MODULAR FIELD PROGRAMMABLE GATE ARRAY IMPLEMENTATION OF A MIMO TRANSMITTER

Richa Shekhar (Student) and Kurt Kosbar (Advisor)
Telemetry Learning Center
Department of Electrical and Computer Engineering
Missouri University of Science and Technology

ABSTRACT

Multiple-Input Multiple-Output (MIMO) systems have at least two transmitting antennas, each generating unique signals. However some applications may require three, four, or more transmitting devices to achieve the desired system performance. This paper describes the design of a scalable MIMO transmitter, based on field programmable gate array (FPGA) technology. Each module contains a FPGA, and associated digital-to-analog converters, I/Q modulators, and RF amplifiers needed to power one of the MIMO transmitters. The system was designed to handle up to a 10 Mbps data rate, and transmit signals in the unlicensed 2.4 GHz ISM band.

Keywords: Multiple-Input Multiple-Output Channels, Bandwidth Efficient Modulation, Scalable System Design, Field-Programmable Gate Arrays

I. INTRODUCTION

Multiple Input Multiple Outputs (MIMO) communication systems use multiple antennas at both the transmitter and receiver to increase data rates, or to provide more reliable communication than Single Input Single Output (SISO) systems [4]. The greatest improvement in performance occurs when the systems are used in rich multipath and fading environments. It is a significant challenge to model these environments accurately, using analytically tractable mathematical models. Software simulations provide more flexibility, but as with many problems, the true performance of the system is not known until a physical device is constructed and tested in the target environment.

There is a need for a MIMO platform that is reasonably inexpensive, and can be easily reconfigured in the field to test a variety of space-time block codes. It would also be helpful if the number of transmit antennas could be varied. This paper describes the architectural design, and testing plan, for a modular MIMO transmitter. The system can be configured to use an arbitrary number of transmit antennas. The baseband processing is performed with a field programmable gate array (FPGA) device. As its name implies, this device can be programmed in the field to produce arbitrary baseband in-phase (I) and quadrature phase (Q) signals. This allows the end user to quickly alter the modulation and coding format used for the MIMO

system. In addition, the transmitter can be configured to implement a SISO transmitter, a series of independent SISO transmitters, or it can implement beam forming algorithms and other modulation and coding schemes.

The system is intended to work with data rates up to 10 Mbps, and is currently configured to use the 2.4 GHz unlicensed ISM band.

The following section describes the architecture of each module in the system. The testing plan for the modules is discussed in the next section, followed by a discussion of how the modules can be interconnected to form multiple SISO transmitters, beam forming transmitters and a MIMO transmitter.

SYSTEM ARCHITECTURE

The system is designed to use a set of identical modules. Each module will drive one, and only one, of the transmit antennas. A high level block diagram of the module is shown in Fig.1. Data is fed into an FPGA, which performs the necessary coding and waveform generation operations. The FPGA may perform digital filtering, and arbitrary waveform synthesis. Two digital waveforms are generated by the FPGA, and sent to the digital-to-analog converter (DAC). The DAC output waveforms are then fed to a modulator, and the RF waveform sent through a power amplifier before being transmitted.

