A DESIGN FLOW FOR MULTIPLIERLESS LINEAR-PHASE FIR FILTERS: FROM SYSTEM SPECIFICATION TO VERILOG CODE

Kai-Yuan Jheng, Shyh-Jye Jou*, and An-Yeu Wu

Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 106, Taiwan, ROC

*Department of Electrical Engineering, National Central University, Zhongli 320, Taiwan, ROC

ABSTRACT

This paper presents a design flow for the multiplierless linear-phase FIR filter synthesizer, which combines several research efforts. We propose a local search algorithm with variable filter order to reduce the number of adders further. In addition, several design techniques are adopted to reduce the hardware complexity of the system. By using this synthesizer, the system designers can design a filter efficiently and a chip can be successfully finished in a very short time.

1. INTRODUCTION

Recent rapid progress in very large scale integrated (VLSI) circuit technology has led to an emerging theme – “System-on-a-Chip” (SoC). With the increase in the density and complexity in VLSI circuit, the design costs for the development of a VLSI chip are also increased. It calls for rapid prototyping and design reuse of major silicon intellectual property (SIP) modules to alleviate the designer's effort and to speed up the design process. Therefore, computer aided design (CAD) tools play an important role in decreasing the design cycle time and accurately simulating the correctness of the circuit design.

The synthesizer we presented can automate the FIR filter design from the system specification to the corresponding synthesizable Verilog hardware description language (HDL) code. Because the synthesizer only requires the system-level specification, the synthesizer allows system designers, who are inexperienced in VLSI design, to design filters easily and concentrate on system design and performance evaluation. Therefore, by using this synthesizer, an efficient design of a chip can be successfully completed in a few minutes.

The rest of this paper is organized as follows. In Section 2, the design flow of the filter synthesizer and several hardware reduction methods are presented. An experimental result of a filter design example synthesized with our automatic design tool is then shown in Section 3. Finally, some conclusions will be given in Section 4.

This work was supported by MediaTek Inc., under NTU-MTK wireless research project.

Fig. 1 Design flow of the synthesizer.

Table 1 Input data of system specification.

Filter Type (low-pass, high-pass, band-pass, band-stop)	
Normalized Passband and Stopband Edge Frequencies	ω_p, ω_s
Passband Ripple in Magnitude or dB	δ_p / A_p
Stopband Attenuation in Magnitude or dB	δ_s / A_s
Input and/or Output Data Word Length (bit)	W_{in} / W_{out}
Signal to Noise Ratio (dB)	SNR

2. SYNTHESIZER IMPLEMENTATION

The system configuration and dataflow of the synthesizer are shown in Fig. 1. The synthesizer consists of many subprograms. The main subprograms are the coefficient optimization, the word length estimation, and the synthesizable Verilog code generation. All programs are written in C++ language.

In this system, the input is the system-level specification, which is listed in Table 1. In addition, the architecture uses the symmetric transposed direct form filter structure with the MSB Fix technique [1], which is frequently adopted by high-speed designs.

2.1 Coefficient Optimization

● Coefficient Calculation

In this subprogram, we integrate the MATLAB engine [2] into our synthesis tool. The floating-point filter coefficient set is calculated by the generalized Reméz method [3] as given in the MATLAB gremez.m function.

● Optimization Algorithm

Numerous search algorithms for the design of multiplierless filters with canonic signed digit (CSD) or signed powers-of-two (SPT) coefficients have been proposed. However, they did not explore the possibility of further reduction of nonzero digits by taking the filter order as a variable parameter. In general, if a filter gradually increases the tap length N , its frequency response will become severer. Thus, we can allow more margins for coefficient quantization error by increasing the filter tap length. Besides, an observation [4] shows that one can start with a filter, which exceeds the given criteria that may involve acceptable level of increase in the filter order, but with much lesser total nonzero digits than the initial design. Therefore, we adopt a two-step local search algorithm proposed by Samuelli [5] and exploit variable filter order to improve the method. The number of total nonzero digits typically decreases with N . However, there is a limit to N since the overhead increases with N .

