Area-Effective FIR Filter Design for Multiplier-less Implementation

Tay-Jyi Lin, *Tsung-Hsun Yang*, and *Chein-Wei Jen*

Department of Electronics Engineering
National Chiao Tung University, Taiwan

ABSTRACT

The hardware complexity of digital filters is not controllable by straightforwardly rounding the coefficients to the quantization levels. In this paper, we propose an effective alternative that distributes a pre-defined addition budget to the multiplier-less FIR filters, which takes into account the common sub-expression sharing inside the computations. We successfully integrate a heuristic common sub-expression elimination (CSE) algorithm and the coefficient quantization by successive approximation proposed by Li et al. Besides, we also propose an improved search algorithm for an optimal scale factor to settle the coefficients collectively into the quantization space. Simulation results show that CSE effectively reduces 29.1%~31.5% budgets for comparable filter responses. Besides, the improved scale factor exploration helps to find an identical or a better (never worse) quantization result with only 32.67%~44.53% run time, whether or not CSE is applied.

1. INTRODUCTION

Most DSP kernels are linear transforms with fixed coefficients, such as FIR filters, discrete cosine transforms (DCT), and fast Fourier transforms (FFT), etc. Adders and shifters can thus replace multipliers for area-efficient implementations, where common sub-expression elimination (CSE) [1]-[3] can further reduce the required additions and shifts. Besides, given an acceptable gain variance (e.g. ±3dB), an optimal scale factor (SF) can be found such that the number of additions after CSE is minimized. The filter characteristics in terms of pass-band ripple and stop-band attenuation are preserved with the magnitude gain equal to the SF [4].

The hardware complexity of the FIR implementation is not under direct control by straightforwardly rounding the coefficients to the quantization levels. An iterative algorithm was proposed by Li et al. [1] to quantize the filter coefficients by successive approximation, which controls the hardware complexity with a pre-defined budget of non-zero terms. The budget gives a rough estimate for the required additions in the implementation, which may lead to a less optimal design, especially when CSE is applied. In this paper, we propose an area-aware algorithm to quantize the FIR coefficients, which approximates the ideal coefficients under an exact addition budget instead of the non-zero terms. An optimal scale factor is explored to settle the coefficients into the quantization space. Besides, common sub-expressions in the filter computation are considered as zero-overhead terms that consume no addition budget.

* This work was supported by the National Science Council, Taiwan under Grant NSC91-2218-E009-011

The rest of this paper is organized as follows. Section 2 covers the background materials that lay the foundations of the proposed area-aware coefficient quantization, which is detailed in Section 3. Simulation results are available in Section 4, and Section 5 concludes this work and outlines our future research.

2. BACKGROUND

2.1 Coefficient Quantization by Successive Approximation

Straightforward quantization on the ideal filter coefficients by truncation or rounding is not aware of the hardware complexity. Successively approximating the ideal coefficients is an effective alternative [5], which gradually assigns non-zero terms to the quantized coefficients (QC) under a pre-defined budget. Besides, it applies an additional scale factor (SF) to settle the coefficients collectively into the optimal quantization space. Fig 1(a) shows the algorithm. The ideal coefficients (IC) are first normalized as the maximum magnitude to be one. An optimal SF is searched from 0.4375 to 1.13 with a fixed step-size of 2^{-w}, where w denotes the word-length including the sign bit. For each SF, the QC's are initialized to zeros and a non-zero term is gradually allocated to the QC that differs most from the scaled and normalized IC, until the budget is exhausted or the differences between IC & QC pairs are all less than 2^{-w}. Finally, the compensated QC (for previous scaling and normalization) with the least error is chosen. Fig 1(b) is an illustrating example of the quantization algorithm when SF=0.5.

