Departament de Teoria
del Senyal i Comunicacions

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Ph.D. Thesis

# Multi Look-Up Table Digital Predistortion for RF Power Amplifier Linearization

Author: Pere Lluis Gilabert Pinal

Advisors: Dr. Eduard Bertran Albertí
Dr. Gabriel Montoro López

Control Monitoring and Communications Group
Department of Signal Theory and Communications
Universitat Politècnica de Catalunya

Barcelona, December 2007

# Chapter 5

# Adaptive Digital Baseband Predistortion Linearization

## 5.1 Introduction

As it has been discussed in previous Chapters, the use of linearizers permits finding a compromise between linearity and power efficiency. PA linearization allows the use of more efficient PAs despite the nonlinear distortion introduced by them, since linearization techniques are aimed at compensating this nonlinear distortion. Moreover, as reported in Chapter 2, since modern communication standards use high speed envelope signals presenting significant bandwidths, memory effects have to be taken into account. Among PA linearizers, digital predistortion takes advantage of the already existing digital signal processing (DSP) devices in transmitters (to cope with signal coding protection and modulation required in modern standards), thus reducing the RF hardware adjustment problems. In addition, as discussed in Chapter 3, DPD is a versatile linearization technique that takes advantage of the software defined radio (SDR) solutions, allowing reconfigurability and being more independent of a particular RF front-end.

Memory effects do not affect equally to all linearization techniques, for example, feedback or feedforward linearizers are less sensitive to PA behavior than DPD. Digital predistortion linearization is quite sensitive to memory effects, which can be a drawback when trying to cancel distortion in wideband signals, since its linearization performance is reduced. Thanks to the envelope filtering technique it is possible to reproduce the inverse memory effects that are generated inside the PA aiming at a later cancelation. For that reason, an overview on PA behavioral models capable of reproducing PA nonlinear behavior and memory effects has been previously presented in Chapter 4.

As discussed in Chapter 4, the nonlinear auto-regressive moving average (NARMA) model is capable of reproducing both PA nonlinear distortion and dynamics. Moreover, it has the advantage of introducing a nonlinear feedback path that may permit relaxing the complexity

of the PA model, in comparison to a model using only FIR terms. However an additional test on its stability is a mandatory previous step that has to be performed in order to prevent the whole system from instabilities.

The main contribution of this thesis, and the scope of the present Chapter, is to propose a *new digital baseband adaptive predistorter* based in a NARMA structure whose parameters can be easily obtained from a closely related NARMA model of the PA. This kind of model embeds both linear and nonlinear parts, thus avoiding a cascaded Linear – Nonlinear decomposition as in Hammerstein or Wiener models. In addition, the NARMA DPD is well suited for being implemented by using a set of simple look-up tables (LUTs). The adaptation process of the proposed DPD relies just on the PA NARMA model, and its stability, despite its nonlinear feedback structure, may be assessed and ensured.

In the following, the design of a digital baseband adaptive predistorter aiming at a final hardware implementation is presented. The experimental set-up for validating this new structure of DPD is described in Chapter 6, while in this Chapter some previous issues related to the identification and adaptation of the DPD are presented. Thus, this Chapter is organized as follows:

- First, a brief discussion on possible strategies for organizing and indexing look-up tables.
- Then, a detailed presentation of two identification approaches: Indirect learning and Predictive Predistortion.
- Finally the description of two possible configurations to perform the DPD adaptation process: External and Real time adaptation.
- Simulation results showing the linearization performance achieved with the proposed predictive NARMA based DPD will be provided for both adaptive configurations.

As an advance of the issues that will be later discussed in this chapter, Fig. 5.1 graphically presents a general block diagram of a digital baseband adaptive predistorter. Digital baseband predistortion is performed in a Field Programmable Gate Array (FPGA) platform by means of a set of LUTs. The adaptation process consists in periodically calculating new predistortion gain values that update LUT contents in the FPGA. This adaptation process can be carried out by an external digital processing device such a personal computer (PC) or a specifically programmed Digital Signal Processor (DSP); or alternatively, taking advantage of the parallel processing characteristics of the FPGA device, can be performed in the same FPGA using simple algorithms that allow real-time adaptation.

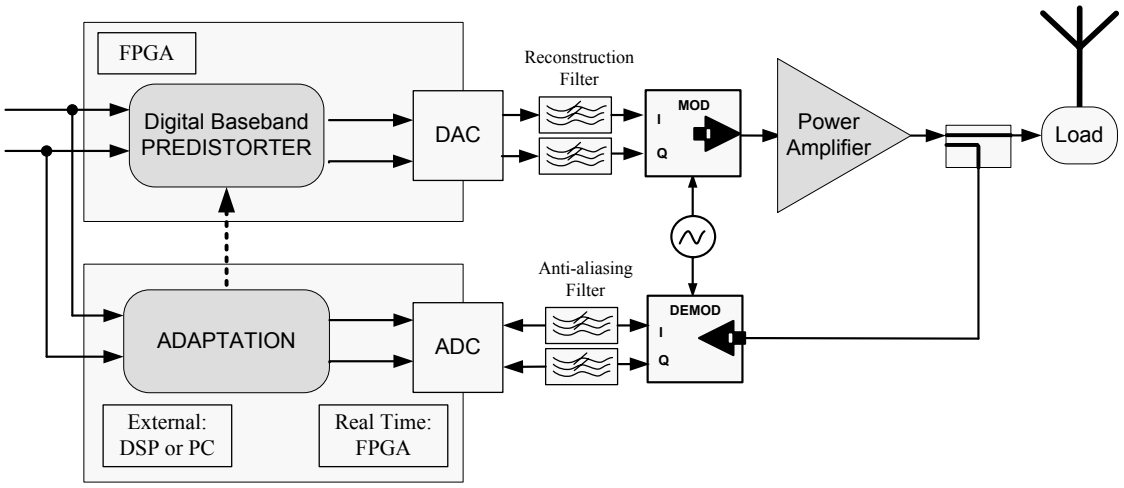


**Figure 5.1:** Simplified block diagram of a transmitter with digital baseband adaptive predistortion linearization.

## 5.2 Look-up Table Organization

Many years of research have dealt with predistortion techniques for memoryless PA. Recently, several solutions already include memory effects compensation, since those are of significant concern when considering high bandwidths with multilevel and multicarrier modulation formats. Digital predistortion solutions have to implement the predistortion function, usually based in a particular PA behavioral model, in a digital signal processor. An efficient way to implement the predistortion function without an excessive computational cost is by using *Look-up Tables* (LUTs) .

In order to map the predistortion function into a LUT, some considerations regarding LUT organization have to be taken into account, such as:

- The LUT architecture (one-dimensional or bidimensional LUTs) to deal with the discrete complex signal's envelope
- The optimum size of the LUT (trade-off between accuracy and memory size)
- The LUT indexing and spacing between entries within the LUT

Another issue that has to be taken into account despite it is not exclusively an issue related to LUT organization, is the complexity of the adaptation algorithm, since it is closely related to the frequency of the LUT updates that the DSP will allow.

In the following subsections, these issues regarding LUT organization are discussed more in deep.

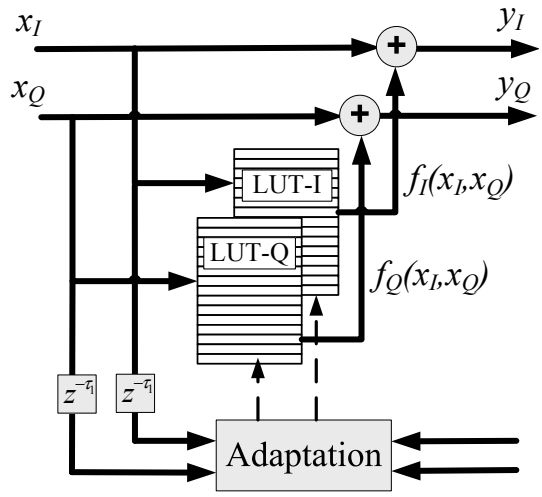


**Figure 5.2:** Mapping predistortion.

### 5.2.1 LUT Schemes

Digital baseband predistortion handles the complex envelope of the PA input and output RF signals. Therefore, the LUT architecture depends on the way this complex envelope is treated in order to be predistorted. LUT based predistorters may be classified by its LUT approach in [Sun95]:

- Mapping predistorters (2-Dimensional LUTs)
- Polar predistorters (1-Dimensional LUT)
- Complex gain based predistorters (1-Dimensional LUT)

In *Mapping Predistortion* the complex input signal is represented by its Cartesian in-phase ($x_I$) and phase-quadrature ($x_Q$) components. By using a two-dimensional LUT both input Cartesian components are mapped to a new constellation of Cartesian components: $y_I = x_I + f_I(x_I, x_Q)$ and $y_Q = x_Q + f_Q(x_I, x_Q)$. Fig. 5.2 shows the block scheme of a mapping predistorter architecture. Some results using mapping predistortion are reported in [Nag89, Min90, Man94]. The major drawback with the mapping predistorter is the size of the two-dimensional LUTs, which results in long adaptation times.

*Polar Predistortion* was presented by Faulkner et al. in [Fau94] and uses two one-dimensional LUTs containing magnitude gain and phase rotation, respectively. The principle is illustrated in Fig. 5.2. The input signal amplitude, $R_{in}$, is used to point an address of the LUT containing an amplitude gain factor $R_{out} = F_R(R_{in})$. Then, this factor is used to multiply the original input signal amplitude (amplitude correction). Moreover, this gain factor is also used to multiply the

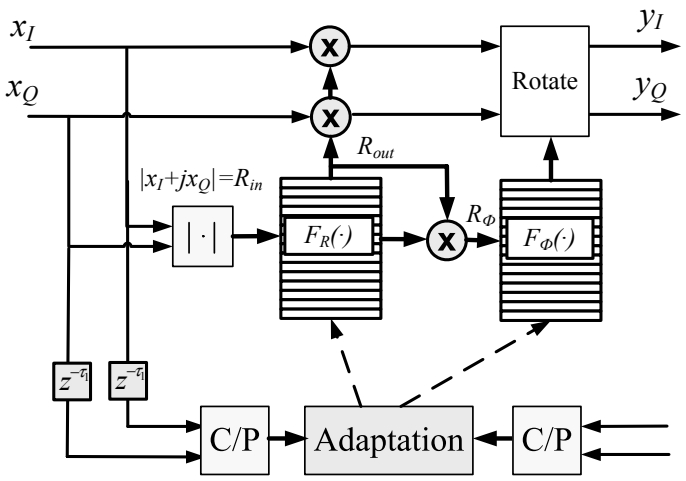


**Figure 5.3:** Polar predistortion.

input amplitude and then addressing a second table containing the predistortion phase, $\Phi_{out} = F_\Phi(R_\Phi)$. This phase is used to rotate the signal previously predistorted in amplitude (phase correction). Both tables are one-dimensional, so the access time and the memory requirements are reduced respect to the previous Mapping Predistortion.

*Complex Gain Predistortion* was presented by Cavers in [Cav90a, Cav90b] and later has been used by many other authors, such as in [Kim05, Hel06, Gil06a, Mon07]. Instead of having two tables with amplitude gain and phase rotation, this approach (see Fig. 5.3) has a LUT containing complex-valued gain factors given in Cartesian form. The complex gain predistorter uses the power of the input signal ($R = |x|^2$) to point a unique LUT containing the complex gains of a predistortion function, $G_{LUT}(R)$. The complex gain ($G_{LUT}(R)$) that results from the LUT addressing is used to predistort the input signal $x$ by computing the complex product $y = x \cdot G_{LUT}(R)$. With only one one-dimensional table the complex gain predistortion reduces the complexity and adaptation time in comparison to mapping predistortion.

### 5.2.2 LUT Size and Word Length

As we have seen before, digital predistortion schemes uses two-dimensional (2-D) or one-dimensional (1-D) LUTs. The 2-D table method (mapping predistortion) builds a table that is indexed by the I and Q inputs of the predistorter and stores the appropriate predistorter output. The disadvantage of 2D tables is their large memory requirement and, in adaptive systems, the large number of samples required before the table is full. The advantage of the 2-D table is that no polar-to Cartesian or Cartesian-to-polar conversions are necessary. Complex Gain or Polar methods uses two 1-D tables (amplitude-phase or I-Q gains) to correct amplitude and phase distortion. A disadvantage of the 1-D table is that it often requires conversions between Carte-

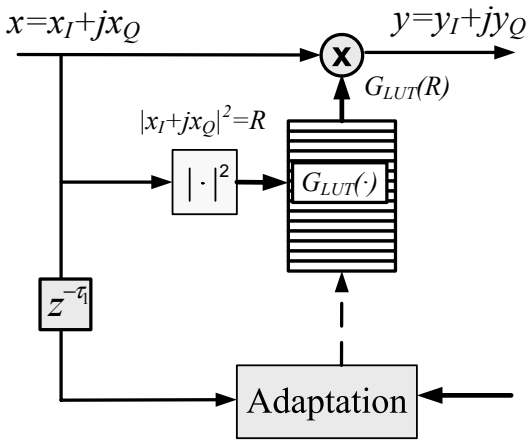


**Figure 5.4:** Complex Gain Predistortion.

sian and polar representations. Another drawback is that only phase-invariant nonlinearities can be corrected without adding additional LUTs in parallel.

If the number of bits used to quantize the signal is represented by n:

- a full two dimensional table memory requires $(2n)^2$ entries, while
- a one dimensional table requires two tables of 2n values.

The *table size* has an inverse relationship with the adjacent channel interference. Each doubling of the table size decreases the ACPR by 6 dB up to a limit after which increasing the table size no longer reduces the adjacent channel interference, and may indeed degrade performances [Shi03]. Furthermore, in adaptive systems, LUT size has an inverse relationship with the convergence speed of the adaptation, so it is preferred to keep the LUT as small as possible.

*Word length* of parameters stored in memory is related to noise and accuracy of the whole system. Since a reduction in the adjacent channel interference is desired, then the dominant noise must not be due to quantization. The desired level of adjacent-channel-interference suppression will set the minimum signal to quantization noise ratio (SNRq) for the table entries. SNRq first depends on word length and then on the ratio of the peak amplitude to the RMS amplitude of the signal. Further details on this topic can be found in [Sun96].

### 5.2.3 LUT Spacing

How to organize the LUT spacing has been an interesting topic of discussion for several years [Cav97, Cav99, Muh99], since a uniform or non-uniform spacing of the LUT is closely related to the linearization performance achieved by DPD linearizers. The so called *companding function* is

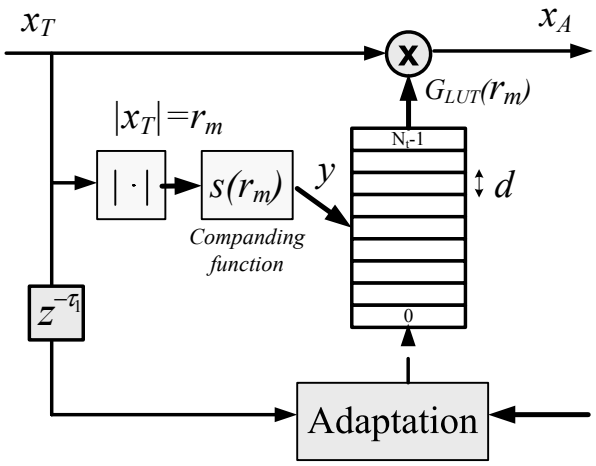


**Figure 5.5:** Structure of a LUT based predistorter with companding function.

the responsible for deriving the spacing of the input levels in the LUT. It performs a processing of input data for pointing the LUT in different resolution ranges (i.e, concentrating most of memory registers for predistortion operation near to the PA compression point). Figure 5.5 shows the basic structure of a LUT based DPD with a companding function $s(\cdot)$ that is responsible for the LUT uniform or non-uniform spacing.

