Evaluating the Linearity of RF-DAC Multiband Transmitters

Researchers at Bell Labs show how to create a flexible platform for rapid evaluation of RF DACs using Xilinx FPGAs, cores and MATLAB.

by Lei Guan
Member of Technical Staff
Bell Laboratories, Alcatel Lucent Ireland
lei.guan@alcatel-lucent.com

The wireless communications industry has entered into a new, all-in-one era, with every network operator seeking more-compact and multiband infrastructure solutions. The emerging RF-class data converters—namely, RF DACs and RF ADCs—architecturally make it possible to create compact multiband transceivers. But the nonlinearities inherent in these new devices can be a stumbling block.

For instance, nonlinearity of the RF devices has two faces in the frequency domain: in-band and out of band. In-band nonlinearity refers to the unwanted frequency terms within the TX band, while out-of-band nonlinearity is the undesired frequency terms out of the TX band.

For system engineers who are prototyping multiband transmitters using RF DACs, it is critical to ensure this key component meets the linearity requirement of the standards. Therefore, at the early prototyping stage, a flexible testing platform is fundamentally required to properly evaluate the nonlinear performance of the RF DACs regarding the multiband applications.

Here at Bell Labs Ireland, we have created a flexible software-and-hardware platform to rapidly evaluate the RF DACs that are potential candidates for next-generation wireless systems. The three key elements of this R&D project are a high-performance Xilinx® FPGA, Xilinx intellectual property (IP) and MATLAB®.

A few words here before starting this engineering story. In the design, we tried to minimize the FPGA resource usage while keeping the system as flexible as possible, so we were only interested in implementing the necessary functions. For setting up the whole testing system, we picked the latest Analog Devices RF-DAC evaluation boards (AD9129 and AD9739a) and the Xilinx ML605 evaluation board. The ML605 board comes with a Virtex®-6 XC6VLX240T-1FFG1156 FPGA device, which contains fast-switching I/Os (up to 710 MHz) and serdes units (up to 5 Gbps) for interfacing the RF DACs.

Now, let's take a closer look at how to use Xilinx FPGAs, IP and MATLAB to create this simple but powerful testing platform.

SYSTEM-LEVEL REQUIREMENT AND DESIGN

The key purpose of this evaluation platform is to stimulate the RF DAC with various user-customized testing-data sequences. For this purpose, we designed two testing strategies: a continuous-wave (CW) signals test (xDDS) and a wideband signals test (xRAM).

Multitone CW testing has long been the preferred choice of RF engineers for characterizing the nonlinearity of RF components. Keeping the same testing philosophy, we created a tunable four-tone logic core based on a direct digital synthesizer (DDS), which is actually using a pair of two-tone signals to stimulate the RF DAC at two separate frequency bands. By tuning the four tones independently, we can evaluate the linearity performance of the RF DAC—that is, the location and the power of the intermodulation spurs in the frequency domain.

CW signal testing is an inherently narrowband operation. To further evaluate the RF DAC regarding wideband performance, we need to drive it with concurrent multiband, multimode signals, such as dual-mode UMTS and LTE signals at 2.1 GHz and 2.6 GHz, respectively. For that purpose, we created an on-chip BRAM array-based data storage core that has two subgroups in which to store the respective dual-band user data for repeated testing.

Figure 1 illustrates the simplified design diagram at the system level. As you can see, we are using a straightforward

