RAPID PROTOTYPING OF A SDR BASED RECONFIGURABLE MIMO-OFDM TESTBED

Sabares Moola, S.M. Shajedul Hasan, Carl B. Dietrich, Jeffrey H. Reed
({sabares,hasan,cdietric,reedjh}@vt.edu), Wireless@Virginia Tech, Virginia Tech,
Blacksburg, 24061

ABSTRACT

This paper describes the rapid prototyping of multiple-input multiple-output -orthogonal frequency division multiplexing (MIMO-OFDFM) created on reconfigurable platforms like field programmable gate arrays (FPGAs) using the model based design (MBD) techniques. The work is based on model based design tools employed for the creation of baseband signal processing components for MIMO-OFDM to be run on an auxiliary FPGA, with a universal software radio peripheral (USRP2) acting as radio frequency (RF) front end. Thus the paper leverages the USRP2 platform with the additional FPGA for wide bandwidth processing capability for baseband. Moreover the integration of MIMO-OFDM testbed with the USRP2 platform validates the capability to use FPGAs as a potential target platform for performing high bandwidth signal processing applications.

1. INTRODUCTION

The term software defined radio (SDR) has become ubiquitous in the wireless communication industry because of the description of physical layer components in software [1]. It is in complete contrast to the design of conventional radios. The use of SDR provides flexibility in terms of reuse of components on multiple platforms and use of the same platform for multiple SDR based communication systems [1]. Other advantages include instant re-configurability in the field and ease of maintenance, within the physical limits of the underlying hardware [1].

The recent proliferation of SDR approaches in the defense and civilian community has necessitated the SDR radio to be built using re-configurable hardware to support high bandwidth applications. This wide bandwidth processing helps in performing high data rate symbol level processing whose latency and processing requirements may not be met with the normal digital signal processor (DSP) and general purpose processor (GPP). This is the primary motivation behind this work. In this paper, we describe the signal processing components built for an FPGA-based MIMO-OFDM testbed on. The outcome of this work can be used as a prototype for developing baseband signal processing components of the other waveforms that can run on an FPGA with a tunable wideband USRP2 acting as the RF front end. The work presented in this paper will help in providing a FPGA platform for the open source community like GNURadio and OSSIE. The model of design employed in this work for creating components on the FPGA will promote the creation of open source intellectual property (IP) cores developed on the lines of GNURadio blocks and OSSIE waveforms.

Rapid changes in underlying hardware platforms and the introduction of higher capability hardware at a rapid pace dictate that development and deployment of signal processing algorithms must support portability and scalability. The work presented in this paper attempts to promote model based design flow as a potential tool for rapid prototyping of wireless applications for SDR platforms. The significance of those models is based on their portability to other platforms. The models are also helpful in scaling an implementtion or adding enhanced features.

The paper is organized as follows: section 2 briefly describes model based design employed to create signal processing components, its implications for SDR and tools employed in this work to create it; section 3 describes signal processing components created for MIMO-OFDM application using these tools; section 4 describes integration of the FPGA platform with the USRP2 and section 5 presents the results obtained using model based design.

2. MODEL BASED DESIGN (MBD)

The motivation for model based design for FPGA's rose from the necessity of designing complex DSP systems that involve dedicated multipliers, compute units for specialized operations like Galois field arithmetic, Add compare select unit for Viterbi decoding, etc. These dedicated and complex compute units obviated the requirement for finer optimizations of FPGA design circuit generally associated with traditional digital design for communication systems. Model based design is used to express the characteristics of

a system like timeliness that describes the way a system is going to handle the concurrency, heterogeneity, interfacing and reactivity [2]. Model based system design is basically an attempt to describe the way a system will interact with the real time analog world.

The aim of model based system design is to convert the system model from its mathematical specification to executable specification. This executable specification represents a platform independent model (PIM)[2]. The PIM is then used to create an elaborate hardware and software specific model for the completion of the system through the automated code generation tool for the underlying platform. Thus the platform independent model (PIM) is converted into a platform specific model (PSM).