Therefore, if a LUT of $N_t$ entries (see Fig. 5.5) is considered, the width of the LUT bins ($d$) in the $y$ domain will be:

$$d = \frac{1}{N_t \cdot s'(r_m)} \tag{5.1}$$

where $r_m$ is defined as $r_m =\mid x_T \mid$.

The most common companding functions reported in literature are:

- Amplitude:

$$s(r_m) = r_m \rightarrow s'(r_m) = 1 \tag{5.2}$$

- Power:

$$s(r_m) = r_m^2 \rightarrow s'(r_m) = 2 \cdot r_m \tag{5.3}$$

- $\mu-$law (as the one used in American companders for voice telephony):

$$s(r_m) = \frac{\ln(1+\mu \cdot r_m)}{\ln(1+\mu)} \rightarrow s'(r_m) = \frac{\mu}{\ln(1+\mu)} \cdot \frac{1}{1+\mu \cdot r_m} \tag{5.4}$$

  where $\mu = 255$ (8 bits) in the North American and Japanese standards.

- Caver's optimum indexing :

$$s'(r_m) = \left( \int w(r_m)^{1/3} dr \right)^{-1} \cdot w(r_m)^{1/3} \tag{5.5}$$

where $w(r_m)$ is the weighting function defined by:

$$w(r_m) = \frac{r_m^2}{12 \cdot N_t^2} \cdot \frac{|g'(r_m)|^2}{|g'(r_m)|^4} \cdot p_r(r_m) \tag{5.6}$$

where $g(r_m)$ $\big(f(r_m) \cdot g(r_m) = K\big)$ is the complex gain of the PA and $p_r(r_m)$ the signal amplitude *probability density function* (pdf).

- Sub-optimum indexing [Muh99]: Considering Caver's optimum indexing:

$$s'(r_m) = \left( \int w(r_m)^{1/3} dr \right)^{-1} \cdot w(r_m)^{1/3} \tag{5.7}$$

but defining $w(r_m) = r_m^2 \cdot p_r(r_m)$.

The best linearity performance is achieved with Cavers *optimum companding function.* However, due to its computational complexity and its dependence on signal's pdf, PA nonlinearities and back-off can make it unsuitable. On the other hand, results reported advert that spacing by *$\mu$-law* and *power* suffer from significant intermodulation power generation at high and low signal levels, respectively, because their table entries are unnecessary concentrated at the other end of the amplitude range. *Amplitude* spacing (uniform spacing) provides good enough results in comparison to the *optimal companding function* to be a serious candidate that provides good linearity performance with little complexity. The main drawback regarding the amplitude companding approach is that the spacing distribution is fixed, while by using a *sub-optimal* approach the less complex algorithm permits considering the signal characteristics (i.e. pdf).

## 5.3 Identification Approaches

Behavioral models describe the PA nonlinear dynamic behavior but, in order to linearize the PA, it is necessary to obtain its inverse characteristic (envelope filtering technique). In this section two identification approaches oriented at obtaining the PA inverse characteristic are presented. The first one, named *postdistortion and translation method*, is commonly used in literature to extract the inverse function (DPD function) of the PA in two steps. The second one is a new identification method, the *predictive predistortion method*, oriented at overcoming some formal issues that present the former one.

### 5.3.1 Indirect Learning: Postdistortion and Translation Method

The indirect learning or post-distortion and translation method is a commonly used technique to identify the predistortion function [Mar03, Kim06, Din04, Cho05, Gil06a]. Its basic functioning is schematically depicted in Fig. 5.6. In the indirect learning approach, a first postdistortion

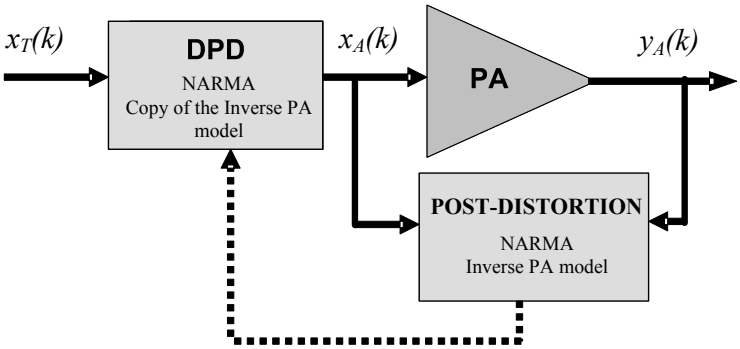


**Figure 5.6:** Block diagram of the indirect learning: postdistortion and translation method.

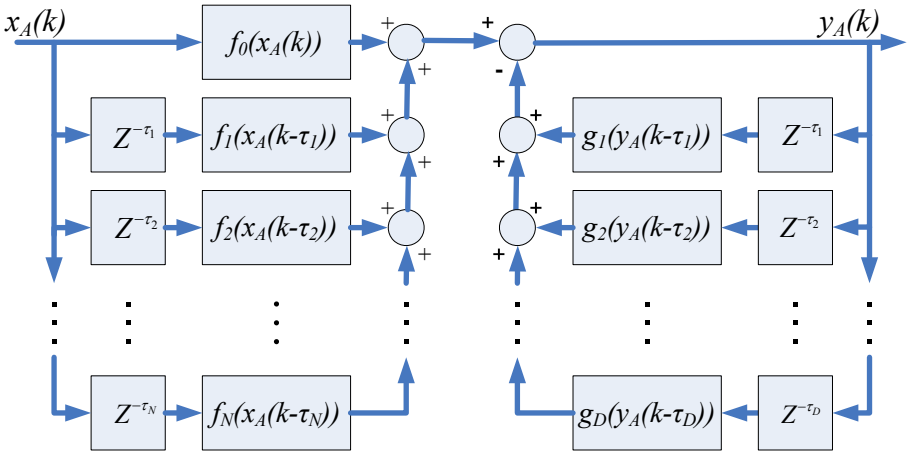


**Figure 5.7:** General block diagram of a NARMA structure.

function is estimated using the power amplifier's input ($x_A(k)$) and output ($y_A(k)$) baseband modulated data. Once the coefficients defining the postdistortion function are estimated, they are copied to an identical model that is used to predistort the input signal that will later fed to the PA.

The main advantage of this identification approach consists in the fact that the postdistortion function is obtained from direct input and output PA observations,

$$F_{Post}\left(G\left(x_T\right)\right) = K \cdot x_T \tag{5.8}$$

with $F_{Post}\left(\cdot\right)$ and $G\left(\cdot\right)$ being the postdistortion and the PA nonlinear functions respectively, and $x_T$ the discrete complex envelope of the signal to be transmitted.

From now on, the NARMA model presented in Chapter 4 (and depicted again in Fig. 5.7) will be considered for the post and predistortion purposes. Following the notation shown in Fig.

5.6, the general NARMA expression is defined again as

$$\hat{y}_A(k) = \sum_{i=0}^{N} \hat{f}_i(x_A(k-\tau_i)) - \sum_{j=1}^{D} \hat{g}_j(y_A(k-\tau_j)) \tag{5.9}$$

Therefore, the postdistortion function in the indirect learning approach (see Fig. 5.6) can be expressed as the inverse model:

$$\hat{x}_A(k) = \sum_{i=0}^{N} \hat{f}_i(y_A(k-\tau_i)) - \sum_{j=1}^{D} \hat{g}_j(x_A(k-\tau_j)) \tag{5.10}$$

where $\hat{f}_i$ and $\hat{g}_j$ are estimated nonlinear functions that can be implemented with polynomials or by using look-up tables (LUTs). In addition, $\tau_0 = 0$ while $\tau_i$ and $\tau_j$ ($\tau \subset \mathbb{N}$) are the most significant sparse delays of the input and the output respectively, contributing at the description of the PA memory effects. As explained in Chapter 4, it is possible to extract these optimal delays by means of the SA heuristic search algorithm.

To implement the predistorter function in the FPGA it is necessary to map nonlinear functions $\hat{f}_i$ and $\hat{g}_j$ into a set of LUTs. But first these nonlinear functions defining the postdistorter and predistorter nonlinear operation have to be calculated. For that reason, these static nonlinear functions in (5.10) are implemented with polynomials,

$$\begin{aligned} \hat{f}_i(y_A(k-\tau_i)) &= \sum_{p=0}^{P} \alpha_{pi} \cdot y_A(k-\tau_i)\,|y_A(k-\tau_i)|^p \\ \hat{g}_j(x_A(k-\tau_j)) &= \sum_{p=0}^{P} \beta_{pj} \cdot x_A(k-\tau_j)\,|x_A(k-\tau_j)|^p \end{aligned} \tag{5.11}$$

with $P$ being the polynomial order, $\alpha_{pi}$ and $\beta_{pj}$ their complex coefficients respectively, $i$=(0, 1,. . . ,N), $j$=(1,. . . ,D) and $\tau_0 = 0$.

Expanding (5.10) with (5.11) and expressing it in a more compact matrix notation, it is possible to rewrite (5.10) as

$$\hat{x}_A(k) = \lambda^H \theta \tag{5.12}$$

where $\lambda = \begin{pmatrix} \alpha_{00}, \alpha_{10}, \cdots \alpha_{P0}, \alpha_{01}, \cdots, \alpha_{P1}, \cdots \alpha_{0N}, \cdots \alpha_{PN}, \\ -\beta_{01}, \cdots, -\beta_{P1}, \cdots, -\beta_{0D}, \cdots, -\beta_{PD} \end{pmatrix}^T$,

$$\theta = \begin{pmatrix} y_A(k),\, y_A(k)\,|y_A(k)|, \cdots, y_A(k)\,|y_A(k)|^P, y_A(k-\tau_1), \cdots, y_A(k-\tau_1)\,|y_A(k-\tau_1)|^P, \\ \cdots, y_A(k-\tau_N), \cdots, y_A(k-\tau_N)\,|y_A(k-\tau_N)|^P, x_A(k-\tau_1), \\ \cdots x_A(k-\tau_1)\,|x_A(k-\tau_1)|^P, \ldots, x_A(k-\tau_D), \cdots, x_A(k-\tau_D)\,|x_A(k-\tau_D)|^P \end{pmatrix}^T$$

and where the superindex $^H$ denotes Hermitian.

The cost function to be minimized in order to extract the $\alpha_{pi}$ and $\beta_{pj}$ complex coefficients describing the postdistorter NARMA based nonlinear function is defined by the following equation:

$$J\left(|e(k)|^2\right) = J\left(|x_A(k) - \hat{x}_A(k)|^2\right) = J\left(\left|x_A(k) - \lambda^H \theta\right|^2\right) \tag{5.13}$$

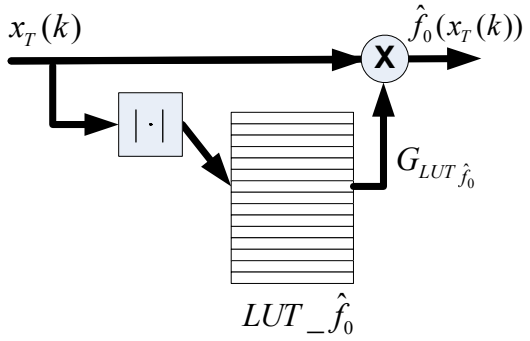


**Figure 5.8:** Basic Predistortion Cell for FPGA implementation.

The estimation error is defined as the difference between the original PA input data and the estimated output data of the postdistorter. Different identification algorithms can be used to extract the postdistorter parameters, such *gradient algorithms* (Newton method, Steepest Descent, LMS, Fast Kalman) or *parametric estimation methods* (Least Squares, Recursive LS, Extended LS).

Once we have the postdistorter function identified, it is possible to map the predistortion function in the FPGA, but to fulfill this objective it is first necessary to express the predistortion function as a combined set of LUTs. With $x_T$ being the predistorter input and $x_A$ the predistorter output (see Fig. 5.6), the predistortion function remains as an exact copy of the estimated postdistortion function, that is:

$$x_A(k) = \sum_{i=0}^{N} \hat{f}_i(x_T(k-\tau_i)) - \sum_{j=1}^{D} \hat{g}_j(x_A(k-\tau_j)) \tag{5.14}$$

Now is possible to conceive the input-output relation of the predistortion function as a Cartesian complex product between the input/output sample $\big(x_T(k), x_A(k)\big)$ and a complex gain ($G_{LUT}$) that depends on the envelope of the signal. Therefore we can rewrite the static nonlinear functions in (5.11) as:

$$\begin{aligned} \hat{f}_i(x_T(k-\tau_i)) &= x_T(k-\tau_i) \cdot G_{LUT-\hat{f}_i}\left(|x_T(k-\tau_i)|\right) \\ \hat{g}_j(x_A(k-\tau_j)) &= x_A(k-\tau_j) \cdot G_{LUT-\hat{g}_j}\left(|x_A(k-\tau_j)|\right) \end{aligned} \tag{5.15}$$

This generalization is the crucial step towards a FPGA practical DPD implementation, since it enables the DPD to be stated in terms of a *Basic Predistortion Cell* (BPC). A simplified block diagram of a BPC based in a complex gain DPD with *uniform spacing* is shown in Fig. 5.8. Therefore, BPCs are the fundamental building blocks from which DPD functions derived from the NARMA model (also Volterra derivations) can be quickly mapped into a FPGA device.

