Adaptive Predistortion Using the ISL5239

AN1028
Rev.0.00
September 2002

Introduction

This application note describes adaptive predistortion using the ISL5239 Predistortion Linearizer Evaluation Board. A Matlab algorithm is presented and validated using a hardware-in-the-loop simulation where the Power Amplifier (PA) is modeled in Matlab and the Predistortion (PD) function is implemented using the ISL5239 Evaluation Board Hardware. Next, the algorithm is used with a laboratory setup to linearize an amplifier. The transmit waveform is four-carriers of UMTS comprising a 20MHz signal bandwidth. The adaptive algorithm is used with a closed-loop hardware architecture to improve Adjacent Channel Leakage Ratio (ACLR) by 15dB, meeting the 3GPP ACLR specification with 12% efficiency.

It is important to note that while a basic algorithm is provided as an example, users are free to add enhancements and develop their own algorithms. This allows users to differentiate their products from their competition.

This application note is organized as follows. The section “Linearization Fundamentals” presents linearization fundamentals and defines the PA model used in the section “Hardware-in-the-Loop Simulation”. We describe the adaptive algorithm and present results from the “Hardware-in-the-Loop Simulation” section on page 3. In the “Laboratory Results” section on page 8, we describe the closed-loop hardware architecture and present laboratory results. A summary is given in the Conclusions section.

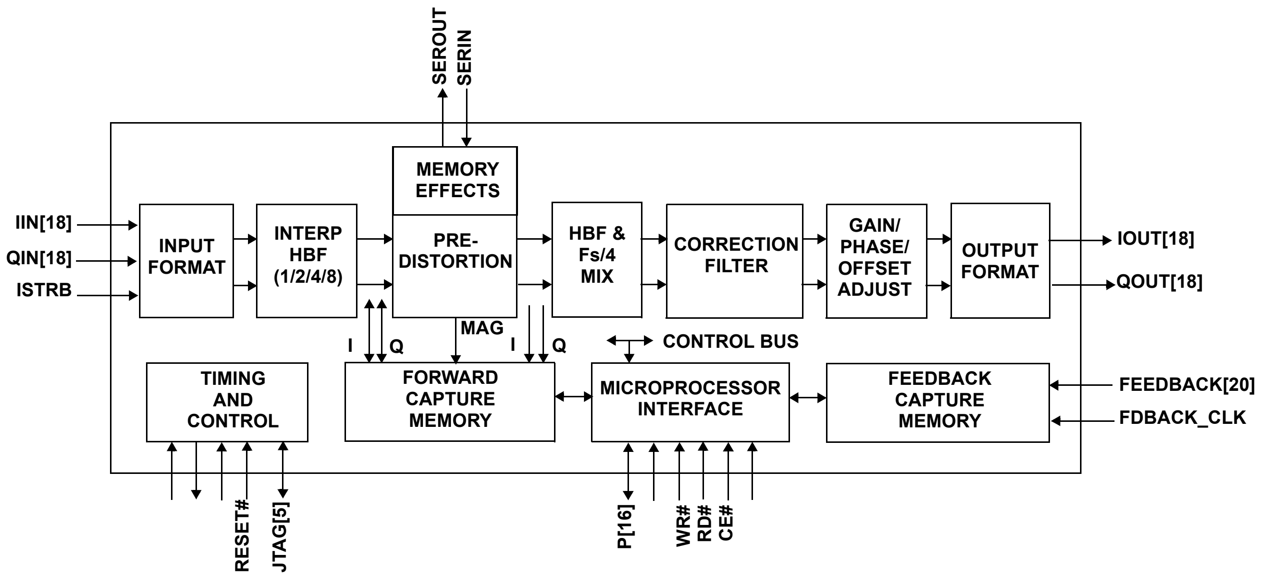

FIGURE 1. TOP-LEVEL BLOCK DIAGRAM OF ISL5239

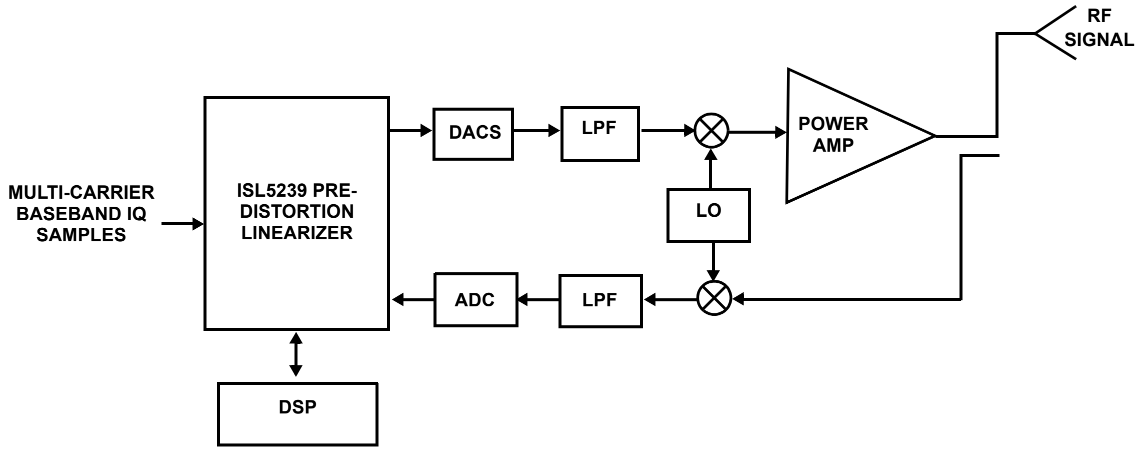
RF SIGNAL
DACS
LPF
POWER AMP
MULTI-CARRIER BASEBAND IQ SAMPLES
ISL5239 PRE-DISTORTION LINEARIZER
LO
ADC
LPF
DSP