2.2 Word Length Estimation

● Overflow Prevention

If the final output is within the range of the word length, overflow in partial sums are unimportant. This is a desirable property of 2's complement arithmetic. However, if the final output exceeds the range of the word length, the value of the output sample will be wrong and methods should be taken to prevent this. An approach is to avoid or allow limited overflow by scaling the coefficients. The coefficients $h(k)$ may be scaled in the following way:

$$\hat{h}(k) = h(k) \cdot 2^{-R} \quad (1)$$

where

$$R = \left\lceil \log_2 \left(\sum_{k=0}^{N-1} h^2(k) \right)^{\frac{1}{2}} \right\rceil \quad (2a)$$

or

$$R = \left\lceil \log_2 \sum_{k=0}^{N-1} |h(k)| \right\rceil \quad (2b)$$

where R denotes right shift bit(s). The method given in (2a) probably lead to shorten internal word length than (2b) but this form of scaling will occasionally occur overflow which result in performance degradation. Therefore, the method in (2b) is adopted which never cause overflow because it is based on the worst-case conditions for overflow. Hence, the coefficient word length increases R bit(s) and the coefficients are then shifted right R bit(s) to prevent overflow.

● Internal Word Length Reduction

In digital signal processing, the finite word length has a strong effect on the system performance since it dominates the precision of the output signals. The increment of

Fig. 2 Internal word length reduction flowchart.

Fig. 3 SNR evaluation block.

internal word length will lead to a better signal-to-noise ratio (SNR), but it would also increase the hardware complexity, consume more power, and slow down the system operation frequency. Therefore, it is a trade-off that the designer should take care of.

It is observed that if designer is willing to accept some deviation from the given specifications, the decrement of internal word length enable a reduction of hardware complexity. In this subprogram, we involve a deviation index SNR that is defined in (3)

$$SNR = 10 \log \left[\frac{E(y^2(n))}{E(e^2(n))} \right] = 10 \log \left[\frac{E(y^2(n))}{E(y(n) - \hat{y}(n))^2} \right] \quad (3)$$

The internal word length reduction flowchart and SNR evaluation block are shown in Fig. 2 and Fig. 3 respectively. The initial internal word length will be evaluated for the result that does not introduce any error first. Then the internal word length will be decreased to the value that its SNR value still fits the specification. Finally, the minimum internal word length, which fulfills the specification, will be obtained.

2.3 Synthesizable Verilog Code Generation

Finally, we will generate three types of the symmetric transposed direct form FIR filters as shown in Fig. 4.

Fig. 4 Transposed direct form filter structure written in (a) behavioral code, (b) DesignWare components and, (c) DesignWare components with pipelining.

Fig. 5 The Structure C with an input buffer.

- Structure A: Fig. 4(a)

The transposed direct form filter structure is adopted and written in behavior level synthesizable Verilog-HDL code, which allows the synthesis tool to select the appropriate architecture for user's constraints.

- Structure B: Fig. 4(b)

The transposed direct form filter structure is utilized with carry save adders (CSA) written in DesignWare components [6] provided by Synopsys.

- Structure C: Fig. 4(c)

We exploit structure B with pipelining to achieve a two-CSA delay critical path. Moreover, the nonzero digits of most CSD coefficient sets is generally less than three so

Table 2

Minimum number of SPT terms required to attain -50dB NPR		
Algorithm	#SPT	N
Max. SPT per coeff. = 4		
MILP [7]	68	28
Samueli [5]	66	28
Our Work #1	54	29
Our Work #2	52	30
Max. SPT per coeff. = 3		
MILP [7]	68	28
Samueli [5]	cannot reach -50 dB	
Our Work #3	57	29

Table 3 Word length estimation results.