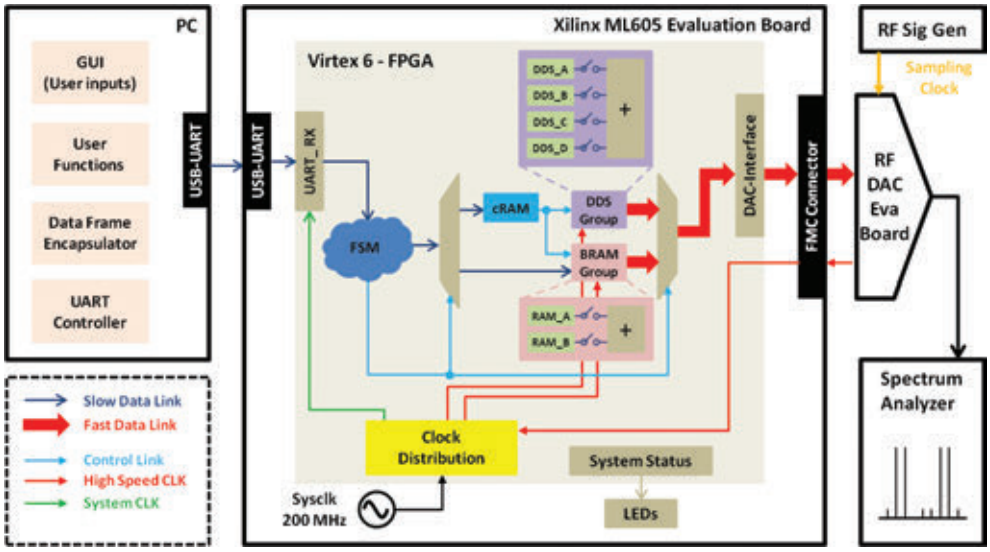

Figure 1 – Simplified block diagram of the platform at the system level

design strategy here, building up the platform as simply as possible and modularizing it with upgrading capability.

HW DESIGN: XILINX FPGA CORE

The FPGA portion of Figure 1 outlines the implemented logic units the system fundamentally required. They include a clock distribution unit, a state machine-based system control unit and a DDS core-based multitone generation unit, along with two units built around Block RAM: a small BRAM-based control message storage unit (cRAM core) and a BRAM array-based user data storage unit (dRAM core). Also included are a UART serial interface toward the PC and a high-speed data interface toward the RF DAC.

The clock is the life pulse of the FPGA. In order to ensure that multiple clocks are properly distributed across FPGA banks, we chose Xilinx's clock-management core, which provides an easy, interactive way of defining and specifying clocks.

A compact instruction core built around a state machine serves as the system control unit. As shown in Figure 2, at the initial state (S0), a header detector unit is active, monitoring and filtering the incoming data byte from the UART receiver. The data bytes were generated and encapsulated into frames in MATLAB, as shown in Figure 3.

Basically, there are two types of data frames in the system. The frame with header "FF01" (cRAM frame) was used for transmitting phase-incremental values for DDSes and system control messages. The other frame, with the header "FF10" or "FF11" (dRAM frame), is used for delivering the user-customized data. States "S1x" process only the data with the header "FF01" for updating the phase-incremental values and executing the control instructions. States "S2x" and "S3x" are utilized for receiving and storing the user-customized data for two bands, respectively. The busy signal is used for continuously latching the data in until seeing the last stop bit at the end of the data sequences. The control messages, for example invoking single/multiple DDS or the user data sequences, are stored in the last two bytes of the cRAM frame. They will be executed at the rising edge of the `cRAM_rd_done` signal.