FIGURE 2. CLOSED-LOOP DIGITAL PREDISTORTION USING THE ISL5239

Linearization Fundamentals

While the ISL5239 includes features for memory compensation, this application note considers the linearization of power amplifiers that can be modeled by a memoryless nonlinearity. Figure 3 on page 2 shows the input-output block diagram of a power amplifier where $v_i(t)$ and $v_o(t)$ are the input and output signals respectively. We consider the complex baseband model in which the output can be expressed as:

$$v_o(t) = g(|v_i(t)|^2)\, v_i(t) = g_a(|v_i(t)|^2)\, e^{jg\phi(|v_i(t)|^2)}\, v_i(t) \qquad \text{(EQ. 1)}$$

where $g_a(|v_i(t)|^2)$ and $g^{\phi}\,(|v_i(t)|^2)$ are the amplifier's amplitude and phase transfer characteristics, commonly referred to as AM-to-AM and AM-to-PM characteristics.

The transfer characteristics can be measured by applying an input pulse and measuring the output amplitude and phase. The amplitude and phase values can be plotted as a function of input power as shown in Figure 4A and Figure 4B. The PA characteristic curves shown in these figures are typical of actual AM-AM and AM-PM measurements, but this data was generated from the polynomial model defined by

$$g_a(|v_i(t)|^2) = G(1 - 1|v_i(t)|^2) \qquad \text{(EQ.2)}$$

where G corresponds to a gain of 35dB and

$$g_\phi(|v_i(t)|^2) = -0.3|v_i(t)|^2 \qquad \text{(EQ.3)}$$

The amplifier's linear characteristic if defined by

$$\bar{g}_\alpha(|v_i(t)|^2) = G$$

and

$$\bar{g}_\phi(|v_i(t)|^2) = 0$$

The linear region is defined as that set of inputs for which $g_\alpha \cong \bar{g}_\alpha$ and $g_\phi \cong \bar{g}_\phi$. From the data in Figure 4, the upper limit of the linear region falls between 5 and 10dBm. From the linear region of Figure 4A, it is evident that the amplifier exhibits a 35dB gain. At higher input power levels, the amplifier cannot sustain this gain. The 1dB compression point is defined as the input power at which the amplifier's output power is 1dB below the linear response. In this case, the 1dB compression point occurs at 20dBm; that is, with the input power at 20dBm the output power is 54m, which is 1dB less than the 55dBm output power required for linear operation. The saturation point corresponds to the input level that results in the largest output. In this case, the saturation point is at 25dBm.

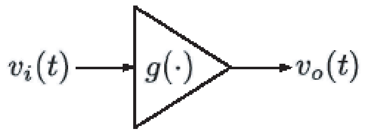

FIGURE 3. POWER AMPLIFIER INPUT-OUTPUT DIAGRAM

A predistortor preceding the amplifier can be used to linearize the amplifier. Figure 5 shows the PD-PA cascade where $v_d(t)$ is the PD output. Expanding Equation 1 gives the PA output expression

$$v_o(t) = g\left(\left|f(|v_i(t)|^2)v_i(t)\right|^2\right) f(|v_i(t)|^2)\, v_i(t)$$

Clearly, the PA is linearized when

$$G = g\left(\left|f(|v_i(t)|^2)v_i(t)\right|^2\right) f(|v_i(t)|^2)$$

For input levels in the linear range of the amplifier, the predistortor applies a gain of 0dBm and has no effect on the signal. In the nonlinear range, the predistortor applies a gain to either amplify or attenuate the input signal. The value of the gain depends on the input level. For example, with an input of 20dBm, the predistortor applies a gain of approximately 2dB.

The amplifier input is now 22dBm, which results in the desired 55dBm output. The saturation point defines the upper limit for linearization.

Although sometimes overlooked, the predistortor must also linearize the amplifier's phase response. Figure 4B shows that the phase distortion is approximately 2 degrees at the 1dB compression point, and nearly 6 degrees at saturation. The predistortor introduces a phase shift that is equal and opposite to that of the amplifier. In the next section, we describe an adaptive algorithm that generates the predistortion characteristic $f(|v_i(t)|^2)$ to linearize a PA modeled by $g(|v_i(t)|^2)$ defined by Equation 2 and Equation 3.

