Sign-extension avoidance and word-length optimization by positive-offset representation for FIR filter design

Huang, Ruimin; Chang, Chip-Hong; Faust, Mathias; Lotze, Niklas; Manoli, Yiannos

2011

Huang, R., Chang, C.-H., Faust, M., Lotze, N., & Manoli, Y. (2011). Sign-extension avoidance and word-length optimization by positive-offset representation for FIR filter design. IEEE transactions on circuits and systems II : express briefs, 58(12), 916-920.

https://hdl.handle.net/10356/96292

https://doi.org/10.1109/TCSII.2011.2168130

Sign-extension Avoidance and Word-length Optimization by Positive Offset Representation for FIR Filter Design

Ruimin Huang, Chip-Hong Chang, *Senior Member, IEEE*, Mathias Faust, *Student Member, IEEE*, Niklas Lotze, Yiannos Manoli, *Senior Member, IEEE*

***Abstract*—This brief proposes a new approach to utilizing positive-offset representation for sign-extension avoidance in shift-and-add implementation of FIR filter. Affine arithmetic is used to model the excess offsets in order to curtail the word-length expansion problem. Tighter probabilistically justified word-length bounds are determined to enable further offset to be removed from each tap. The approach is applicable even after the redundant adders in the multiplier block of the filter have been minimized. Our simulation results show an average power reduction of about 19% over and above the savings achieved by sharing of adders in multiple constant multiplication.**

Index Terms— **finite-impulse response (FIR) filters, common sub-expression space, sign extension, positive-offset**

I. Introduction

Finite-impulse response (FIR) filters are widely used in many mobile applications to perform versatile functions, such as spectral shaping, adaptive equalization, etc. Very often the area and power budgets are so stringent that they can only be met by dedicated hardwired implementations of significantly higher computational density than programmable processor based implementations. Efficient VLSI implementations of FIR filters have been proposed, with configurable coefficients and filter order to adapt to different communication standards [1]. These filters are mainly designed based on direct form structure and additional pipelined registers have to be added to increase the throughput rate.

For high-order filters, which are usually designed based on constant coefficients, the transposed-direct form structure, as shown in Fig. 1 is preponderant because the throughput rate is independent of the filter order. The coefficients are represented in signed-power-of-two (SPT) terms and canonical signed digit (CSD) representation [2]-[5] to decompose the multiplier block into a shift-and-add network. The complexity of the multiplier block can be reduced by either common subexpression elimination (CSE) [2]-[5] or graph dependence (GD) [6]-[9] algorithms, making fixed-coefficient custom filters 15- to 30-fold computationally denser than the configurable implementations.

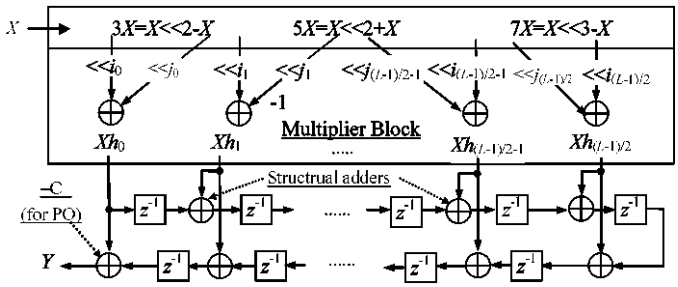

Figure 1 An L-tap (L is even) linear phase transposed-direct form FIR filter based on sub-expression space: '<<' represents left-shift; h_0X is the product of input sample X and the constant coefficient h_0; z^{-1} denotes the delay register.

It has been accustomed to use 2's complement arithmetic in filter implementation without being aware of the potential increase in dynamic power dissipation. Even though the number of adders can be reduced by CSE and GD algorithms, the addends must be sign extended up to the most significant bit (MSB) of the sum for each adder. This increases the asymmetry in the path delays of each filter tap. More glitches are generated and propagated, especially when the signals frequently switch around zero and induce spurious transitions around the MSB. Modifications to the number representation have been proposed to avoid the sign extension in 2's complement [10]. The use of sign-magnitude, gray code, logarithm and other number representations to reduce signal switching activities often complicates the arithmetic operations. Other techniques that preserve the readily use of 2's complement arithmetic in FIR filter design include distributed arithmetic (DA) that stores the nonnegative partial products in lookup table for sign-extension avoidance [11] and MSB-fix with compensation vectors inserted before the filter [12]. In [13], Gustafsson et al. proposed three techniques to avoid sign-extensions in the multiplier block. Among them, the addition of input bias is the most energy efficient. However, these methods did not address the concomitant word-length (WL) growth stems from the expanded operand ranges. While the adder ranges within the multiplier block are determined by the input and coefficients, the ranges of the structural adders increase monotonically along the tap-delay line as their inputs are uncorrelated. This brief

Manuscript received April 14, 2011, revised July 10, 2011.