Finally the input-output relation of the predistorter in (5.14) can be expressed as the com-

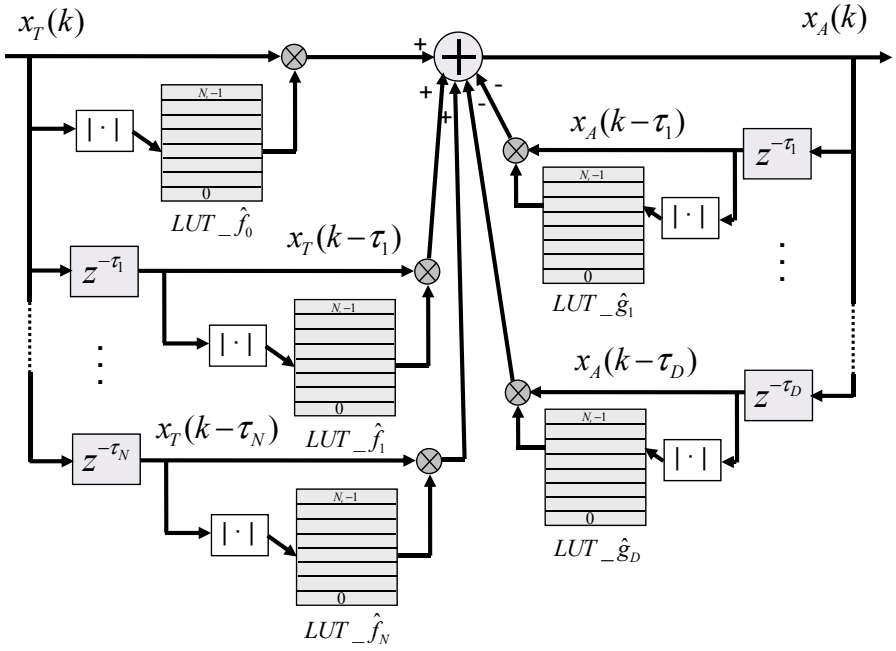


**Figure 5.9:** Multi-LUT implementation of a NARMA based Predistorter with the indirect learning approach.

bination of a set of LUTs,

$$
\begin{aligned}
x_A(k) = x_T(k)\cdot \underbrace{\left[\alpha_{00}+\alpha_{10}|x_T(k)|+\cdots+\alpha_{P0}|x_T(k)|^P\right]}_{LUT\_\hat{f}_0=G_{LUT-\hat{f}_0}(|x_T(k)|)} + \\
+x_T(k-\tau_1)\cdot \underbrace{\left[\alpha_{01}+\alpha_{11}|x_T(k-\tau_1)|+\cdots+\alpha_{P1}|x_T(k-\tau_1)|^P\right]}_{LUT\_\hat{f}_1=G_{LUT-\hat{f}_1}(|x_T(k-\tau_1)|)} + \\
\vdots \\
+x_T(k-\tau_N)\cdot \underbrace{\left[\alpha_{0N}+\alpha_{1N}|x_T(k-\tau_N)|+\cdots+\alpha_{PN}|x_T(k-\tau_N)|^P\right]}_{LUT\_\hat{f}_N=G_{LUT-\hat{f}_N}(|x_T(k-\tau_N)|)} - \\
-x_A(k-\tau_1)\cdot \underbrace{\left[\beta_{01}+\beta_{11}|x_A(k-\tau_1)|+\cdots+\beta_{P1}|x_A(k-\tau_1)|^P\right]}_{LUT\_\hat{g}_1=G_{LUT-\hat{g}_1}(|x_A(k-\tau_1)|)} - \\
\vdots \\
-x_A(k-\tau_D)\cdot \underbrace{\left[\beta_{0D}+\beta_{1D}|x_A(k-\tau_D)|+\cdots+\beta_{PD}|x_A(k-\tau_D)|^P\right]}_{LUT\_\hat{g}_D=G_{LUT-\hat{g}_D}(|x_A(k-\tau_D)|)}
\end{aligned}
\tag{5.16}
$$

The predistorter input-output relation expressed in (5.16) by means of BPCs is graphically shown in Fig. 5.9, where we can see the multi-LUT implementation that would be mapped in the FPGA. Results showing a multi-LUT NARMA based predistorter using the indirect learning approach can be found in [Gil06a].

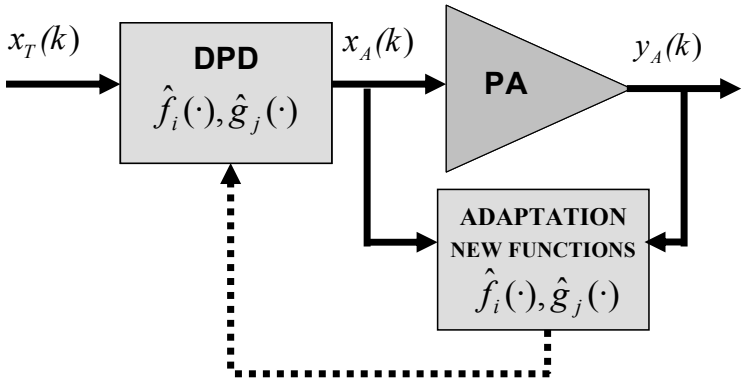


**Figure 5.10:** Block diagram of the Predictive Predistorter approach.

### 5.3.2 Predictive Predistortion Method

Despite the good results shown by the indirect learning approach (regarding linearity improvement), the postdistortion and translation method assumes the commutative property for cascading nonlinear systems, which is not rigorous at all. For that reason we proposed in [Mon07] an alternative method for identifying and adapting the predistortion function, the *predictive predistortion method*. This method permits an identification and later adaptation of the predistortion function that relies just on the PA NARMA model. The stability of the NARMA model, as it has been discussed previously, may be, in any case, assessed and ensured.

The *digital predictive predistortion* approach follows the block diagram shown in Fig. 5.10, where digital predistortion linearization is carried out at baseband by adaptively forcing the PA to behave as a linear device.

The functioning of the digital PD is quite simple and intuitive. First it is necessary to perform an identification of the low-pass complex envelope PA behavioral model, which in our case is the NARMA model described in equation (5.9). From the NARMA model a first set of nonlinear functions ($\hat{f}_i$ and $\hat{g}_j$) defining the PA behavior are calculated at baseband using the PA input ($x_A$) and output ($y_A$) discrete complex envelope data. Once we have identified the $\hat{f}_i$ and $\hat{g}_j$, we now consider $y_D$ as the desired linearized PA output. This desired output is defined as the signal to be transmitted ($x_T$) multiplied by a linear gain ($G_{linear}$),

$$y_D(k) = x_T(k) \cdot G_{linear} \tag{5.17}$$

As shown in Fig. 5.10, $x_A(k) = x_T(k)$ if no baseband predistortion is considered. From the PA NARMA model expression in (5.9) it is possible to derive (5.18),

$$\hat{f}_0(x_A(k)) = y_A(k) - \sum_{i=1}^{N} \hat{f}_i(x_A(k-\tau_i)) + \sum_{j=1}^{D} \hat{g}_j(y_A(k-\tau_j)) \tag{5.18}$$

and solving (5.18) it is possible to obtain the necessary amplifier input $\hat{x}_A(k)$ in order to achieve a certain output $y_A(k)$. Since the desired output $y_D(k)$ is evaluated a priori (see (5.17)), then in (5.18) $y_A$ is replaced by $y_D$ (and the same for all delayed output samples). In other words, the desired output $y_D(k)$ is considered as a prediction of the future value of $y_A(k)$ (current output) and consequently, the input value of the PA ($\hat{x}_A(k)$) that permits achieving the desired output $\big(y_A(k) = y_D(k)\big)$ is calculated. Finally, the digital PD output $\hat{x}_A(k)$ (and PA input) can be expressed as:

$$\hat{x}_A(k) = \hat{f}_0^{-1}\left(y_D(k) - \sum_{i=1}^{N} \hat{f}_i(\hat{x}_A(k-\tau_i)) + \sum_{j=1}^{D} \hat{g}_j(y_D(k-\tau_j))\right) \tag{5.19}$$

In order to implement the predistortion function in a FPGA, it is necessary to express it a combination of LUTs. For that reason, similarly to what has been done in the previous subchapter, we now consider the NARMA model expressed in (5.9), expanded by means of their polynomial expression. Hence it results

$$\begin{aligned}
y_A(k) = {} & x_A(k) \cdot \underbrace{\left[\alpha_{00} + \alpha_{10}\,|x_A(k)| + \cdots + \alpha_{P0}\,|x_A(k)|^P\right]}_{LUT_\widehat{f}_0 = G_{LUT-\hat{f}_0}(|x_A(k)|)} + \\
& x_A(k-\tau_1) \cdot \underbrace{\left[\alpha_{01} + \alpha_{11}\,|x_A(k-\tau_1)| + \cdots + \alpha_{P1}\,|x_A(k-\tau_1)|^P\right]}_{LUT_\widehat{f}_1 = G_{LUT-\hat{f}_1}(|x_A(k-\tau_1)|)} + \\
& \vdots \\
& x_A(k-\tau_N) \cdot \underbrace{\left[\alpha_{0N} + \alpha_{1N}\,|x_A(k-\tau_N)| + \cdots + \alpha_{PN}\,|x_A(k-\tau_N)|^P\right]}_{LUT_\widehat{f}_N = G_{LUT-\hat{f}_N}(|x_A(k-\tau_N)|)} - \\
& y_A(k-\tau_1) \cdot \underbrace{\left[\beta_{01} + \beta_{11}\,|y_A(k-\tau_1)| + \cdots + \beta_{P1}\,|y_A(k-\tau_1)|^P\right]}_{LUT_\widehat{g}_1 = G_{LUT-\hat{g}_1}(|y_A(k-\tau_1)|)} - \\
& \vdots \\
& y_A(k-\tau_D) \cdot \underbrace{\left[\beta_{0D} + \beta_{1D}\,|y_A(k-\tau_D)| + \cdots + \beta_{PD}\,|y_A(k-\tau_D)|^P\right]}_{LUT_\widehat{g}_D = G_{LUT-\hat{g}_D}(|y_A(k-\tau_D)|)}
\end{aligned} \tag{5.20}$$

Therefore, in a similar manner as in the indirect learning approach, it is possible to rewrite the predistortion function in (5.19) in a more convenient DPD expression, in terms of the (delayed) complex inputs and outputs multiplied by its corresponding complex gain ($G_{LUT}$),

$$\hat{x}_A(k) = z(k) \cdot G_{LUT-\hat{f}_0^{-1}}(|z(k)|) \tag{5.21}$$

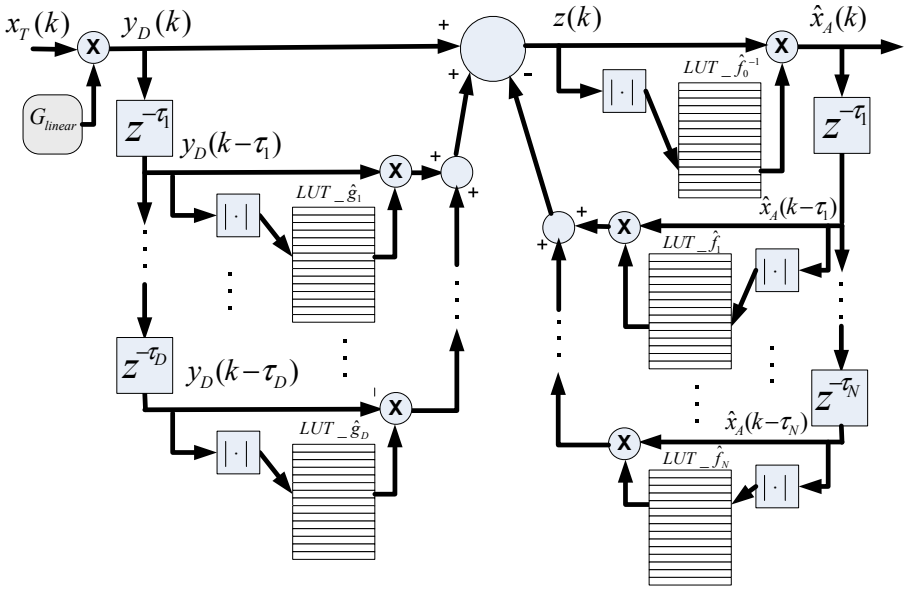


**Figure 5.11:** Predictive digital predistorter implementation with multi-LUTs.

where $z(k)$ is defined as:

$$z(k) = y_D(k) + \sum_{j=1}^{D} y_D(k-\tau_j) \cdot G_{LUT-\hat{g}_j}\left(|y_D(k-\tau_j)|\right) - \\ - \sum_{i=1}^{N} \hat{x}_A(k-\tau_i) \cdot G_{LUT-\hat{f}_i}\left(|\hat{x}_A(k-\tau_i)|\right) \tag{5.22}$$

Thanks to (5.21) and (5.22), the DPD can be mapped into the FPGA as a set of parallel and cascade BPCs, as depicted in Fig. 5.11. Furthermore, suitable DPD operation is obtained by just downloading the appropriate complex gain values into each BPC LUT.

## 5.4 Adaptation Process: Look-up Table Updates

As it has been presented in previous subsection, the predistortion function is described by the Multi-LUT configuration in Fig. 5.11 that results from applying the predictive NARMA predistortion method. The hardware implementation of the predistortion function is carried out in a FPGA board, in charge of the DPD processing at the actual sample rate and thus allowing high data throughput. The FPGA considered in the experimental set-up that will be presented in Chapter 6, is a Xilinx Virtex-IV XC4VSX35, with the developed DPD core in charge of predistortion running at 105MHz. An overview on the Virtex-IV family specifications can be found in [vir07], while an insight of the Nallatech XtremeDSP Development Board-IV is depicted in Fig. 5.12. The linearization process in itself is open loop controlled, and works separately of the adaptation process.

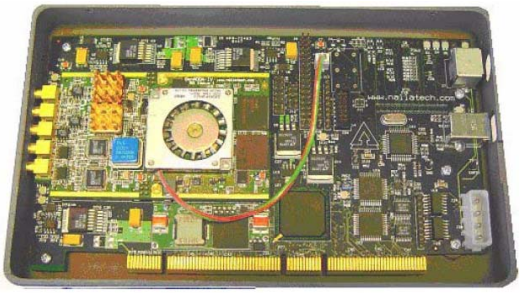

**Figure 5.12:** Nallatech XtremeDSP Virtex IV Development Board.

On the other hand, the adaptation process, consisting in periodically calculating new predistortion values to fill the LUT contents, can be performed in both a host PC running Matlab or in the same FPGA in charge of the real-time predistortion. Therefore, provided that the DPD function is carried out in a FPGA device, two possible configurations to carry out the adaptation process of the LUTs can be considered:

- External adaptation; where algorithms that will provide new updated LUTs to the DPD function are programmed in an external device (PC or DSP).
- Real time FPGA adaptation; where the LUT contents are continuously updated (real-time) in the same FPGA board and thus taking advantage of the parallel processing capabilities of this device.

In the following, both adaptation configurations are presented and discussed. Besides, simulation results showing the linearity performance achieved by the predictive-NARMA DPD will be provided for both external and real-time adaptation scenarios

### 5.4.1 External Adaptation Description

In this kind of configuration, while predistortion is carried out in a FPGA, the more complex adaptation algorithms are computed in an external device, such a DSP board as in [Váz06,BN05], and thus having more relaxed adaptation time constants. For simplicity and to enhance flexibility during the prototyping procedures, the DSP device can be replaced by an external host PC in which Matlab is in charge of the adaptation. As shown in Fig. 5.13, it is necessary to have a feedback loop from the PA output towards the FPGA, through the demodulator and A/D converters, to capture the necessary data enabling the adaptation process. In the proposed implementation, the FPGA provides the external host with buffers of predistorted and PA output data, of 2048 I/Q samples each, from which the NARMA model is derived. The D/A and A/D converters handle 14 bits data, at 105 MSPS as well.