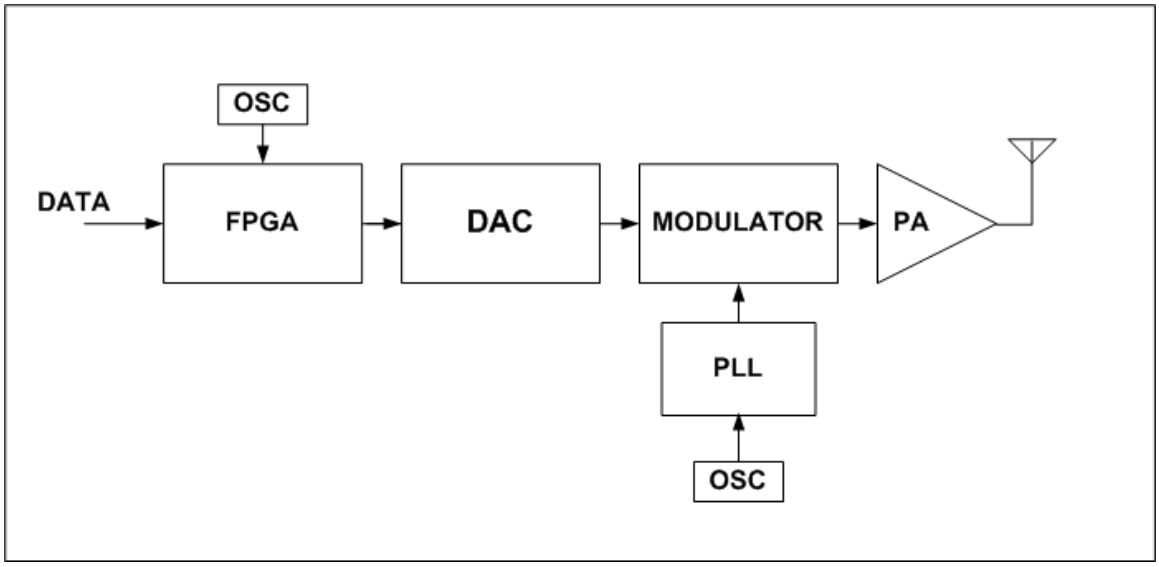

Figure 1 Architecture of One Module

The modulator uses an RF reference produced by a phase-locked loop device, which is in-turn controlled by a crystal time base. This particular test bed is based on an Altera Cyclone II FPGA [5], which includes an embedded processor. The processor will be used to not only filter, and code, the data – but can also be used to generate test patterns for testing the system, and pilot symbols for measuring the channel state information during transmission.

A more detailed block diagram is shown in Fig. 2. Beginning from the left hand side, we see the FPGA is clocked by oscillator 2. The goal is to perform pulse shaping using digital signal processing algorithms in the FPGA. The target data rate is 10 Mbps, and we would like to have at least 10 samples per symbol, so the FPGA needs to be clocked at a rate of at least 100 MHz [2].

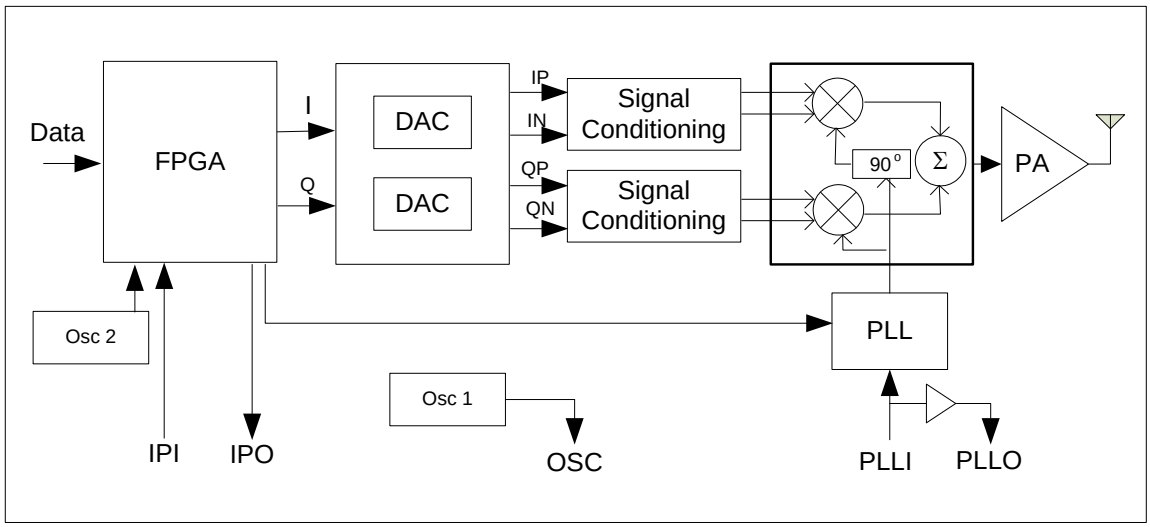

Figure 2 Detailed Module Architecture

The FPGA must generate two digital signals, and pass these to a pair of DAC to generate the I and Q baseband waveforms. These differential analog signals are then fed to signal conditioning electronics before being passed to the quadrature modulator. This was necessary in the design because the DAC selected generates a signal which is symmetric about found, whereas the quadrature modulator requires a single sided input.

