Optimization of structural adders in fixed coefficient transposed direct form FIR filters

Mathias Faust and Chip-Hong Chang
Centre for High Performance Embedded Systems, Nanyang Technological University, Singapore

***Abstract*—Over the last two decades, fixed coefficient FIR filters were generally optimized by minimizing the number of adders required to implement the multiplier block in the transposed direct form filter structure. In this paper, an optimization method for the structural adders in the transposed tapped delay line is proposed. Although additional registers are required, an optimal trade-off can be made such that the overall combinational logic is reduced. For a majority of taps, the delay through the structural adder is shortened except for the last tap. The one full adder delay increase for the last optimized tap is tolerable as it does not fall in the critical path in most cases. The criterion for which area reduction is possible is analytically derived and an area reduction of up to 4.5% for the structural adder block of three benchmark filters is estimated theoretically. The saving is more prominent as the number of taps grows. Actual synthesis results obtained by Synopsys Design compiler with 0.18μm TSMC CMOS libraries show a total area reduction of up to 13.13% when combined with common subexpression elimination. In all examples, up to 11.96% of the total area saved were due to the reduction of structural adder costs by our proposed method.**

I. Introduction

The inherit stability makes FIR filters a preferred choice in digital signal processing. As wireless technology advances, FIR filters with shorter transition bands, more stringent stopband attenuation requirement and higher sampling rate, are in great demand. To achieve these goals, ASIC implementation is necessary. The Transposed Direct Form (TDF) structure is preferred over direct form structure for higher order ASIC filters due to its shorter critical path delay. In the direct form structure, the input is delayed before the coefficient multiplication and the register length of each tap is fixed by the input bit width. In the TDF structure, the partial sums generated by the outputs of the coefficient multiplier, are delayed. Thus, the lengths of the registers increase monotonically along the taps to hold the correct precision of the partial sums. Consequently, the number of registers needed for the TDF structure is larger than that for the direct form. Fig. 1 shows a generic TDF fixed coefficient FIR filter. For long filters, the shorter critical path of the TDF is more significant than the costs of the registers.

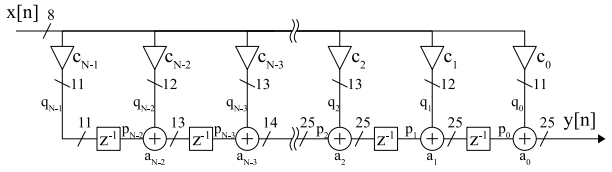

Fig. 1. Transposed direct form FIR Filter.

Over the last two decades, the complexity of implementing TDF FIR filters has been minimized by modelling the multiplier block [1]–[5] as a Multiple Constant Multiplication (MCM) problem [2]. MCM optimization algorithms emphasize on the minimization of the number of adders in the multiplier block and assume that the cost for each addition is the same. Recently, more accurate results have been reported by using the Full Adder (FA) count as the cost metric [6], [7]. The structural adder (SA), a_i, $(i = N-2, N-3, \ldots, 0)$ is defined as the adder that adds the output of the register, p_i to the output of the coefficient multiplier, q_i in the tapped delay line. The structural adders have been omitted by most MCM algorithms, as it is not possible to share these adders. In general, the operands to the structural adders are binary numbers in two's complement representation. Thus, the smaller addend needs to be sign extended to match the bit width of the other operand. This leads to a higher fanout for the Most Significant Bit (MSB) of the smaller input. For positive coefficients, the sign of the coefficient multiplier output follows the sign of the input value. As the bit width of the coefficient multiplier, q_i, is always smaller than that of the partial sum, p_i, a high load is imposed on the sign bit of the input.

In [8], a technique called MSB-Fix is used to avoid sign extension. It uses constant correction vectors as in the partial product addition of two's complement multiplication. These constant correction vectors are grouped together and lumped at the last tap. Due to the delay of the partial sums in the TDF FIR filters, the first $N-1$ outputs are incorrect as the cumulative error has to propagate to the last tap. In [8], the FIR filter in the all-digital modem has only $N=20$ taps. This system needs more than 20 clock cycles to start up, so the first 19 incorrect outputs are not critical. For generic FIR filters, in particular high order filters with large N, the flushing of $N-1$ incorrect output samples may not be acceptable. Therefore, an alternative approach to simplifying the structural adders is necessary.

