Research Repository UCD

Title	Low-Cost FPGA Implementation of Volterra Series-Based Digital Predistorter for RF Power Amplifiers
Authors(s)	Guan, Lei, Zhu, Anding
Publication date	2010-04
Publication information	Guan, Lei, and Anding Zhu. "Low-Cost FPGA Implementation of Volterra Series-Based Digital Predistorter for RF Power Amplifiers." IEEE, April 2010. https://doi.org/10.1109/TMTT.2010.2041588.
Publisher	IEEE
Item record/more information	http://hdl.handle.net/10197/8645
Publisher's statement	© 2010 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Publisher's version (DOI)	10.1109/TMTT.2010.2041588

Downloaded 2026-05-01 23:36:32

The UCD community has made this article openly available. Please share how this access benefits you. Your story matters! (@ucd_oa)

© Some rights reserved. For more information

Low-Cost FPGA Implementation of Volterra Series-Based Digital Predistorter for RF Power Amplifiers

Lei Guan, *Student Member*, *IEEE*, Anding Zhu, *Member*, *IEEE*

***Abstract*— This paper presents a low-complexity and low-cost hardware implementation of a Volterra series-based digital predistorter. This is achieved by introducing two novel model complexity reduction techniques into the system, namely, look-up table assisted gain indexing and time-division multiplexing for multiplier sharing. Experimental results show that this novel DPD implementation uses much less hardware resources but still maintains excellent performance, compared to conventional approaches.**

***Index Terms*— linearization, look-up table, power amplifier (PA), predistorter, Volterra series**

I. Introduction

WITH the goal of simultaneously achieving high efficiency and high linearity, digital predistortion (DPD) techniques along with methods for memory effect compensation have been proposed to linearize radio frequency (RF) power amplifiers (PAs). Examples include techniques based on memory polynomials [1][2], Hammerstein and Wiener models [3], augmented Wiener model [4], nonlinear auto-regressive moving average (NARMA) model [5], etc. The principle of DPD is simple: a nonlinear distortion function is built up within the digital domain that is the inverse of the distortion function exhibited by the amplifier. A highly linear and low distortion system can be achieved in principle when these two nonlinear systems, the DPD and the PA, are combined in series. An attraction of this approach is that the nonlinear PA can be linearized by a standalone add-on block, freeing vendors from the burden and complexity of manufacturing complex analog/RF circuits [6].

Previously, an efficient digital predistorter for RF PAs was proposed in [7]. This DPD approach is derived from the *dynamic deviation reduction*-based Volterra series [8], which allows simultaneous compensation for nonlinear distortion and memory effects with a small number of parameters. Based on the *p*th-order post-inverse theory, the parameters of this digital predistorter can be directly estimated from the measured input and output of the PA with a simple off-line characterization process. This eliminates the requirement for real-time closed-loop adaptation, and removes the necessity to implement the parameter estimation algorithms in real digital circuits, which significantly reduces system complexity and implementation cost. Experimental results showed that this truncated Volterra DPD model can achieve superb performance in linearizing various types of RF PAs.

However, only the model structure and the system characterization methodology were introduced in [7], The DPD algorithm was just implemented in a software environment, e.g., MATLAB, and important issues related to practical hardware implementation were not discussed. Although it may look straightforward, different methodologies or implementation structures will lead to very different results in terms of complexity and cost from the viewpoint of hardware implementation. In this paper, we present a low-complexity and low-cost solution for hardware implementation of this advanced DPD technique using real digital circuits. This is achieved by introducing two novel model complexity reduction techniques in the system, namely, look-up table (LUT) assisted gain indexing and time-division multiplexing for multiplier sharing.

Since the field programmable gate-array (FPGA) has many advantages in digital signal processing, including high density integration, parallel operation mechanisms, high speed processing, and flexible implementation, the FPGA has become one of main choices for implementing baseband digital predistorters for RF power amplifiers. In this work, we illustrate how to implement the proposed Volterra DPD model on FPGA chips, but the proposed technique is not solely limited to FPGAs. The same structure can also be readily implemented in other types of digital circuits, e.g., general digital signal processing (DSP) chips or application-specific integrated circuits (ASICs).

The paper is organized as follows. In Section II, we briefly reintroduce the proposed Volterra DPD system. Section III presents details of hardware implementation of the DPD. The experimental results are given in Section IV, with a conclusion in Section V.

This work was supported by the Science Foundation Ireland under the Principal Investigator Award Scheme.