The I and Q signals modulate an RF carrier produced by a phase locked loop (PLL). The PLL is stabilized by a crystal oscillator, labeled "Osc 1" in Fig. 2. The ratio between the PLL output frequency and the oscillator is controlled through internal registers which are set by the FPGA as part of its power-up sequence.

When multiple modules are used, it may be necessary to synchronize the PLL. To accommodate this, the crystal oscillators are not connected directly to the PLL. The two must be connected via an external jumper, and may be daisy chained if necessary.

TRADEOFFS

Three architectures were considered for the baseband signal processing in this device: programmable digital signal processing (DSP) devices, application specific integrated circuit (ASIC) design and FPGA. The tradeoffs associated with these approaches are discussed below.

There are many inexpensive DSP devices available; however some of these may have difficulty maintaining the necessary throughput to meet the target data rate of 10 Mbps. The data must undergo pulse shaping, which will require a sampling rate significantly higher than the data rate. High performance DSP devices have the necessary throughput, but this comes at a potentially substantial increase in cost and power consumption. The lowest power consumption would come from an ASIC solution [3]. However there is substantial non-recurring development costs associated with this. In addition, ASIC designs are inflexible, so it would not be possible to experiment with a large number of different coding and modulation formats.

To achieve the necessary data throughput, while still keeping with a design that can be rapidly, and inexpensively, changed, we selected an FPGA approach [1]. The FPGA selected for this design allows a soft-implementation of a DSP core, which will simply the design process. There are also a number of public domain cells which can be used to speed the development of the system. The baseband processor will be implemented in VHDL, which will allow migration to an ASIC at a later date, should that be needed. In some applications a FPGA may be useful for other parts of the system design, which will help minimize the overhead of adding a MIMO transmitter.

The FPGA approach is not without drawbacks. The FPGA design will consume more power than ASIC and low-end DSP solutions. The unused gates on the FPGA will consume power, and the reasonably long interconnects on the FPGA will also increase power consumption. But the flexibility of reprogramming the device, along with the ability to process data at the target rate, and the ability to implement other portions of the system design on the FPGA provide strong incentives for selecting this approach.

The particular device selected can be programmed in either of two ways. A JTAG interface allows a user to quickly inspect and program volatile memory. We plan to use this during the development and testing phase in the laboratory. Once some candidate designs have been developed, the device will be programmed using an asynchronous (AS) interface that allows the program to be stored into non-volatile memory, for loading on power-up.

TESTING AND VALIDATION PLAN

To verify the design, and also to help debug the coding and modulation algorithms to be analyzed, a number of test features will be built into the base modules. The FPGA portion of the design should be reasonably low risk. Some of the unused general purpose I/O pins will drive LEDs, which can be toggled by the user to indicate when particular portions of the code are executed. There will be a number of low impedance outputs which drive the DAC, which will be brought to headers to allow easy connection of a logic analyzer. In addition, the LSB and MSB of these outputs will drive LEDs, to allow the user to quickly observe when there is activity on the output data busses.

We considered connecting the FPGA to the DAC through jumpers, which could be removed so the user could inject signals directly to the DAC. However given the number of digital lines involved, the lower reliability of jumper connections in the final design, and what we anticipate

will be a reasonably low risk FPGA design, we elected not to use that approach. Once the design has been validated, the LEDs can be removed from future implementations to help reduce the power consumption.

The DAC can potentially inject a wide variety of imperfections that could impact system performance, including non-linearity and timing jitter or delays. To measure these effects, the DAC outputs will be brought to test points which will allow the connection of oscilloscopes, spectrum analyzers, and other test equipment. During initial testing, the FPGA will be programmed to generate sinusoids at the DAC outputs. Imperfections such as nonlinearities can then be measured through harmonic analysis of the output signals. The impedances will be selected so that it will be possible to also inject signals at this point. Should the FPGA be unable to generate the signals required to test the remainder of the module, the DAC will be disabled and signals injected at the DAC outputs.