R. M. Huang is with School of Information Science and Engineering, Huaqia University, China, 361021 (e-mail: huangruimin@hqu.edu.cn).

C. H. Chang and M. Faust are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: {echchang, faus0001}@ntu.edu.sg).

N. Lotze and Y. Manoli, are with the Fritz-Hüttinger-Professur für Mikroelektronik, University of Freiburg, Germany (e-mail: {lotze, manoli}@imtek.de).

formalizes and expands the original notion of input bias [13] to further restrict the ranges of the operands in positive offset (PO) form. One important and distinctive contribution of this brief is the range analysis of PO expressions to remove the excess offsets in transposed-direct form filter after the number of arithmetic operations in the multiplier block has been minimized by either CSE or GD algorithm. A means to preset the tap registers is also suggested to avoid the first L invalid output samples of input bias [13]. Our simulation results show that the proposed PO based sign-extension avoidance with legitimate WL optimization can lead to an overall reduction of area, delay and power consumption.

The rest of this brief is organized as follows. Section II introduces the PO representation. Section III analyzes the excess offsets of affine transformations in PO representation to be removed for WL optimization. Section IV presents the experimental results and the brief is concluded in Section V.

II. PO Variable for Sign Extension Avoidance

A PO number Z is obtained from an n-bit 2's complement number X by inverting the MSB of X. Z is a positive integer variable obtained by adding to the integer variable X an offset of 2^{n-1} corresponding to the input bias suggested by [13].

$$Z = X + 2^{n-1} = -x_{n-1}2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i + 2^{n-1} = \overline{x}_{n-1}2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i \quad (1)$$

where $\overline{x} \in \{0,1\}$ denotes the complement of the binary variable x.

By restricting the input signal range to $[-2^{n-1}+1,\ 2^{n-1}-1]$ as in sign-magnitude representation [13], it will simplify, without loss of generality, the excess-offset subtraction to be discussed in Section III.*B*. By adding an offset of $2^{n-1}-1$ to the negation of X, we have

$$\begin{aligned} -X + \left(2^{n-1}-1\right) &= x_{n-1}2^{n-1} + \sum_{i=0}^{n-2}(1-x_i)2^i \\ &= x_{n-1}2^{n-1} + \sum_{i=0}^{n-2}\overline{x}_i 2^i = \overline{Z} \end{aligned} \quad (2)$$

where $\overline{Z}$ denotes the 1's complement of Z. In other words, negation of a 2's complement number can be obtained simply by subtracting an offset of $2^{n-1}-1$ from the 1's complement of its PO representation.

Let an n_c-bit coefficient, h_i, $i = 1, 2, \ldots, L$, of an L-order FIR filter be expressed as a sum of SPT terms, where $j_i^+, j_i^- \in [0, n_c-1]$ denote the exponents of the positive and negative SPT terms of h_i, respectively. Using (1) and (2),

$$\begin{aligned} Y &= \sum_{i=0}^{L-1} h_{L-i}X_i = \sum_{i=0}^{L-1}\left(\sum_{j_{L-i}^+} 2^{j_{L-i}^+} - \sum_{j_{L-i}^-} 2^{j_{L-i}^-}\right)X_i \\ &= \sum_{i=0}^{L-1}\sum_{j_{L-i}^+} 2^{j_{L-i}^+} Z_i - 2^{n-1}\sum_{i=0}^{L-1}\sum_{j_{L-i}^+} 2^{j_{L-i}^+} \\ &\quad + \sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-}\overline{Z_i} - (2^{n-1}-1)\sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-} \\ &= \left\{\sum_{i=0}^{L-1}\sum_{j_{L-i}^+} 2^{j_{L-i}^+} Z_i + \sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-}\overline{Z_i}\right\} \\ &\quad -2^{n-1}\sum_{i=0}^{L-1}\sum_{j_{L-i}^+} 2^{j_{L-i}^+} - (2^{n-1}-1)\sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-} \\ &= \hat{Y} - 2^{n-1}\sum_{i=0}^{L-1}\sum_{j_{L-i}^+} 2^{j_{L-i}^+} + 2^{n-1}\sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-} - (2^n-1)\sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-} \\ &= \hat{Y} - 2^{n-1}\sum_{i=0}^{L-1} h_{L-i} - (2^n-1)\sum_{i=0}^{L-1}\sum_{j_{L-i}^-} 2^{j_{L-i}^-} = \hat{Y} - C \end{aligned} \quad (3)$$