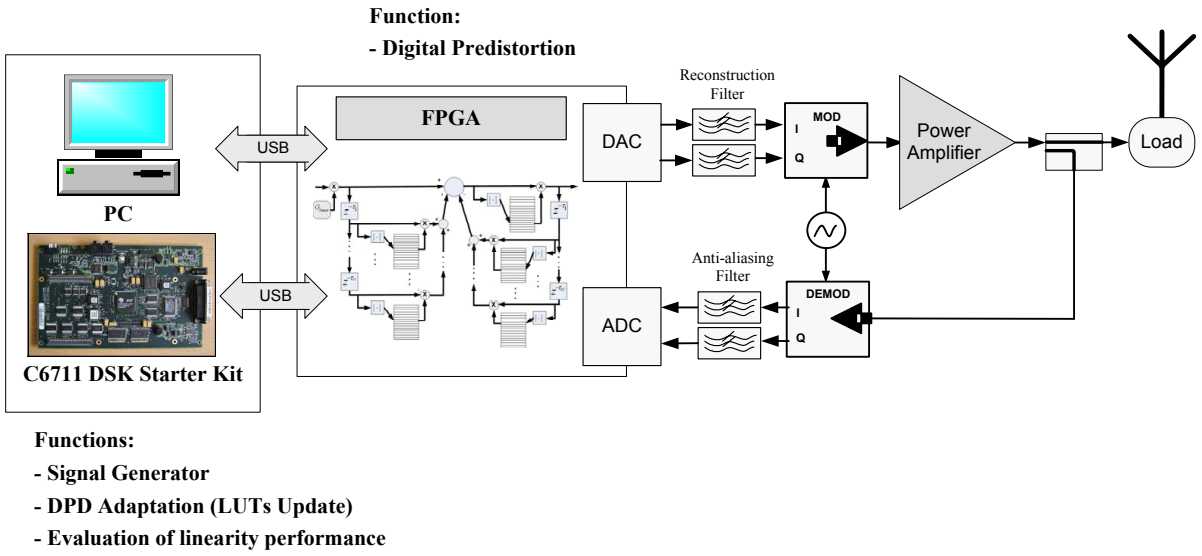


**Figure 5.13:** Block diagram of a DPD with external adaptation, carried out in a PC or DSP board.

The update of the LUT contents describing the DPD relies just on the PA NARMA model defined in (5.9), since the DPD nonlinear functions of the Predictive-NARMA DPD in (5.19) are the same as in (5.9) except for $\hat{f}_0^{-1}$. The PA model identification procedure is performed via the least squares (LS) algorithm, taking advantage of using buffers of data. In the next subchapter where real-time adaptation in the same FPGA is pretended, sample-per-sample identification algorithms such as the LMS or Fast Kalman [Hay91] will be considered.

Therefore, considering $\mathbf{x_A}$ the complex data vector at the DPD output (PA input - see Fig. 5.10) and $\mathbf{y_A}$ the corresponding time-aligned complex data vector of the PA output (and normalized by the linear PA gain to allow signals comparison), both vectors of L samples length, we define

$$x_A^{pi}(k) = x_A(k-\tau_i)\,|x_A(k-\tau_i)|^p \tag{5.23}$$

$$y_A^{pj}(k) = y_A(k-\tau_j)\,|y_A(k-\tau_j)|^p \tag{5.24}$$

Then, the NARMA input-output relation in (5.9) can be expressed in a matrix notation as

$$\mathbf{y_A} = \mathbf{Q}\hat{\delta} \tag{5.25}$$

where $\mathbf{y_A} = [y_A(0), \cdots, y_A(L-1)]^T$,

$\mathbf{Q} = \left[\mathbf{x_A^{00}}, \cdots, \mathbf{x_A^{P0}}, \cdots, \mathbf{x_A^{0N}}, \cdots, \mathbf{x_A^{PN}}, \mathbf{y_A^{01}}, \cdots, \mathbf{y_A^{P1}}, \cdots, \mathbf{y_A^{0D}}, \cdots, \mathbf{y_A^{PD}}\right]$,

$\mathbf{x_A^{pi}} = \left[x_A^{pi}(0), \cdots, x_A^{pi}(L-1)\right]^T$, $\mathbf{y_A^{pj}} = \left[y_A^{pj}(0), \cdots, y_A^{pj}(L-1)\right]^T$ and

$\hat{\delta} = \left[\begin{array}{l} \alpha_{00}, \cdots, \alpha_{P0}, \cdots, \alpha_{0N}, \cdots, \alpha_{PN}, \\ -\beta_{01}, \cdots, -\beta_{P1}, \cdots, -\beta_{0D}, \cdots, -\beta_{PD} \end{array}\right]^T$.

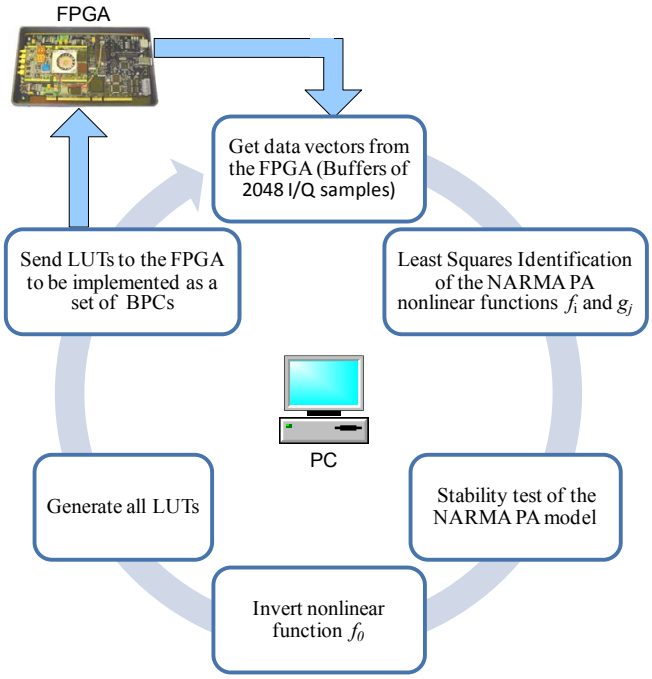


**Figure 5.14:** Flow diagram of the external adaptation carried out in the host PC.

Hence, the Least Square (LS) solution for (5.25) is:

$$\hat{\delta} = (\mathbf{Q}^H\mathbf{Q})^{-1}\mathbf{Q}^H\mathbf{y_A} \tag{5.26}$$

where superindex H denotes complex conjugate transpose.

The adaptive process performed in the host PC is schematically depicted in Fig. 5.14 and described by the following steps:

- In a first step, identifying by means of the LS algorithm in (5.26), new $\hat{f}_i$ and $\hat{g}_j$ nonlinear functions (described in (5.9)) by monitoring current PA input ($x_A$) and output ($y_A$) data vectors, as it is shown in Fig. 5.10.
- Second step: test the stability of the resulting PA NARMA model (bounds given by the small-gain theorem) described in Chapter 4.
- Third step consists in inverting the $\hat{f}_0$ memoryless nonlinear function to obtain the digital predistorter output as it is shown in (5.19).
- The last step consists in generating all necessary LUTs to implement the predistorter function described in (5.19). From (5.20), (5.21) and (5.22) complex gains ($G_{LUT}$) are computed and fed into the FPGA in the BPC convenient LUT form. Then, back to step one.

Note that at every adaptation routine, all LUT contents (BPC complex gains) are recalculated, since so do their associated nonlinear functions defining the NARMA PA model. Besides, the computational complexity introduced by the LS algorithm, handling 2048x2048 matrixes of complex data, makes unfeasible a real-time adaptation.

### 5.4.2 External Adaptation Simulation Results

To validate the predictive NARMA DPD with external adaptation, a FPGA DPD simulator has been implemented in Matlab. This simulator emulates the DPD function as it is performed in a FPGA, that is, by means of a set of BPCs. Moreover, a PA behavioral model based in a Hammerstein structure has been used. This Hammerstein based PA model has been extracted from input-output complex data obtained from a final stage 170 W peak power PA based on the Freescale MRF21170 MOSFET transistor. A medium power PA based on the MRF21010 transistor (10 W peak power), acting as a driver, precedes the main output amplifier. This RF chain is actually the one used for the experimental results that will be presented in Chapter 6.

By adjusting the PA quiescent point, and in order to see the compensation capabilities of the predictive NARMA DPD, two PA modes of operation have been considered: class AB and class B operation. Moreover, in order to handle signals with different PAPR, both multicarrier (OFDM) and Single-Carrier (SC) test signals with root-raised cosine (RRC) filtered M-QAM modulation schemes (roll-off of 0.22) have been taken into account.

In a first approach, the linearization performance achieved by a simple memoryless DPD and our proposed predictive NARMA DPD have been compared. Simulation results have been obtained after 5 iterations (LUT updates) considering buffers of 2048 samples (for both I-Q) of modulated complex data. The predictive NARMA DPD, whose basic structure is depicted in Fig. 5.11, has been configured with 7 BPCs, in concrete: 3 FIR LUTs ($LUT\_\hat{g}_1, LUT\_\hat{g}_2, LUT\_\hat{g}_3$), 3 IIR LUTs ($LUT\_\hat{f}_1, LUT\_\hat{f}_2, LUT\_\hat{f}_3$) and the inverse LUT associated to $\hat{f}_0$ ($LUT\_\hat{f}_0^{-1}$). In addition, back-off amplification (without DPD) has been considered in all results here presented to guarantee a fair comparison of the linearization performance achieved with DPD. Thus, both backed-off and linearized signals will present the same power at the PA output.

The SC 16-QAM modulation test signal presents a PAPR of around 6.5 dB. The AM-AM characteristics for both class AB and class B PA operation modes when memoryless DPD is applied are shown in Fig. 5.15. While in Fig. 5.16 the same AM-AM characteristics are shown when DPD with 7 BPCs is included, that is, activating the multi-LUT configuration of our predictive NARMA based DPD.

As shown in Fig. 5.15 and Fig. Fig. 5.16, the nonlinear behavior in a class AB PA is manifested as a compression of the linear AM-AM characteristic near PA saturation. On the other hand, the nonlinear behavior of a class B like PA is manifested at low input value levels and it is related

to crossover distortion.

It is possible to observe how the use of DPD derives in a linearized AM-AM characteristic (red line) independently whether memoryless or dynamic DPD is applied. Nonlinear distortion compensation can also be seen in the frequency domain as a reduction of the spectral regrowth at the PA output. Fig. 5.17 and Fig. 5.18 show the out-of-band compensation achieved for both memoryless and multi-LUT DPD, which turn to be similar, around 5 dB reduction in class AB amplification and at least 14 dB in class B amplification.

However, memoryless DPD cannot compensate for dispersion shown in the AM/AM characteristic, while using DPD with 7 BPCs, dispersion is reduced. This dispersion compensation in the AM-AM characteristic is directly translated as in-band compensation in the demodulated constellation, as it is shown when comparing Fig. 5.19 and Fig. 5.20. A significant amount of EVM reduction is achieved only when using DPD with memory compensation (7 BPCs).

Now, a test signal presenting a higher PAPR is considered. In concrete, an OFDM (with 256 subcarriers) 16-QAM modulated signal with a PAPR of around 9.5 dB. Analogously, the AM-AM characteristics for both class AB and class B PA operation modes when considering memoryless DPD and DPD with 7 BPCs are shown in Fig. 5.21 and Fig. 5.22, respectively.

Apparently, no significant differences can be noted with respect the results achieved with the SC 16-QAM modulation. As it can be observed in Fig. 5.23 and Fig. 5.24 both memoryless and multi-LUT based DPD perform well with out-of-band distortion compensation. However, 16-QAM constellations and their corresponding EVM metrics in Fig. 5.25 and Fig. 5.26 evidence again the advantage of using a DPD with memory compensation capabilities with respect a memoryless one. Moreover, in-band distortion when using OFDM appears to be more critical to PA nonlinear behavior, since backed-off amplified signals without DPD (blue) present significantly worse EVM than SC ones.

As it turns out, a memoryless DPD, solely aimed at pre-expanding the signal to compensate the compression effect, cannot fully pre-compensate the signal in a manner that permits dealing with PA nonlinear dynamics. Moreover, in-band distortion cannot be equalized by memoryless DPD unless some kind of filtering in the time domain is considered together with the nonlinear compensation. For that reason, a significant amount of improvement in PA linearization can be expected by using the predictive NARMA based DPD to minimize or cancel memory effects.

On the other hand, there are some significant issues derived from an external adaptation that require attention. The external update process, where new LUTs are calculated in host PC using the LS algorithm, is schematically described in Fig. 5.14. As it has been explained before, at every adaptation routine, all LUT contents (BPC complex gains) are recalculated, what may introduce some discontinuity and reliability problems to DPD. Fig. 5.27 and Fig. 5.29 show the evolution of the EVM, considering memoryless and dynamic DPD, along several