In previous student projects, the phase locked loop has proved to be a challenging device to work with. The circuit design for the PLL chip is reasonably straight forward. However during power up the device must be programmed, so that the output frequency and power level will fall in the correct range for the quadrature modulator. To help observe, diagnose, and potentially overcome these problems, the serial communication link that connects the FPGA to the PLL will pass through headers/jumpers, and to test points. This will allow a logic analyzer to be connected to the FPGA output, to observe the sequence of programming bits sent to the PLL. Should the FPGA fail to generate the proper signals, the jumpers will be removed, and an external logic source will be used to program, or reprogram, the PLL. The output of the PLL will be passed through a power splitter, with one of the outputs going to a test point, and the other to the modulator. This test point can be used to confirm the power level, and frequency, of the PLL output – along with measuring phase jitter, and other imperfections in the carrier signal.

The quadrature modulator device cannot be probed; however its output will be split between the power amplifier, and a test point. During testing, constant levels will initially be applied to the I, then Q, signals to verify that both sides of the quadrature modulator are functional. This will be followed by sinusoidal signals on both I and Q, to essentially perform single side band modulation. Phase and gain imbalances can then be measured by observing the unwanted sideband at the modulator output. Finally, BPSK and QPSK signals will be generated, and their time domain and spectral domain signatures analyzed at the modulator output.

Once the modulator output has been validated, the final test will be to observe the output of the RF power amplifier. Initially sinusoidal signals of varying frequencies and amplitudes will be generated by the modulator, to verify the power level of the output. In addition, spectral analysis will be used to measure the nonlinear characteristics of the power amplifier.

CONNECTION OF MULTIPLE MODULES

The modular design of this system allows it to be configured in a variety of ways. In the most basic configuration, the boards can be used simply as independent transmitters, each operating on a different data pattern, as shown in Figure 3. Each module will need to have jumper

connecting the oscillator output to the PLL input. In this configuration, there will be no synchronization between the modules. This would be appropriate if the modules were used with frequency division multiple access systems, or code division multiple access.

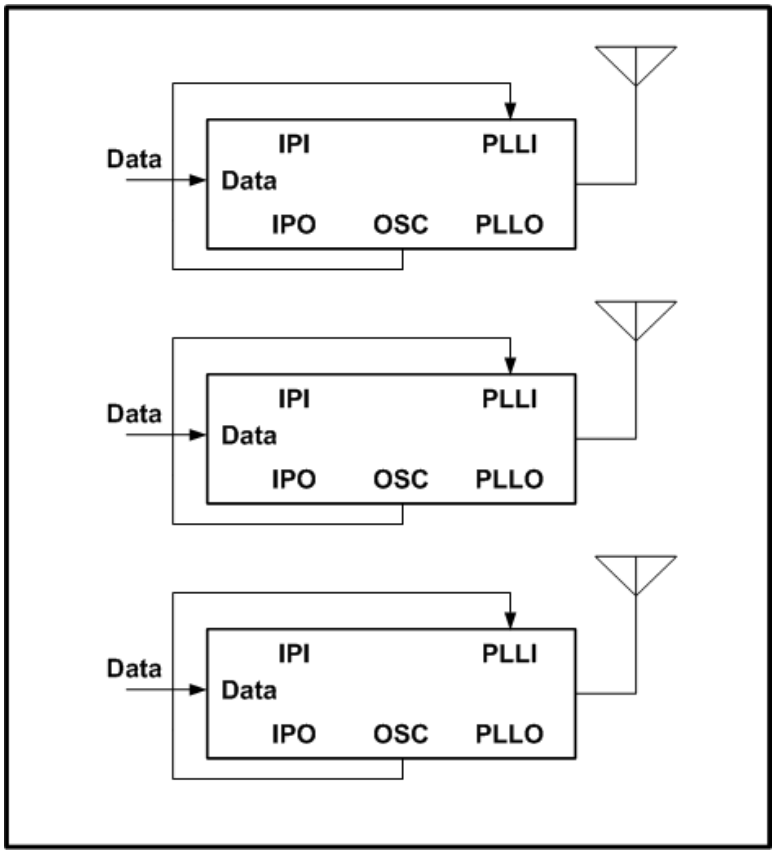

Figure 3 Multiple SISO Transmitter Configurations

In the multiple SISO operation, there is no centralized control of the transmit frequency of the various modules. Each module references its carrier frequency to an on-board crystal oscillator. These oscillators drift with time and temperature, so the corresponding carrier frequencies will also not remain synchronized. There are applications where it may be desirable to have all transmitters phase locked to the same reference oscillator. This can be accomplished using the "PLLO" outputs from the modules.