The authors are with the School of Electrical, Electronic and Mechanical Engineering, University College Dublin, Dublin 4, Ireland (e-mail: lei.guan@ucd.ie; anding.zhu@ucd.ie).

II. DPD ALGORITHM

With the rapid development in digital signal processing techniques, most predistorters today are implemented in the digital baseband. In these systems, the original baseband signals are first predistorted by a digital predistortion unit and then passed through a digital-to-analog converter (DAC), up-converted to RF frequency and finally passed to the PA. In order to extract and update the coefficients of the DPD, a small fraction of the transmit signal is fed back and transferred to baseband via a down-converter and an analog-to-digital converter (ADC). The model parameter extraction unit compares the input and the output captured data, and extracts the coefficients for the DPD. This model extraction unit is normally only operational during the initial setup of the system or whenever the characteristics of the system are changed. The block diagram of a baseband digital predistortion system is illustrated in Fig. 1.

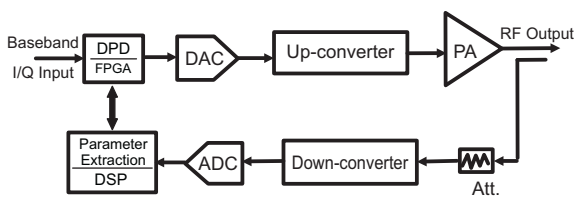

Fig. 1 Block diagram of digital predistortion

In [7], the *dynamic deviation reduction*-based Volterra model is used for constructing the DPD function. For instance, the 1st-order dynamic truncation of the Volterra model can be written as

$$\begin{aligned}\tilde{u}(n) = &\sum_{k=0}^{\frac{P-1}{2}}\sum_{i=0}^{M}\tilde{g}_{2k+1,1}(i)\left|\tilde{x}(n)\right|^{2k}\tilde{x}(n-i)\\ &+\sum_{k=1}^{\frac{P-1}{2}}\sum_{i=1}^{M}\tilde{g}_{2k+1,2}(i)\left|\tilde{x}(n)\right|^{2(k-1)}\tilde{x}^2(n)\tilde{x}^*(n-i)\end{aligned} \tag{1}$$

where $\tilde{x}(n)$ and $\tilde{u}(n)$ are the input and the output I/Q envelopes, respectively; $\tilde{g}_{2k+1,j}(\cdot)$ is the complex Volterra kernel of the DPD, $(\cdot)^*$ represents the complex conjugate operation and $|\cdot|$ returns the magnitude. P is the order of nonlinearity and M represents the memory length. Only odd order nonlinearities are included in this baseband representation in (1), i.e., P is an odd number, since the effects from even order kernels can be omitted in a band-limited modulation system.

The implementation of this DPD system mainly includes two parts: (i) the DPD unit, which predistorts the input I/Q signal; and (ii) the parameter extraction unit, which is used to initialize and update the coefficients. In [7], by employing the pth-order post-inverse theory [9], the coefficients of the DPD can be estimated from the measured input and output of the PA with an off-line characterization process. The model extraction unit can thus be implemented in a software environment, which can be embedded in general digital signal processing circuits in the system. Although both units, the DPD and the parameter extraction, must be implemented efficiently in the system, the most critical part centers on implementing DPD unit itself. Because real-time on-line signal processing is required, the DPD unit must be implemented in high speed digital hardware circuit, e.g., FPGAs or ASICs. As price pressures become ever more demanding, in order to make it feasible and competitive in the market, the proposed DPD structure must be implemented in transmitters with very low cost. However, direct implementation of the DPD function, i.e., the equation (1), requires a large number of multipliers and adders to construct the nonlinear and memory terms. For example, 60 multipliers will be needed when P=5, M=3. This number increases significantly when P and M become larger. Because the multiplier is one of the most complex and expensive components in FPGA hardware [10], the large number of multipliers required dramatically increases system complexity and the final cost of the DPD system.

III. NEW DPD IMPLEMENTATION

In this paper, we seek an alternative and efficient way to implement this Volterra DPD, as discussed in the following.