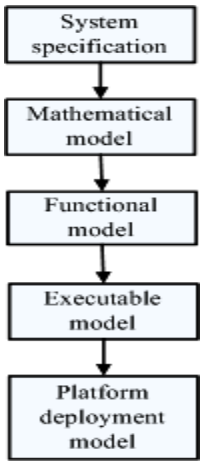

Figure 1. Model Based Design Flow

2.1. Implications of MBD for Software defined radio

A typical SDR platform will contain a combination of a general purpose processor (GPP), embedded DSP processor and FPGA[1]. Generally, the GPP handles the control and configuration of the software, the DSP processor handles the low level signal processing algorithms while the FPGA handles advanced computationally intensive signal processing algorithms. The baseband processing required for multiple waveforms can be implemented either on a DSP processor or FPGA. But choice of an FPGA as a platform for SDR arises from the basic requirements of SDR like *flexibility*, *scalability* and *performance[1]*. With current wireless standards offering higher data rates and using advanced processing techniques like MIMO, STC coding techniques, adaptive modulation and coding techniques, the processing bandwidth required to perform such highly intensive operations are easily provided by an FPGA. Moreover, performance of computational operations on an FPGA minimizes the latency in the system thereby increasing the system performance.

The developmental approach of SDR on FPGAs is driven towards MBD because of the necessity to support multiple waveforms in a single SDR platform or to support a single waveform on multiple platforms. An MBD design approach will allow the designers to carry out the rapid prototyping of waveforms on FPGA. It provides a viable approach to implementation and reuse of intellectual properties. It allows the designers to reuse the same high level design model for porting to different FPGA platforms. For example, the Joint Tactical Radio System (JTRS) specifies twenty different waveforms to be supported in the underlying SDR platform[3]. The use of MBD design flow allows the designer to focus on algorithm development and optimization rather than concentrating on system implementation issues.

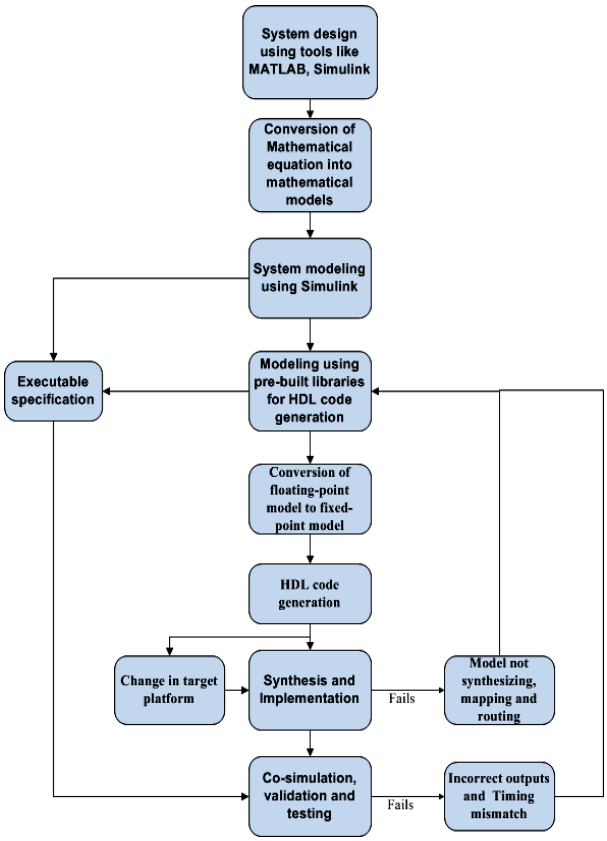

Figure 2. Model Based Design Flow for FPGA

2.2. Tools used in this work

The MBD tools that were used in this work are Simulink® HDL Coder™ and Xilinx System Generator for DSP™. The former is used to generate bit-accurate synthesizable HDL code that truly generates a PIM model whereas the later is used for generating IP cores necessary for mapping the PIM model onto Xilinx specific FPGA devices.