For fixed coefficient FIR filters, the bit widths of the input and all coefficients are known. This enables the bit width of the coefficient multiplier to be determined from its dynamic range. As the partial sums are delayed before they are added with the coefficient multiplier outputs in the structural adders, the bit widths of the structural adders increase monotonically from the first structural adder towards the output. Careful analysis revealed that for most filters, the bit width of the adder increases only from coefficient N–1 to about $N/2$, after which the bit width stays relatively constant and increases by no more

than two bits. As the bit width of the coefficient multiplier output reduces towards the last tap, longer sign extension is required for these structural adders. This paper proposes an addition scheme to reduce the bit widths of these structural adders so that the total combinational logic is reduced at the expense of some register overhead. To determine if the area reduction is able to offset the overheads of additional adders and registers, a lower bound for the difference between the adder bit width and the coefficient multiplier output bit width is established analytically.

II. Proposed structural adder optimization

As the proposed method optimizes the trade-off between the structural adders and registers, a ratio of these two component areas is required. A structural adder adds two two's complement numbers using Carry Propagate Adder (CPA). The simplest CPA is a Ripple Carry Adder (RCA). Faster adders like carry look-ahead or carry select adder require more area than a RCA. For simplicity, the complexity of structural adder is assessed based on the number of FAs required to implement a RCA. A delay element is used to hold the output of the structural adder for one clock cycle and is typically implemented by delay Flip-Flop (FF). The authors of [9] suggest a ratio of $1:0.6\sim0.8$ for FA:FF. The TSMC 0.18μm standard cell library [10], which is used for our simulation, contains a variety of FFs with areas measured from 53.2 to $86.5\mu\mathrm{m}^2$ and a standard FA cell has an area of $69.9\mu\mathrm{m}^2$. These documented numbers and the simulation results showed an average area of $69.2\mu\mathrm{m}^2$ for FF and FA in typical FIR filters synthesized with this standard cell library. Therefore a FA:FF ratio of $1:1$ $(\rho=1)$ is adopted in this paper.

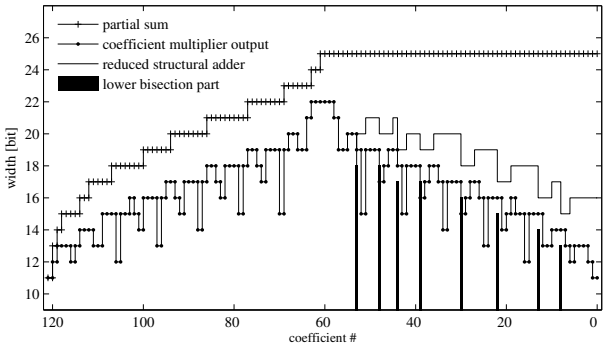

Fig. 2. Bit width analysis for tapped-delay line signals of Filter 1 of [11].

An area of several successive structural adders can be optimized by exploring the aggregate difference in the bit widths between the delayed signals and the outputs of the coefficient multipliers. Fig. 2 shows the analysis of Filter Example 1 of [11]. The bit width of the partial sum is the minimal bit width based on the conventional range estimation. It remains constant after c_{60} whereas the coefficient multiplier output's bit width reduces towards both ends. As the difference between the bit widths of the coefficient multiplier output and the partial sum increases towards the output tap, longer sign extension is required for these structural adders. Six or more bits are required for the sign extension for the structural adders, a_{52} to a_0. The other two bit widths marked as reduced structural adder and lower bisection part in the legend are obtained from our proposed structural adder reduction scheme, which will be explained in what follows.

The fundamental concept of our proposed method can be illustrated by an example in decimal. Let $\{610,-274,2,258\}$ be a set of coefficient multiplier outputs to be accumulated to a large partial sum 1234567 by the structural adders in a tapped delay line. A downright approach is to add one number at a time from the set of smaller integers to the large integer. Alternatively, the integers in the set are summed and then added to the large integer. The latter accumulation scheme, when implemented in hardware, requires the large integer and the smaller integers to be stored at each tap. This incurs a large register overhead, which can be reduced if the large number is split into two smaller integers as shown in Fig. 3.