FIGURE 4. CHARACTERIZATION OF A MEMORYLESS NONLINEARITY

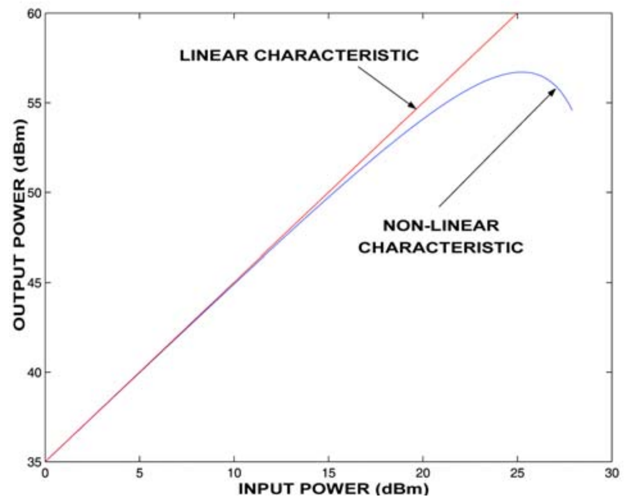

FIGURE 4A. AM-AM CHARACTERISTICS

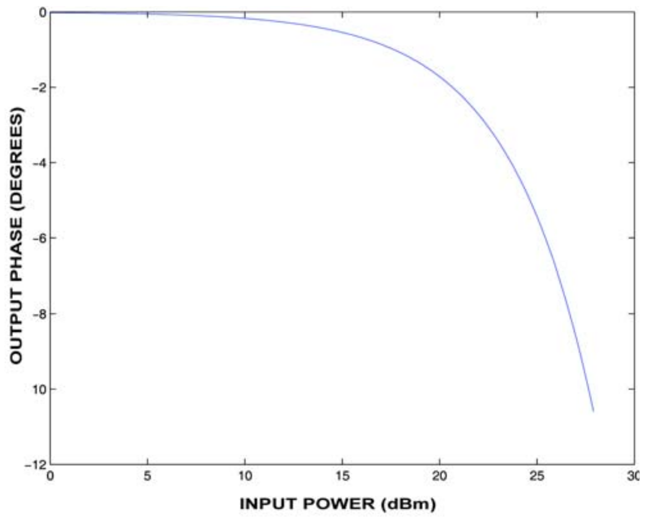

FIGURE 4B. AM-PM CHARACTERISTICS

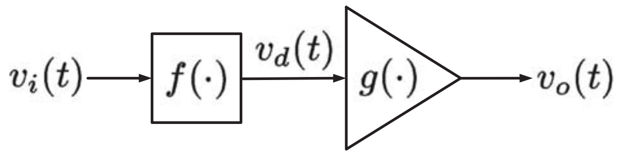

FIGURE 5. POWER AMPLIFIER INPUT-OUTPUT DIAGRAM

Hardware-in-the-Loop Simulation

In this section we describe the hardware-in-the-loop simulation that is included with the ISL5239 Matlab Interface Library software. Figure 6 shows a block diagram of the simulation functional blocks. The ISL5239 Evaluation Board is used as the predistortion engine while Matlab is used to run the adaptive algorithm and simulate the PA. The ISL5239 hardware is controlled through the USB interface. Full programmatic control of the hardware is provided by the Matlab Interface Library. Noise is added to the PDL output as shown in the figure to simulate a DAC and to the feedback to simulate an ADC.

The hardware-in-the-loop simulation requires Matlab with the Signal Processing Toolbox. Also required are the Matlab Interface Library for the ISL5239 and the ISL5239 Matlab Tools found in the folder “C:/ISL5239/MATLAB TOOLS AND INTERFACE LIBRARY”. The demo routines are described in Table 1. The adapt_LUT.m routine implements the adaptive algorithm and has been applied in laboratory experiments to linearize a real-world PA with a dual 10-bit ADC in the feedback path with performance similar to that shown by the simulation.

TABLE 1. SIMULATION FILES (SEE FOLDER “C:/ISL5239/MATLAB TOOLS AND INTERFACE LIBRARY/CLOSED-LOOP DEMO”)

FILENAME	DESCRIPTION
demo.m	Main routine for closed-loop demo
mkPA_model.m	Defines coefficients for simulated PA model
configHW.m	Reset and configure the ISL5239 hardware
runPD.m	Controls I/O to ISL5239 hardware
adapt_LUT.m	Adaptive algorithm to generate new LUT