As shown in Figure 4, the crystal oscillator from the top module may be used to drive the phase locked loops of an unlimited number of other modules. Each module contains a buffer amplifier which will reproduce a copy of the PLLI signal at the PLLO port. By daisy chaining the modules together, all of them can be frequency locked to the same crystal oscillator. There will be a phase shift introduced by the interconnecting cables, in addition to the buffer amplifiers. Perhaps even more importantly, when it comes to phase synchronization, there will be an unresolved phase ambiguity in the PLL of each module.

Because of the issues listed above, it will not be possible to develop phase synchronous RF carriers on each of the modules. However the common crystal time base should maintain the

modules in frequency lock. In the configuration shown in Figure 4, the modules are still using independent data paths. These could be connected together at the user's discretion, to create a system which had frequency diversity.

The final configuration, and the one for which the system was originally planned, is shown in Figure 5. This is the MIMO configuration. As with the previous arrangement, all of the modules are frequency locked to a common low frequency crystal time base. It is not critically important that the top module's time based be used for this, it should work equally well to use the time base from any of the modules.

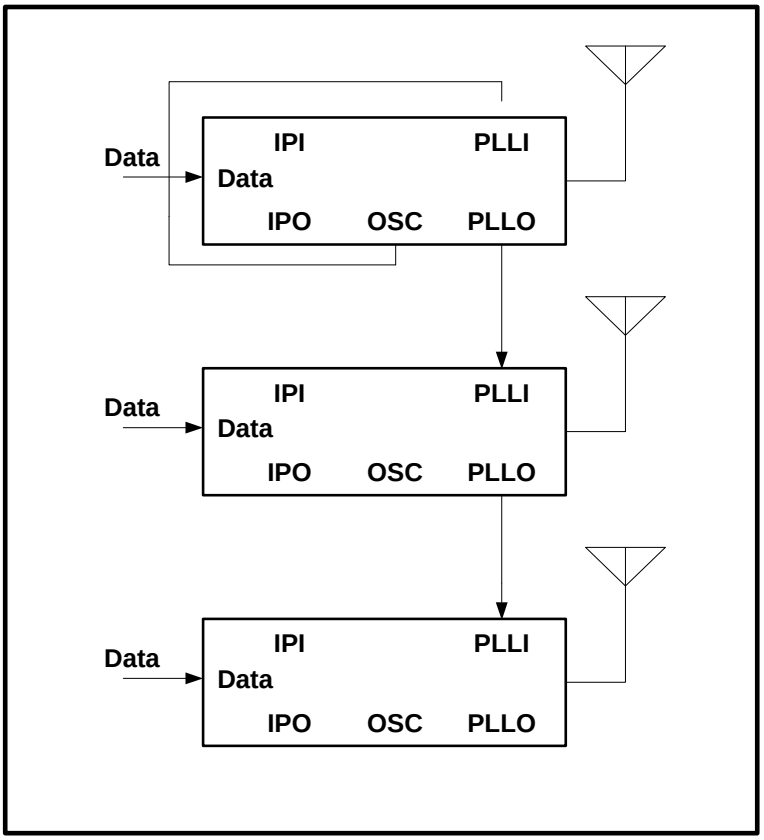

Figure 4 Multiple SISO Transmitters with Synchronized Carriers

The new feature in Figure 5 is that only one of the modules is fed the data to be transmitted. This module examines the data, to determine the waveform which it must transmit. The module then sends the same data over the interprocessor (IP) data bus to the second module. This module will use a slightly different algorithm for determining its transmitted waveform. The details of the algorithms used will depend on the particular space time block code used.