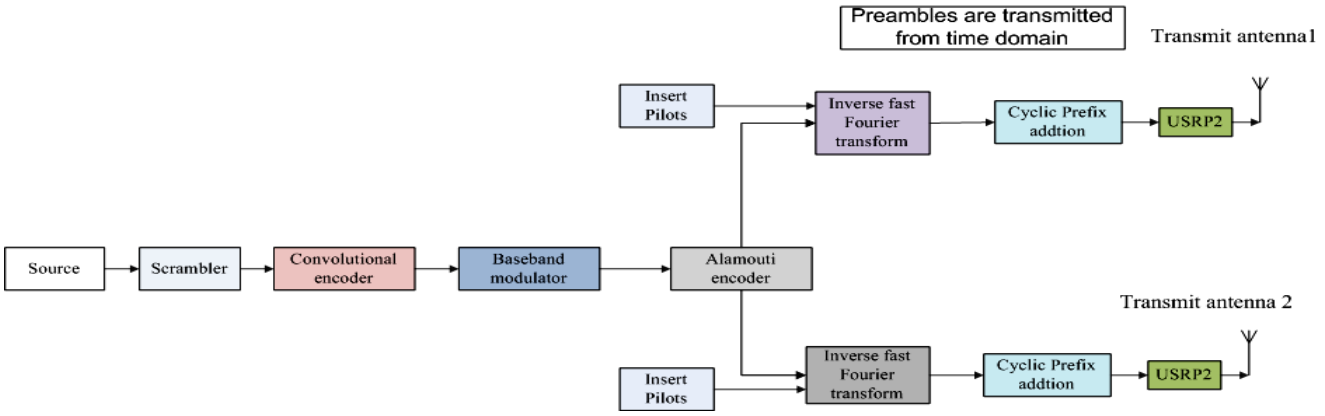

Figure 3. Block Diagram of MIMO-OFDM Transmitter

3. IMPLEMENTATION

This section is going to present the implementation details of the MIMO-ODFM testbed. The signal processing blocks required for implementing the transceiver chain of MIMO-OFDM on FPGA are created using the MBD tools. The following sections are going to present the implementation details involved in creating this whole testbed.

3.1. System Design

The design presented here uses a USRP2 as an RF front end and a Xilinx ML403 development board as a baseband signal processing platform. The FPGA present on the USRP2 has limited space for additional signal processing algorithms. The USRP2 interfacing with the ML403 platform is achieved through the Gigabit Ethernet cable. The interfacing details are presented as a separate section. The USRP2s present in the transmitter section are synchronized with the common clock. This common clock can be achieved either by a MIMO connector cable provided by the Ettus Research or by the external common reference clock. In the same manner, the USRP2s present in the receiver can be synchronized. The brief details about the USRP2 and ML403 development board is presented in the following sub-sections.

3.1.1 USRP2

The USRP2 is an SDR platform useful for building radios with a wide range of applications. There are different kinds of daughter boards available to target different kinds of applications. USRP2 is a successor to the USRP with added capabilities in terms of a larger FPGA and a Gigabit Ethernet connection. It uses a Xilinx Spartan-3 FPGA with its ADC producing samples at the rate of 100 MS/s and its DAC is capable of producing 16-bit 400 MS/s.

3.2.2 Xilinx ML403 development board

ML403 development board is powered by Virtex®-4FX12 FPGA and used for embedded system development. Its base clock runs at the speed of 100MHz[4]. It has a PowerPC®-405 core that can run at the speed of 300MHz. It is a simple embedded platform for testing and prototyping. It does not have an analog-to-digital (ADC) converter or a digital-to-analog (DAC) converter. It includes peripherals such as Ethernet, USB, and UART.

3.3 Module Implementation

The individual blocks required for the implementation of the MIMO-OFDM transceiver chain are shown in the Figure 3 and Figure 4. Most of the blocks are implemented using the MBD technique.

3.3.1 Module development

In this section, the steps involved in the design and implementation of individual blocks/modules are explained. Each block influence on the complete system has to be pre-decided for the proper designing of the system. The reference parameters required for designing the individual blocks were taken from IEEE 802.11a and IEEE 802.16 standards. Some of these parameters were rate of the convolutional encoder,