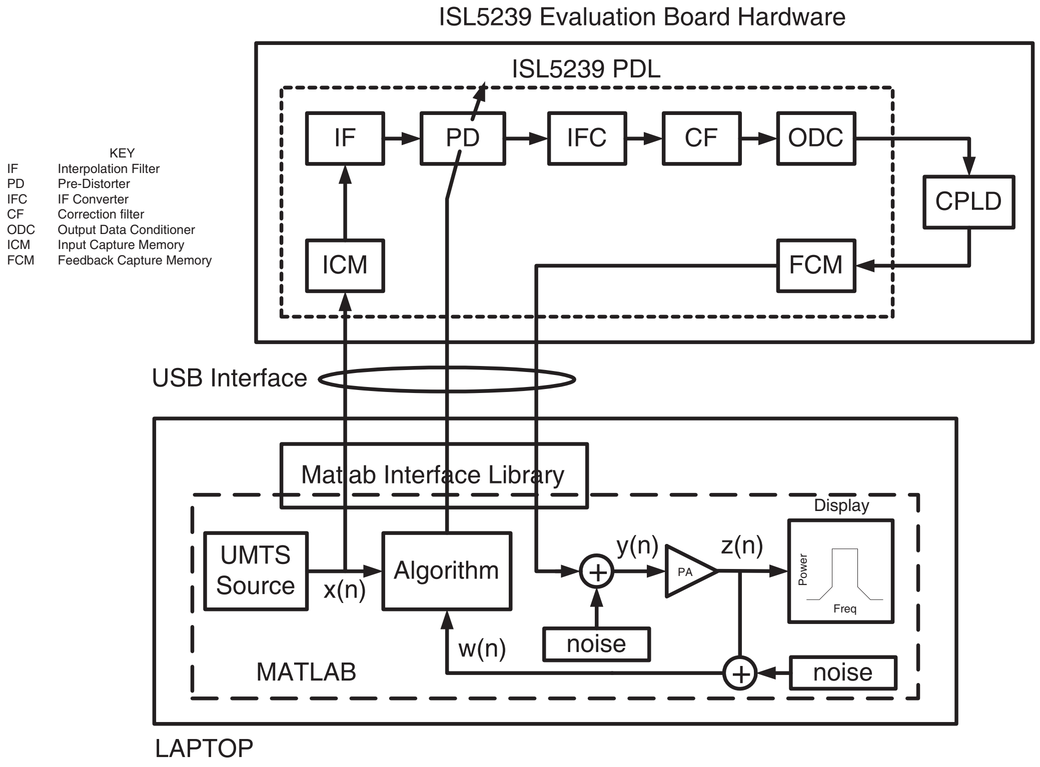

FIGURE 6. HARDWARE IN-THE-LOOP SIMULATION

Running the Simulation

Make sure that the folder “C:/ISL5239/MATLAB TOOLS AND INTERFACE LIBRARY” is included in the Matlab path. In Matlab, set the working folder to “C:/ISL5239/MATLAB TOOLS AND INTERFACE LIBRARY/CLOSED-LOOP DEMO”. Then at the Matlab prompt simply type “demo.”

Figure 7 shows the output after 10 iterations of the algorithm. The upper four axes show the signals x(n), y(n), z(n), and w(n) as identified in Figure 6. The PDL input x(n) is a four-carrier UMTS waveform shown on the top left. The PDL output y(n) is shown on the top right.

This spectra shows the PDL output after DAC quantization noise is added. The PA output z(n) is shown in the middle right. The red curve shows the output without predistortion. The feedback signal w(n) is shown middle left. This signal includes ADC quantization noise. Notice that the feedback signal has a higher noise floor than the PA output. The integration time constants in the adaptive algorithm are sufficient to tolerate a noisy ADC on the feedback path. The bottom left and right axes show the LUT gain and phase values along with scatter plots of the gradient updates used by the algorithm. The simulation can be stopped by clicking on the “running” menu at the top of the figure. Also, the PD function can be disabled by clicking on the “PD on” menu. Next we describe the details of the software simulation and algorithm.