The output Y of the filter can be obtained as a sum of a variable $\hat{Y}$ and a constant C. From (3), $\hat{Y}$ being the sum of two positive variables is always positive. Since all variables involved in the computation are positive, sign extension is obviated. C is a constant that can be computed directly from the filter coefficients and compensated once at the final tap. Since $Y = \sum_{i=0}^{L-1} h_{L-i}X_i = 0$ when $X_i = 0\ \forall i \in [0, L-1]$, $Y = \hat{Y} - C = 0 \Rightarrow \hat{Y} = C$. C is equal to the first output sample after flushing the filter with a stream of L zero inputs. To avoid the flushing period [13], the registers at the tap-delay line can be preset to the values equal to the steady-state values after flushing L zero inputs, which is also the offsets relative to the implementation using 2's complement representation with sign-extension.

III. Word-Length Optimization for PO Operations

Multiplying a variable by a constant in PO representation is an affine transformation defined by $\mathrm{PO}(h_i X) = h_i Z = h_i(2^{n-1} + X)$ for $h_i > 0$ and $\mathrm{PO}(h_i X) = |h_i|\overline{Z} = |h_i|(2^{n-1} - 1 - X)$ for $h_i < 0$. The range is $[h_i, (2^n-1)h_i]$ for positive h_i and $[0, (2^n-2)|h_i|]$ for negative h_i. Thus, the WL required to represent the product is $W = n + \lceil \log_2 |h_i| \rceil$, where $\lceil w \rceil$ denotes the smallest integer greater than or equal to w. This is also the minimum WL needed to represent the 2's complement product. This affine transformation has a variable span and a fixed positive offset of at least half the span to keep the integer within the positive range. While the span is elastic when two operands are added or subtracted, the offset cumulates monotonically. Offset in excess of the minimum dynamic range calls for unnecessarily larger operator, increases the critical path delay and wastes power. In this section, affine arithmetic [14] is used to analyze the dynamic range of the intermediate results and the product of each coefficient multiplier, with filter coefficients expressed in SPT terms and input variable in PO form. Aggregate offsets in excess of the minimum dynamic range are detected and subtracted timely before they are allowed to escalate into the next operation. The aggregate offset removed can be determined by flushing the filter with zero inputs and compensated at the last tap.

A. Range of PO coefficient multipliers

By restricting X to its maximum symmetric range, it can be expressed in an affine form as follows:

$$X = \left(2^{n-1} - 1\right)\varepsilon \tag{4}$$

where $\varepsilon \in [-1,1]$ models the uncertainties in X, and $n \geq 2$.

The filter coefficient h_i is usually expressed as a sum of SPT terms in CSD representation. Fewer partial products are generated when it is multiplied by an input variable but negation of input variable is needed due to the presence of signed constant –1. In CSD representation, the coefficient h_i can be decomposed into positively and negatively weighted sums of power-of-two terms as follows.

$$h_i = \sum_{j_i^+} 2^{j_i^+} - \sum_{j_i^-} 2^{j_i^-} = h_i^+ - h_i^- \tag{5}$$

where $h_i^+, h_i^- \geq 0$.

It follows from (1) (2) and (4) that

$$\begin{aligned} \mathrm{PO}(h_i X) &= h_i^+ Z + h_i^- \overline{Z} \\ &= 2^{n-1} h_i^+ + \left(2^{n-1} - 1\right)\varepsilon h_i^+ + \left(2^{n-1} - 1\right)(1-\varepsilon) h_i^- \\ &= \left(2^{n-1} - 1\right)\varepsilon\left(h_i^+ - h_i^-\right) + \left(2^{n-1} - 1\right) h_i^- + 2^{n-1} h_i^+ \end{aligned} \tag{6}$$

For $h_i > 0$, $|h_i| = h_i = h_i^+ - h_i^-$. By substituting $h_i^+ = h_i + h_i^-$ in (6), we have

$$\begin{aligned} \mathrm{PO}(h_i X) &= \left(2^{n-1} - 1\right)\varepsilon h_i + 2^{n-1}\left(h_i + h_i^-\right) + \left(2^{n-1} - 1\right) h_i^- \\ &= \left(2^{n-1} - 1\right)(1+\varepsilon) h_i + \left(2^n - 1\right) h_i^- + h_i \\ &= \left(2^{n-1} - 1\right)\left(|h_i| + \varepsilon h_i\right) + \left(2^n - 1\right) h_i^- + h_i \end{aligned} \tag{7}$$

For $h_i < 0$, $|h_i| = -h_i = h_i^- - h_i^+$. By substituting $h_i^- = |h_i| + h_i^+$ in (6), we have

$$\begin{aligned} \mathrm{PO}(h_i X) &= \left(2^{n-1} - 1\right)\varepsilon h_i + 2^{n-1} h_i^+ + \left(2^{n-1} - 1\right)\left(|h_i| + h_i^+\right) \\ &= \left(2^{n-1} - 1\right)\left(|h_i| + \varepsilon h_i\right) + \left(2^n - 1\right) h_i^+ \end{aligned} \tag{8}$$

The range and WL of the product are $[(2^n - 1)h_i^- + h_i, (2^n - 1)h_i^+]$ and $n + \lceil \log_2(h_i^+) \rceil$, respectively for positive h_i and $[(2^n - 1)h_i^+, (2^n - 1)h_i^- + h_i]$ and $n + \lceil \log_2(h_i^-) \rceil$, respectively for negative h_i. These WLs deviate from W by no more than one bit.

B. Excess offset removal from coefficient multipliers

The ranges of the first terms in (7) and (8) are both equal to $[0, |h_i|(2^n - 2)]$. Thus $(2^n - 1)h_i^- + h_i$ in (7) and $(2^n - 1)h_i^+$ in (8) represent the excess offsets that can be removed to avoid undesirable WL expansion for their accumulation in the tap-delay line. The excess offset can be reduced by subtracting $2^n h_i^-$ from (7), which requires a much shorter adder than removing it completely. The remnant $h_i - h_i^- = h_i^+ - 2h_i^- > 0$ because there is no consecutive nonzero digit in CSD representation. On the other hand, subtracting $2^n h_i^+$ from (8) will cause the result to underflow 0. Instead, (8) can be generated by inverting the corresponding positive product. By definition of 1's complement,

$$\begin{aligned} &\mathrm{PO}(-h_i X) = \overline{h_i Z} \\ &= 2^W - 1 - \left\{\left(2^{n-1} - 1\right)\varepsilon\left(h_i^+ - h_i^-\right) + \left(2^{n-1} - 1\right) h_i^- + 2^{n-1} h_i^+\right\} \\ &= 2^W - 1 - \left(2^{n-1} - 1\right)\varepsilon h_i - \left(2^{n-1} - 1\right)(h_i^+ - |h_i|) - 2^{n-1} h_i^+ \\ &= \left(2^{n-1} - 1\right)|h_i| - \left(2^{n-1} - 1\right)\varepsilon h_i + 2^W - 1 - (2^{n-1} + 2^{n-1} - 1) h_i^+ \\ &= \left(2^{n-1} - 1\right)\left(|h_i| - \varepsilon h_i\right) + 2^W - 1 - \left(2^n - 1\right) h_i^+ \end{aligned} \tag{9}$$

The range of (9) becomes $[\, 2^W - 1 - (2^n - 1)h_i^+ , 2^W - 1 - h_i^+ - (2^n - 2)h_i^- \,]$. When $h_i^- = 0$, the upper bound is close to 2^W. There is a tendency to overflow 2^W when another PO number is added. For example, if $-39Z$ is obtained by $\overline{40Z} + Z$, its upper bound $2^{n+6} - 1 - 40 + 2^n - 1$ exceeds 2^{n+6} when $n \geq 6$. Since the first term of (9) has a range of $[0, (2^n - 2)|h_i|]$, the excess offset of $2^W - 1 - (2^n - 1)h_i^+$ can be easily curtailed by subtracting from it $2^W - 2^n h_i^+$, which is equivalent to adding $2^n h_i^+$ to (9) modulo 2^W. This will reduce the range of (9) to $[h_i^+ - 1, (2^n - 2)|h_i| + h_i^+ - 1]$, which has a much closer upper bound to $(2^n - 2)|h_i|$.

C. Excess offset removal from addition of partial products

Based on the above analysis, the WL of each adder in the multiplier block of Fig. 1 can be optimized by removing its excess offset. A multi-root reduced adder graph is used to represent the shift-and-add network, where each node represents an adder [4]. The output $A(u,v)$ of each adder can be expressed as:

$$A(u,v) = \left(2^{s_1} u + v\right) 2^{s_0} \tag{10}$$

where $s_0, s_1 \geq 0$ are integer constants representing the amount of shifts applied to the output and the operand u of the adder, respectively. The operands, $u = \pm\lambda X$ and $v = \pm\eta X$ are the partial products of a coefficient multiplier, where λ and η are positive integer constants, i.e., $\lambda, \eta \geq 0$.

Let $tX = 2^{-s_0} A(u,v)$ so that the output shift s_0 after the operation can be ignored in the range analysis of $t = \pm 2^{s_1}\lambda \pm \eta$. If u and v are both positive or both negative, according to the analysis of Section III.*B*, it can be proved that after the subtraction of the excess offsets of $2^n \lambda^-$ from λZ and $2^n \eta^-$ from ηZ for u, $v > 0$ or $2^{W_\lambda} - 2^n \lambda^+$ from $\overline{\lambda Z}$ and $2^{W_\eta} - 2^n \eta^+$ from $\overline{\eta Z}$ for u, $v < 0$, the residual excess offset are negligible compared to 2^n, where λ^- and λ^+ are the negatively and positively weighted sum of power-of-two terms of λ, η^- and η^+ are the negatively and positively weighted sum of power-of-two terms of η, and W_λ and W_η are the truncated word-lengths of PO(λX) and PO(ηX), respectively. On the other hand, if only $v = -\eta X$ is negative, i.e., $t = 2^{s_1}\lambda - \eta$, then

$$\mathrm{PO}(tX) = 2^{s_1}\left(\lambda Z - 2^n \lambda^-\right) + \overline{\eta Z} - 2^{W_\eta} + 2^n \eta^+ \tag{11}$$

Using the results from (7) and (9), (11) can be written as:

$$\begin{aligned}&\mathrm{PO}(tX)\\&=2^{s_1}\left(\left(2^{n-1}-1\right)\left(\lambda+\varepsilon\lambda\right)+\lambda-\lambda^{-}\right)+\left(2^{n-1}-1\right)\left(\eta-\varepsilon\eta\right)+\eta^{+}-1\\&=\left(2^{n-1}-1\right)\left(t+\varepsilon t\right)+2\eta\left(2^{n-1}-1\right)+2^{s_1}\left(\lambda-\lambda^{-}\right)+\eta^{+}-1\\&=\left(2^{n-1}-1\right)\left(t+\varepsilon t\right)+2^{n}\eta+t-2^{s_1}\lambda^{-}+\eta^{-}-1\end{aligned} \tag{12}$$

If only $u = -\lambda X$ is negative, i.e., $t = -2^{s_1}\lambda + \eta$, then

$$\mathrm{PO}(tX) = 2^{s_1}\left(\overline{\lambda Z} - 2^{W_\lambda} + 2^{n}\lambda^{+}\right) + \eta Z - 2^{n}\eta^{-} \tag{13}$$

Similarly, using $2^{s_1}\lambda = |t| + \eta$ and the results from (7) and (9), (13) can be rewritten as:

$$\begin{aligned}&\mathrm{PO}(tX)\\&=\left(2^{n-1}-1\right)\left(|t|+\varepsilon t\right)+2\eta\left(2^{n-1}-1\right)+2^{s_1}\left(\lambda^{+}-1\right)+\eta-\eta^{-}\\&=\left(2^{n-1}-1\right)\left(|t|+\varepsilon t\right)+2^{n}\eta+2^{s_1}\left(\lambda^{+}-1\right)-\eta-\eta^{-}\end{aligned} \tag{14}$$

The excess offsets after the first terms of (12) and (14) are in the order of 2^n. If $\lceil \log 2(|t|+\eta)\rceil = \lceil \log 2|t|\rceil$, the WL will not exceed W bits and the offset needs not be removed. The offset may accumulate to cause the range of the partial result at some point along the tap-delay line to exceed the WL required by the 2's complement method. The above formulas can then be used to analyze the range of the structural adder to determine if the excess offset needs to be subtracted to bring the WL back to W.

D. Tighter word-length control

Ideally, the range of PO(h_iX) with no excess offset comprises two components, $|h_i|$ and εh_i, as shown in the first term of (7) and (9). The former provides a constant positive offset to ensure that the result is always kept within a legitimate range of $[0, |h_i|(2^n-2)]$. Assume that the ideal coefficient multipliers, PO(h_iX) are accumulated up to tap k. The sum $\hat{S}_k$ at tap k is:

$$\hat{S}_k = \left(2^{n-1}-1\right)\left(\sum_{i=0}^{k}|h_i| + \sum_{i=0}^{k}\varepsilon_{L-i}h_i\right) \tag{15}$$

The term $\sum_{i=0}^{k}|h_i|$ guarantees the legitimate positive range for the worst case condition when all variables ε_{L-i} in (15) are simultaneously maximal with the opposite sign to h_i. However, this is unlikely to happen. Based on similar analysis as [14] by assuming that the variables ε_{L-i} in (15) are independent random variables uniformly distributed in [−1, 1], the maximal swing of the term $S_k = \sum_{i=0}^{k}\varepsilon_{L-i}h_i$ can be estimated by the central limit theorem and $S_k \big/ \sqrt{k\sigma^2(S_k)} \sim \mathrm{N}(0,1)$, where $\sigma^2(x)$ is the variance of x and $x \sim \mathrm{N}(0,1)$ implies that the random variable x has a standard normal distribution with mean 0 and variance 1. The probabilistic bound B_k of S_k can be derived by using the inverse cumulative density function for k = 1, 2 and 3, and Gaussian approximation for k > 3 [14]. Hence, the excess offset of $\sum_{i=0}^{k}|h_i| - B_k$ can be subtracted from $\hat{S}_k$ at appropriate tap k for WL optimization based on the empirical analysis that overflow is rare and has insignificant effect on the accuracy of the final output. In the next section, B_k is obtained by simulation using randomly generated signals on several filter examples.

IV. Experimental Results

Table I shows the partial product generation of frequently encountered odd fundamentals for FIR filter design. The first column lists the odd fundamentals, the second and third columns list the PO expressions of positive and negative partial products, respectively upon removal of excess offsets, where '<<' denotes the left-shift operation and n the WL of input X.

TABLE I
PO Representations of Regular Odd Fundamentals

c	PO(cX)	PO($-cX$)
3	Z << 1 + Z	$\overline{\mathrm{PO}(3X)} + (11)_2 << n$
5	Z <<2 + Z	$\overline{\mathrm{PO}(5X)} + (101)_2 << n$
7	$Z << 3 + \overline{Z} + (111) << n$	$\overline{Z << 3 + \overline{Z}}$
9	Z << 3 + Z	$\overline{\mathrm{PO}(9X)} + (1001)_2 << n$
11	Z << 3 + Z << 1 + Z	$\overline{\mathrm{PO}(11X)} + (1011)_2 << n$
13	Z << 3 + Z << 2 + Z	$\overline{\mathrm{PO}(13X)} + (1101)_2 << n$
15	$Z << 4 + \overline{Z} + (1111)_2 << n$	$\overline{Z << 4 + \overline{Z}}$

Five FIR filters, Xu07_28 (Xu07) [3], Yeung04_40 (Ye04) [15], L2 [2], Maskell07_A108 (Ma07) [16] and Aksoy07_A200 (Ak07) [17] were implemented with 8 bit input using 2's complement with sign-extension and PO with the worst-case WL as well as WL optimization presented in Section III.*D*. All but L2 were optimized using the multi-root binary partition graph [4]. The odd fundamentals of L2 are obtained from [2] and mapped to PO representations according to Table I. The partial products of identical polarity were maximally exploited in L2 so as to reduce the logic needed for excess offset removal as presented in Section III.*C*.

Fig. 2 shows the growth of WL along the tap-delay line in the worst case (denoted by 'WW') and under tighter control (denoted by 'TW') for L2 and Ye04. The tighter word lengths were obtained from the maximum swing of each tap by simulating the filters with random input signals generated by an *m*-sequence of length 12. Total WL reductions of 2, 7, 45, 81 and 168 bits for Xu07, Ye04, L2, Ma07 and Ak07, respectively, are observed with TW, and the worst-case overflow probability based on the central limit theorem are 0.388%, 1.132%, 0.353%, 0.038% and 0.344%, respectively.

All designs were synthesized by Synopsys Design Compiler using UMC-Faraday 0.13-μm cell library. The synthesis results for WW and TW constraints are shown in Table II, where the area is expressed in the number of cell units (1 cell unit ≅ 1.2 μm^2), the critical path delay in ns and the power in mW simulated at 100MHz with randomly generated input signals. Each filter in the first column is specified with the number of taps, the coefficient WL, and the number of adders in the multiplier block. The methods are identified by S for 2's complement with sign-extension and P for PO and B for input-bias [13] in the second column. For each filter, the best results of each performance metric for WW and TW are

highlighted in bold print.

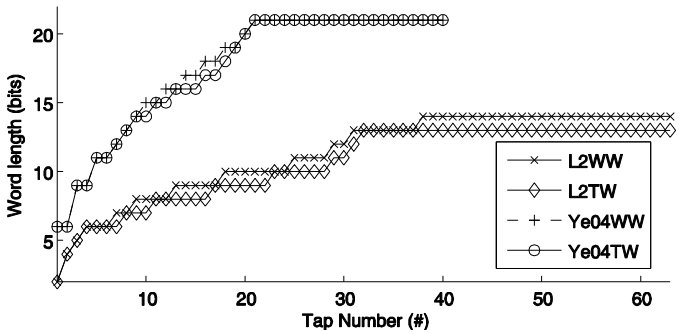

Figure 2 WL of filter tap for L2 and Ye04 under the worst-case and tighter WL constraints.

TABLE II
COMPARISON BETWEEN CONVENTIONAL, INPUT BIAS AND PO APPROACHES

FIR Filter		Area (No. of cells) (1 cell =1.2μm^2)		Delay (ns)		Power (mW)	
		WW	TW	WW	TW	WW	TW
Xu07[3] 28, 11, 9	S	22622	22522	4.19	**4.19**	3.41	3.39
	P	**20777**	**20684**	5.07	5.07	**3.26**	**3.31**
	B	22619	-	**4.18**	-	3.26	-
Ye04[15] 40, 19, 42	S	57644	57315	6.68	6.87	11.10	11.17
	P	**52583**	**52254**	**5.90**	**6.00**	**8.95**	**9.01**
	B	56022	-	6.01	-	9.77	-
L2[2] 63, 11, 17	S	57444	55458	4.86	4.76	9.07	8.77
	P	**53417**	**51517**	**4.67**	**4.59**	**7.92**	**7.94**
	B	56906	-	4.87	-	8.08	-
Ma07[16] 108, 9, 31	S	96297	92754	6.13	5.29	16.80	15.30
	P	**90716**	**87912**	**4.84**	**5.04**	**13.30**	**13.30**
	B	97012	-	4.86	-	14.60	-
Ak07[17] 200, 15, 129	S	252303	244704	8.66	8.36	50.80	49.50
	P	**234597**	**227119**	**6.80**	**6.90**	**40.60**	**40.10**
	B	250103	-	7.16	-	42.50	-

'-' TW was not considered in the input bias method of [13]

Both sign-extension avoidance techniques reduce the critical path delay and power consumption, particularly for long filters and filters with large coefficient WL. For Ma07, 'B' consumes slightly more area than 'S' because the saving in combinational area by sign-extension avoidance is not sufficient to offset the increase in the sizes of the registers and accumulators due to the additional bit needed after the first negative coefficient. PO outperforms the input bias as subtractions are also avoided and transitions due to the negation by full length adders have been reduced. The area reduction is more significant under TW as more registers and adders can be removed for long filters. For low order filters with short coefficient WL, like Xu07, the improvement due to PO is less. Due to the final constant subtraction, the critical path delay is higher. Owing to the longer sign-extension for filters with higher coefficient WL, like Ye04, the power reduced by PO is significant, even though the filter has only 40 taps. With TW, PO saves the most area, but at the expense of slight increase in power consumption because of the higher switching activity incurred by the removal of more excess offset. For the worst-case WL, the area, critical path delay and power consumption of FIR filters using PO representation are on average 7.9%, 10.8% and 18.9%, respectively, lower than those using 2's complement representation with sign extension.

V. CONCLUSION

Sign-extension problem associated with 2's complement variable in FIR filter design has been overcome by using PO representation. The ranges of various affine transformations arising from the partial product additions for the multiplication of a coefficient and a PO variable have been analyzed, from which excessive offsets are removed with small hardware premium to optimize the WLs of subsequent operations. Besides the reduction in area and critical path delay, our experimental results show that the proposed technique has also led to a discernible power reduction of about 19% on average.

REFERENCES

[1] K. H. Chen and T. D. Chiueh, "A low-power digit-based reconfigurable FIR filter," *IEEE Trans. Circuits Syst. II*, vol. 53, no. 8, Aug. 2006, pp. 617-621.

[2] Y. J. Yu and Y. C. Lim, "Design of linear phase FIR filters in subexpression space using mixed integer linear programming," *IEEE Trans. Circuits Syst. I*, vol. 54, no. 10, Oct. 2007, pp. 2330-2338.

[3] F. Xu, C. H. Chang and C. C. Jong, "Design of low-complexity FIR filters based on signed-powers-of-two coefficients with reusable common subexpressions," *IEEE Trans. on CAD*, vol. 26, no. 10, Oct. 2007, pp. 1898-1907.

[4] C. H. Chang, J. Chen and A. P. Vinod, "Information theoretic approach to complexity reduction of FIR filter design," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 8, Sep. 2008, pp. 2310-2321.

[5] F. Xu, C. H. Chang and C. C. Jong, "Contention resolution – a new approach to versatile subexpressions sharing in multiple constant multiplications," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 2, Mar. 2008, pp. 559 – 571.

[6] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Trans. Circuits Syst. II*, vol. 42, no. 9, Sep. 1995, pp. 569–577.

[7] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, no. 2, May 2007.

[8] L. Akosy, E. O. Gunes and P. Flores, "Search algorithms for the multiple constant multiplications problem: exact and approximate," *Microprocessors and Microsyst*., vol. 34, no. 5, Aug. 2010, pp. 151-162.

[9] O. Gustafsson, "A difference based adder graph heuristic for multiple constant multiplication problems," in *Proc. IEEE Int. Symp. Circuits and Syst.*, New Orleans, USA, May 2007, pp. 1097-1100.

[10] O. Salomon, J. M. Green, and H. Klar, "General algorithms for a simplified addition of 2's complement numbers," *IEEE J. Solid-State Circuits*, vol. 30, no. 7, Jul. 1995, pp. 839–844.

[11] S. Hwang, G. Han, S. Kang and J. Kim, "New distributed arithmetic algorithm for low-power FIR filter implementation," *IEEE Signal Process. Lett.*, vol. 11, no. 5, May. 2004, pp. 463-466.

[12] 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, Dec. 1991, pp. 1970–1980.

[13] O. Gustafsson, K. Johansson and L. S. DeBrunner, "Techniques for avoiding sign-extension in multiple constant multiplication", in *Proc. 43th Asilomar Conf. Signals, Syst. and Computers*, Pacific Grove, CA , Nov. 2009, pp. 740-743.

[14] C. Fang, R. A. Rutenbar, M. Püschel and T. Chen, "Toward efficient static analysis of finite-precision effects in DSP applications via affine arithmetic modeling," in *Proc. Design Automation Conf*, Anaheim, CA, Jun. 2003, pp. 496-501.

[15] K. S. Yeung and S. C. Chan, "The design and multiplier-less realization of software radio receivers with reduced system delay," *IEEE Trans. Circuits Syst. I*, vol. 51, no. 12, Dec. 2004, pp. 2444–2459.

[16] D. L. Maskell, "Design of efficient multiplierless FIR filters," *IET Circuits, Devices and Syst.*, vol. 1, no. 2, Apr. 2007, pp. 175–180.

[17] L. Aksoy, E. O. Günes, E. Costa, P. Flores, and J. Monteiro, "Effect of number representation on the achievable minimum number of operations in multiple constant multiplications," in *Proc. IEEE Workshop on Signal Processing Syst*., 2007, Shanghai, China, Oct. 2007, pp. 424–429.