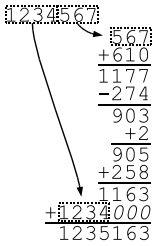

Fig. 3. Example of adder size reduction for decimal number accumulation.

By partitioning the large integer into two halves, the register overhead is greatly reduced as only the fourth overlapping digit has to be saved twice. The additional adder at the last step needs only a four-digit addition as the three least significant digits are all zeros. Besides, the reduction of the dynamic ranges of the operands also simplifies the structural adder implementation and reduced the length of sign extension.

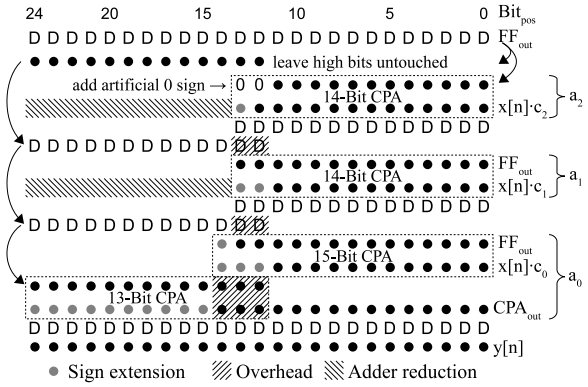

Fig. 4. Binary example on the optimization of last three adders of Fig. 2.

Fig. 4 shows the binary implementation of the proposed scheme on the last three coefficients of the filter example from Fig. 2. The reduction of adder lengths is observed to be several times more than the register and adder overheads it incurred. Furthermore, the delays through the structural adders, a_2 and a_1 have been reduced, while the delay through a_0 is increased by one FA delay. The slight increase in the delay through a_0 is not an issue as in most cases, there exists at least one tap $(i>0)$ for which $\mathrm{delay}(x\cdot c_0)<\mathrm{delay}(x\cdot c_i)$. The FAs reduction for the structural adders can be offset by the increase in FF overhead. Therefore, information about the minimal difference between the addends of the structural adders is of interest.

With reference to Fig. 1, p_i and q_i are the inputs to the ith structural adder, a_i, where q_i is the ith coefficient multiplier's output and a_0 is the last adder that produces the output y. Let $\mathrm{n}(z)$ be the bit width of an arbitrary signal z and let $\left[v_i^-, v_i^+\right]$ be the dynamic range of q_i. The selected partial sum, p_i is bisected into two halves, i.e., $p_i = p_i^u \left| p_i^l \right.$ at bit position, h_i such that $\mathrm{n}\left(p_i^l\right) = h_i$ and $\mathrm{n}(p_i^u) = \mathrm{n}(p_i) - h_i$. If p_i^u is added, after m delays, to the sign extended p_{i-m+1}^l where $\mathrm{n}(p_{i-m+1}^l) > h_i$, then the original structural adders, a_{i-j+1} will be replaced by the reduced structural adders, $a_{i,j}$, for $j = 1, 2, \ldots, m$. The minimal bit width $\mathrm{n}(a_{i,j})$ of each reduced adder can be obtained by the following range estimation

$$\mathrm{n}(a_{i,j}) = \left\lceil \log_2 \left(\max \left(2^{h_i} - 1 + \sum_{k=0}^{j-1} \left(v_{i-k}^+\right), -\sum_{k=0}^{j-1} \left(v_{i-k}^-\right) \right) \right) \right\rceil + 1 \tag{1}$$