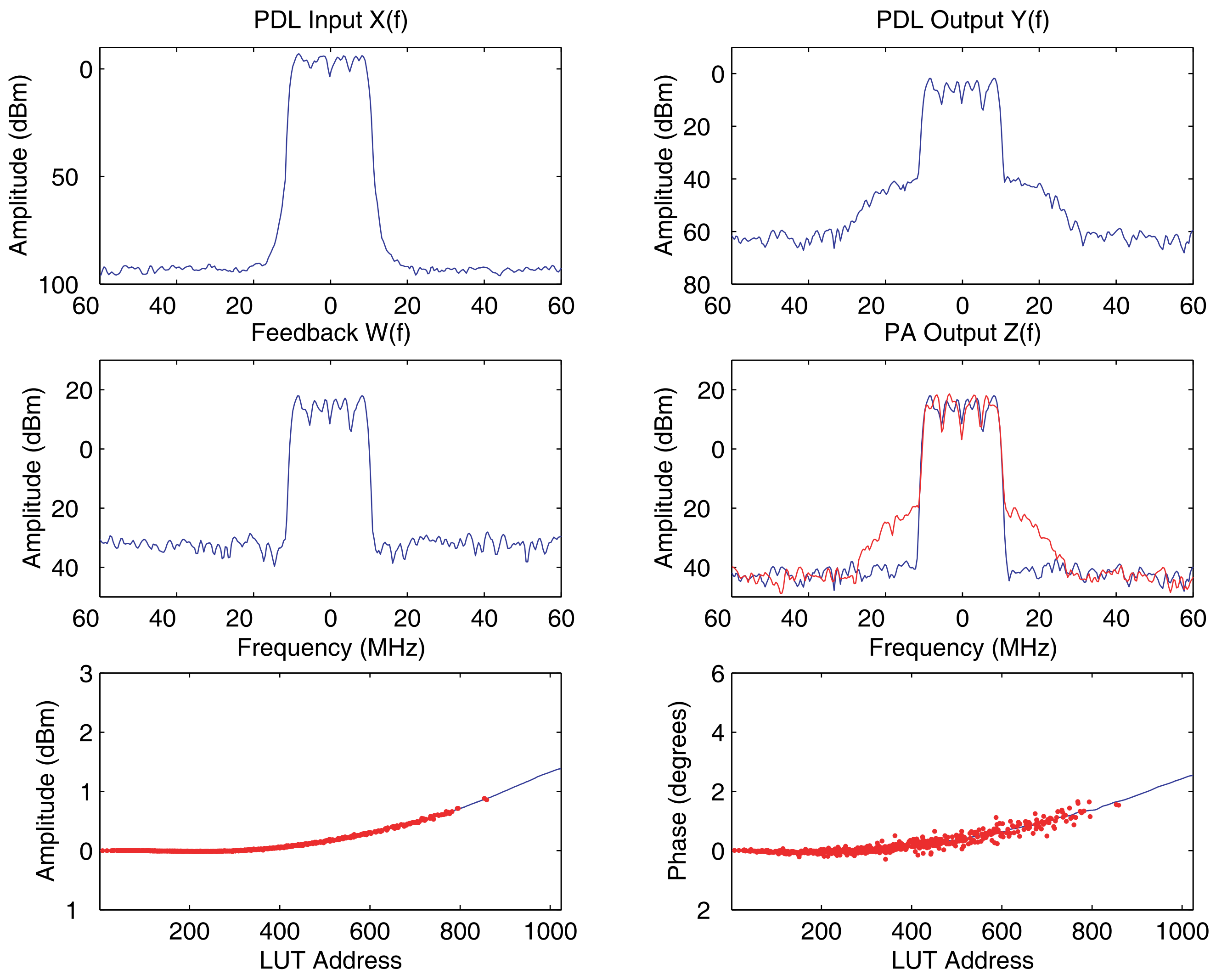

FIGURE 7. HARDWARE-IN-THE-LOOP SIMULATION

Demo.m

The main file in the simulation is demo.m. This file includes the core loop shown below:

```
while (running)
i = i+1;
k = rem(i-1,2)+1;
% Form PD input
x(k,:) = (data(1:2:end,rem(i,10)+1) +
j*data(2:2:end,rem(i,10)+1)).’;
% Pre-distort using ISL5239 Evaluation Board
Hardware
y(k,:) = runPD(x(k,:));
% Add DAC quantization noise
y(k,:) = y(k,:) + randn(1,1024)*10.^(-
65/20)+j*randn(1,1024)*10.^(-65/20);
% Scale for proper dBm level
y(k,:) = 10.^(-5.0756/20) *y(k,:);
% Run signal through PA model
z(k,:) = pa_model(1,y(k,:),0).’;
% Add ADC quantization noise to feedback
w(k,:) = z(k,:) + randn(1,1024)*10.^(-
40/20)+j*randn(1,1024)*10.^(-40/20);
% Scale feedback
w1(k,:) =
[w(k,:)/max(abs(w(k,:)))*max(abs(x(k,:)))
zeros(1,1024)];
% Adapt the LUT
updateLUT(k,x,w1,i)
% PD on/off control
PD_on;
% Update plots
updatefig(i,k,h,w,x,y,z);
drawnow
i
end
```

The variables x (k, n), y (k, n), z (k, n), and w (k, n) correspond to those identified in Figure 6 and Figure 7. The command y (k,:) = runPD (x (k, :)); returns values less than 1.0 in magnitude so that the scaling implemented by the command y (k,:) = 10. A ^(-5.0756/20) *y (k, :) ; results in values that have a peak around 25dBm, which is near the saturation point shown in Figure 4A. The PA is simulated by the command z (k, :) = pa_model (1, y (k, :) ,0) .' ; and the PD LUT is updated by the command updateLUT (k, x, w1, i). In the sections that follow, we describe the PA model and the adaptive algorithm.

PA Model

Toward the top of the file demo.m appear the commands

```
fprintf('Constructing PA model \n');
mkPA-model
%Configure the PA_model
pa_model (0, ' simpa.pap' ) ;
```

The Matlab code shown below is taken from mkPA_model and uses PA_poly.m and mkPATable.m to generate the PA model defined by Equation 2 and Equation 3. PA_poly takes input arrays of polynomial coefficients and PA input voltage and generates an array defining the PA output voltage. The routine mkPATable takes the PA input and output voltage arrays and constructs the simPA.pap file used by the function pa_model.

```
% MODEL COEFFICIENTS
% amplitude
a(1,1) = 10.0;
a(1, 2) =10.0;
a(1, 3) = 0.0;
a(1, 4) = 0.0;
a(1, 5) = 0.0;
% phase
b(1 ,1) = 0.0;
b(1 , 2) = 0.3;
b(1 , 3) = 0.0;
b(1 , 4) = 0.0;
b(1,5) = 0.0;
% PA INPUT VOLTAGE
dB_step = 0.1;
dBm_start = -35;
dBm_stop = 28.4;
L = (dBm_stop - dBm_start)/dB-step;
dBm = dBm_start;
for i = 1:L
Vi_PA.._model(i) = sqrt (1/1000*10. ^(dBm/10);
dBm = dBm + dB-step;
end
% PA OUTPUT VOLTAGE
[Vo~PA_model] = PA_poly(Vi_PA_model,a,b,l);
% P~ LUT
mkPATable(Vi_PA_model,Vo_PA_model,'SimPA.pap')
;
```

The function pa_model is used to calculate the PA output and is implemented by the file pa_model.dll. The syntax for its use is:

```
pa_model (0, filename)
z(k, :) = pa_model(l,y(k, :),temperature)
pa_model(2);
```

The first argument can take on values 0, 1, or 2. The value 0 indicates initialization mode. In this mode the second argument is the filename of the file containing the PA characteristic data. When the mode value is I, the second argument is the PA input and the third is the normalized PA temperature, which can take values between 0 and I. The PA output is returned in this mode. The last mode is used to release persistent memory allocated at initialization. This mode is used at the close of the demo.m routine.

Adaptive Algorithm

The adaptive algorithm is implemented by the file adapt_LUT.m. The algorithm is a variation on that presented in [3]. This routine is called with the syntax:

```
newLUT =
adapt_lut(x,w,oldLUT,iteration_index);
```

The routine takes the PD input, the feedback, old PD LUT, and iteration index and generates a new PD LUT. The first step in the process involves aligning the x and y data in time using the commands:

```
x = interp(x,InterpFactor);
y = interp(y,InterpFactor);
corrData = xcorr(abs(x),abs(y));
```

Second, the data is aligned in magnitude such that the mean-square error is minimized for those samples below a threshold.

```
k = 0;
for idx = start_index:1024
if abs(ydata(idx))<threshold
k = k+1;
tmpy(k) = abs(ydata(idx));
tmpx(k) = abs(xdata(idx));
end
end
tmp1 = tmpx*tmpx.';
tmp2 = tmpx*tmpy.';
A = tmp2/tmp1;
ydata = ydata/A;
```

For each input sample we calculate the corresponding LUT address using calc_addr below.

```
k = zeros(1024,1);
for idx = 1:1024
k(idx) = ca1c_addr(real(xdata(idx))*2.^4,
imag(xdata(idx))*2.4,1,256,0)+1;
end
```

The syntax for the calc_addr function is:

```
address =
calc_addr(I_ch_value,Q_ch_value,address_mode,s
cale, and offset)
```

The I- and Q-channel values are fixed point values between -2^{19} and 2^{19} -1. The address mode can be 0 for log mode, 1 for linear voltage, and 2 for linear power. The scale and offset values are described in the ISL5239 datasheet.