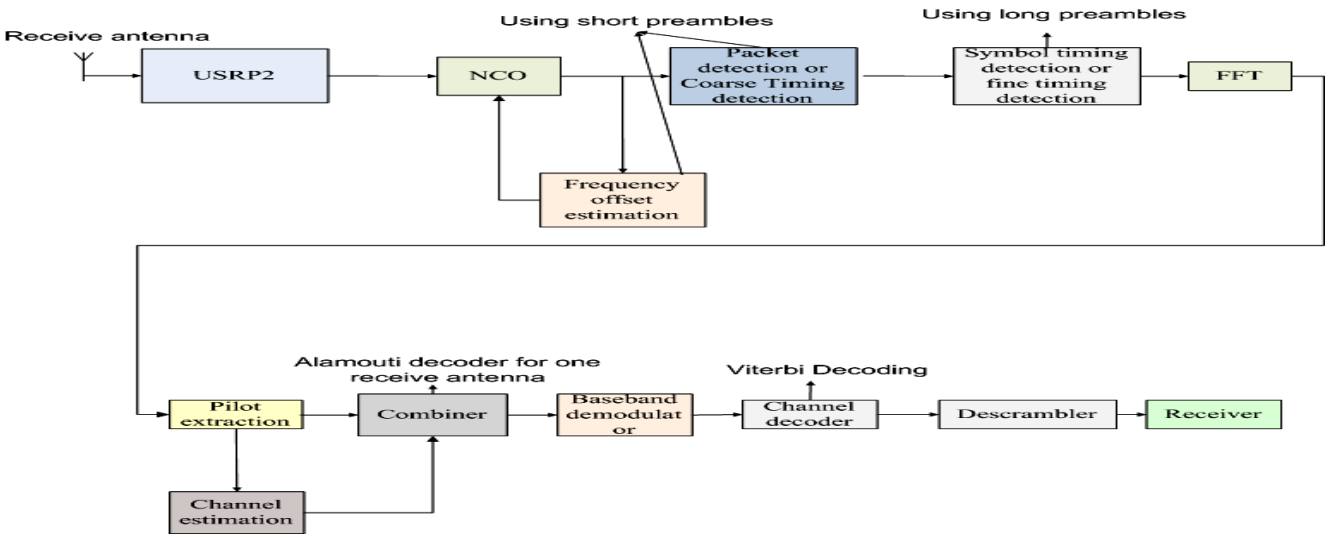

Figure 4. Block Diagram of MIMO-OFDM Receiver

scrambler/de-scrambler design, number of points in fast Fourier transform/inverse fast Fourier transform (FFT/IFFT), length of the cyclic prefix, preamble structures for initial synchronization, pilot arrangement for the channel estimation and phase error tracking, etc. The data format and wordlength of the system are also pre-decided. This is important for the wordlength analysis of the individual blocks. Based on the analysis, the scaling factor overflow limit and saturation limits are determined. A typical design flow involved in the development of modules for transmit and receive side is explained in the next few sections.

3.3.2 Design flow

As shown in Figure 2, the module development started with the system specification and simulation. This step involves the simulation of the end-to-end system with parameters taken from the standards like IEEE 802.11a and IEEE 802.16e. This step is performed using the MATLAB® functions and/or Simulink® blocks. This step helps in the validation of the system and the test vectors obtained from this step are helpful in testing the system after its implementation on the FPGA. During the simulation stage, the level of re-configurability of the blocks along with the scalable parameters is determined. This helps in making a PIM model to achieve scalability and flexibility while converting into PSM.

The results obtained from the simulation step are used for building the models using the MBD tools. For example, the performance of the various synchronization algorithms has to be analyzed under various conditions before constructing a model for it. The mathematical equations resulted from the previous step have to be represented using a mathematical model now. The tool used for constructing these models was Simulink® HDL coder™. Simulink® does not allow the conversion of the built-in blocks directly into HDL code. It has certain pre-defined libraries/blocks that can only be converted into a synthesizable HDL code. Most of these pre-defined blocks are primitive blocks like arithmetic blocks, delay elements, random access memories, shifters, comparators, logic elements, port selector, switching elements, etc. Most of the signal processing blocks mentioned in this paper were implemented using these primitive elements. This reduces the reliance on third party IP cores required for building the system. The models created as building blocks of the MIMO-OFDM transceiver help in achieving the goal of PIM. These PIM models can be released as *open source models* that will help the open source community, for building radios, to create the open source cores.