Normalize IC so that the maximum coefficient magnitude is 1
FOR SF=0.4375: 2^{-w} : 1.13
{ **Scale** the normalized IC with SF
WHILE (budget >0 & difference between QC & IC >2^{-w})
{ **Allocate** a non-zero term to QC }
Estimate the error between IC & the normalized QC }
Choose the QC that has the minimum error

(a)

IC = [0.26 0.131 0.087 0.011]
Normalized IC (NIC) = [1 0.5038 0.3346 0.0423]
SF = 0.5
Scaled NIC = [0.5 0.2519 0.1673 0.0212]
QC_0 = [0 0 0 0]
QC_1 = [0.5 0 0 0]
QC_2 = [0.5 0.25 0 0]
QC_3 = [0.5 0.25 0.125 0]
QC_4 = [0.5 0.25 0.15625 0]
QC_5 = [0.5 0.25 0.15625 0.015625]

(b)

Fig. 1 Quantization by successive approximation

The fixed 2^{-w}-step SF exploration may step through multiple candidates that result in an identical QC result, while omitting significant SF's at finer granularity, i.e. the higher computational complexity may have a worse quantization result. In this paper, we propose an improved searching algorithm in Section 3, which explores significant SF's only with a variable step-size. Besides, we also take account of the common sub-expressions sharing of the filter computation, where the non-zero terms require no extra addition and thus consume no budget.

2.2 Common Sub-expression Elimination

For brevity, this paper considers two common sub-expressions in the direct-form FIR filters, namely, the common sub-expressions across coefficients (CSAC) and the common sub-expressions within coefficients (CSWC) [1]. Consider a 4-tap FIR filter with the coefficients: h_0=0.0111011, h_1=0.0101011, h_2=1.0110011, and h_3=1.1001001, which are four fractional numbers in the 8-bit 2's complement form. The output is computed as

$$y_n = h_0 \cdot x_n + h_1 \cdot x_{n-1} + h_2 \cdot x_{n-2} + h_3 \cdot x_{n-3}.$$

Additions and shifts can be substituted for the multiplications as

$$\begin{aligned} y_n = {} & x_n\text{»}2 + x_n + x_n\text{»}4 + x_n\text{»}6 + x_n\text{»}7 \\ & + x_{n-1}\text{»}2 + x_{n-1}\text{»}4 + x_{n-1}\text{»}6 + x_{n-1}\text{»}7 \\ & - x_{n-2} + x_{n-2}\text{»}2 + x_{n-2}\text{»}3 + x_{n-2}\text{»}6 + x_{n-2}\text{»}7 \\ & - x_{n-3} + x_{n-3}\text{»}1 + x_{n-3}\text{»}4 + x_{n-3}\text{»}7 \end{aligned} \quad (1),$$

where "»" denotes arithmetic right shift (i.e. with sign extension). Each output requires 17 additions (including 2 subtractions), and 16 shifts. The h_0 and h_2 multiplications, i.e. the first and the third rows in Equation (1), have four terms with the same shifts. Restructuring the equation by adding x_n and x_{n-2} first effectively eliminates the redundant CSAC between h_0 and h_2 as

$$\begin{aligned} y_n = {} & (x_n+x_{n-2})\text{»}2 + (x_n+x_{n-2})\text{»}3 + (x_n+x_{n-2})\text{»}6 + (x_n+x_{n-2})\text{»}7 \\ & + x_n\text{»}4 - x_{n-2} \\ & + x_{n-1}\text{»}2 + x_{n-1}\text{»}4 + x_{n-1}\text{»}6 + x_{n-1}\text{»}7 \\ & - x_{n-3} + x_{n-3}\text{»}1 + x_{n-3}\text{»}4 + x_{n-3}\text{»}7 \end{aligned} \quad (2),$$

where the total additions and shifts are reduced to 14 and 12, respectively. The extraction and elimination process of CSAC can be more concisely manipulated in tabular form as Fig 2.