The error between the input and output samples is calculated and used to update the PD LUT. A window is used to weight the correction as a function of amplitude. This technique increases the integration time constant for LUT addresses adapted by data with low SNR.

```
errorSum = abs(xdata(index(idx))) -
abs(ydata(index(idx)));
error = window(LUT_addr)*errorSum/sumCnt;
phi = atan2 (imag (oldLUT (LUT_addr) ), real
(oldLUT (LUT_addr) ) ) ;
newLUT(LUT_addr) = real(oldLUT(LUT_addr) +
stepSize*error*cos(phi) + . . .
j*(imag(oldLUT(LUT_addr)) +
stepSlze*error*sln(phi));
```

Similar calculations are performed on the unwrapped phase to update the PD LUT's AM-PM characteristic.

```
error = window(LUT_addr)*errorSum/sumCnt;
phi = error * stepSize;
tmpR = real(oldLUT(LUT_addr)) * cos(phi) -
imag(oldLUT(LUT_addr)) *sin(phi);
tmpI = real(oldLUT(LUT_addr)) *sin(phi) +
imag(oldLUT(LUT_addr)*cos(phi);
newLUT(LUT_addr) = tmpR + j*tmpI;
```

In order to smooth the LUT and fill in those addresses that were not updated, a polynomial fit is applied.

```
PI = polyfit(x_addr,yI,6);
YI2 = polyval(PI,x_addr(1) :x_addr(end) )
PQ = polyfit (x_addr, yQ, 4) ;
YO2 = polyval (PQ, x_addr (1 : x_addr(end) );
```

Laboratory Results

Figure 8 shows the laboratory setup. Both forward and feedback paths employed direct RF conversion. The Intersil ISL5217 evaluation board was used in a polyphase mode to generate a four-carrier UMTS digital waveform. This waveform was predistorted using the Intersil ISL5239 evaluation board. The ISL5239 is a baseband lookup-table (LUT) predistortion part. It was clocked at 125MHz, affording a full 100MHz processing bandwidth. An on-chip clock divider provided a 62.5MHz clock to drive the ISL5217. On-chip interpolating filters were used to upsample the ISL5217 output to the 125MHz rate.Figure 9 shows the CCDF of the ISL5217 output. The PAR at IE-8 was approximately 8dB.

