sets of fixed point coefficients are generated by rounding the result of the floating point coefficients multiplied by a scale factor different for each set. The coefficient sets are then converted to NPT and a frequency response for each set is obtained. Based on the frequency response, the algorithm selects the best set. This set is then used as input for a CSE algorithm, which eliminate all common subexpressions among the multipliers and generates a hardware description of the filter in VHDL for synthesis purpose. The results show significant reduction in the number of adders and logic depth of the multiplier block with a minimal degradation in the filter transfer characteristics, showing the usefulness of the proposed method for low power design of parallel filters.

Categories and Subject Descriptors

B.2.1 [**Arithmetic and Logic Structures**]: Design Styles – *Parallel.*

General Terms

Algorithms, Performance, Experimentation.

Keywords

Parallel FIR filter, Power-of-two, Common Subexpression Elimination, FPGA Synthesis.

1. INTRODUCTION

Finite Impulse Response (FIR) filters are of great importance in the digital signal processing (DSP) world. Their characteristics of linear phase and feed forward implementation make it very useful for building high performance filters.

There are two main aspects to be considered when designing a hardwired parallel filter, namely the number of bits required for the signal and the required transfer function of the filter. The former one determines the word length of the entire datapath and the later one are determined by two parameters, namely the number of taps, and the number of bits in each coefficient. In this work we are addressing optimizations of the number of adders, by adequately selecting the best coefficient set taking into account the transfer function of the filter. Our methodology explores the reduction of the complexity of the multiplier block reducing the coefficients to a maximum number of power-of-two (NPT) terms. A coefficient scaling approach is adopted to generate the best NPT coefficient set. With this methodology we are able to reach a significant reduction of the number of adders in the multiplier block. It is possible to reach a reduction of up to 100% in the number of adders in the multiplier block, for the case when we find it possible to approximate to only one power-of two (PT) term for each coefficient.

We present a brief review of the related work on power-of-two coefficients and common subexpression elimination in section 2. In section 3 we present our proposed algorithm, and in section 4 its implementation. Section 5 shows the results obtained and section 6 summarizes the conclusions and presents our proposals for future work.

2. RELATED WORK

A FIR filter can be mathematically expressed by the equation (1) [10]:

$$Y[n] = \sum_{i=0}^{N-1} H[i]X[n-i] \ , \qquad (1)$$

where X represents the input signal, H the filter coefficients, Y the output signal, n is the current output sample, and N is the number of coefficients (or taps) of the filter. This is a convolution operation of the filter coefficients along the signal. The coefficients of the FIR filter are obtained by the Discrete Fourier Transform (DFT) of the required frequency transfer function, applying some known windowing method. In the sequential implementation a set of multiply-and-add (MAC) operations is performed for each sample of the input data signal, multiplying the N delayed input samples by coefficients and summing up the results together to generate the output signal. In parallel implementations, we can have two main architectures. The first one consists of unrolling of MAC loop where we have several delayed versions of the input signal entering in a fully parallel multiplier block, followed by a summation block. The other one consists of a multiplier block, which takes the same input signal and delivers each output to an input of a delayed summation block. The former (Fig. 1a) is the direct form parallel FIR and the last (Fig 1b) is the transposed form of the FIR.

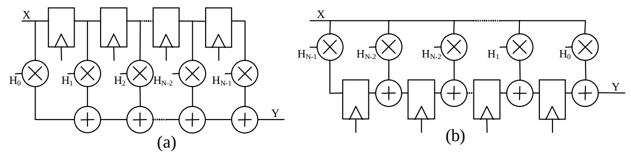


Figure 1. Parallel FIR filters in (a) direct form or (b) transposed form.

Both the direct form and transposed architectures of the FIR filter have the same complexity [10], but for some multiplier block optimization algorithms, the transposed form is preferred [1,2,3].

Several techniques for optimizing the multiplier block of parallel FIR filters were proposed in the literature. All of them consider the use the fixed-point representation and most [1-3] consider the transposed form implementation, because it is easier to obtain common sub expressions to be shared along two or more multipliers in this form. Many consider the use of some kind of signed digit (SD) representation [2,3], mainly the canonical signed digit (CSD) representation [2,3], which results in fewer non-zero digits in each coefficient, usually resulting in a smaller multiplier block. Previous research has been shown reductions of more than 50% [3] in the number of adders by using these techniques. The great advantage of these techniques is that the optimized filter has the same behavior of the original non-optimized one (i.e. same impulse response or transfer function). Other optimization techniques consist of the modification of the coefficients in order to generate sets of coefficients, which have a lower implementation, cost. Scaling and coefficient perturbations are examples of those techniques. Another approach consists of representing each coefficient as a sum of power-of-two terms and limiting the number of power-of-two terms in each coefficient [4,5,7,8]. That means the reduction of the number of bits in '1' state in each coefficient, reducing the number of adders needed to implement the multiplier for that coefficient. The best case is when we have just one power-of-two term in each coefficient, eliminating additions in the multiplier block at all, requiring operand shifting only (we are considering a hardwired implementation, where the sifting operation have no cost). We name this NPT (N-Power-of-Two), where N is the number of power-of-two terms. This approach has the advantage of preserving the full dynamic range of the coefficients and limiting the number of adders necessary to make the multiplication operation (leading to low power and high speed). The disadvantage of this approach is that the transfer function of the filter is not the same as obtained with the original fixed-point representation. In [4] an extensive review of the power-of-two technique is presented. In this work we combine these approaches in a improved way. The key point is to use scaling for an improved coefficient reduction, and later optimizing the resulting filter with CSE for eliminating common subexpressions in the multiplier block for an efficient hardwired implementation.

3. PROPOSED ALGORITHM

In this work we propose an algorithm to select the best NPT coefficient set based on scaling of the coefficients before the conversion to fixed point format followed by common sub expression elimination (CSE). The algorithm will search a wide range of discrete scaling factors and store the resulting transfer function of each NPT coefficient set associated and later select the best coefficient set based on the characteristics of the transfer function. We adopt different criteria from the published literature for selecting the best coefficient set [4,5,7,8]. We use the in-band ripple as a constraint, selecting only the transfer functions for which the entire pass band are within the specified ripple, and select the coefficient set in which the minimum attenuation in the stop band is the maximum among all the resulting transfer functions. The algorithm 1 shows the NPT coefficient selection process.

Algorithm 1: NPT coefficient selection by transfer function analysis

Step 1: Obtain FIR filter parameters: Taps; Bits; N PT elements; transfer function; pass and stop bands region; in-band ripple; Scale factors region and increment.

Step 2: Obtain the floating-point coefficients for the specified transfer function.

Step 3: For each element in scale factor vector, generate a new set of coefficients by multiplying each coefficient in floating point the current scale factor; make the coefficients positive and save the signal in of each coefficient in a set of signals for later use; get the fixed point representation of this set of coefficients; convert the fixed point coefficients to NPT; obtain a transfer function of the filter with these NPT coefficients. Add the set of coefficients and transfer function to a set of filters.

Step 4: From the set of filters, eliminate those that do not respect the in-band ripple constraint.

Step 5: From the results of Step 4, find out the coefficient set that generates a filter with the highest minimum attenuation in the stop band and select this set as the solution of the NPT phase.

Step 6: Make the common subexpression elimination of the solution of the NPT phase.

The Step 1 of the algorithm is only an initialization step. We have to guarantee that the floating-point coefficients generated satisfy the required specifications in a way we could not find a NPT solution otherwise. The number of PT (power-of-two) bits has to be selected by trial and error, once the solution found by the algorithm may not satisfy the specifications. As a rule of thumb, we use at least 1 PT for each 20dB step between the pass-band and the stop band of the filter. For the scale factor vector, the smaller is the step, the greater is the possibility of finding the best NPT coefficient set, once the variation of NPT coefficients is very non-linear. The number of bits of the fixed point determines the dynamic range of the coefficients (and the width of the adders and registers in the final summation block). The Step 2 of the algorithm calculates the filters coefficients from the specification using some windowing method, generating a floating point coefficient set for the filter, which the transfer function is not exactly the specified transfer function, but an approximation which is limited by the number of taps of the filter. The Step 3 consists in generating one NPT coefficient set and the associated filter transfer function for each specified scaling factor. This step makes an additional calculation, using the remaining bits (not selected by the NPT step) to round the NPT representation, so we can have a more accurate representation of the fixed point by the NPT and potentially reducing the number of PT digits. For example, if we have a 16 bit coefficient, say 0100011100001110, then the truncated 2PT representation will be 0100010000000000 and the rounded 2PT representation will be 01001000000000. If we had chosen 3PT, the truncated 3PT will be 0100001100000000 and the rounded 3PT will be 01001000000000. In the step 4 all the coefficient sets whose associated transfer function does not fit in the in-band ripple specification along the entire specified pass band are eliminated. The Step 5 determines the minimum attenuation point in the stop band for all the transfer functions and selects the one with the lowest value. The NPT coefficient set which generates this transfer function is then returned as result of this step. The coefficient set selected in Step 5 of the algorithm can be called transfer-function optimized, but we are treating each multiplier in the multiplier block separately. As our target is a hardwired implementation, a further optimization phase is done, namely the common sub expression elimination (CSE). The CSE phase will search sub expressions which are common between two or more multipliers and generate the hardware only once for these sub expressions, usually reducing the hardware size necessary to build up the entire multiplier block. The Step 6 will get the coefficient set selected in Step 5 and make the common sub expression elimination task. The output is a graph of the multiplier block that can easily be used to generate a hardware description of the filter. The algorithm 2 presents the process of eliminating common subexpressions.

Algorithm 2: Common Sub Expression Elimination

Step 1: Create a matrix C_{NxW} filled with the coefficients, where *W* is the Width of the coefficient and *N* is the number of coefficients; create a matrix F_{2xW} with all values -1.

Step 2: Create a set of triples *X(a,b,c)*, referencing the two columns of the matrix *C* and the number of bits in state '1' in both bit positions of these two columns.

Step 3: Sort *X* by descending order of the element *c*

Step 4: Get the first element of the set *X*. If $c<1$ in this element, go to step 6

Step 5: Create a new row in matrix *C* with the bits that are common in both columns of the element selected in Step 4; create a new row in matrix *F*, making the two values referencing the number of the columns that formed this one; go to step 2

Step 6: Sweep the matrix *F* backwards from the last index down to the index *W*, generating a tree of adders. Bits in '1' in the matrix *C* are used to mark interconnections from the multiplier block to the summation block.

This algorithm is a variation of the algorithm proposed in [1] for binary coefficients in the sense it treats only positive numbers, leaving the signals to control the final summation block in the architecture.

4. IMPLEMENTATION

The algorithm for NPT coefficient selection by transfer function analysis (Algorithm 1 described in the Section 3) was implemented in Matlab and its DSP and visualization toolboxes. It takes a set of parameters from a configuration file, process the algorithm, show the search space and the selected solution graphically, and write a file with the selected filter coefficients. Since the filter coefficients can be positive or negative, and we deal only with positive numbers, our method saves signals to be treated separately. This was helpful for the task of optimizing the multiplier block. Figure 2 shows the architecture developed for this implementation, where the signals S1..N of the coefficients C1..N were saved to be control signals in the final summation block (the signal actually selects between add or subtract functions). The algorithm for common subexpression elimination (Algorithm 2 described in the Section 3) was implemented in C. The input is a file with the filter parameters N and W and the binary filter coefficients, one in each row. The output is a VHDL file describing the CSE optimized filter. Figure 2 shows the architecture of the filter, which is generated and described in VHDL by the improved methodology proposed.

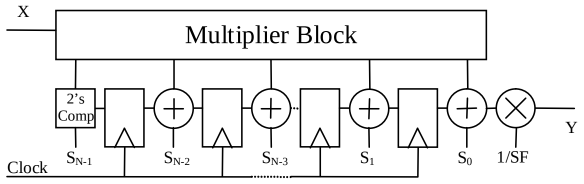


Figure 2. Architecture of the hardware description output

This is a transposed form FIR filter where the multiplier block receives the input signal X, and delivers N (taps) multiplied outputs. A multiplier just before the output Y is needed if we need to maintain the unity gain in the pass band, since our method scaled all the coefficients to find a better representation in the NPT phase. This multiplier can be eliminated if the gain in the pass band is not critical.

5. RESULTS

After implementing the algorithm described in the section 3, we analyzed the behavior of the NPT rounding technique for several filter specifications. Our experiments showed that it is very hard to get more than 20dB per PT digit (a similar result was stated in [5] for CSD coefficients), and this is very dependent on the frequency response shape and on the number of taps. Fig. 3 shows graphical results for the Low Pass (LP) filter LP1 specified in Table 2.

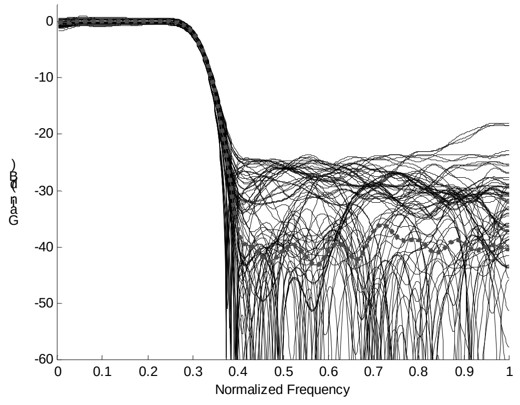
Dotted line in Fig. 3(a) and 3(b) shows the transfer function for the FIR filter LP1 in fixed point. Solid lines in Fig. 3(a) show the transfer functions for the 2PT coefficient sets for all the 76 scale factors tested (0.5 to 2 in steps of 0.02). Fig. 3(b) compares the filter transfer function for the LP1 with fixed point coefficients and the optimized version with 2PT coefficient set selected by our method. The insert in Fig. 3(b) shows that the effect of the optimization is negligible in the in-band ripple.