A. LUT-Assisted Gain Indexing

From (1), we can see that the number of multipliers required mainly results from the generation of the power and the memory terms, and their multiplications with the coefficients. However, if we re-write (1) as

$$\begin{aligned}\tilde{u}(n) &= \tilde{u}_1(n)+\tilde{u}_2(n)\\ &= \sum_{i=0}^{M}C_{i,1}(n)\tilde{x}(n-i)+\sum_{i=1}^{M}C_{i,2}(n)\tilde{x}^2(n)\tilde{x}^*(n-i)\end{aligned} \tag{2}$$

where

$$C_{i,1}(n)=\sum_{k=0}^{\frac{P-1}{2}}g_{2k+1,1}(i)\left|\tilde{x}(n)\right|^{2k} \tag{3}$$

$$C_{i,2}(n)=\sum_{k=1}^{\frac{P-1}{2}}g_{2k+1,2}(i)\left|\tilde{x}(n)\right|^{2(k-1)} \tag{4}$$

we can find that, for a given memory tap i, $C_{i,1}$(n) and $C_{i,2}$(n) can be considered as a constant complex gain to $\tilde{x}(n-i)$ or $\tilde{x}^*(n-i)$, respectively, while $C_{i,1}$(n) or $C_{i,2}$(n) depends on polynomial power terms of $|\tilde{x}(n)|$ only, since $\tilde{g}_{2k+1,1}(i)$ or $\tilde{g}_{2k+1,2}(i)$ is a constant for each power term after model

extraction or in time-invariant operations, when i is fixed. Furthermore, in a real system, even if the individual samples of the signal, $\tilde{x}(n)$ and $\tilde{x}(n-i)$ are randomly generated, the range of the input signal is normally known in advance. This means that it is possible to calculate the high order power terms of $|\tilde{x}(n)|$ and thus the values of $C_{i,1}$(n) and $C_{i,2}$(n) in advance, and store them in a Table and retrieve them later. For example, if the maximum value of the input magnitude $|\tilde{x}(n)|$ is normalized to unity and the LUT size is 1024, the interval value for the LUT, Δ, will be 1/1024. A set of values of $C_{i,1}$(n) and $C_{i,2}$(n) for each interval Δ can be calculated in advance. For example, for i=2, $C_{2,1}$(n), can be obtained by adding all power terms and multiplying with the coefficients together for each given interval values, as shown in Fig. 2. In the same way, similar tables can be built for different memory taps for $C_{i,1}$ and $C_{i,2}$, where i=0, 1, …, M. These values can then be retrieved directly from the tables when a specific input value of $|\tilde{x}(n)|$ is given. Using this approach, only 2M+1 LUTs are required for the whole system, and this number no longer increases with the nonlinear order P since all higher order terms can be built in one table, no matter how many orders there are.

One may note that from (2), for $\tilde{u}_1(n)$, only the memory terms are separated out and all other terms are built into the LUT; while for $\tilde{u}_2(n)$, $\tilde{x}^2(n)$ is also left outside the Table along with the conjugated memory terms. This is because the input $\tilde{x}(n)$ is a complex number including In-phase (I) and Quadrature (Q) values, but the LUT address is generated from the magnitude only, with the same magnitude value, the power term $\tilde{x}^2(n)$ can have different values since $\tilde{x}(n)$ may have different phases for each sample, which leads that one address can correspond to multiple output values. To avoid this problem, $\tilde{x}^2(n)$ must be left outside the LUT for $\tilde{u}_2(n)$, although this requires two more multipliers to implement the system.

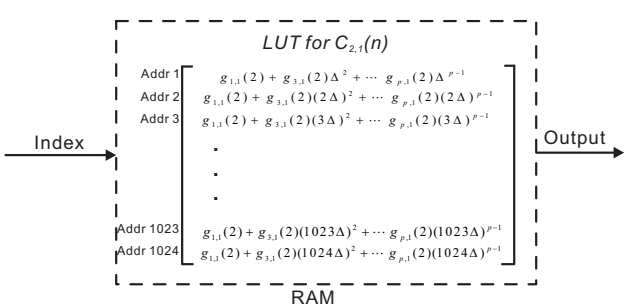

Fig.2 An example of LUT contents

This LUT mapping approach is more efficient than a direct polynomial-based implementation of equation (1), since it no longer requires multiplication between input samples to produce the power terms. This structure also eliminates the requirement for building separate Tables for each nonlinear power terms as is the case in traditional LUTs [11-13]. Consequently, the hardware resource requirement for the DPD is significantly reduced.

Furthermore, unlike most conventional LUT-based DPDs where the gain values are often obtained directly through curve fitting between input and output data, which often leads to low accuracy, in this approach, an analytical function-based model, i.e., the truncated Volterra series, is employed to represent the characteristics of the DPD and a Least Squares algorithm is utilized to extract the coefficients. This allows us to accurately control the nonlinear order and the memory length to be compensated for, and guarantees high accuracy in the model extraction. Although during the calculation of $C_{i,1}$ and $C_{i,2}$, high precision signal processing is required to produce accurate results, e.g., producing high order power terms, this does not affect the final LUT structure, because the intermediate values are calculated offline in a software environment and they are not stored in the LUTs. Only the final results of $C_{i,1}$ and $C_{i,2}$, are stored in the LUTs. The precision requirement for these final values are much lower than that required for the intermediate terms, and therefore, they can be easily scaled to suit a small LUT size, significantly saving memory storage in the DPD chip. In summary, this proposed approach takes advantages of the best features of both analytical function-based and LUT-based solutions: (i) high accuracy in model extraction is maintained by using an analytical function-based DPD model; while (ii) complexity and cost in hardware implementation is reduced by the LUT.

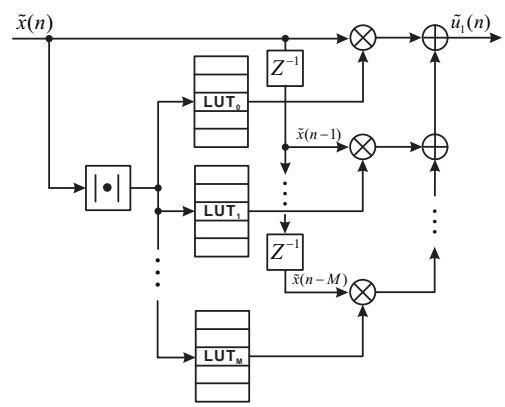

Fig.3 DPD implementation example

The next step of the implementation is to include the memory terms, i.e., $\tilde{x}(n-i)$, $\tilde{x}^*(n-i)$, and the terms left outside the LUTs, $\tilde{x}^2(n)$. One solution is to multiply the values retrieved from the LUTs with these terms by using multipliers. For example, an implementation of the first part of the output, $\tilde{u}_1(n)$, is shown in Fig. 3. This is similar to the "Basic Predistortion Cell" structure proposed in [14], but only one address index is needed to retrieve the values from LUTs in this case, because the power terms in all LUTs are the same, in other words, they all only depend on the

instantaneous samples $\tilde{x}(n)$. Multiple CORDICs (COordinate Rotation DIgital Computers) [15] are therefore avoided, which can further save hardware resources.

B. Time-Division Multiplexing for Multiplier Sharing

Compared to direct analytical function based DPDs or conventional LUT approaches, the complexity and cost of the implementation shown in Fig. 3 have been dramatically reduced. However, this structure is not the simplest one because a large number of multipliers are still involved, and this number will increase with the memory length. To further reduce the complexity, time-division multiplexing can be employed as described below.

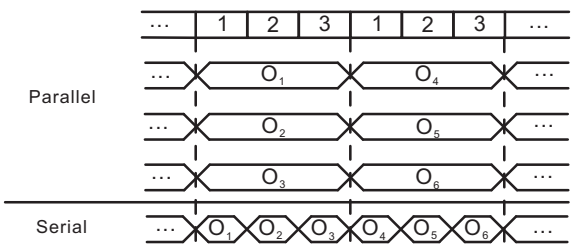

Fig. 4 Time sequence view of multiplication operations

In Fig. 3, multipliers are operated in parallel and each operation occupies a whole time slot. However, if we convert the parallel operation to a serial operation, for example, by using a multiplexer, all multiplication operations can be conducted in a serial way with one multiplier. The time sequence of these operations is shown in Fig. 4.

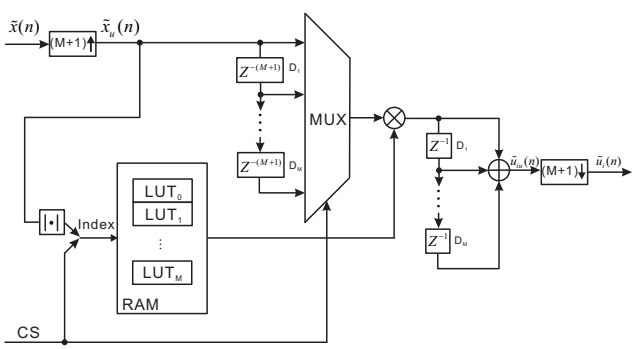

Fig. 5 DPD implementation using multiplexing

Transforming from parallel to serial processes, the time duration for each operation becomes shorter, which requires that the original data must be up-sampled and the signal processing needs to be conducted in a higher speed. The block diagram of this implementation of $\tilde{u}_1(n)$ is shown in Fig. 5, where the original data are up-sampled first and a multiplexer is then used to redistribute the data for conducting the multiplication operations serially. At the end of operations, the output data are recombined and down-sampled back to the original sampling rate to produce the final output. The details of each composite block are given below:

1) *Input Up-sampling*: this block increases the operation frequency by a factor of (M+1) with data repeating units, i.e., the input data are repeated (M+1) times at a higher sampling rate.

2) *Input Magnitude Calculation*: The CORDIC algorithm [15] is used to calculate the magnitude of input complex I/Q data, and this is used in conjunction with a selection signal CS as the address for the LUTs.

3) *LUT Mapping*: Separated LUTs are mapped into different parts of a standalone RAM (random access memory) on board so that indexing the contents is easier; the LUTs parts are indexed with the magnitude of respective input signals and the multiplexed CS signal. This saves the hardware cost for implementing addressing buses. Each LUT contains two parts, one for storing the real part of the coefficients and the other for the imaginary part.

4) *Data Multiplexing*: This part contains a multiplexer with a selection signal CS and M delay-units, and each delay-unit delays M+1 time cycles, This structure converts parallel multiplications using multiple multipliers to serial operations sharing only one multiplier.

5) *Complex Multiplication*: The symbol $\otimes$ in Fig. 5 represents the complex multiplication (CM), which is used to conduct multiplication between the output of MUX and the value retrieved from LUTs.

6) *Output Accumulation*: This block is used to implement the accumulation (ACC) operation which is used to compensate for the memory effects distortion.

7) *Output Down-sampling*: Not all the results obtained from the accumulator are useful for producing the final output of the DPD; some values are just the intermediate processing results which can be discarded. The output is only obtained from every M+1 time cycles in the serial data sequences. Thus, a down-sampling block is employed to conduct such "selection".

The second part of the DPD, $\tilde{u}_2(n)$, can be implemented using the similar structure as in Fig. 5, and with another two complex multipliers to include the $\tilde{x}^2(n)$ term. Combining the two parts, $\tilde{u}_1(n)$ and $\tilde{u}_2(n)$, together with proper delays to compensate processing delays between two branches, we can form the final structure of the DPD, as shown in Fig. 6. In this structure, only four complex multipliers are required in total, and the number of multipliers used does not increase with the memory length M or the nonlinearity order P. This significantly reduces the cost of the system, especially for the systems with long term memory effects or high order nonlinearities.

Furthermore, in [7], in order to further reduce the system complexity, under-sampling and interpolation were used. The original input and output signals are sampled with a low sampling rate (lower than the Nyquist rate) at the model extraction stage, while the input signal is then up-sampled and the parameters are interpolated by inserting zeros to avoid aliasing effects at the implementation stage. As such case, the

DPD function (1) becomes

$$\bar{u}(n)=\sum_{k=0}^{\frac{P-1}{2}}\sum_{i=0}^{M}\tilde{g}_{2k+1,1}(i)\left|\bar{x}(n)\right|^{2k}\bar{x}(n-i\times N)+\sum_{k=1}^{\frac{P-1}{2}}\sum_{i=1}^{M}\tilde{g}_{2k+1,2}(i)\left|\bar{x}(n)\right|^{2(k-1)}\tilde{x}^{2}(n)\bar{x}^{*}(n-i\times N) \quad (5)$$

where $\bar{x}(n)$ is the over-sampled input, and $\bar{u}(n)$ the new output of the DPD. N is the ratio of the sampling rate required for DPD output signal reconstruction versus that of the under-sampled model extraction data. Equation (5) can be easily implemented with the same structure as Fig. 6, where only the duration of the time delay in each delay unit of the multiplexers needs to be changed from M+1 to (M+1)×N. This proposed structure can also be easily adopted to implement the memory polynomial or other truncated Volterra series-based DPD models, although that is outside the scope of this paper.

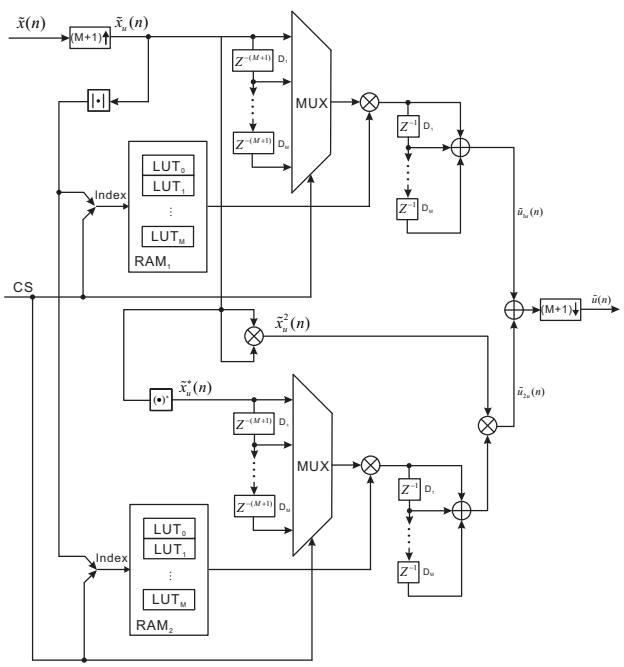

Fig. 6 Proposed DPD implementation

IV. Results

In order to validate the proposed DPD implementation, an experimental test bench was set up, in which the RF chain is the same as that presented in [7], where the baseband I/Q signal is first modulated to a digital intermediate frequency (IF) and then converted to the analog domain and up-converted to the RF band, and finally sent to the power amplifier. To capture the output data, the output of the PA was first down-converted to the IF band, and then converted to the digital domain and finally demodulated to the baseband. The core digital predistortion function was implemented on a Xilinx Virtex-4 XC4VSX35 FPGA chip and the model parameter extraction was conducted in MATLAB.

A. System Performance Evaluation

The performance of this truncated Volterra series-based DPD model has already been tested and presented in [7]. To avoid repetition, in this paper, we only focus on evaluating the accuracy of the hardware-implemented DPD compared to the software-implemented one in [7], that is, comparing the output predistorted by the hardware DPD with the one produced by the software DPD when given with the same input. To facilitate this comparison, the same power amplifier and signal source were used as in [7], where a high power Gallium Nitride (GaN) amplifier was operated at 1.94 GHz, and excited by a WCDMA signal with a chip rate at 3.84 MHz and with 9.57 dB PAPR. The WCDMA I/Q samples were recorded with a low sampling rate at 15.36 MHz for parameter extraction and the input signal was then over-sampled by a factor of four to a sampling rate at 61.44 MHz in the final system to avoid aliasing. Equation (5) with N=4 was implemented on the FPGA board.

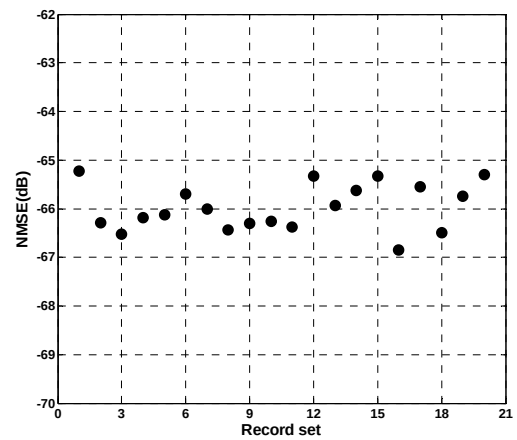

Fig. 7 Sample waveform errors in the time domain

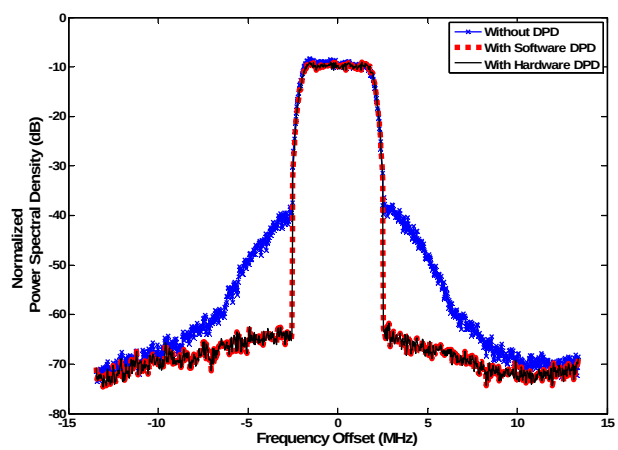

Fig. 8 Spectra plots of the system with and without DPD

The nonlinearity order P was chosen as 7 and memory length M=4. There were 32 parameters in total. It required four complex multipliers (one for $\tilde{u}_1(n)$ and three for $\tilde{u}_2(n)$) and nine LUTs (five for $C_{i,1}$ and four for $C_{i,2}$) to implement the