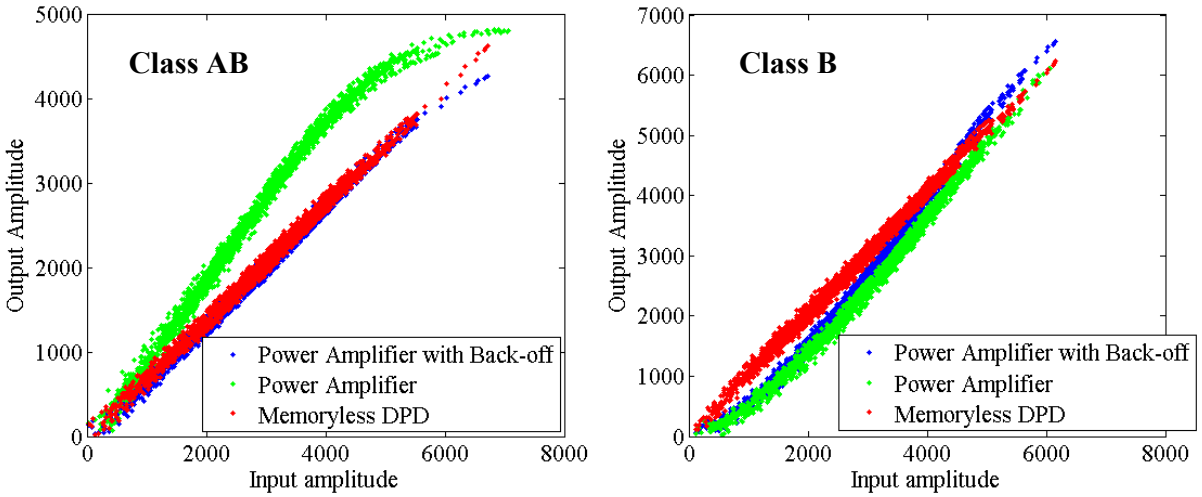


**Figure 5.15:** AM-AM characteristics for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal SC 16-QAM).

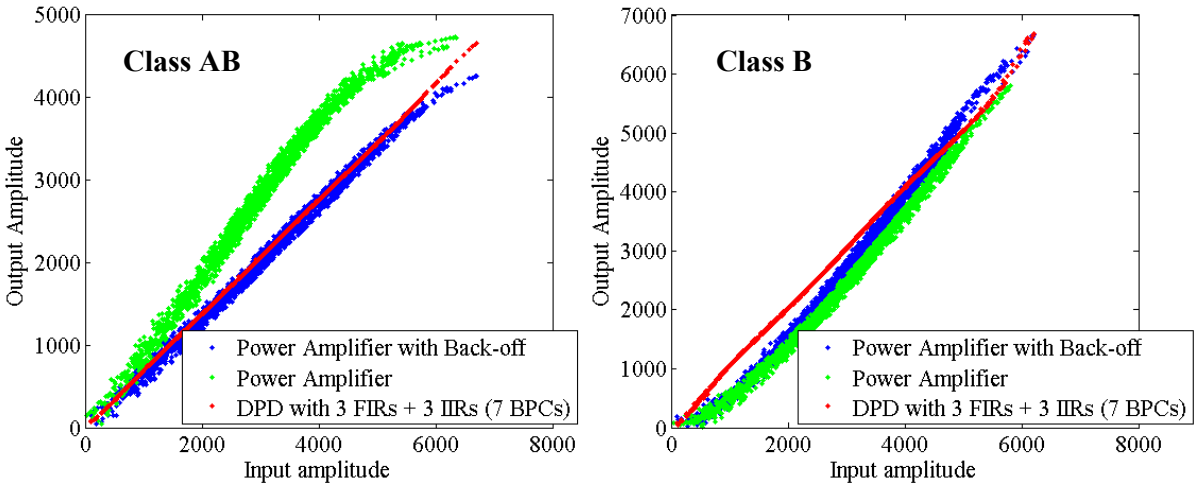


**Figure 5.16:** AM-AM characteristics for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal SC 16-QAM).

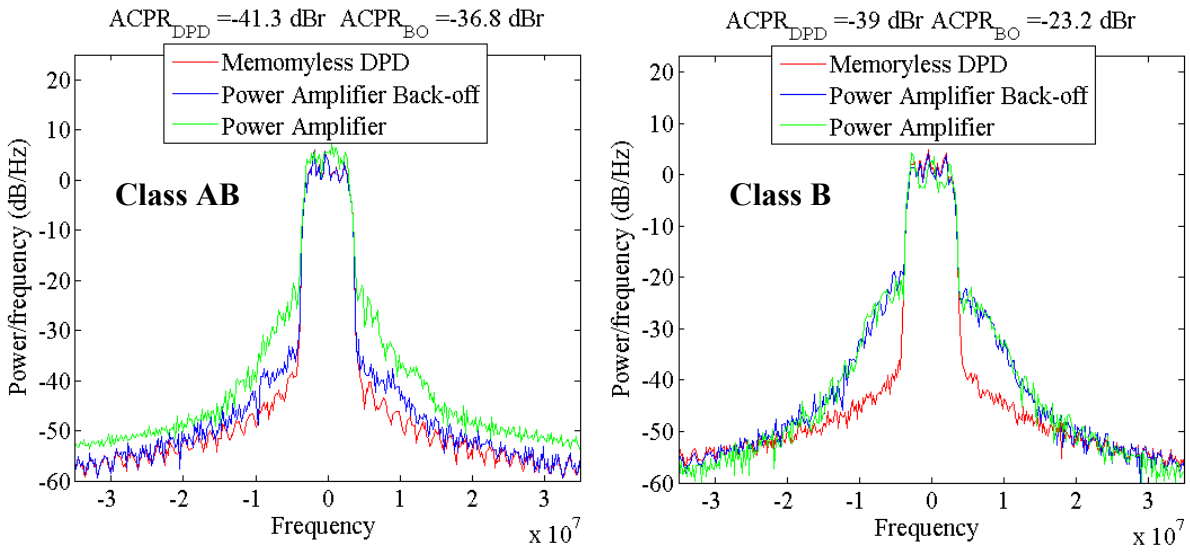


**Figure 5.17:** Output power spectra for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal SC 16-QAM).

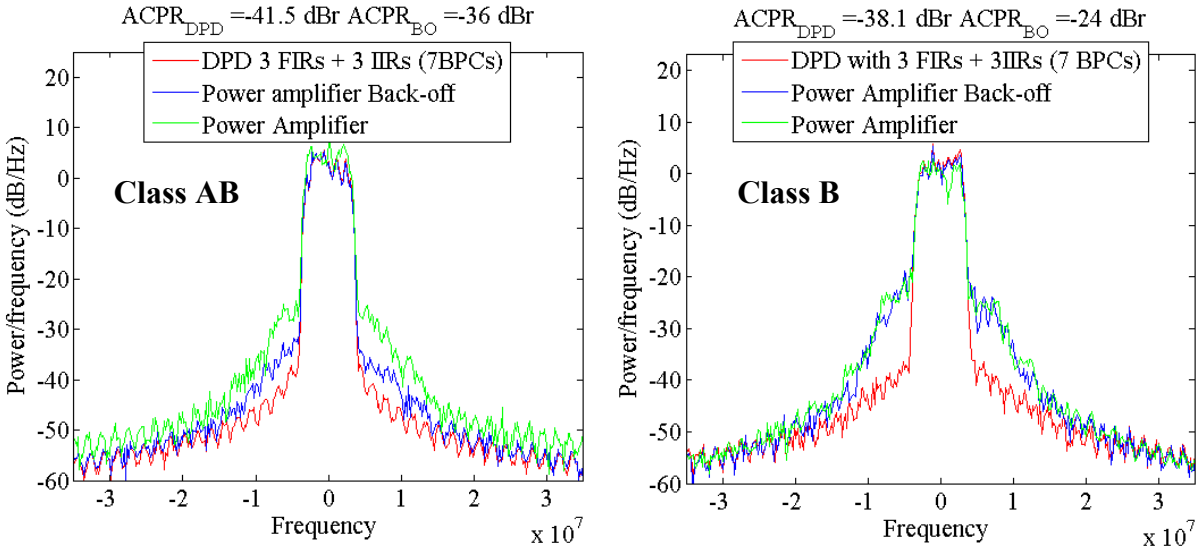


**Figure 5.18:** Output power spectra for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal SC 16-QAM).

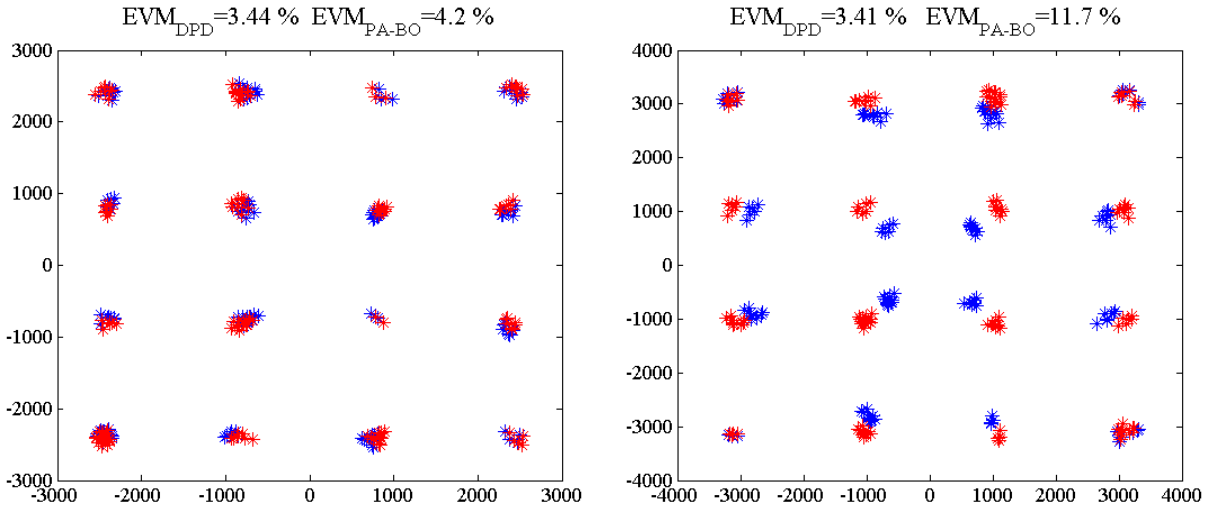


**Figure 5.19:** 16-QAM constellation for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal SC 16-QAM).

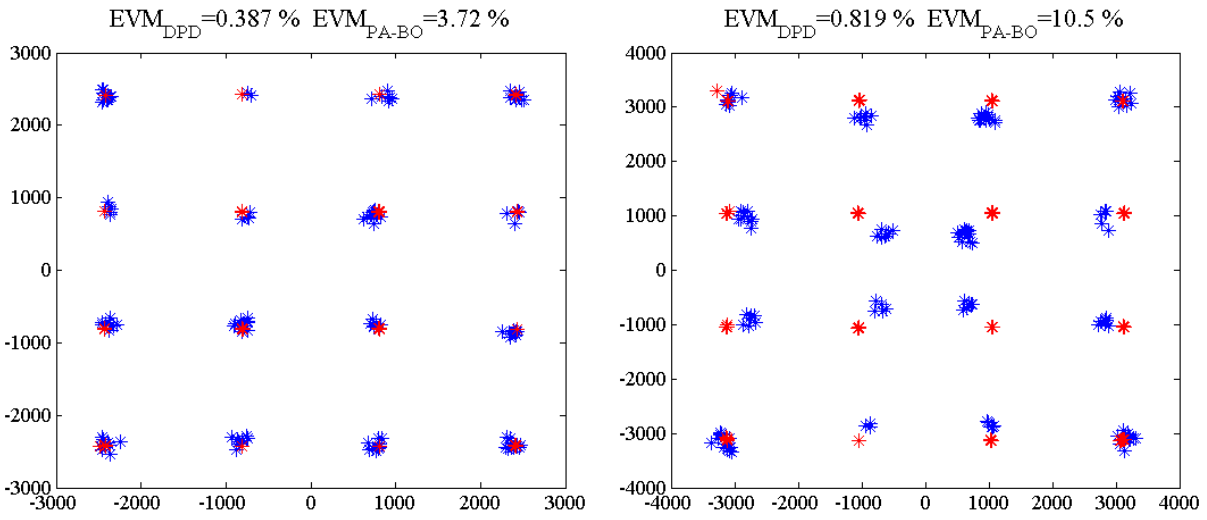


**Figure 5.20:** 16-QAM constellation for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal SC 16-QAM).

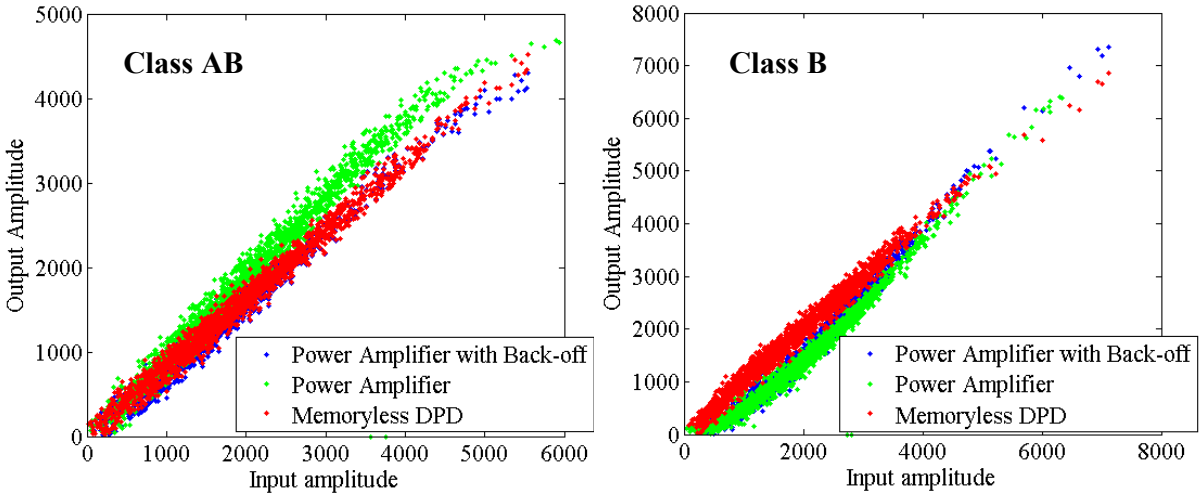


**Figure 5.21:** AM-AM characteristics for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal OFDM 16-QAM).

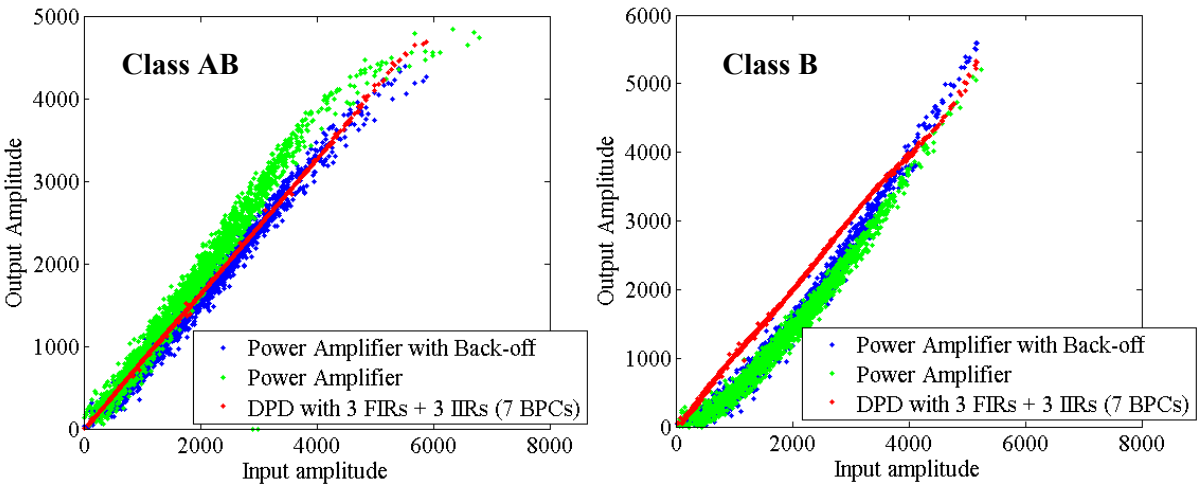


**Figure 5.22:** AM-AM characteristics for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal OFDM 16-QAM).

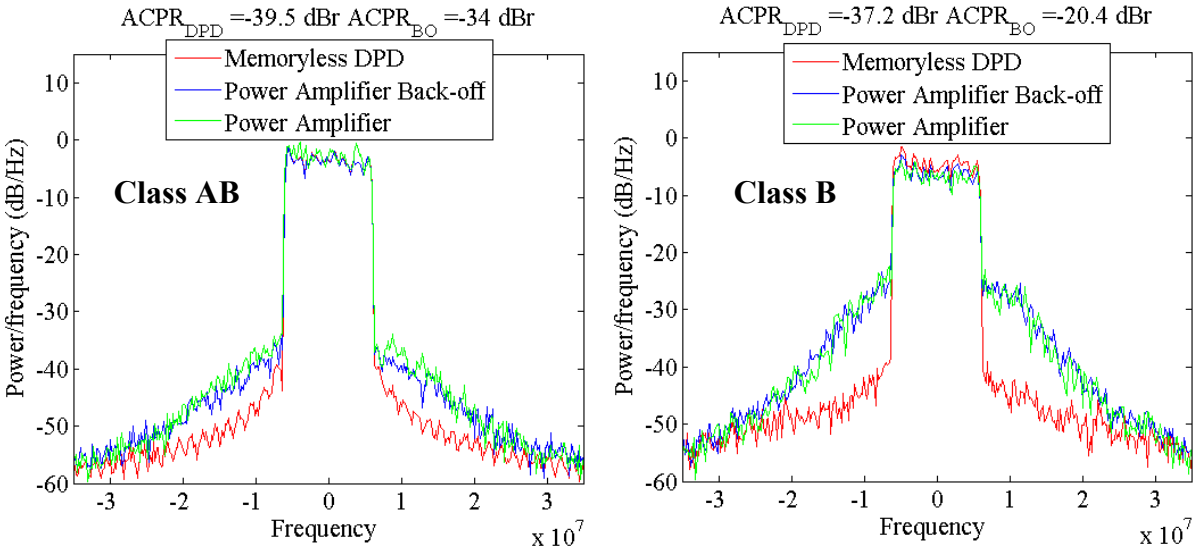


**Figure 5.23:** Output power spectra for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal OFDM 16-QAM).

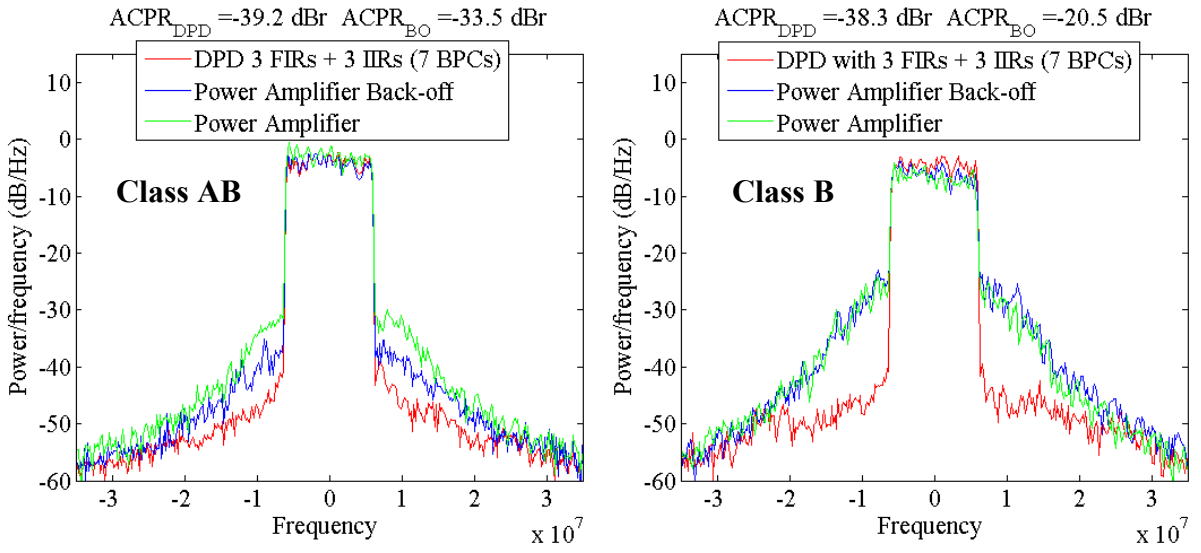


**Figure 5.24:** Output power spectra for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal OFDM 16-QAM).

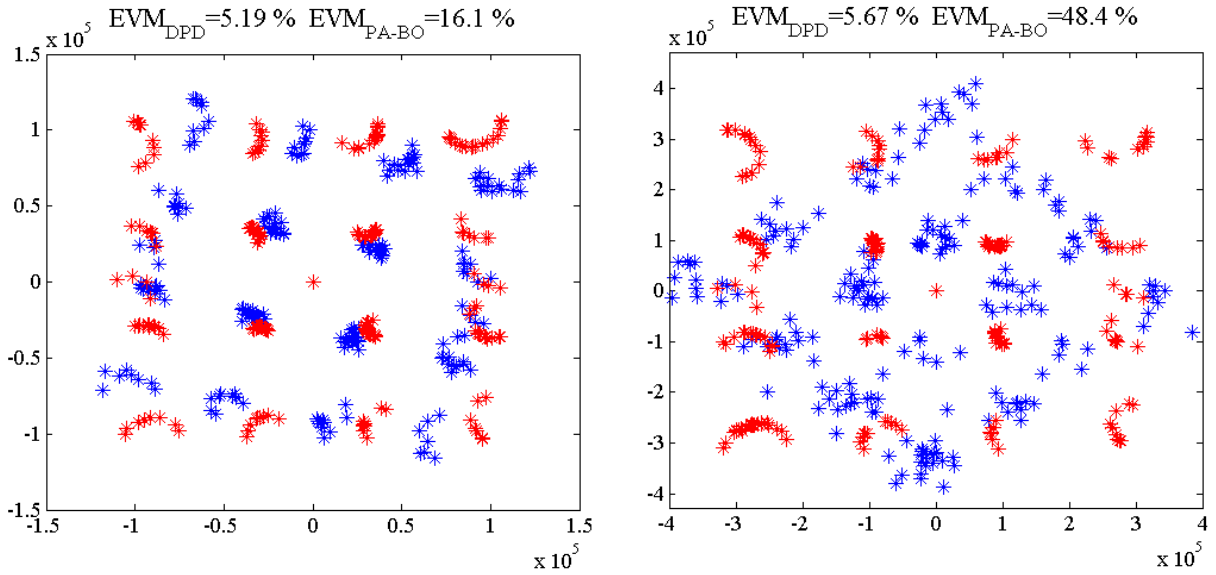


**Figure 5.25:** 16-QAM constellation for both class AB (left) and class B (right) PA operation modes with memoryless DPD (test signal OFDM 16-QAM).

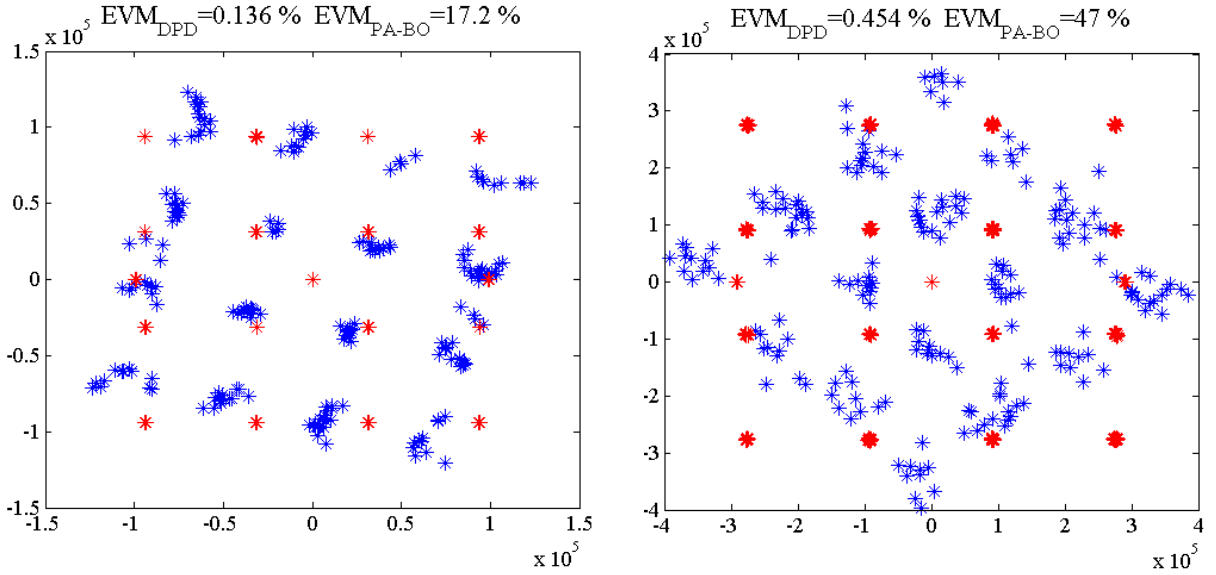


**Figure 5.26:** 16-QAM constellation for both class AB (left) and class B (right) PA operation modes with predictive NARMA DPD with 7 BPCs (test signal OFDM 16-QAM).

update iterations for both SC and OFDM M-QAM modulation schemes, respectively.

On the other hand, Fig. 5.28 and Fig. 5.30 show the evolution of the out-of-band distortion compensation in terms of ACPR, for both SC and OFDM configurations. Note that, independently on the test signal used (SC or OFDM), at certain update iterations there are some discontinuities or imbalances (marked with a red circle) in the ACPR and EVM evolution that affect both memoryless and multi-LUT DPD.