In (1), the value, $2^{h_i} - 1$ is the maximum possible value of the lower bisected partial sum, p_i^l. Since the first term provides the bit width of the magnitude from the dynamic range, the constant '1' is added to account for the sign bit. As j increases, the values of $\mathrm{n}(a_{i,j})$ grow monotonically as $v_i^- \leq 0$ and $v_i^+ \geq 0$. The total number of FAs saved by the m successive reduced structural adders starting from the ith tap is given as follows:

$$\Delta_{i,m} = \sum_{k=1}^{m-1} \left[\left(\mathrm{n}(p_i) - \mathrm{n}(a_{i,k})\right) - \rho \left(\mathrm{n}(a_{i,k}) - h_i\right) \right] - \left(\mathrm{n}(a_{i,m}) - h_i\right) \tag{2}$$

where ρ is the FA:FF ratio. In (2), the term $\mathrm{n}(p_i) - \mathrm{n}(a_{i,k})$ denotes the adder cost reduction and the term $\mathrm{n}(a_{i,k}) - h_i$ represents the register overhead. The last term, $\mathrm{n}(a_{i,m}) - h_i$ refers to the overlapping bits for the addition of p_i^u. As $\mathrm{n}(a_{i,m}) > h_i$, $\forall c_i \neq 0$, it is apparent from (2) that there is no saving if $m = 1$.

Hypothetically, h_i can be any arbitrary value smaller than $\mathrm{n}(p_i)$. However, as $\mathrm{n}(a_{i,j})$ is dependent on v_i^+ and v_i^-, it can be optimally deduced. If $h_i > \max\left(\left\lceil \log_2\left(v_i^+\right)\right\rceil, \left\lceil \log_2\left(-v_i^-\right)\right\rceil\right)$, the FA saving in the structural adders is reduced. If it is chosen to be smaller, the register overhead and the FA cost of the final addition are higher. As $\mathrm{n}(q_i)$ is not monotonically decreasing with i, h_i needs to be selected by looking ahead into several coefficients. It was empirically determined that a look ahead of 5 coefficients is sufficient. This leads to

$$h_i = \max\left(\left\lceil \log_2\left(v_k^+\right)\right\rceil, \left\lceil \log_2\left(-v_k^-\right)\right\rceil\right), \forall\, k = i, i-1, \ldots, i-5 \tag{3}$$

Combining (1) and (3) it can be derived that $\mathrm{n}(a_{i,j}) \geq h_i + 2, \forall j > 0$. Defining $\delta_i = \mathrm{n}(p_i) - h_i$, from (2) we have

$$\Delta_{i,j} \leq (m-1) \cdot (\delta_i - 2 - 2\rho) - 2 \tag{4}$$

In order to not increase the area, the following lower bound can be derived from (4).

$$\delta_i \geq 2\frac{m\,(1+\rho) - \rho}{m-1} \tag{5}$$

If $m \to \infty$, $\delta_i \geq 2\,(1+\rho)$. Usually, m is small and for TSMC 0.18μm standard cell library, $\rho \approx 1$, $\delta_i \geq \frac{2(2m-1)}{m-1}$. For $m = 2$, $\delta_i \geq 6$ and for $m = 3$, $\delta_i \geq 5$.

The above formulae hold only if no zero valued coefficient is involved. Since no structural adder is required to add a zero coefficient, no saving can be gained from the structural adder but the overhead due to the overlapping bits remains. Experiment results showed that in many cases it is not beneficial to span the bisected partial sum addition process over zero coefficients. It was also observed that $\mathrm{n}(a_{i,j}) = h_i + 2$ is valid only for $j \leq 3$. This observation suggests the use of $\mathrm{n}(p_i) - \mathrm{n}(q_i) \geq 6$ to select the partial sum for bisection. In Fig. 2, it can be seen that the first bisection is made at the partial sum of the tap where the partial sums of the tap and all successive taps have $\mathrm{n}(p_i) - \mathrm{n}(q_i) \geq 6$. h_i is always one bit below $\mathrm{n}(q_i)$, as the sign bit is not included in the range $\left[v_i^-, v_i^+\right]$ calculated in (3).