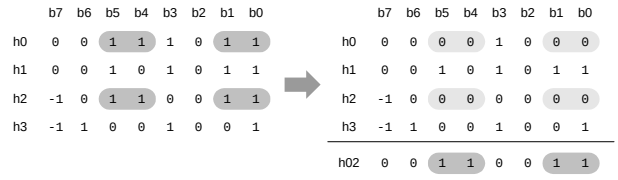

Fig. 2 CSAC extraction & elimination

Within a single coefficient or CSAC, bit-pairs with identical bit displacement are recognized as CSWC, where the redundant CSWC can be eliminated by reusing the computation result. For example, the sub-expression in the first row of Equation (2) can be simplified as $(x_{02}+x_{02}\text{»}1)\text{»}2+(x_{02}+x_{02}\text{»}1)\text{»}6$ to reduce both the addition and the shift by one, where x_{02} stands for x_n+x_{n-2}.

The CSE quality depends on the elimination order. A steepest-descent approach [1] is applied as a heuristic to reduce the search space, which first eliminates the redundant CSAC between the coefficient pair with the most non-zero terms in identical bit-positions. One-level look-ahead can further distinguish the candidates of the same weight. A similar strategy is applied to eliminate redundant CSWC, too. Fig 3 shows the CSE algorithm for CSAC and CSWC of direct-form FIR filters [1].

Eliminate zero coefficients
Merge coefficients with the same value (e.g. linear-phase FIR)
Construct a coefficient matrix of size N×W, where N is the number of coefficients for CSE and W is the word-length (Fig 2)
WHILE (highest weight > 1) // CSAC elimination
{ **Find** the coefficient pair with the highest weight
Update the coefficient matrix }
FOR each row in the coefficient matrix // CSWC elimination
{ **Find** bit-pairs with identical bit displacement
Extract the distances between those bit-pairs
Update the coefficient matrix with the shift information }
Output the SDFG with addition, shift and negation

Fig. 3 CSE for direct-form FIR filters [1]

3. Proposed Area-Aware Quantization

Allocating a non-zero term to the quantized coefficient (QC) set either forms an isolated term to sum up, or just enlarges a common sub-expression without any addition overhead. In other words, the number of additions in the multiplier-less FIR filters is non-decreasing during the successive approximation. The property guarantees the termination conditions in Fig 1 are valid when CSE is incorporated.

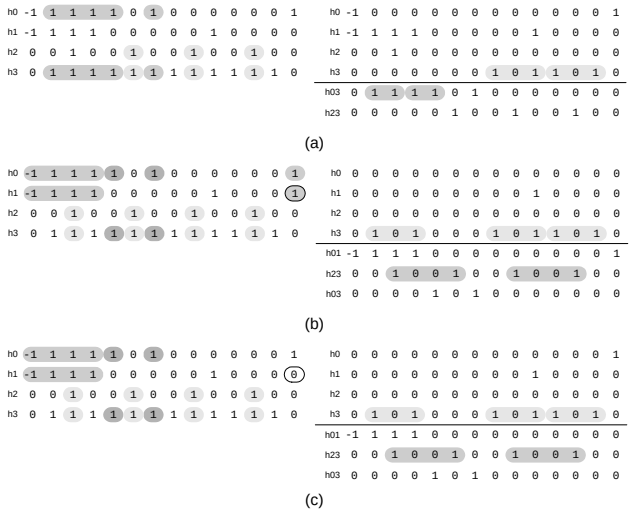

Fig. 4 Insert a non-zero term but reduce the additions

In fact, the number of additions is not always 'non-decreasing' because of the steepest-descent CSE heuristic. For example, if the optimum CSE for a coefficient set does not start with the pair with the most CSAC terms and thus the steepest-descent heuristic cannot find the optimum result. Allocating a non-zero term increases the weight of the allocated pair and possibly alters the CSE order, which may lead to a better CSE result. Fig. 4 is an example where the number of additions decreases after inserting one non-zero term. The left matrices are original coefficients with eliminated CSAC terms marked. The right matrix in (a) is the steepest-descent CSE result with the CSWC highlighted, which requires 19 additions. Then, a zero-overhead

term is allocated to the LSB of h_1 as shown in (b), which changes the CSE order and introduces a new budget of two additions (17 additions are required here). Applying the CSE order in (b) for (a), we can find a better result as (c), which also requires 17 additions only. The following is our approach to integrate CSE and the successive approximation with improved SF exploration.