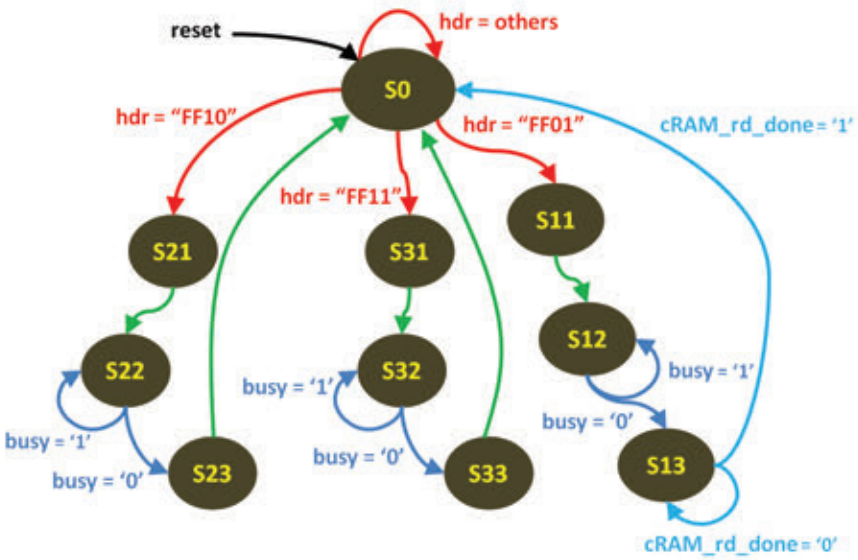

Figure 2 – Detailed design chart of the key state machine

We then instantiated four independent tone-generation units using Xilinx DDS cores, which we configured as the phase-incremental mode. The phase-incremental values for specific frequencies were generated in MATLAB and downloaded to the FPGA via the cRAM frame. We combined multiple tones via adders and then pipelined these tones to the next stage. Since the output of the DDS core was in two's complement binary format, a format-conversion unit may be needed if the RF DAC requires another data format, such as offset binary.

In general, high-performance on-chip BRAMs are always the first choice for creating small to medium-size user storage. For example, in this platform, we utilized Xilinx's Block Memory Generator core to build up two independent data RAMs for two frequency bands. Each of them is 16 bits in width and 192k in depth.

For communicating between the PC and the FPGA, we created a UART serial interface unit and configured it at a relatively low speed, 921.6 kbps, equivalent to 115.2 kbytes per second. It takes around 0.16 milliseconds and 3.33 seconds to transfer cRAM frames (18 bytes) and dRAM frames (~384k bytes), respectively.

Device vendors usually provide an example design of the high-speed data interface toward their chip in VHDL or Verilog format. It is not very difficult for an experienced FPGA engineer to reuse or customize the reference design. For example, considering our system's AD9739a and AD9129 RF DACs, a reference design of the parallel LVDS interface is available from Analog Devices. By the way, if there is no available example design from chip vendors, Xilinx has several very handy cores for the high-speed interfaces, such as CPRI and JESD204B.

SW DESIGN: MATLAB DSP FUNCTIONS AND GUI

We chose MATLAB as the software host, simply because it has many advantages in terms of digital signal processing (DSP) capability. What's more, MATLAB also provides a handy tool called GUIDE for laying out a graphical user interface (GUI). So now, what do we need from MATLAB for this project?

We actually need a user interface associated with lower-level DSP functions and data flow control functions. The required DSP functions are a phase-incremental values calculator, baseband data

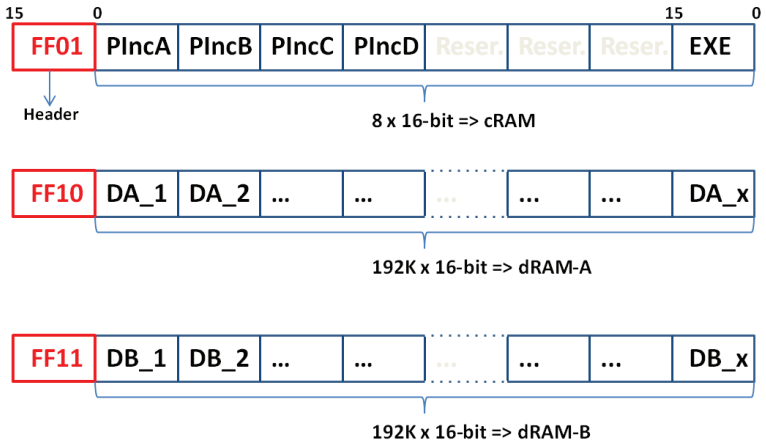

Figure 3 – Illustration of the data frame encapsulation

sequence generator and digital upconverter. The control functions are a data frame encapsulator, UART interface controller and system status indicators.

Figure 4 illustrates the GUI that we created for the platform. The key parameter of the RF DAC—namely, the sampling rate—should be defined first, and then you can select either xDDS mode or xRAM mode for stimulating the device. Next, at each subpanel, we can customize the parameters on the go to invoke the corresponding MATLAB signal-processing functions. In xDDS mode, you can calculate the phase-incremental value for the frequency tone `fc` with sampling rate `fs` by means of a simple equation, `phase_incr = fc*2^nbits/fs`, where `nbits` represents the number of binary bits used by DDS for synthesizing the frequencies. By pressing the "start" button, the generated phase-incremental values will be converted to the fixed-point format, encapsulated into a 2-byte data frame with different headers and control messages as shown in Figure 3, and then sent to the cRAM unit via UART and executed in the FPGA.