These imbalances or uncertainties that degrade the DPD reliability are inherent to the methodology used for estimating LUT contents. As explained previously, the polynomial functions that model the PA are obtained from a LS linear regression. These polynomial functions are later directly mapped into the BPC-LUTs to achieve the suitable DPD operation. However, as a consequence of the high PAPR of current signals, the peak probability is low and it is difficult to get knowledge of the PA characteristic at high amplitudes. For instance, if the data from which the polynomial coefficients are being derived does not cover all PA dynamic range, but only a certain low-input region, the LS estimation is underdetermined. That is, there is no reliable way to ensure that the PA behavior described by the polynomials is accurate beyond that low-input range. Clearly, this may result in non reliable DPD operation as soon as the input signal reaches amplitudes beyond the well estimated PA regions, for which the BPC-LUT values are not trustworthy. Therefore the PA model estimation during the adaptation/update procedures has to be somehow reengineered.

A possible solution to avoid uncertainties consists in performing a selective adaptation procedure, in which only data buffers presenting input PA values above a certain power threshold are taken into account to perform the adaptation. Otherwise, data buffers are rejected, and a new set of data buffers are recorded and so on. In such a way, the PA model functions are estimated when the stimuli are complete enough, in the sense that they cover a wide part of the PA dynamic range, thereby reducing the uncertainty and resulting in a reliable DPD operation. Besides, it is possible to dynamically adjust the threshold to tradeoff between accuracy and adaptation rate. A low threshold lowers the chances of data buffer rejection, but at the risk of underdetermination. Inversely, an excessive value for the threshold will result in a high buffer rejection rate, postponing the estimation.

Further details on the adaptation policy finally implemented will be provided and discussed in the following Chapter of this thesis.

### 5.4.3 FPGA Real-Time Adaptation Description

Another possible configuration consists in performing both predistortion and adaptation functions within the same FPGA. This is possible if we consider an iterative update of the individual predistortion complex gains ($G_{LUT}$) constituting the set of LUTs or BPCs. Therefore, taking