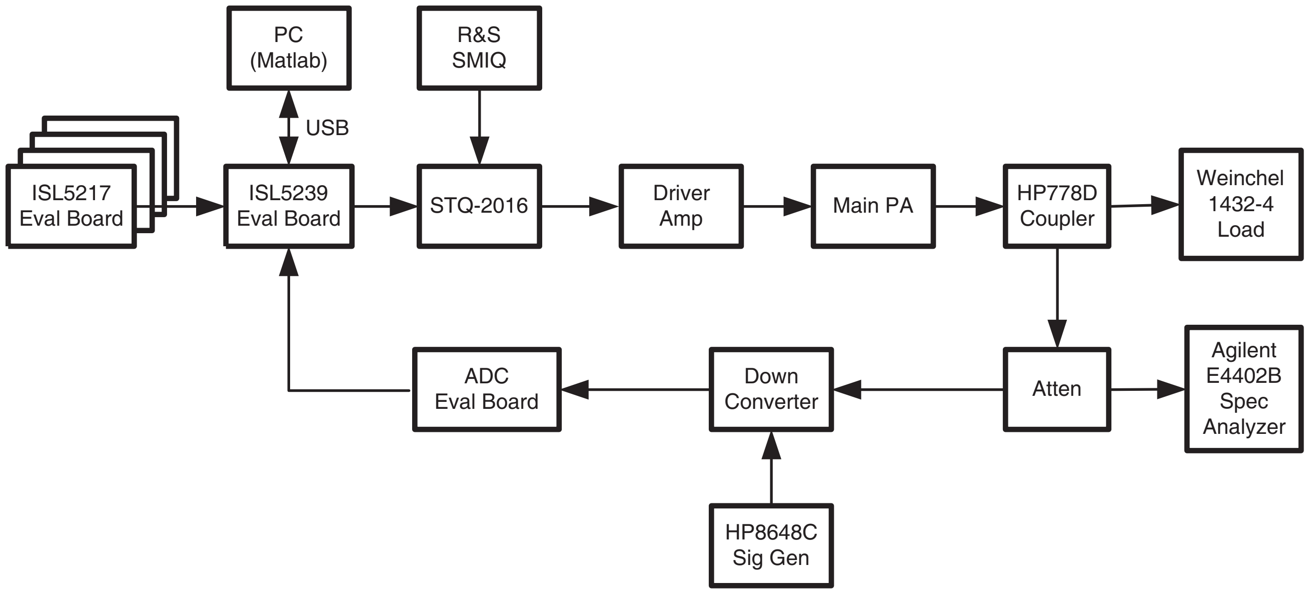

FIGURE 8. LABORATORY SETUP

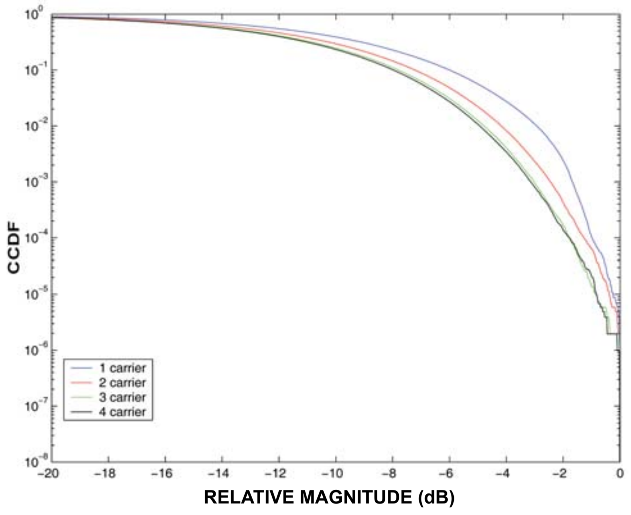

FIGURE 9. CCDF FOR MULTI-CARRIER UMTS

The ISL5239 evaluation board includes the Intersil ISL5929 dual DAC and interfaces with the Sirenza STQ-2016 direct upconverter to generate the 2.14GHz predistorted RF waveform. The ISL5239 IC includes programmable FIR filters, which were used to equalize $\frac{\sin(x)}{x}$ distortion and frequency response imbalance between quadrature paths -- in particular group delay mismatch. We also used the ISL5239's gain, phase, and offset correction to improve the image rejection and carrier leakage of the direct upconverter. The file TxCal.m was used to aid in the calibration of the correction filter and gain, phase, and offset adjustment.

Figure 10 shows the improvement in image rejection and carrier leakage using the gain, phase, and offset correction capabilities of the ISL5239. By digitally correcting for these imbalances, the low-cost STQ-2016 performs like a very expensive direct upconverter.

FIGURE 10. PERFORMANCE WITH AND WITHOUT GAIN, PHASE, AND DC OFFSET CORRECTION

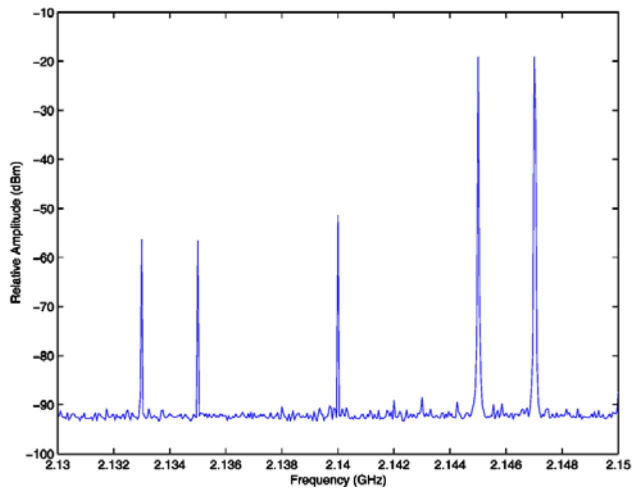

FIGURE 10A. TWO-TONE RF INPUT WITHOUT CORRECTION

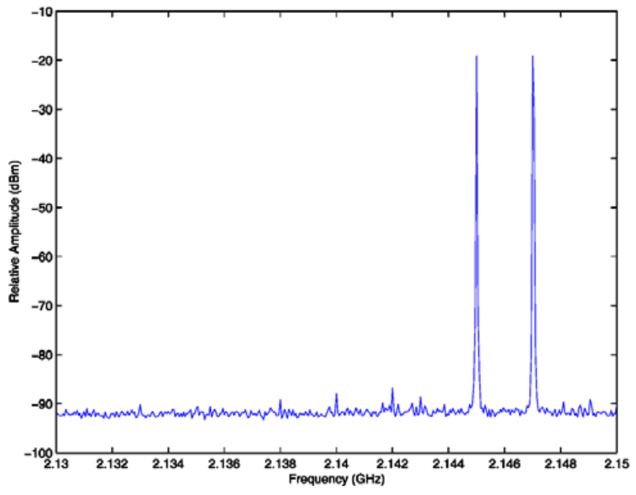

FIGURE 10B. TWO-TONE RF INPUT WITH CORRECTION

Figure 11 shows the frequency response of the analog transmit path with and without the correction filter. The correction filter was designed to equalize the frequency response across an 80MHz bandwidth. The figure shows that the frequency response variation was reduced from 3dB to around 0.1dB across this band.

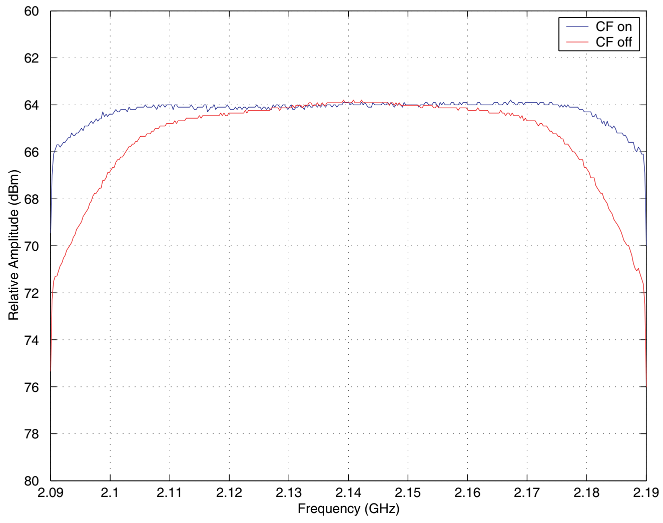

FIGURE 11. FREQUENCY RESPONSE WITH AND WITHOUT CORRECTION FILTER

The output of a ZFL25000VH amplifier from Mini-Circuits was downconverted and fed back to the ISL5239 through a dual 10-bit ADC evaluation board running at 125MHz. The input and feedback waveforms were captured using the ISL5239 input- and feedback-capture memories.