DPD function of (5). The size of the LUTs was 1024, and each LUT had 32-bit storage capacity, in which low 16-bit was for the real part values and high 16-bit for the imaginary part. The import and export dual data port RAMs were 32-bit in width and 8640 in size.For system validation, the data processed by FPGA was captured from the hardware platform and compared with the results produced in the ideal software environment MATLAB. 2000 samples per set and 20 sets of independent data samples were recorded and compared. The normalized mean square errors (NMSEs) are depicted in Fig. 7, where we can see that the average error is around -66dB. This indicates that the FPGA hardware implementation produced a very high precision for the signals. We then passed the predistorted signal through the whole RF chain. As expected, with the DPD, the nonlinear distortion was almost completely removed. The ACPRs (Adjacent Channel Power Ratios) in both the first and the second adjacent channels were reduced to under -60 dBc and the NRMSE (normalized root mean square error) was reduced to 1.8%. The output spectrum of the PA was almost identical with that produced by the software-implemented DPD, as shown in Fig. 8. Tests with other PAs and signals also produced similar results as presented in [7].

B. *Comparison of Different Implementations*

For comparison, two other DPD structures were also implemented: (i) Scheme 1 (S1) is the direct structure, in which the DPD function is directly implemented by using the multipliers and adders; (ii) Scheme 2 (S2) is the structure as shown in Fig. 3, in which the coefficients and nonlinear power terms are built into LUTs while the memory terms are implemented by using multipliers. The proposed structure in which the coefficients and nonlinear power terms are built into LUTs while the memory terms are implemented by using shared multipliers with time-division multiplexing, is referred to as Scheme 3 (S3).

The hardware utilization for different nonlinear orders and memory lengths for three schemes above are shown in Table I, where $\otimes$, $\oplus$ and LUT represents complex multipliers, two-input adders and look-up-tables, respectively. It can be clearly seen that the cost of Scheme 1 is highest and its complexity significantly increases with the nonlinear order or the memory length. The number of multipliers is reduced in Scheme 2, but it increases with the memory length. Scheme 3 uses much fewer resources and the number of multipliers is the same for all systems with different nonlinear orders and memory lengths, which provides significant benefit when the nonlinear order and the memory length increase.

The resource utilization difference between three schemes becomes more evident when implementing this DPD on an FPGA board. Table II shows the Xilinx FPGA resource utilization in implementing the three schemes for the nonlinear order P=7 and memory length M=4. It can be seen that the proposed scheme use much fewer resources on the board, especially the DSP48 slices, which are dedicated high performance signal processing blocks.

TABLE I: HARDWARE UTILIZATION COMPARISON

	P=5,M=2			P=7,M=2			P=7,M=4		
	$\otimes$	$\oplus$	LUT	$\otimes$	$\oplus$	LUT	$\otimes$	$\oplus$	LUT
S1	27	12	0	39	17	0	65	31	0
S2	8	4	5	8	4	5	14	8	9
S3	4	4	5	4	4	5	4	8	9

TABLE II: FPGA RESOURCE UTILIZATION COMPARISON

	DSP48	Slice Flip Flops	Slice	4-input LUTs	RAM 16s
S1	260	8132	5934	9512	17
S2	56	2027	1399	1388	18
S3	16	1282	1220	1418	18

By employing multiplexers and conducting serial operations, the system clock in the proposed scheme must be M times higher than those in the parallel processing, such as in Scheme 1 and 2. This does not create a problem for digital circuits since the clock rate of digital chips is continuously pushing higher, for example, 600 MHz signal processing capability is currently available in Xilinx Virtex-6 [10]. The price of such chips is continuously falling. However, although significant improvements have been made over last decades, the complex multipliers are still complicated to implement, requiring dedicated DSP slices and occupying a large area of silicon chip. The number of these special designed DSP units, e.g., DSP48, is limited on FPGA boards. Therefore, by employing the LUT and multiplexing combined approach, the implementation structure of the DPD is much simpler and the overall cost will be much lower compared to conventional approaches. The benefit of the proposed approach will become more evident when the FPGA implementation is transformed to stand-alone integrated circuits or ASICs for mass production.