Table 1 shows a comparison between the original fixed-point coefficients and the reduced 2PT coefficients for LP1 FIR filter (using the same scale factor for both coefficient sets). As the coefficients are symmetrical, the table presents only the first $\lfloor N/2 \rfloor$+1 coefficients of the 49-tap filter.

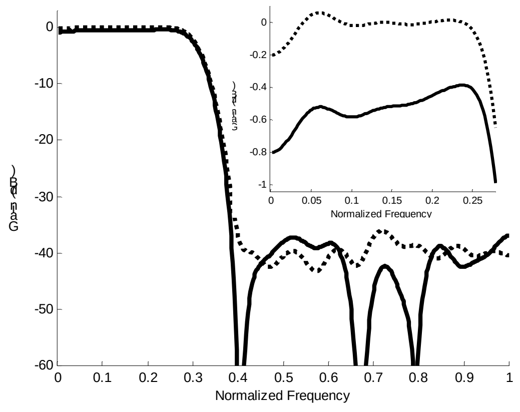
Table 1. Comparison of the coefficients before and after the NPT phase

Tap	Fixed Point	Num Adders	Logic Depth	Optimized 2PT Coefficients	Num. Adders	Logic Depth
1	000000001	0	0	000000001	0	1
2	000000001	0	0	000000001	0	1
3	000000001	0	0	000000001	0	1
4	000000000	0	0	000000000	0	1
5	000000010	0	0	000000010	0	1
6	000000001	0	0	000000001	0	1
7	000000011	1	1	000000011	1	1
8	000000110	1	1	000000110	1	1
9	000000100	0	0	000000100	0	1
10	000000011	1	1	000000011	1	1
11	000001010	1	1	000001010	1	0
12	000000111	2	2	000001000	0	1
13	000000111	2	2	000001000	0	1
14	000010100	1	1	000010100	1	0
15	000010001	1	1	000010001	1	1
16	000001000	0	0	000001000	0	0
17	000100011	2	2	000100100	1	1
18	000100000	0	0	000100000	1	1
19	000001011	2	2	000001100	1	0
20	001000010	1	1	001000010	1	1
21	001001010	2	2	001001000	1	0
22	000001100	1	1	000001100	1	1
23	010101110	4	3	010100000	1	1
24	101001110	4	3	101000000	1	1
25	110011110	5	3	110000000	1	1
Total		31	3 (max)		15	1 (max)

The results presented in Table 1 show the capability of the NPT phase in reducing the multiplier block in terms of number of adders with small changes in the transfer function, with the methodology adopted. As stated in section 4, the sign of the coefficients are treated separately in the final summation block (not considered here), so all coefficients are positive. Note that the NPT technique not only reduces the total number of adders but also the logic depth in terms of number of adders needed to implement the multipliers. The last phase of the algorithm is the common subexpression elimination. Table 2 presents the specifications some low pass (LP) and high pass (HP) filters used to test our methodology. The parameters have been selected to cover the 1PT to 4PT-reduced coefficients and 10 to 16 bits fixed point. Table 3 summarizes the results for the filter specifications presented in Table 2, showing the number of adders for each case.



(a)



(b)

Figure 3. (a) Fixed point (dotted) x 2PT for each scale factor tested (solid), and (b) fixed point (dotted) x 2PT selected (solid). Ripple in pass band (detail).

Table 2. Some filters used to test the proposed methodology

Parameter	LP1	LP2	LP3	HP
# of Taps	49	31	71	31
Scale Range (increment)	0.5-2 (0.02)	0.5-2 (0.02)	0.5-2 (0.02)	0.5-2 (0.02)
Bits Fixed Point	10 (sign+9)	16 (sign+15)	16 (sign+15)	12 (sign+11)
NPT digits	2	4	3	1
Pass Band (normalized)	0-0.3	0-0.3	0-0.05	0.6-1
Max. Pass Band Ripple	0.1	0.01	0.1	0.1
Stop Band (normalized)	0.35-1	0.35-1	0.07-1	0-0.4
Stop Gain (dB)	-40	-60	-60	-20
Window Type	Hamming	Blackman	Blackman	Hamming

Table 3. Optimization results for the filters in Table 2.