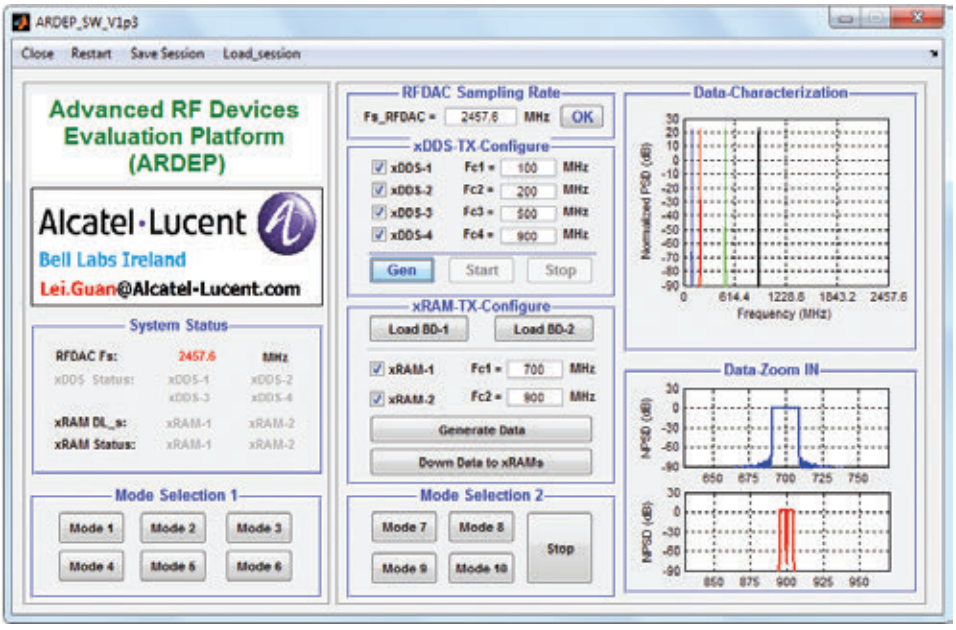

Figure 4 – Snapshot of the graphical user interface

In xRAM mode, we generated the baseband data sequences, normalized them to full scale (signed 16-bit) and upconverted them to the required frequency in MATLAB. After downloading the processed data to the dRAMs via UART, we can invoke the wideband signal testing by pressing the start button. Don't forget to configure the UART serial interface in MATLAB with the same protocol parameters used on the FPGA side.

Finally, we used a signal generator, the R&S SMU200A, to provide the sampling clock to logically "turn on" the RF DAC. And we connected the output of the RF DAC to a spectrum analyzer for evaluating the linearity performance of the RF DAC in the frequency domain.

QUICK EVALUATION

At the early-prototyping stage, evaluation of the key RF components regarding linearity performance can be a critical problem, but with our hardware/software platform you can manage this evaluation quickly without compromising performance. Then you can add in your RF power amplifier and use the proposed platform to evaluate the linearity of the cascaded system. After identifying the nonlinearities, you can implement some digital-predistortion algorithms to eliminate the unwanted nonlinearities of the cascaded system.

Properly using Xilinx IP cores in your FPGA design will significantly reduce the development cycle and increase the robustness of your digital system. Looking forward, we expect to upgrade the data interface module in the platform to the JESD204B standard to support the higher data rates required by multiple synchronized RF DACs. Meanwhile, we are migrating the FPGA host from Xilinx's ML605 to the Zynq®-7000 All Programmable SoC ZC706 evaluation kit. The Zynq SoC design will be a good choice for creating a standalone solution that does not need any external DSP and control functions on a separate PC.

For more information regarding the platform and digital predistortion, contact the author at *lei.guan@alcatel-lucent.com*.