From (1), $\mathrm{n}(a_{i,j})$ grows monotonically with increasing j. This implies that the FF overhead is increasing as the FA reduction in the structural is declining. The gain diminishes with increasing m and can become negative. Furthermore, as $\mathrm{n}(q_i)$ decreases towards c_0, a smaller h_i can be chosen for better gain. This suggests that it is beneficial to make repeated bisections. The bisection can be repeated in two ways. If only area reduction is of interest, it is more optimal to merge p_i^u at a tap and then bisect the partial sum again on the next tap. This will increase the path delay at each merging structural adder. To avoid the delay penalty, the merging and bisecting can be performed simultaneously at the same tap. For this concurrent merge/bisect structural adder a_i, the operand, p_i^l from the previous bisection is bisected again at h_i such that $p_i^l = p_i^{lu} \left| p_i^{ll} \right.$. p_i^{lu} is sign extended and added to p_i^u while p_i^{ll} is added to q_i. The register overhead will be increased by two FFs, but since the additions are independent, the delay through the structural adder is also reduced as the operands are shortened due to the simultaneous merging and bisection. Experiment results showed that the delay reduction is more prominent than the cost of two FFs. Therefore, simultaneous merging and bisection is adopted. For the last tap, one FA delay increment due to the final merging is inevitable as shown in Fig. 4. In most cases, this is not a problem as c_0 does not fall in the critical path. In case the minimal delay is affected, the final merging could be moved to an earlier tap.

For the most optimal trade-off, it is suggested that (2) is calculated for every $i > 2$ for which $\mathrm{n}(p_i) - \mathrm{n}(q_i) \geq 6$ is valid. Every $\{i, j\}$ pair will be checked to determine if a repeated bisection would lead to a better result than the best result obtained from the previous runs. This exhaustive search finds the most optimal cost reduction. The complexity of this search is $\mathrm{O}(n^2)$. On an Intel P4, 3.0 GHz PC with 2 GB RAM, the runtime is below 10 seconds for filters with $N > 1000$.

TABLE I
TEST FILTER DATA

Filter	ω_P [π]	ω_S [π]	A_p [dB]	A_s [dB]	Tap #	Word Length		
						c_n	In	Out
A	0.042	0.14	0.2	-60	60	13	8	25
B	0.8	0.74	0.1	-80	121	14	8	25
C	0.2	0.2125	0.05	-50	441	14	8	26

TABLE II
EXPERIMENTAL RESULTS

Filter	Estimated SA Reduction	Delay [ns]	Total Filter Area [μm^2]			Area Reduction			Compilation Time [s]		
			CSD	[3]	Prop. & [3]	[3]/CSD	Opt./CSD	Prop./[3]	CSD	[3]	Prop.
A	2.03% (51 FA)	∞	188 078	186 608	182 510	0.78%	2.96%	2.19%	65	66	60
		3	212 760	216 798	201 014	-1.90%	5.52%	7.28%	137	131	105
B	2.63% (138 FA)	∞	397 778	396 351	381 684	0.36%	4.05%	3.70%	159	158	141
		3	468 028	463 397	432 645	0.99%	7.56%	6.63%	389	373	285
C	4.58% (900 FA)	∞	1 389 870	1 385 835	1 317 281	0.29%	5.22%	4.94%	535	495	380
		3	1 666 463	1 644 279	1 447 606	1.33%	13.13%	11.96%	1780	1566	975

III. IMPLEMENTATION RESULTS

The proposed method was evaluated using Example 2 from [12] (Filter A), Example 1 from [11] (Filter B) and Filter 3 from [5] (Filter C). The details are given in Table I. All examples used signed digit to optimize the coefficients. The estimated FA reduction of each filter is shown in the second column of Table II. A CSD implementation and a non- pipelined NRSCSE [3] optimized implementation of the MCM block are used to demonstrate the effectiveness of our structural adder optimization. The proposed structural adder optimization scheme was combined with the optimized MCM block and all designs were coded in VHDL and synthesized by Synopsys Design Compiler (Ver. 2007.12) using TSMC 0.18μm standard cell library. The designs were optimized under two different delay constraints, unconstrained (∞) and fixed constraint (3ns). The synthesized area and the percentage area reduction over the baseline CSD implementation were shown in Table II. The compilation time was reported by Design Compiler on Red Hat Linux (Ver. 4.1.2-13) running on a Core2 2.66GHz workstation with 3.3GB RAM.