If there are more than two transmit antennas in the MIMO system, the second module will relay the data to the third, and subsequent, modules through a daisy chained interprocessor data bus, as shown in Fig. 5. Each module will need to know which portion of the space time block code it is assigned to generate.

As an alternative, the interprocessor data bus could be used to send configuration information in addition to the data. A framing protocol could be used, where in addition to the data bit to be transmitted, each processor would insert a digital word to indicate which part of the space time block code it is implementing. The subsequent modules could then read this data, and determine which portion of the code it should implement.

Another portion of the interprocessor communication must be a synchronization signal to indicate when each processor should begin transmitting a new symbol. It is important that all modules begin, and end, the symbols at the same time. The PLL bus only serves to synchronize the carrier frequencies, and would not normally be used for symbol synchronization.

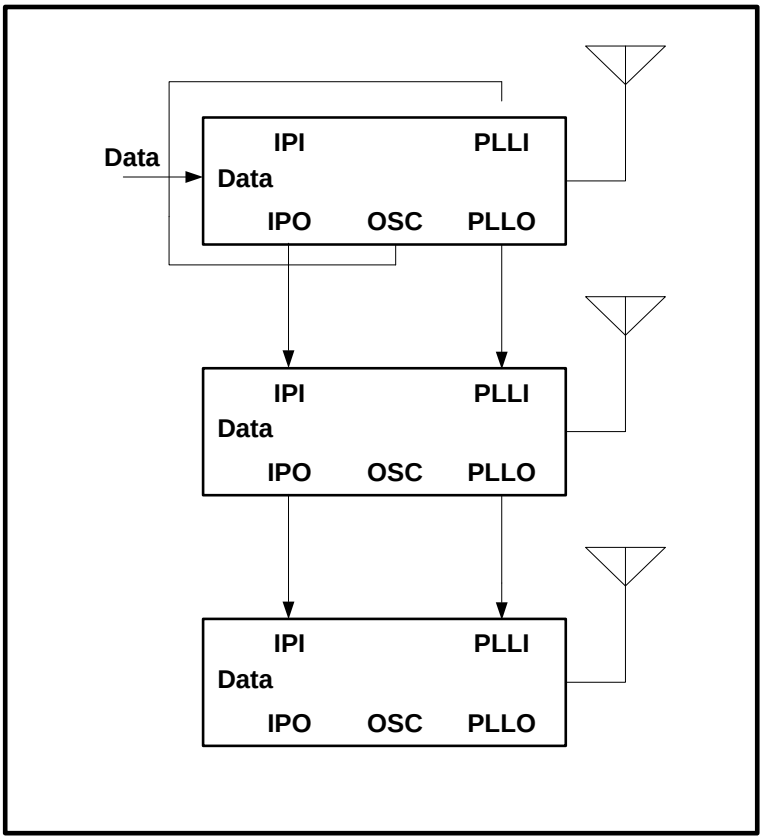

Figure 5 MIMO Configuration

CONCLUSION

The paper presented an architectural description of a modular MIMO test bed. Each module consists of a FPGA based baseband processor, frequency reference, PLL, quadrature modulator and RF amplifier. The configuration of the modules will allow a system to be constructed as a series of SISO transmitters, a series of frequency-synchronized SISO transmitters, or a MIMO transmitter. The system is scalable, so that an arbitrary number of transmit antennas can be used.

REFERENCES

[1] R. Joost, R. Salomon, “Advantages of FPGA-based multiprocessor systems in industrial applications,” Proc. of 31st Annual Conference of IEEE on Industrial Electronics society, 2005.

[2] Y. Palaskas, A. Ravi and S. Pellerano, “MIMO techniques for high data rate radio communications,” Proc. of IEEE Custom Integrated Circuits Conference, pp.141-148, 2008.

[3] R. C. Restle, “Choosing between DSP’s , FPGAs, µPs, and ASICs to implement digital signal processing,” Conference proceedings of ICSPAT:DSP world, Autumn 2000.

[4] D. Bates, S. Henriksen, B. Ninness, S.R. Weller, “A 4x4 FPGA-based wireless testbed for LTE applications,” Proc. of the IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communication, pp.1-5, 2008.

[5] Altera, Cyclone II Device Handbook, 2008.