As the algorithm shown in Fig 1, our proposed area-aware quantization algorithm first normalizes the ideal coefficients (IC) such that the magnitude of the maximum coefficient is one. An optimal scale factor (SF) is then explored to collectively settle the coefficients into the quantization space, but with a more efficient searching strategy. Instead of the fixed-size stepping from the lower bound, the next SF is calculated as

$$SF' = SF \times [\min |QD| + |coef(\min QD)|] / |coef(\min QD)|,$$

where QD denotes the distance of a coefficient to its next quantization level, depending on the approximation strategy (e.g. rounding to the nearest value, toward 0, or toward -∞, etc). |coef(min QD)| denotes the magnitude of the coefficient with the minimum QD. In short, the next SF is chosen as the minimum value to scale up one coefficient to its next quantization level. For 16-bit wordlength and ±3dB acceptable gain, the number of candidates ranges from 14,986 to 20,429 in our simulation, which depends on the filter coefficients. By contrast, the fixed step-size exploration has 45,875 candidates for all coefficients.

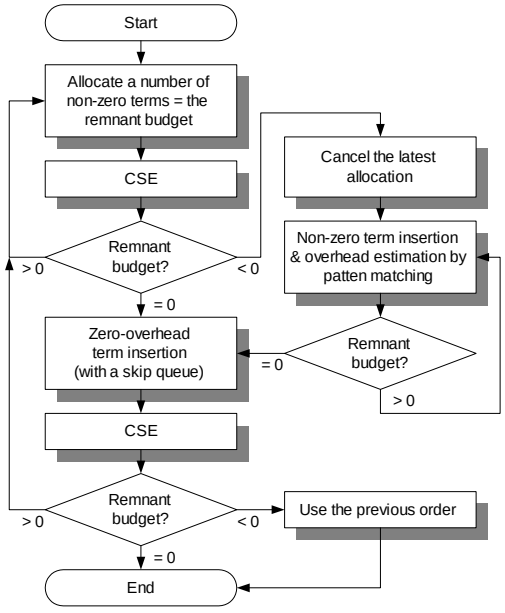

Fig. 5 Proposed area-aware coefficient quantization

For each SF, the QC's are initialized as zeros and a non-zero term is gradually allocated to a QC that differs most from the scaled and normalized IC. Once the allocated terms amount to the remnant budget, CSE is performed to introduce a new budget. The iterations continue until no additional budget exists. Zero-overhead terms (i.e. those enlarge the common sub-expressions without any addition overhead) are inserted as a post-processing step. Note that the zero-overhead insertion can introduce new budgets as the illustrating example shown in Fig. 6. A queue is needed to insert the skipped (i.e. with addition overheads), but more significant terms, once such a new budget is available. The already-allocated zero-overhead terms that are less significant should be removed. Finally, additional CSE is performed to check if there exists another CSE order for a better result. The successive approximation resumes if the skip queue is empty and a new budget is available.

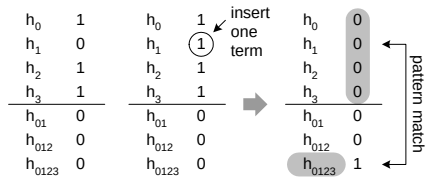

Fig. 6 Zero-overhead insertion by pattern matching

The steepest-descent heuristic CSE can have a worse result after the insertion of a group of non-zero terms, and the remnant budget will be negative (i.e. the required additions exceed the pre-defined budget). We save this situation just by canceling the latest allocation and using the previous CSE order as the steps shown in the right-hand side of Fig 5. With the original CSE order, the addition overhead is estimated with pattern matching to use up the remnant budget, which is very similar to the zero-overhead insertion except no queue is required here. By the way, the approximation stops, of course, whenever all the difference between QC and IC pairs are all less than 2^{-w}, because the QC cannot improve anymore.