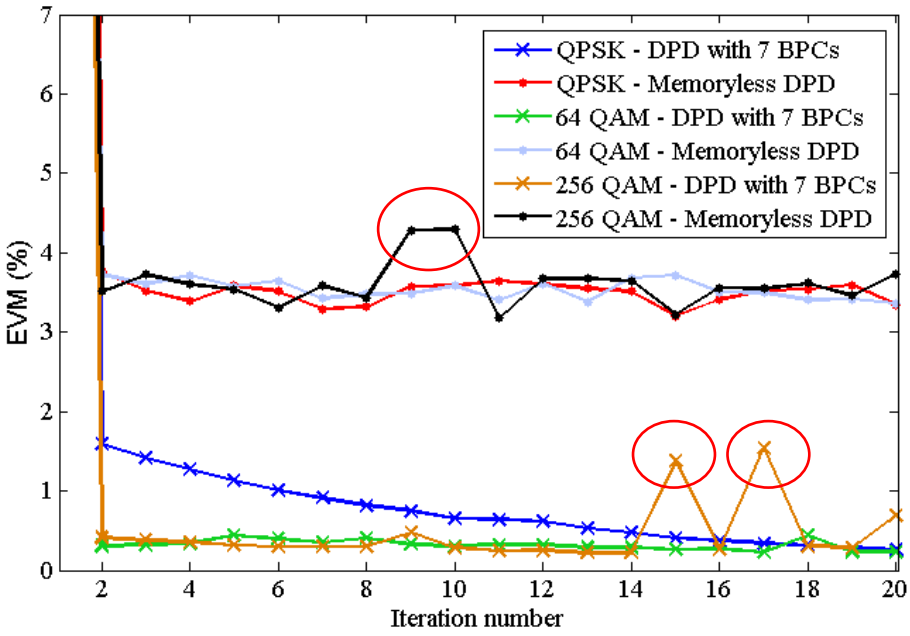


**Figure 5.27:** Evolution of the EVM for different SC M-QAM modulation schemes considering a class AB PA.

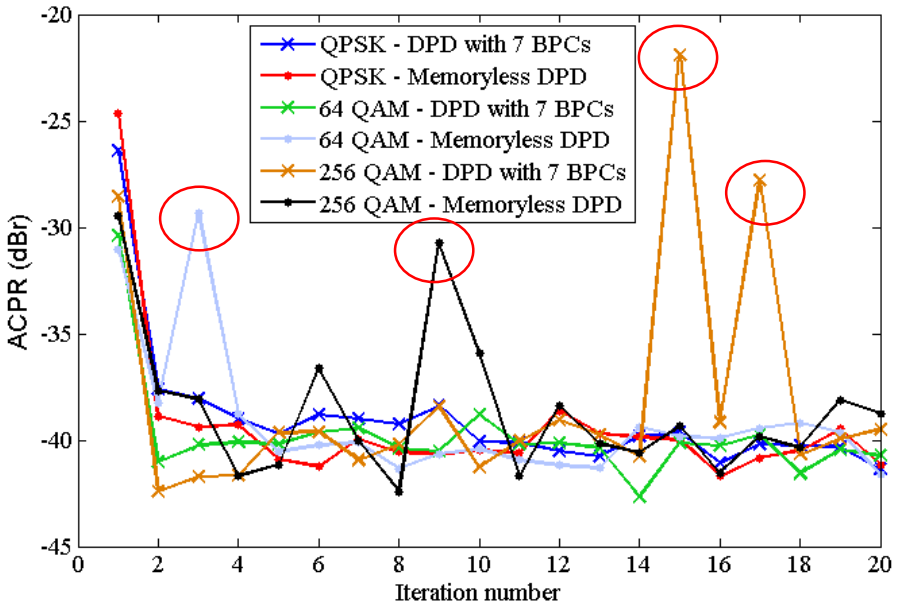


**Figure 5.28:** Evolution of the ACPR for different SC M-QAM modulation schemes considering a class AB PA.

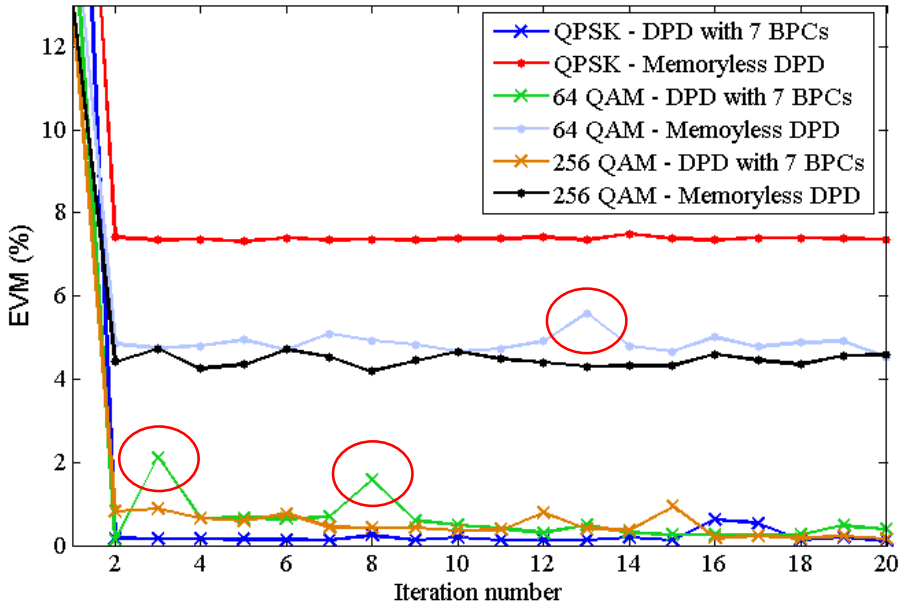


**Figure 5.29:** Evolution of the EVM for different OFDM M-QAM modulation schemes considering a class AB PA.

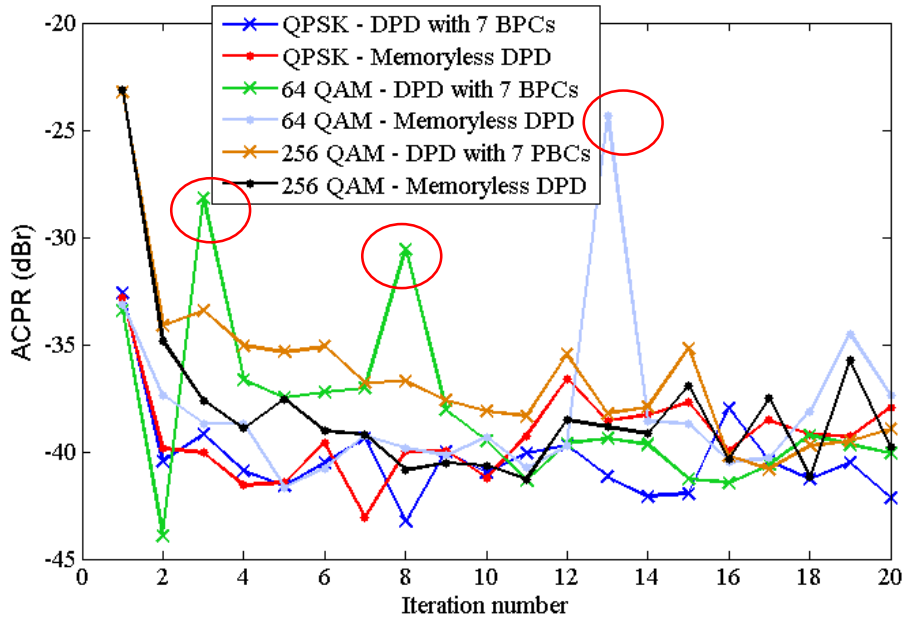


**Figure 5.30:** Evolution of the ACPR for different OFDM M-QAM modulation schemes considering a class AB PA.

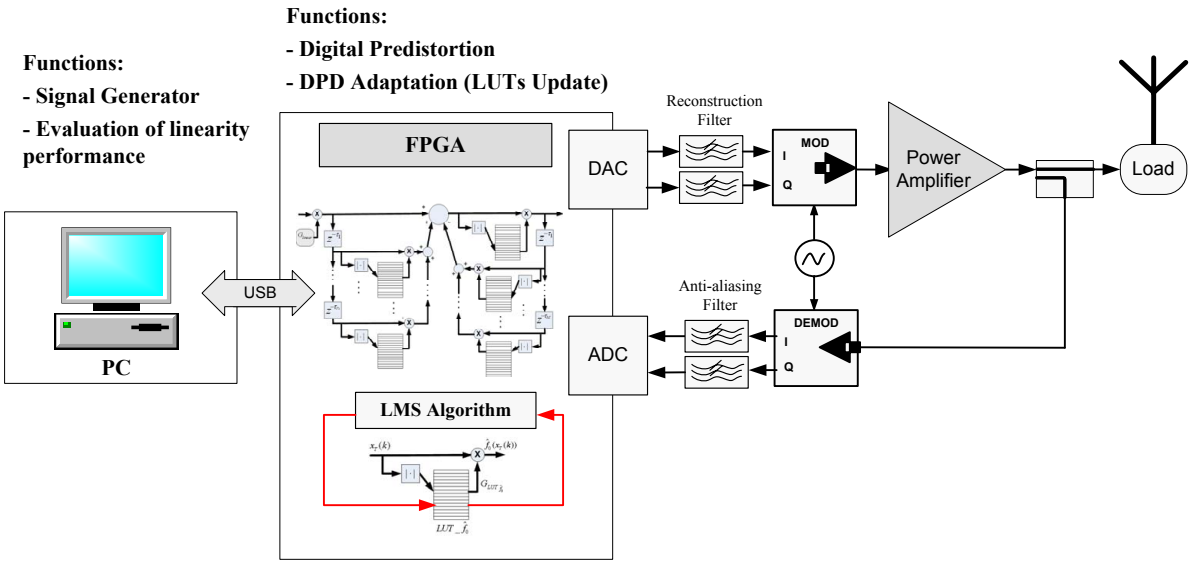


**Figure 5.31:** Block diagram of a DPD with real-time adaptation carried out in the FPGA.

advantage of the parallel computing capabilities of the FPGA, it is possible to use the least mean squares (LMS) algorithm [Hay91] to simultaneously update all gains involved (within BPCs) in a predistortion execution, leaving the rest of the complex gains of the BPCs unchanged since they have not been involved. Therefore, unlike external adaptation where all BPCs complex gains were recalculated at every adaptation execution, it is possible to perform real-time updates in an FPGA device without the need of a DSP or any other kind of advanced coprocessor for doing it. This real-time adaptation configuration is schematically shown in Fig. 5.31.

A first study of the best sparse delays defining the PA model in (5.9) and a stability test has to be carried out whatever the adaptive configuration is. Once the best sparse delays are identified and it is ensured that nonlinear functions related to the recursive part of the NARMA structure are consistent (stable), the PA and DPD NARMA based structures (see Fig. 5.11 and Fig. 5.7 respectively) are mapped into the FPGA.

Similarly to (5.21) and (5.22), the NARMA PA model in (5.9) can be expressed as set of complex products between input/output samples and their corresponding BPC complex gain:

$$\begin{aligned} y_{A\_NARMA}(k) &= x_A(k) \cdot G_{LUT-\hat{f}_0}\left(|x_A(k)|\right) + \\ &+ \sum_{i=1}^{N} x_A(k-\tau_i) \cdot G_{LUT-\hat{f}_i}\left(|x_A(k-\tau_i)|\right) - \sum_{j=1}^{D} y_A(k-\tau_j) \cdot G_{LUT-\hat{g}_j}\left(|y_A(k-\tau_j)|\right) \end{aligned} \tag{5.27}$$

Where, initially, complex gains ($G_{LUT}$) of the BPCs are filled with 0's or 1's.

The update of the complex LUT Gains is performed by means of the complex LMS algorithm,

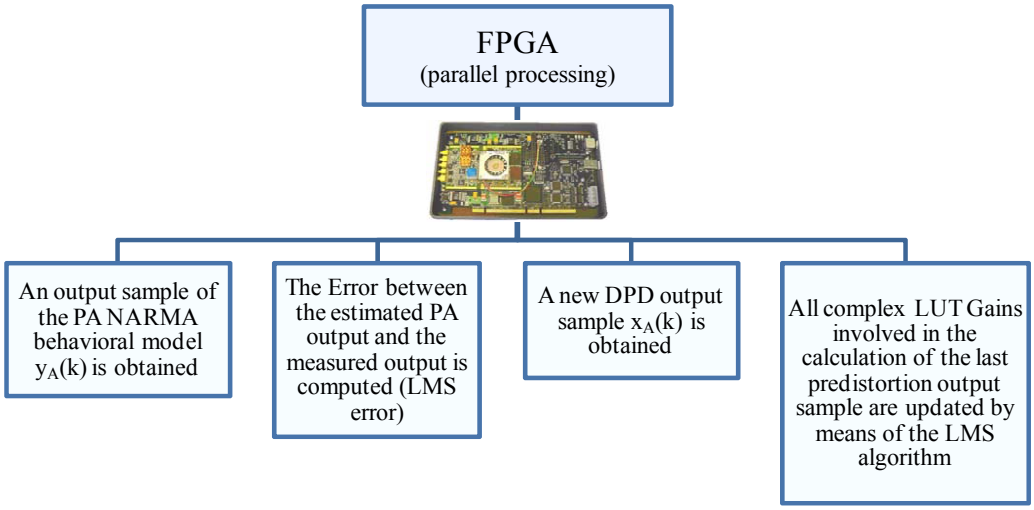


**Figure 5.32:** Flow diagram of the real-time adaptation carried out in the FPGA.

where the identification error is described by

$$e(k) = y_{A\_Measured}(k) - y_{A\_NARMA}(k) \tag{5.28}$$

and the LMS algorithm can be expresses as:

$$\begin{aligned} \Delta G_{LUT-\hat{f}_i}(|x_A(k-\tau_i)|) &= \mu_i^f \cdot x_A(k-\tau_i) \cdot \bar{e}(k) \\ \Delta G_{LUT-\hat{g}_j}(|y_A(k-\tau_j)|) &= \mu_j^g \cdot y_A(k-\tau_j) \cdot \bar{e}(k) \end{aligned} \tag{5.29}$$

where $y_{A\_Measured}$ is the measured amplifier output, $y_{A\_NARMA}$ is the PA NARMA model output, $\bar{e}(k)$ is the conjugate of the complex error and $\mu$ ($\mu_i^f$, $\mu_j^g$) is the step size that determines the learning rate of the LMS algorithm.

Therefore, as depicted in Fig. 5.32, at every iteration step the following actions are performed simultaneously (parallel processing) in the FPGA:

- Applying the algorithm defined in (5.27), an output sample of the PA NARMA model ($y_{A\_NARMA}(k)$) is obtained in order to create the LMS error in (5.28).
- Applying the algorithm defined in (5.21) a new DPD output sample ($x_A(k)$) is obtained.
- All complex LUT Gains involved in the calculation of the last predistortion output sample are updated by means of the LMS algorithm in (5.29).

After a transient period, in which all LUT Gains are continuously being updated, the PA output converges to the desired output, achieving then the desired linear amplification.

### 5.4.4 FPGA Real Time Adaptation Simulation Results

The LMS based adaptation in the FPGA has been simulated in Matlab in a similar manner as the external adaptation solution presented before. However, while the external adaptation has been implemented providing experimental results that will be exposed and discussed in next Chapter, the implementation of the real-time FPGA based adaptation remains as future work to be done.

It is feasible to implement the NARMA based Multi-LUT structure and its real-time adaptation in a commercial FPGA board as the Nallatech XtremeDSP, consisting in a Xilinx Virtex XCV4SX35 connected to two analog-to-digital and two digital-to-analog converters running at a clock of 105 MHz. Each LUT correspond to an addressable memory table which contains 512 addresses-gains, and as explained previously, every time that the content of a LUT is addressed their content will be updated.

For testing and debugging purposes the developed predictive NARMA DPD has been assessed in Matlab, making use, as in the external adaptation simulations, of a Hammerstein PA model based in a Freescale PA (MRF21170 MOSFET transistor). The actual NARMA based predictive DPD is formed by 3 FIR, 3 IIR terms and 2 additional LUTs, related to $f_0$ and $f_0^{-1}$ nonlinear functions. A total of 8 LUTs of size $2^9 = 512$ addresses corresponds to a total of 4096 complex Gains. Therefore, with a sample rate of 10 MSamples/second, a complex Gain update is performed each $10^{-7}$ seconds.

Now, assuming an access to a LUT uniformly distributed, the update of a particular Gain can be executed approximately every $512 \cdot 10^{-7}$ seconds. That corresponds to 512 iterations, that is, 512 data samples. Nevertheless, the distribution of the access to a LUT follows the input signal probability density function. For that reason, in order to ensure that all LUTs have been updated at least once, a training signal is used during 5000 iterations. That represents 10 times the minimum number of required iterations to update all LUT gains in the case of an access to a LUT uniformly distributed.

Figure 5.33 shows the convergence of both identification and linearization NMSE, defined as the error resulting from subtracting the output of the NARMA PA model ($y_{NARMA}$) to the measured output ($y_{out-PA}$); and as the the error resulting from subtracting the desired output ($y_{desired}$) to the measured output ($y_{out-PA}$), respectively.

Once LUTs are trained, two types of test signals have been considered: a SC 16-QAM and an OFDM (256 subcarriers) 16-QAM modulated signal. The PA model presents a nonlinear distortion typical of a class AB PA.

After a short transient period, this multi-LUT DPD with real-time adaptation achieves a steady state where it is possible to observe the achieved linearization performance. For instance,

Fig. 5.34 shows the linearized AM-AM characteristic and dispersion compensation for both SC and OFDM configurations in steady state. In addition, Fig. 5.35 and Fig. 5.36 show the out-of-band and in-band nonlinear dynamic compensation achieved by the DPD with 7 BPCs, stated in terms of ACPR and EVM respectively. Since real-time adaptation is continuously updating (sample by sample) LUT contents, problems related with possible discontinuities or uncertainties, typical of the aforementioned LS based external adaptation, are completely avoided. This can be observed in Fig. 5.37 and Fig. 5.38, presenting the steady state evolution of the ACPR for both SC and OFDM 16-QAM signals. Complementary Fig. 5.39 and Fig. 5.40 show the steady state evolution of the EVM for both SC and OFDM test signals, respectively.

In order to compare the linearization performance achieved with both real-time and external adaptive configurations, the mean values of ACPR and EVM have been also computed every 2048 samples, which correspond to the size of the buffers that are handled in the external adaptive configuration. Compared to the external adaptation, the real-time adaptation performed in the same FPGA already used for implementing the DPD function, has the advantage of being more robust in front of possible uncertainties derived from a bad LS identification. On the other hand, as we can observe in Fig. 5.38, the variance in the ACPR values is significant, thus, in order to reduce this variance one might consider the use of alternative gradient techniques, such as the Fast Kalman algorithm [Hay91].

## 5.5 Summary

In this Chapter design and implementation aspects related to Digital Predistortion have been presented and discussed. In a first approach some basic issues regarding LUT organization have been presented to finally introduce the particular atomic LUT structures, called BPCs, that will conform the DPD function when it is mapped into a FPGA.

In addition, a new predictive approach to identify the DPD function has been presented, overcoming then the classical indirect learning approach that assumes the commutative property for nonlinear systems. The DPD function has been particularized using a NARMA scheme, extensively presented in Chapter 4. It has been shown how it is possible to derive a NARMA structure, based in the contribution of parallel nonlinear functions associated to delayed samples of the input and output signal, into a set of BPCs ready to be mapped into a FPGA.

In order to obtain some results allowing a preliminary evaluation of the nonlinear and memory effects compensation achieved by our proposed DPD, a Matlab simulator emulating the FPGA functioning has been used. Two different approaches for updating all BPCs in the FPGA have been considered. One performs LUTs update in an external DSP while the alternative approach consists in taking advantage of FPGA parallel processing capabilities to perform the adaptation process in the same FPGA responsible for the real-time DPD functioning. In the

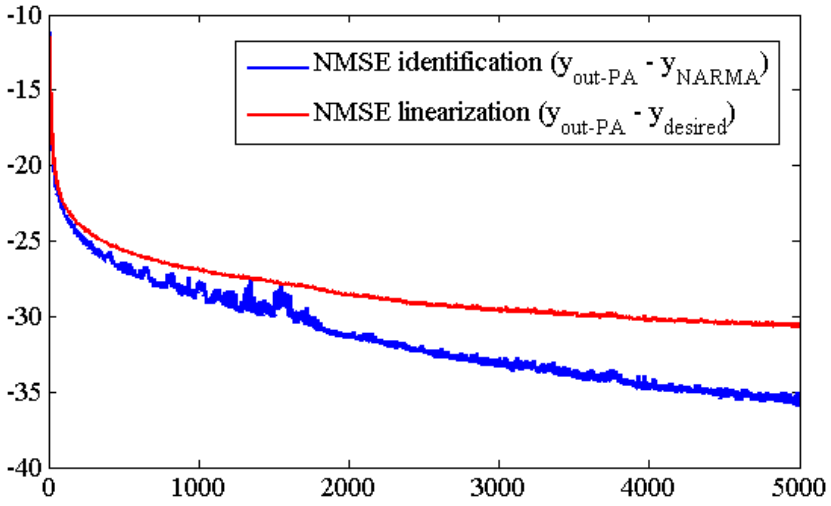


**Figure 5.33:** Identification and linearization NMSE versus number of iterations.

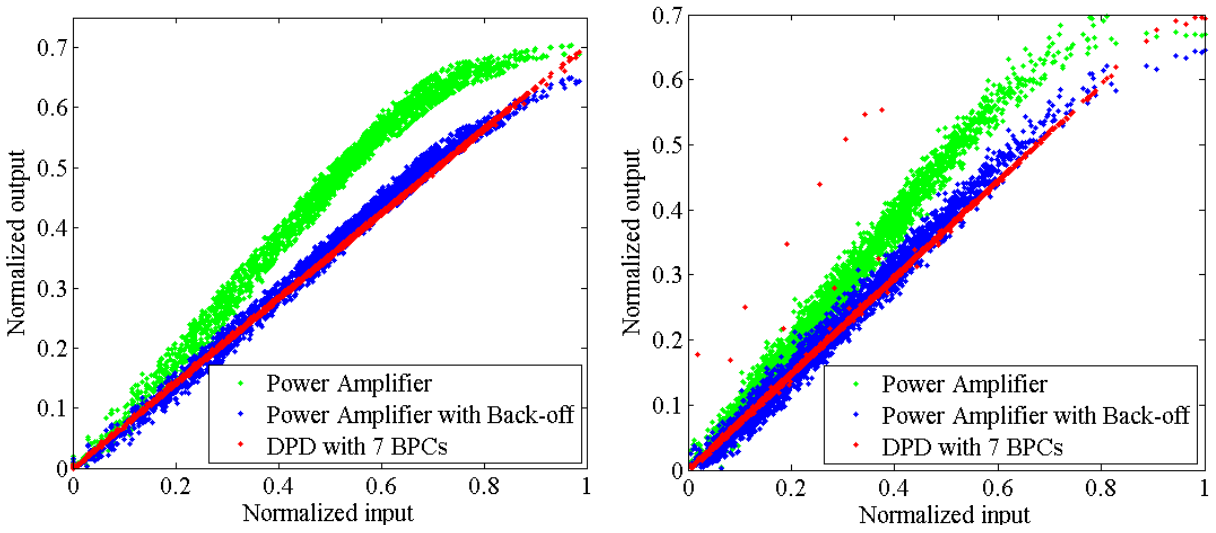


**Figure 5.34:** AM-AM characteristics for a SC (left) and OFDM (right) 16 QAM modulated signal considering Back-off operation and DPD with 7 BPCs for a class AB PA.