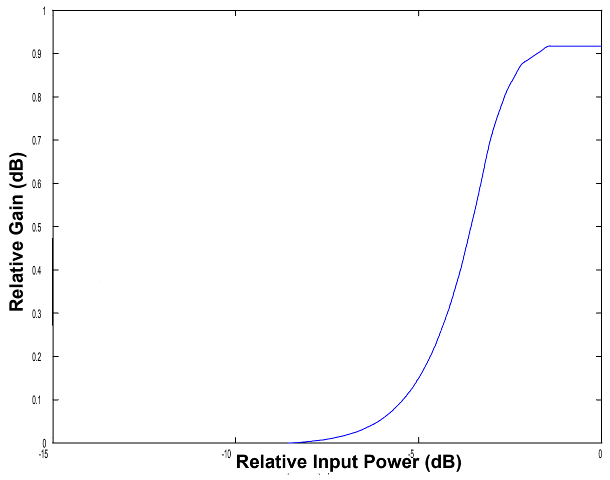

FIGURE 12A. PD AM-AM CHARACTERISTIC

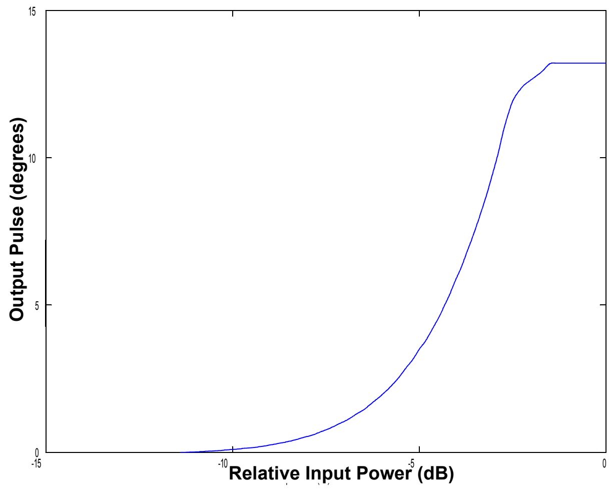

FIGURE 12B. PD AM-PM CHARACTERISTIC

Figure 12 shows the AM-AM and AM-PM characteristics of the PD LUT after the adaptive algorithm converged. The upper region of the curve is flat because that region was beyond the peak level of the input waveform and was not updated by the algorithm. The AM-AM characteristic shows only a small gain expansion of around 0.5dB, while the AM-PM characteristic shows a large phase adjustment of nearly 6 degrees. This data suggests that phase correction is just as important as amplitude correction for linearizing this amplifier.

Figure 13 shows the amplifier output with and without predistortion. In this case, ACLR is improved by 15dB for a 20MHz signal bandwidth. It was observed that as the signal bandwidth is reduced, ACLR improvement climbs to 25dB or more. Similar results were measured using a high-power LDMOS PA operating at 36 watts average output power with an efficiency of 12%. Digital predistortion maintained this high-level of efficiency while meeting the 3GPP ACLR requirements of 45dBc in the adjacent channel and 50dBc in the alternate channel.

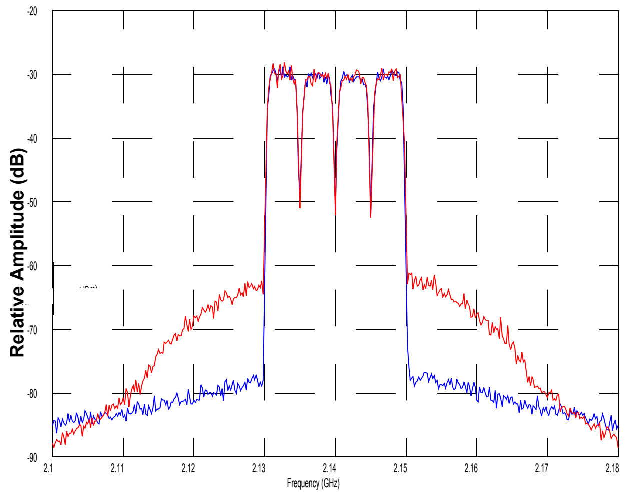

FIGURE 13. PREDISTORTION PERFORMANCE

Conclusions

In this application note we used the ISL5239 to linearize a low power amplifier and a real-world 3G basestation PA. We presented results showing 15dB of ACLR correction on a 20MHz bandwidth four-carrier UMTS waveform. An efficiency of 12% was achieved while meeting the 3GPP ACLR requirements. We presented the hardware architecture and described the adaptive algorithm. The ISL5239 affords a low-cost solution with high performance and is an attractive alternative to feedforward linearization and other more expensive options.

References

[1] H.S. Black, Translating System, U.S. Patent 1686792, issued Oct. 29, and U.S. Patent 2102671, issued Dec. 1937.

[2] S. C. Cripps, RF Power Amplifiers for Wireless Communication, Artech House, 1999.

[3] M. Faulkner, Adaptive Linearization Using Predistortion-Experimental Results, IEEE Trans. on Veh.Tech., vol. 43, pp. 323-332, May,1994.

[4] J.K. Cavers and M.W. Liao, Adaptive Compensation for Imbalance and Offset Losses in Direct Conversion Transceivers, IEEE Trans. Veh. Tech., vol. 42, no.4, pp. 581-588, 1993.

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks