On-Chip Nonlinear Digital Compensation for RF Receiver

Karen M. G. V. Gettings, Andrew K. Bolstad, Show-Yah Stuart Chen, Benjamin A. Miller, Michael Vai
Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA 02420
{karen.gettings, andrew.bolstad, stuart.chen, bamiller, mvai}@ll.mit.edu

Introduction

A system-on-chip (SOC) implementation is an attractive solution for size, weight and power (SWaP) restricted applications, such as mobile devices and UAVs. This is partly because the individual parts of the system can be designed for a specific application rather than for a broad range of them, like commercial parts usually must be. Co-design of the analog hardware and digital processing further enhances the benefits of SOC implementations by allowing, for example, nonlinear digital equalization to further enhance the dynamic range of a given front-end component.

This paper presents the implementation of nonlinear digital compensation for an active anti-aliasing filter [1], which is part of a low-power homodyne receiver design. The RF front-end circuitry and the digital compensation will be integrated in the same chip. Co-design allows the front-end to be designed with known dynamic range limitations that will later be compensated by nonlinear equalization. It also allows nonlinear digital compensation architectures matched to specific circuits and dynamic range requirements—while still maintaining some flexibility to deal with process variation—as opposed to higher power general purpose designs.

Nonlinear Signal Processing Overview

To compensate for the nonlinear behavior of the RF circuitry, we derive a nonlinear inverse function of the system response. Memory effects (i.e., state-dependent behavior) of RF devices often complicate the modeling process. A general nonlinear finite impulse response (FIR) model for systems with significant memory is the Volterra series [2], whose output is given by

$$y_{NL}(n) = \sum_{p=0}^{P} \sum_{m_1=0}^{M} \cdots \sum_{m_p=0}^{M} h_p(m_1, \cdots, m_p) \prod_{\ell=1}^{p} x(n - m_\ell).$$

This model generalizes the linear FIR filter to higher dimensions.

While this representation captures general nonlinear behavior, its complexity grows combinatorially with memory depth (M), and systems requiring real-time performance will typically use a "pruned" version of the kernel. The approach we take is similar to that of [3], in which the full coefficient space is divided into subspaces, only a few of which are selected for use in the compensator. A version of this architecture was recently developed in an ASIC that operates on data sampled up to 4 GSPS [4].

This work is sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

We further constrained the architecture to conserve power. Rather than operate over the entire Volterra kernel coefficient space as in [3], we restrict the coefficients to those used in the generalized memory polynomial (GMP) architecture of [5]. In this model the nonlinear output (neglecting the constant h_0 term) is given by

$$y(n) = \sum_{p=1}^{P} \sum_{m_1=0}^{M_1} \sum_{m_2=0}^{M_2} h_p(m_1, m_2) x(n - m_1) x^{p-1}(n - m_1 - m_2),$$

i.e., it is restricted to the coefficients lying on a 2-dimensional plane within the larger coefficient space. This decision costs us flexibility—we can no longer choose coefficients from arbitrary portions of the space—but allows a simple, power-efficient implementation. We use an optimization procedure to choose a fixed number of coefficients from arbitrary sections of the GMP coefficient space. Using architectures that include several contiguous coefficients as in [3] can be more power-efficient per coefficient in a larger system, but we have empirically observed that allowing the procedure to choose individual coefficients allows a greater initial dynamic range improvement with few coefficients. This constrained implementation is, despite architecture restrictions, sufficient to compensate the RF front-end that is part of this system on chip.

Implementation

The front-end specific implementation was designed in VHDL to be subsequently synthesized into a standard cell design, using IBM 65 nm CMOS technology and focusing on a low power implementation. The VHDL description was written with portability in mind, which allows easy conversion to other technologies or needs.

A block diagram of the equalizer is presented in Figure 1. It consists of the following building blocks: Two's Complement Conversion, Global Exponentiation, Processing Element (PE), Shifter and Final Accumulator.

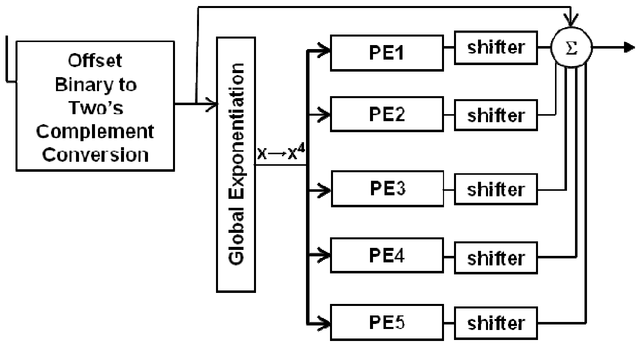

Figure 1: Block diagram of equalizer

Two's Complement Conversion

Our algorithm works on two's complement format. The Two's Complement Conversion block changes the front-end receiver's data format to two's complement if needed.

Global Exponentiation

In order to save power, the input signal from the front-end receiver is truncated to the most significant 8 bits. These 8 bits are subsequently exponentiated to powers ranging from 2 to 4, hence signals $x(n)$ to $x^4(n)$ are available for processing. This centralized multiplication provides large power savings, because instead of creating custom exponentiated signals per Processing Element (PE), these signals are created only once, and the output can feed all of the processing elements, with no additional exponentiation needed. Each PE can select which order it will use for processing via a multiplexer. This order is determined during training, and it is fed to each PE as a control signal.

Processing Element

The processing element is the main building block of the equalization circuitry. Its architecture includes a multiplexer, 2 delay blocks, and 2 multipliers, as depicted in Figure 2.

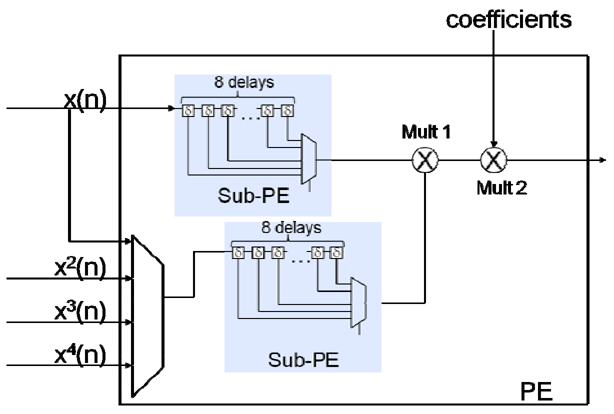

Figure 2: Processing element block diagram

The top delay block (labeled as Sub-PE in Figure 2) takes in the top (MSB) 8 bits of the input signal from the front end receiver. The bottom delay block takes in either the same top 8-bit input signal or one of its exponentiated forms, from $x(n)$ to $x^4(n)$. The signals are delayed in each delay block independently, and then multiplied. The delay required is determined during training, and is fed to each Sub-PE as a control signal. For power savings only the top 8 bits of the product of the 2 Sub-PE outputs are kept, and it is subsequently multiplied with the coefficient calculated for that specific PE during training.

Each PE implements a single term in the GMP series rather than a contiguous block, as described in the "Nonlinear Signal Processing Overview." Simulation showed that 8 delays per delay block and 5 PEs were sufficient to compensate our front-end circuitry while maintaining our low power requirement.

Shifter

The 8-bit output of each PE is sign-extended and shifted (multiplied) before being added to the uncompensated 16-bit input signal $x(n)$. Shifting allows a greater dynamic range of the compensation term for more accurate linearization.

Final Accumulator

After the output of each PE has been shifted to the appropriate position in a 16-bit word, these values are added to the input signal in the Final Accumulator. This can be viewed as subtracting nonlinear effects from $x(n)$, where subtraction is implemented by negating coefficients.

Results

We excited the active filter from [1] with 184 two-tone signals at various frequencies and tone spacings. Of these signals, 60 were used to train a compensator with the architecture described in the previous section, choosing the delays and computing the coefficients. Figure 3 demonstrates the increase in dynamic range when the compensator is applied (in Matlab) to the remaining 124 signals. In over 90% of the signals, the largest distortion is less than -85 dBFS, which is true for less than 17% when no compensation is used. We performed a bit-true fixed point simulation, which, as shown in the figure, attains nearly the same performance as floating point.

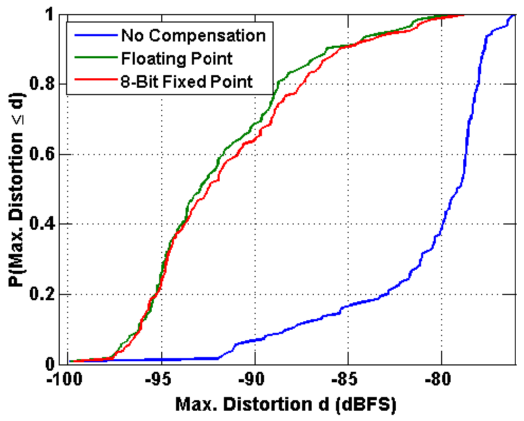

Figure 3: CDF for distortion level in an active filter, and improvement using nonlinear digital compensation

Acknowledgements

The authors wish to thank Helen Kim, Merlin Green, Daniel Santiago and Joel Goodman for their technical contributions to this work.

References

[1] H. H. Kim, M. Green, B. A. Miller, A. Bolstad, D. D. Santiago, "An active filter achieving 43.6dBm OIP$_3$," in *Proc IEEE RFIC Symp.*, 2011. To appear

[2] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, John Wiley & Sons, 2000.

[3] M. Herman, B. Miller and J. Goodman, "Efficient Multidimensional Polynomial Filtering for Nonlinear Digital Predistortion," in *Proc. Twelfth Annual HPEC Workshop*, 2008.

[4] W. S. Song et al., "Nonlinear Equalization Processor IC for Wideband Receivers and Sensors," in *Proc. Thirteenth Annual HPEC Workshop*, 2009.

[5] D. R. Morgan, Z. Ma, J. Kim, M. G. Zierdt, and J. Pastalan, "A generalized memory polynomial model for digital predistortion of RF power amplifiers," *IEEE Trans. Signal Process.*, Vol. 54, No. 10, 2006.