Filter	Fixed Point		CSE		NPT		NPT+CSE	
	Add.	%	Add.	%	Add.	%	Add.	%
LP1(fig.3)	31	100	18	58	15	48	12	38
LP2	58	100	39	67	45	78	31	53
LP3	171	100	85	50	42	25	38	22
HP	16	100	14	88	0*	0	0*	0
Mean		100		66		37		28

*Only one PT term to compute; shift-only operation.

Table 3 shows that significant reduction in the number of adders is achieved by applying either CSE or NPT optimization separately. Our proposed methodology, combining them appropriately, improves the results even more. The great advantage of using the NPT phase is that we greatly simplify the complexity of the multipliers by controllably modifying the coefficients, with small and acceptable changes in the filter transfer function. Also the logic depth is guaranteed to be low in the NPT optimization phase. Limiting the number of power-of-two (PT) terms in each coefficient also reduces the summation tree needed to implement each multiplier [6] (as shown in Table 1 for LP1 filter), reducing the delays, glitches and enabling lower voltage operation, improving the low power characteristics. The CSE phase improves the results further by eliminating any redundancies (common subexpressions among multipliers) in the multiplier block, thus further reducing the number of adders further.

6. CONCLUSION

From the results obtained we conclude that using NPT coefficients representation is very useful for reducing the complexity of the multiplier block of a FIR filter, keeping the side effects in the filter transfer function under control. This is done by means of selecting the appropriate fixed-point representation for the NPT conversion process by means of scaling the floating-point coefficients before its conversion to fixed point. The scale factor that generates the best results is found by exhaustive search along a discrete range of scale factors. The CSE phase improves results further, reducing even more the number of adders in the multiplier block. The reductions in the number of adders and in the logic depth obtained contribute to a great reduction in the required silicon area and power for the FIR filter. The power consumption reduction stems from three factors: first, the smaller number of adders, second the reduced (and controlled) logic depth from NPT phase reduces glitching activity, and third shorter paths may allow voltage reduction for the same data rate.

In this work we are not considering the Signed Digit representation for the coefficients, which can lead to further reductions in the number of adders as observed in literature [2-7]. Also, we are not considering also the fact that the coefficients in the center of the filter have greater value and so they are more significant to the filter transfer characteristics than the ones in the borders of the filter. As future work we will investigate these considerations and evaluate the improvements, which can be reached for low power, low area and high-speed FIR filters.

7. REFERENCES

[1] M. Potkonjak, M. B. Srivastava, and A. Chandrakasan, Efficient substitution of multiple Constant multiplication by shifts and addition using iterative pairwise matching. *In Proc. 31st ACM/IEEE Design Antomation Conf.*, (1994), 189-194

[2] M. Mehendale, S. D, Sherlekar, and G. Venkatesh, Syntesis of multiplier-less FIR filters with minimum number of additions, *in Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, (1995), 668-671

[3] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Iuraekova. A new algorithm for elimination of common subexpressions, *IEEE Trans. Computer-Aided Design*, 18. (Jan 1999), 58-68.

[4] H. Samueli, An improved search algorithm for the design of multiplier-less FIR filters with powers-of-two coefficients, *IEE Trans. Circuits Syst.*, 36 (July 1989), 1044-1047.

[5] K-H Chen, T-D Chiueh, Design and implementation of a reconfigurable FIR filter, *Proc of 2003 Int. Symp. Circuits Systems, ISCAS '03*, 3, (May 2003), 25-28.

[6] K. Hwang, Computer arithmetic Principles, Architecture and Design: *Wiley*, 1979.

[7] C. Lim, J. B. Evans, and B. Liu, Decomposition of binary integers into signed power-of-two terms, *IEEE Trans. Circuits Syst.*, 38, (June 1991) 667-672.

[8] J. Portela, E. Costa. J. Monteiro, Optimal Combination of Number of Taps and Coefficient Bit-Width for Low Power FIR Filter Realization, *IEEE European Conference on Circuit Theory and Design*. (Sep. 2003), 145-148.

[9] ZHAO, Q.; TADOKORO, Y. A Simple Design of FIR Filters with Powers-of-Two Coefficients, *IEEE Transactions on Circuits and Systems*. 35, 5 (May, 1988).

[10] R. W. Hamming, Digital Filters: 3rd ed, *Prentice Hall*, 1989.