The creation of mathematical model may be relatively straight-forward. But there are certain inherent issues in the tools and platform employed in this work. First and foremost, the conversion of PIM to PSM requires the user to take care of certain fixed point conversions, clock manager for multi-clock systems, and timing issues specific to FPGA hardware. The Simulink® HDL coder™ helps in generating the HDL code but it does so only for fixed-point model. The conversion from floating-point model to fixed point model requires the user to take care of fixed point conversion guideline at various points within the system. The inherent conversions specified by the Simulink® Fixed Point Advisor Tool™ can be very dangerous for real time implementation. Moreover, the HDL code generated by the tool should be tested for functionality and timing using the

Modelsim™ tool. Though Modelsim™ is a pre-synthesis tool it helps in checking the validity of the model.

Once, the validation was done the model was mapped to the target platform using the vendor specific tools for synthesizing, mapping and routing onto the FPGA. The mapping on the FPGA fabric requires understanding of certain timing constraints associated with the FPGA and the maximum clock frequency it can handle. The tools required for converting to PSM are very specific to vendor. These tools can be for embedded DSP platforms, GPP, or FPGA. While mapping PIM to PSM, Xilinx System Generator for DSP™ tool is used to handle certain hardware characteristics of the Virtex-4 FPGA. This includes digital clock manager (DCM) for handling multi-clock system and CORDIC IP core for performing certain trigonometric functions. After synthesizing, mapping and routing a bitstream file is generated for running the circuit on the FPGA. Most of these steps are integrated into the Simulink® that helps in performing the hardware co-simulation. This is useful in checking the Simulink® model in tandem with the equivalent circuit running on the hardware.

The next few sections will explain in brief about the blocks created for transmitter and receiver.

3.3.3 MIMO-OFDM transmitter

As explained previously, the USRP2 acts an RF front-end for this system with baseband processing components running on the ML403 development board. In this paper, Alamouti[5] space time block coding is used as the MIMO technique. The transmitter block diagram is shown in the Figure 3. The key module in this transmitter design is the IFFT. It defines the operating frequency of the MIMO-OFDM system. This is because the latency of the IFFT should be equal to the OFDM symbol duration corresponding to IEEE 802.11a or IEEE 802.116e. For example, in this work the IFFT/FFT is targeted for the IEEE 802.11a standard that has OFDM symbol duration as 3.2µs[6]. In order to achieve this with 60 MHz operating clock the latency of the FFT/IFFT module should be equal to 188-192 cycles. The module created in this work meets this target. In the transmitter, the known preambles are stored in the read only memories for periodic transmission from the time domain. The preambles are basically made up of two parts. One is the short training sequence that consists of ten repetitions of short symbols and the other is long training sequence that consists of two repetitive long symbols with cyclic prefix. These training sequences are based on the IEEE 802.11a standard and aid in initial packet detection, fine symbol timing synchronization and automatic gain control. The pilots aiding channel estimation and carrier phase tracking are inserted in the frequency domain i.e. before IFFT. The baseband samples from the FPGA development board are given to the USRP2 that interpolates and up-converts to the carrier frequency. This interfacing is explained in the further section.

3.3.4 MIMO-OFDM Receiver

As shown in Figure 4, the major constituents of the receiver are synchronization blocks, channel decoding block and channel estimation block. Since the Alamouti scheme is used in the transmitter the MIMO decoder block at the receiver is very simple. It just uses the combining scheme to segregate the two orthogonal symbols transmitted over two symbol durations[5]. All types of synchronization are performed in the time domain in order to avoid the latency caused by the FFT module. This is necessary in the burst transmission of IEEE 802.11a and IEEE 802.16e to shorten the time for packet detection[7]. The synchronization algorithms were dependent on the preamble sent before the OFDM data symbols[6]. Basically, it depends on correlating the received signal with the delayed version of the signal to perform packet detection and cross-correlating the received signal with the known preamble for the symbol timing detection i.e. the point at which FFT can be effectively taken[8-10]. Then the frequency offset estimation relying

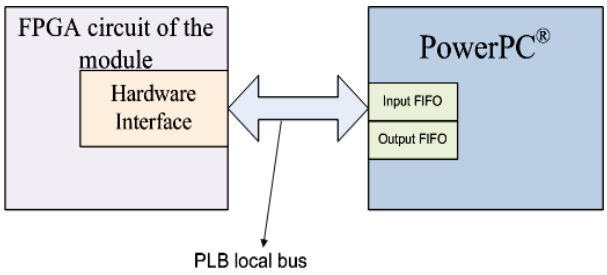

Figure 5. Interfacing between PowerPC and user IP circuit

on the principles of the packet detection and uses the CORDIC IP core of Xilinx, Inc for computing the arctangent angle. The output of the CORDIC IP core drives the numerically controlled oscillator (NCO) whose output will complex multiply the incoming signal for correction[11]. The channel estimation is performed using the pilot frequency symbols and then corrected using *the linear interpolation* technique across the other frequencies[12]. The Viterbi decoding technique is the complex module implemented at the receiver.