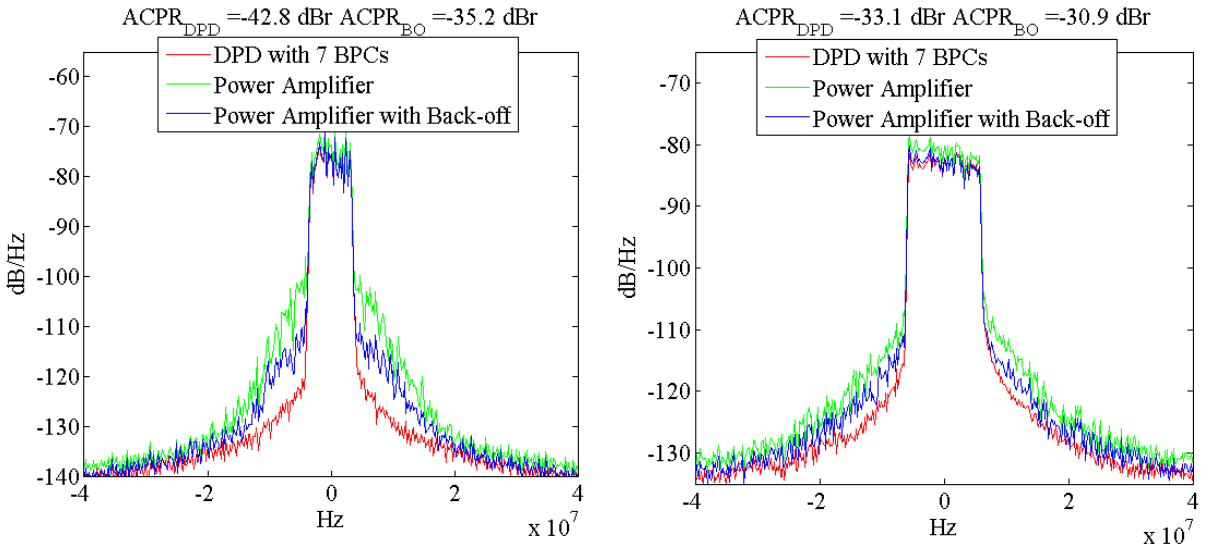


**Figure 5.35:** Power spectra for a SC (left) and OFDM (right) 16 QAM modulated signal considering Back-off operation and DPD with 7 BPCs for a class AB PA.

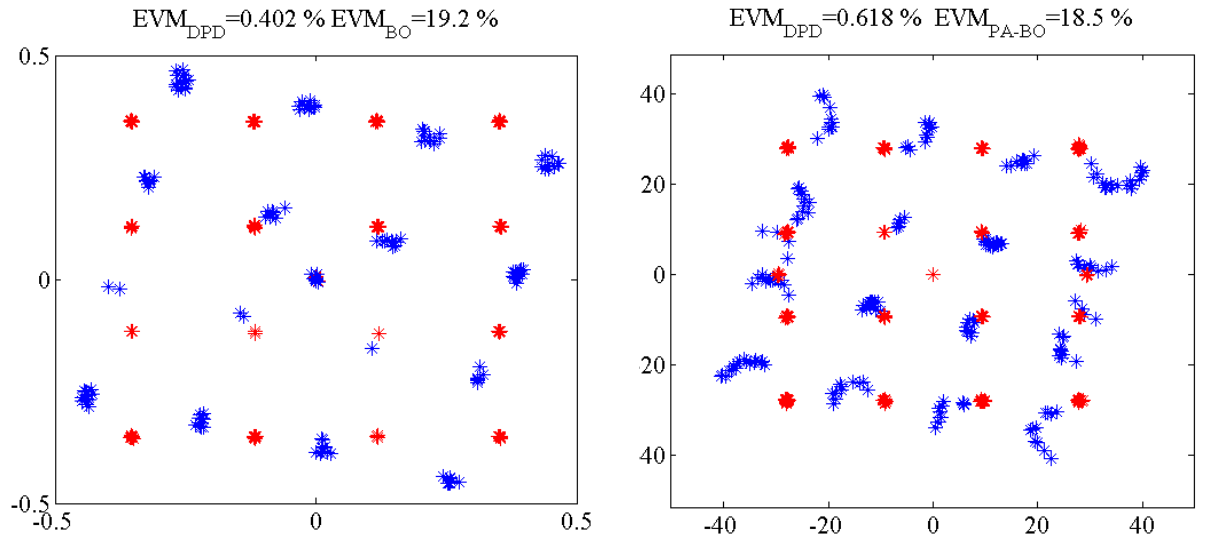


**Figure 5.36:** Constellation of a SC (left) and OFDM (right) 16 QAM modulated signal considering Back-off operation and DPD with 7 BPCs for a class AB PA.

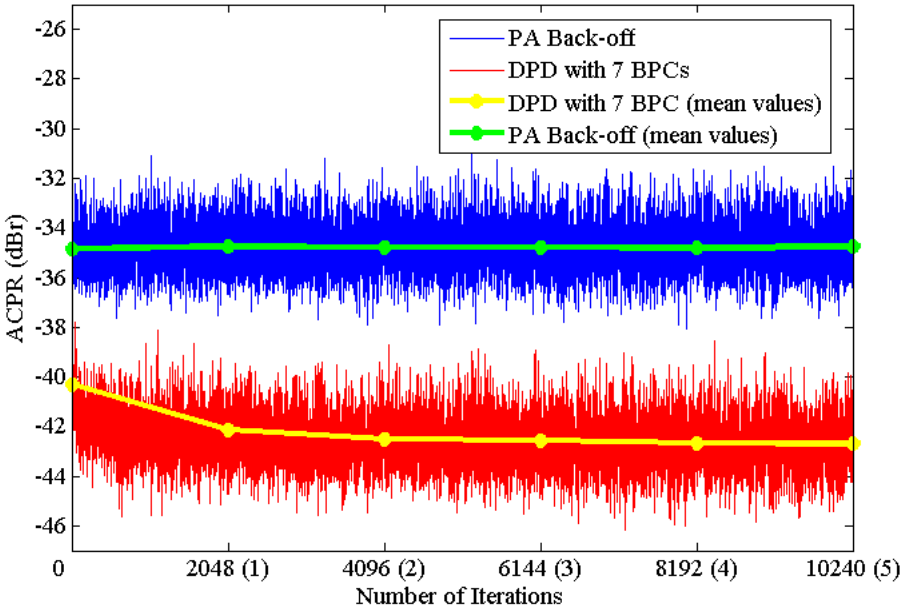


**Figure 5.37:** Steady state evolution of the ACPR for a SC 16-QAM modulation considering a class AB PA.

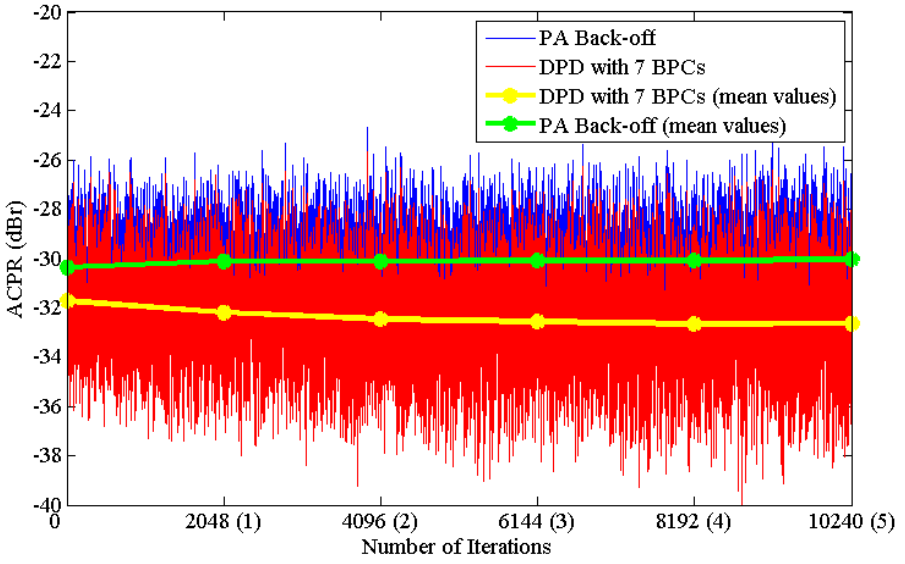


**Figure 5.38:** Evolution of the ACPR for an OFDM 16-QAM modulation considering a class AB PA.

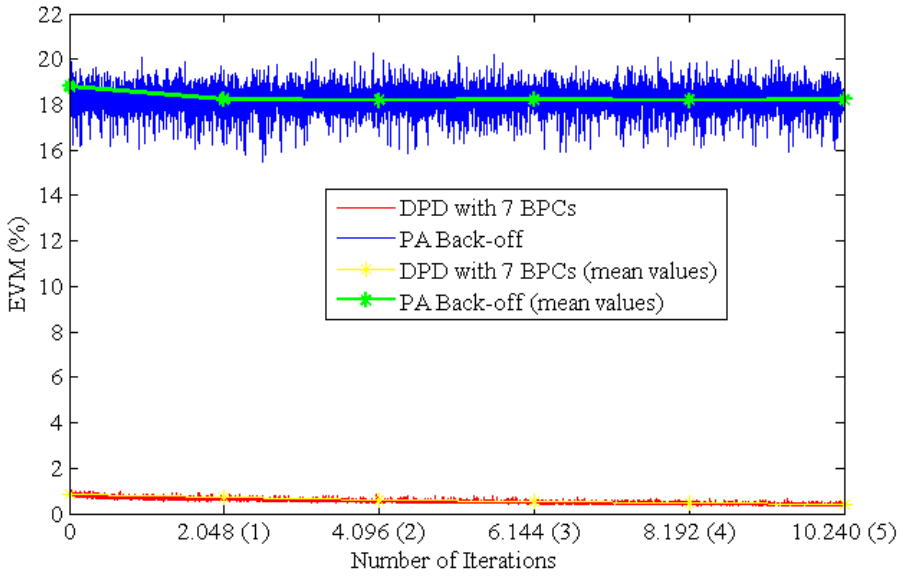


**Figure 5.39:** Evolution of the EVM for a SC 16-QAM modulation considering a class AB PA.

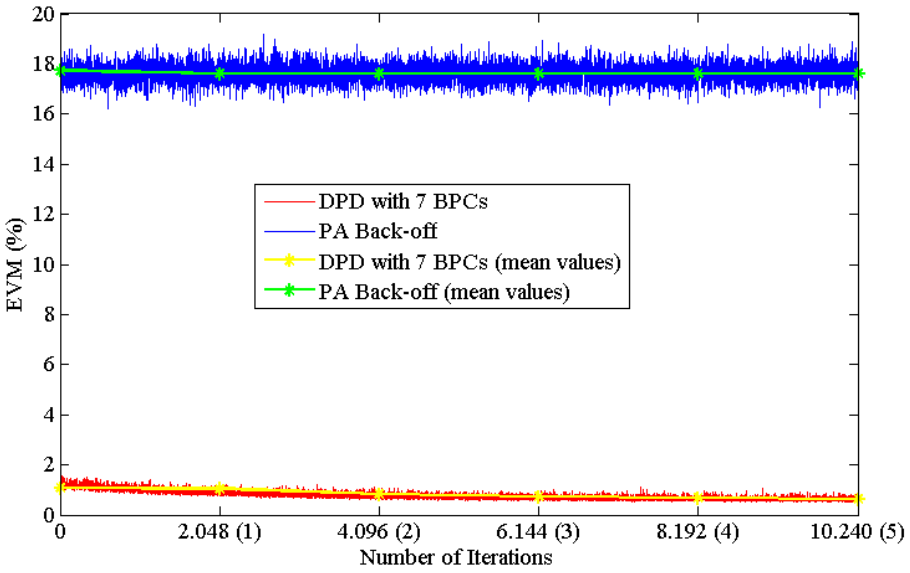


**Figure 5.40:** Evolution of the EVM for an OFDM 16-QAM modulation considering a class AB PA.

external adaptation approach the adaptation is performed independently of the real-time DPD functioning in the FPGA and complex gains of all LUTs are completely updated at every adaption process. This may cause some continuity problems as it has been pointed out in simulation results, making reconsider the inclusion of an adaptation policy aimed at dealing with this continuity problem derived from LUTs update. On the other hand, real time adaptation can be performed in the same FPGA in charge of the DPD, by performing parallel updates of the particular LUT complex gains involved in a DPD iteration. This approach may overcome the continuity problems derived from a non real-time external adaptation.

Despite simulation results have been obtained in a noise-free Matlab environment that also obviates digital signals quantization (no resolution limitations), some preliminary conclusion can be drawn concerning the linearity performance achieved with our proposed DPD. Both memoryless and DPD with dynamic compensation (more than 1 BPC in the multi-LUT based DPD structure) enhance linearity when they are fairly compared with a backed-off signal without DPD presenting the same output power. This nonlinear compensation can be appreciated in the straightening of the AM-AM characteristic, the spectral regrowth reduction and the rotation correction of signal constellation, becoming more evident when considering a class B mode of operation. In addition, the advantages of using a DPD with dynamic compensation are highlighted in the time domain, showing a scattering reduction in both AM-AM characteristics and signal constellation that permits reducing EVM figures by a factor of 5, approximately.

Therefore, after this encouraging preliminary simulation results, next Chapter of this thesis will be focused on an experimental evaluation of the real distortion compensation capabilities that can be achieved by our proposed DPD.