Input Word Length	Coefficient Word Length	Internal Word Length	Output Word Length
14 bits	11 bits	14 bits	14 bits

we can use a single input buffer rather than pipelining at each tap. Referring to Fig. 5, the input $x(n)$ is for the taps whose nonzero digits are more than two and $x(n-1)$ is for less than three.

3. DESIGN EXAMPLE

A linear-phase low-pass FIR filter is designed using our proposed method, the mixed integer linear programming (MILP) algorithm [7], and Samueli's local search algorithm [5]. The pass-band and stop-band edge frequencies are 0.3π and 0.5π , respectively. The normalized peak ripple (NPR) $\delta_{NPR} = -50\text{dB}$. The word length of the input signal is assumed 14 bits.

The minimum number of SPT terms required by the various methods mentioned above is summarized in Table 2. The frequency responses and coefficients of the filter designed by our proposed method are shown in Fig. 6. When the maximum allowed number of SPT terms per coefficient is limited to four, the filter designed by our methods saves 22%(21%~24%) SPT terms and costs 5%(4%~7%) additional tap length. If the application requires us to limit the maximum number of SPT terms per coefficient to three, for a higher throughput rate, the filter designed using Samueli's algorithm failed to reach -50 dB NPR. However, using our proposed method can save 16% SPT terms and costs 4% additional tap length.

Secondly, the design results of the word length estimation are summarized in Table 3. In general, the SNR is set more than 40 dB for practical implementation.

Lastly, the design results are converted into three structures mentioned in Section 2.3. We then use the Synopsys Design Compiler to synthesize the filters with TSMC 0.25 μm process. The synthesis results of Work #1 are summarized in Table 4. The area is measured in equivalents of 2-input NAND gates.

Work #1	
$h(0) = h(28) = -2^{-9} - 2^{-11}$	$h(8) = h(20) = 2^{-5} + 2^{-8} - 2^{-11}$
$h(1) = h(27) = -2^{-9} - 2^{-11}$	$h(9) = h(19) = 0$
$h(2) = h(26) = 2^{-8}$	$h(10) = h(18) = -2^{-4}$
$h(3) = h(25) = 2^{-7}$	$h(11) = h(17) = -2^{-4} + 2^{-7}$
$h(4) = h(24) = 0$	$h(12) = h(16) = 2^{-3} - 2^{-5} - 2^{-7}$
$h(5) = h(23) = -2^{-6} + 2^{-11}$	$h(13) = h(15) = 2^{-2} + 2^{-5} + 2^{-8}$
$h(6) = h(22) = -2^{-6} + 2^{-8} - 2^{-11}$	$h(14) = 2^{-1} - 2^{-3} + 2^{-8} + 2^{-10}$
$h(7) = h(21) = 2^{-6} + 2^{-10}$	
Work #2	
$h(0) = h(29) = -2^{-10}$	$h(8) = h(21) = 2^{-5} + 2^{-7} - 2^{-10}$
$h(1) = h(28) = -2^{-8}$	$h(9) = h(20) = 2^{-5}$
$h(2) = h(27) = 0$	$h(10) = h(19) = -2^{-4} + 2^{-6} + 2^{-8} + 2^{-10}$
$h(3) = h(26) = 2^{-7}$	$h(11) = h(18) = -2^{-3} + 2^{-5}$
$h(4) = h(25) = 2^{-7} - 2^{-10}$	$h(12) = h(17) = 0$
$h(5) = h(24) = -2^{-7} - 2^{-9}$	$h(13) = h(16) = 2^{-2} - 2^{-7} - 2^{-9}$
$h(6) = h(23) = -2^{-5} + 2^{-7} + 2^{-9}$	$h(14) = h(15) = 2^{-1} - 2^{-4} + 2^{-6}$
$h(7) = h(22) = 0$	
Work #3	
$h(0) = h(28) = -2^{-9} - 2^{-11}$	$h(8) = h(20) = 2^{-5} + 2^{-7} - 2^{-9}$
$h(1) = h(27) = -2^{-9} - 2^{-11}$	$h(9) = h(19) = 0$
$h(2) = h(26) = 2^{-8}$	$h(10) = h(18) = -2^{-4} - 2^{-8}$
$h(3) = h(25) = 2^{-7} + 2^{-11}$	$h(11) = h(17) = -2^{-4} + 2^{-8}$
$h(4) = h(24) = 0$	$h(12) = h(16) = 2^{-3} - 2^{-5} - 2^{-9}$
$h(5) = h(23) = -2^{-6} - 2^{-11}$	$h(13) = h(15) = 2^{-2} + 2^{-4} - 2^{-7}$
$h(6) = h(22) = -2^{-6} + 2^{-8} - 2^{-10}$	$h(14) = 2^{-1} - 2^{-3} + 2^{-5}$
$h(7) = h(21) = 2^{-6} + 2^{-9}$	