4. SIMULATION RESULTS

The ideal coefficients in our simulation are generated in Matlab for linear-phase and low-pass FIR filters using the Hamming window (`fir1`). The quantized coefficients are fractional numbers in the 16-bit 2's complement form.

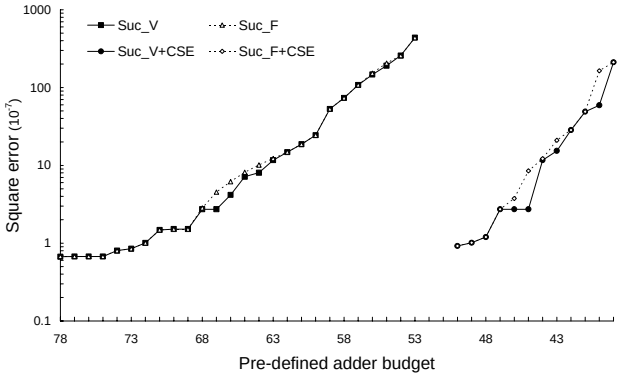

Fig. 7 Effectiveness of CSE & variable step-size exploration

Fig 7 shows the square error versus the given addition budget for a 20-tap filter, which demonstrates the effectiveness of CSE and the proposed SF searching with variable step-sizes. CSE saves 29.1%~31.5% addition budgets for comparable filter responses. The solid lines (variable step-size SF exploration) are always below their corresponding dashed lines (fixed step-size) or just on them. That is, the proposed searching approach never finds a QC result that introduces higher quantization error, whether or not CSE is applied.

Table 1 Comparison of quantization results with CSE

	DIRECT_V[1]			Suc_F[2]		Suc_V[3]		Suc_F_90%[4]		Suc_V_90%[5]	
TAP	SF	Error (10^{-7})	Add	SF	Error (10^{-7})	SF	Error (10^{-7})	SF	Error (10^{-7})	SF	Error (10^{-7})
12	0.994738	0.928092	27	0.994677	0.928092	0.994738	0.928092	0.994540	38.068010	0.994738	38.068010
16	0.981586	1.556596	37	0.981555	1.556596	0.981681	1.556596	0.838504	91.450613	0.838683	91.450613
20	1.121206	2.725406	45	1.123400	**8.482496**	1.121206	2.725406	1.121218	**163.872263**	1.121206	59.289435
24	1.125531	**5.147758**	54	1.125994	3.436487	1.125981	3.436487	1.126605	181.767276	1.126600	181.767276
28	0.743739	10.684378	61	0.743701	10.684378	0.743819	10.684378	0.751602	167.711315	0.751498	167.711315
32	1.263281	4.431486	72	1.263293	4.431486	1.263281	4.431486	1.008407	108.630690	1.008411	108.630690

[1] Straightforward quantization & variable step-size SF exploration
[2] Quantization by successive approximation & fixed step-size SF exploration; adder budget is given by the DIRECT_V result
[3] Quantization by successive approximation & variable step-size SF exploration; adder budget is identical to Suc_F
[4] Suc_F with 90% adder budget
[5] Suc_V with 90% adder budget

Our proposed variable step-size searching algorithm has 18,899 instead of 45,875 candidates of the fixed step-size one, and runs 2.43× faster in average. To be brief, our method finds the same (with identical bit patterns) or better quantization results with only 41.2 % computations.