The area reduction over CSD implementation by [3] is small due to the Boolean optimization made by Synopsys Design Compiler. The combination of the proposed scheme and [3] reduced the area by 2.96% to 13.13% and after discounting the saving due to the MCM block optimization of [3], the actual synthesis results still exceed the expected saving of 2.03%, 2.63% and 4.58% from theoretical estimation for these filters. The discrepancy between the estimated and simulation results can be explained as follows. The proposed optimization trades combinational area against registers and reduces the path delays in several taps. The reduced combinational logic requires less area and the area reduction is more prominent under tighter delay constraints. Moreover, the reduced path delays create more slacks, which enable slower but smaller cells to be used. As filter length increases, the area for the structural adders dominates, up to 96% for Filter C. The time taken by the synthesis tool to optimize the architecture generated by our proposed method has also reduced significantly. The time for unconstrained case was reduced by 7.6%, 11.5% and 29.1% for Filters A, B and C, respectively and 23.2%, 26.6% and 45.2% for the case of fixed delay constrained optimization.

IV. CONCLUSION

This paper presents a new method to reduce the total area of fixed coefficient transposed direct form FIR with a large number of taps by minimizing the bit widths of the structural adders. Sign extensions have been shortened and the delays through the structural adders have been reduced at the expense of some register overhead and a reduced size merged adder for each bisection of a long partial sum. The condition for which area reduction is possible and the optimal bisection were analytically formulated. Theoretical estimate shows an area reduction of up to 4.5% for the structural adder block of the benchmark filters. The proposed method was combined with a MCM optimization algorithm. An area reduction of up to 13.13% (up to 1.33% by MCM optimization and up to 11.8% by the proposed method) was obtained from the synthesis result of long filter.

REFERENCES

[1] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in fir digital filters," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 42, no. 9, pp. 569–577, Sept. 1995.

[2] M. Potkonjak, M. B. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplications: efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 15, no. 2, pp. 151–165, Feb. 1996.

[3] M. Martinez-Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 49, no. 3, pp. 196–203, Mar. 2002, 1057-7130.

[4] F. Xu, C.-H. Chang, and C. C. Jong, "Contention resolution algorithm for common subexpression elimination in digital filter design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 10, pp. 695–700, Oct. 2005.

[5] D. L. Maskell, "Design of efficient multiplierless fir filters," *IET Circuits, Devices & Systems*, vol. 1, no. 2, pp. 175–180, April 2007.

[6] K. Johansson, O. Gustafsson, and L. Wanhammar, "A detailed complexity model for multiple constant multiplication and an algorithm to minimize the complexity," in *Proc. 2005 European Conf. Circuit Theory and Design*, vol. 3, 28 Aug.-2 Sept. 2005, pp. III/465–III/468 vol. 3.

[7] L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Optimization of area in digital fir filters using gate-level metrics," in *Proc. 44th ACM/IEEE Design Automation Conference, 2007. DAC '07*, 4-8 June 2007, pp. 420–423.

[8] B. C. Wong and H. Samueli, "A 200-mhz all-digital qam modulator and demodulator in 1.2-μm cmos for digital radio applications," *IEEE J. Solid-State Circuits*, vol. 26, no. 12, pp. 1970–1980, Dec. 1991.

[9] K. Suzuki, H. Ochi, and S. Kinjo, "A design of fir filter using csd with minimum number of registers," in *Proc. IEEE Asia Pacific Conf. on Circuits Syst. 1996.*, 18-21 Nov. 1996, pp. 227–230.

[10] *TSMC 0.18m Process 1.8-Volt SAGE-XTM Standard Cell Library Databook*, 4th ed. Sunnyvale, CA: Artisan Components Inc., Sept. 2003.

[11] Y. C. Lim and S. Parker, "Discrete coefficient fir digital filter design based upon an lms criteria," *IEEE Trans. Circuits Syst.*, vol. 30, no. 10, pp. 723–739, Oct. 1983.

[12] H. Samueli, "An improved search algorithm for the design of multiplierless fir filters with powers-of-two coefficients," *IEEE Trans. Circuits Syst.*, vol. 36, no. 7, pp. 1044–1047, July 1989.