Fig. 6 Magnitude responses and coefficient sets of the filters.

The synthesis results show that structure A is suitable for the low-speed (133MHz) and area-efficient application; Structure B is suitable for the high-speed (400MHz) application; and Structure C is suitable for the very high-speed (800MHz) application. Therefore, our filter synthesizer can provide flexible hardware implementation for various applications.

Table 4 Synthesis results of Work #1.

(a) Timing Constraint: 7.50(ns)			
Structure	A	B	C
Critical Path (ns)	7.46	4.65	4.65
Total Gate Count	5069	8103	9119
Combinational Area	2824	3613	3907
Noncombinational Area	2245	4490	5212
(b) Timing Constraint: 2.50(ns)			
Structure	A	B	C
Critical Path (ns)	3.86	2.50	2.50
Total Gate Count	8338	8563	9225
Combinational Area	5799	4038	3973
Noncombinational Area	2539	4525	5252
(c) Timing Constraint: 1.25(ns)			
Structure	A	B	C
Critical Path (ns)	3.86	1.57	1.25
Total Gate Count	8338	11520	12862
Combinational Area	5799	5595	5999
Noncombinational Area	2539	5925	6863

4. CONCLUSION

We have implemented a multiplierless FIR filter synthesizer written in C++ language and combined the MATLAB engine with our automatic design tool. We have also shown that the local search algorithm with variable filter order towards further reduction in the number of total nonzero digits. The variable filter order approach can be applied to other coefficient optimization algorithms.

Several design techniques are adopted to reduce the hardware complexity of the system. For flexible hardware implementation, we provide three structures that structure A is suitable for low-power applications and structures B, C are suitable for high-performance applications. We also find that the coefficient sets produced by our tool have many common terms, so common sub-expression elimination (CSE) techniques will be studied in the future.

5. REFERENCES

- [1] B. C. Wong and H. Samuelli, "A 200-MHz all-digital QAM modulator and demodulator in 1.2μm CMOS for digital radio applications," IEEE JSSC, pp. 1970-1979, Dec. 1991.
- [2] "MATLAB external interfaces," MathWorks, Sep. 2003.
- [3] D. J. Shpak and A. Antoniou, "A generalized Reméz method for the design of FIR digital filters," IEEE Trans. Circuits Syst., pp. 161-174, Feb. 1990.
- [4] M. Bhattacharya and T. Saramaki, "Some observations on multiplierless implementation of linear phase FIR filters," IEEE ISCAS, pp. 193-196, May 2003.
- [5] H. Samuelli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," IEEE Trans. Circuits Syst., pp. 1044-1047, Jul. 1989.
- [6] "DesignWare foundation library databook," Synopsys, Inc., Jan. 2002.
- [7] Y. C. Lim and S. R. Parker, "FIR filter design over a discrete powers-of-two coefficient space," IEEE Trans. ASSP, pp. 583-591, Jun. 1983.