Table 1 summarizes the CSE results for different taps. The first column lists the number of taps and the simulation results for the straightforward quantization (DIRECT_V) follow immediately, where the ideal coefficients are rounded to their nearest 16-bit 2's complement fractional numbers. An optimal SF is explored with variable step-sizes to minimize the number of additions. The applied SF, the square error, and the resultant minimum additions are shown respectively in the second, third and fourth columns. By the way, the numbers of additions are used as budgets for the rest experiments. It should be noted that DIRECT_V has no control over the required additions. Suc_F and Suc_V denote the quantization results by successive approximation, with fixed and variable step-size SF exploration respectively.

For the 24-tap case, DIRECT_V finds a QC set with 54 additions by performing CSE on the arbitrarily distributed non-zero terms (i.e. the non-zero terms are not optimally allocated by quantizing the scaled coefficients straightforwardly). Another QC set with 55 additions exists, which significantly outperforms that with 54 additions. DIRECT_V skips this candidate, for it always tries to find the minimum additions. But the QC set has better response after precisely removing the least significant non-zero terms (i.e. this is what successive approximation does), which also needs 54 additions.

The rightmost four columns summarize the Suc_F and Suc_V results with only 90% of the previous addition budgets. In most cases, the two methods have identical results and equivalent square error after normalization, except for the 20-tap one. The variable-step searching finds better quantization results, because the fixed-step SF exploration jumps over some significant SF that leads to smaller error.

5. CONCLUSION

This paper presents an area-aware quantization algorithm to optimize coefficients for non-recursive filters. It integrates the quantization by successive approximation and the heuristic common sub-expression elimination to precisely distribute a pre-defined addition budget to the quantized coefficients. We also propose an improved exploration algorithm with variable step-sizes to find an optimal SF that collectively settles the filter coefficients into the quantization space. The simulation results show that CSE effectively reduces 29.1%~31.5% budgets for comparable responses. Besides, our algorithm finds the same (with identical bit patterns) or better quantization results with only 32.67%~44.53% run time, whether CSE is applied or not.

We have implemented the proposed algorithm in C, which takes the double-precision coefficients as input and generates the coefficient matrix with a compensation factor (i.e. to compensate the normalization, scaling and also for zero-mean quantization error). The automatic generation of bit-serial implementations [6][7] is still on-going. Common sub-expressions for transposed-form FIR (also known as multiple constant multiplication; MCM [2][3]) will be considered in the future, which is more sensitive to the computation order in our bit-serial architectures. Finally, the compensation factor is regarded as free, since multiple factors can be combined and carried out once, or just moved to the analog parts. Self-compensated coefficient scaling that considers this effect will be studied in the future.

6. REFERENCES

[1] M. Mehendale, S. D. Sherlekar, *VLSI Synthesis of DSP Kernels - Algorithmic and Architectural Transformations*, Kluwer Academic Publishers, 2001

[2] M. Potkonjak, M. Srivastava, and A. Chandrakasan, "Multiple Constant Multiplication – Efficient and Versatile Framework and Algorithms for Exploring Common Sub-expression Elimination," *IEEE Trans. Computer-Aided Design Systems*, vol.15, pp.151-165, Feb 1996

[3] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "A New Algorithm for Elimination of Common Subexpressions," *IEEE Trans. Computer-Aided Design*, vol.18, pp.58-68, Jan 1999

[4] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd Edition, Prentice Hall, 1999

[5] D. Li, Y. C. Lim, Y. Lian, and J. Song, "A Polynomial-Time Algorithm for Designing FIR Filters with Power-of-Two Coefficients," *IEEE Trans. Signal Processing*, vol.50, pp.1935-1941, Aug 2002

[6] K. K. Parhi, *VLSI Digital Signal Processing Systems – Design and Implementation*, Wiley, 1999

[7] J. Valls, M. M. Peiro, T. Sansaloni, and E. Boemo, "Design and FPGA Implementation of Digit-Serial FIR Filters," *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, vol.2, pp.191-194, 1998