V. Conclusion

A new technique for implementing Volterra series-based digital predistorters for RF power amplifiers has been proposed in this paper. By combining look-up-table and multiplication, together with time division multiplexing, the hardware resource required by the implementation is significantly reduced and therefore the cost of the system is dramatically lowered. This digital predistorter can be easily implemented on programmable FPGA boards or made as a stand-alone integrated circuit chip, and used for linearizing various types of RF power amplifiers. Because of the very simple structures involved, this digital IC can be made at very low cost in a single unit; however, by adding such a linearization module, the PA in the transmitter can be driven

to saturation to produce much high power, but without introducing severe distortion. Compared to conventional transmitters employing linear amplifiers, this saturated nonlinear PA plus digital predistorter structure can be much more power efficient and a great deal less expensive to implement, which is very attractive for future wireless communications.

REFERENCES

[1] J. Kim and K. Konstantinou, "Digital predistortion of wideband signals based on power amplifier model with memory," *Electron. Lett.*, vol. 37, no. 23, pp. 1417–1418, Nov. 2001.

[2] L. Ding, G. T. Zhou, D. R. Morgan, Z. Ma, J. S. Kenney, J. Kim, and C. R. Giardina, "A robust digital baseband predistorter constructed using memory polynomials," *IEEE Trans. Commun.*, vol. 52, no. 1, pp. 159–165, Jan. 2004.

[3] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, New York: Wiley, 2000.

[4] T. Liu, S. Boumaiza, and F. M. Ghannouchi, "Pre-compensation for the dynamic nonlinearity of wideband wireless transmitters using augmented Wiener predistorters," in *Proc. Asia-Pacific Microw. Conf.* Suzhou, China, Dec. 2005, vol.5, pp. 4-7.

[5] G. Montoro, P. L. Gilabert, E. Bertran, A. Cesari,and D. D. Silveira, "A new digital predictive predistorter for behavioral power amplifier linearization," *IEEE Microw. Wireless Compon. Lett.*, vol.17, no.6, pp.448-450, Jun. 2007.

[6] P. B. Kennington, *High Linearity RF Amplifier Design*. Norwood, MA: Artech House, 2000.

[7] A. Zhu, P. J. Draxler, J.J Yan, T. J. Brazil, D. F. Kinball, and P. M. Asbeck, "Open-loop digital predistorter for RF power amplifiers using dynamic deviation reduction-based Volterra series," *IEEE Trans. Microw. Theory Tech.*, vol. 56, no. 7, pp. 1524-1534, Jul. 2008.

[8] A. Zhu, J. C. Pedro, and T. J. Brazil, "Dynamic deviation reduction-based Volterra behavioral modeling of RF power amplifiers," *IEEE Trans. Microw. Theory Tech.*, vol. 54, no. 12, pp. 4323-4332, Dec. 2006.

[9] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, reprint ed. Malabar, FL: Krieger, 2006.

[10] http://www.xilinx.com.

[11] P. Jardin and G. Baudoin, "Filter lookup table method for power amplifier linearization," *IEEE Trans. Veh. Tech.*, vol. 56, no. 3, pp. 1076- 1087, May 2007.

[12] P. L. Gilabert, A. Cesari, G. Montoro, E. Bertran, and J.-M. Dilhac, "Multi-lookup table FPGA implementation of an adaptive digital predistorter for linearizing RF power amplifiers with memory effects," *IEEE Trans. Microw. Theory Tech.*, vol.56, no.2, pp.372-384, Feb. 2008.

[13] L. Rexberg, "Power Amplifier Pre-distortion," U.S. Patent 2006/0133536, June 2006.

[14] A. Cesari, P. L. Gilabert, "A FPGA based digital predistorter for RF amplifiers with memory effects," in *Proc. 2nd European. Microw. Conf. Integrated Circuits*, Munich Germany, Oct. 2007, pp. 135–138.

[15] Y. H. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, pp.17-34, Jul. 1992.

Anding Zhu (S'00-M'04) received the B.E. degree in telecommunication engineering from North China Electric Power University, Baoding, China, in 1997, and the M.E. degree in computer applications from Beijing University of Posts and Telecommunications, Beijing, China, in 2000, and the Ph.D. degree in electronic engineering from University College Dublin (UCD), Dublin, Ireland, in 2004.

He is currently a Lecturer with the School of Electrical, Electronic and Mechanical Engineering, UCD. His research interests include high-frequency nonlinear system modeling and device characterization techniques with a particular emphasis on Volterra-series-based behavioral modeling for RF PAs. He is also interested in wireless and RF system design, digital signal processing and nonlinear system identification algorithms.