Logic utilization	Alamouti Encoder		Convolutional Encoder		Scrambler/ Descrambler		Resources available in XCV4FX12
	Used	Utiliza-Tion	Used	Utiliza-Tion	Used	Utiliza-tion	
Number of occupied slices	172	3%	172	3%	4	0%	5472
Number of slice flip flops	148	1%	48	0%	7	0%	10944
Number of 4-input LUT(look up tables)	253	2%	24	0%	2	0%	10944
Number of GCLKs	-	-	-	-	-	-	32
Number of DSP48s	-	-	-	-	-	-	32

Table 1. Profiling summary for the signal processing modules created using MBD design.

Logic utilization	Packet detection algorithm		Symbol timing Estimation algorithm		Viterbi decoder		Resources available in XCV4FX12
	Used	Utiliza-Tion	Used	Utiliza-Tion	Used	Utiliza-tion	
Number of occupied slices	501	9%	134	2%	3469	64%	5472
Number of slice flip flops	525	4%	137	1%	3411	30%	10944
Number of 4-input LUT(look up tables)	805	7%	241	2%	4374	41%	10944
Number of GCLKs	1	3%	1	3%	1	3%	32
Number of DSP48s	21	64%	-	-	24	75%	32

Table 2. Profiling summary for the signal processing modules created using MBD design.

Logic utilization	FFT/IFFT		Resources available in XCV4FX60	CORDIC IP		Resources available in XCV4FX12
	Used	Utiliza-Tion		Used	Utiliza-tion	
Number of occupied slices	2862	1%	25280	608	11%	5472
Number of FIFOs/RAMs	86	37%	232	1018	9%	10944
Number of 4-input LUT(look up tables)	4266	8%	50560	1042	9%	10944
Number of GCLKs	-	-	-	1	3%	32
Number of DSP48s	80	62%	128	-	-	32

Table 3. Profiling summary for the signal processing modules created using MBD design.

Timing summary in XCV4FX12	Alamouti encoder	Convolutional encoder	Scrambler/ Descrambler	Packet detection algorithm	Symbol timing estimation algorithm
Minimum period (ns)	2.888	0.759	1.179	16.446	6.485
Maximum Frequency (MHz)	346.278	1317.176	848.033	60.806	146.09
Maximum Delay (ns)	6.554	6.814	6.165	6.987	14.517

Table 4. Timing summary for the signal processing modules created using MBD design.

Timing summary in XCV4FX12	Viterbi decoder	CORDIC IP	Timing summary in XCV4FX60	FFT/IFFT
Minimum period (ns)	14.93	4.554	Minimum period (ns)	10.700
Maximum Frequency (MHz)	60.656	219.569	Maximum Frequency (MHz)	93.458
Maximum Delay (ns)	6.156	-	Maximum Delay (ns)	6.976

Table 4. Timing summary for the signal processing modules created using MBD design

4. USRP2 INTERFACING WITH THE FPGA DEVELOPMENT BOARD

This MIMO-OFDM implementation uses the USRP2 as the RF front end. The interfacing between the USRP2 and ML403 development board is achieved through the Gigabit Ethernet cable. The USRP2 runs with "Universal Hardware Driver" (UHD) that is useful in controlling the USRP2 from any kind of operating system. The FPGA of ML403 development board has a PowerPC™ 405 core that can run Linux and UHD driver for controlling the USRP2. The PowerPC™ acts as a host in this case. There is lightweight Internet protocol (LwIP) running on the PowerPC™ that helps in transmitting and receiving the user datagram protocol from/to USRP2. A simple diagram for interfacing the USRP2 with user defined baseband FPGA circuit is shown in Figure 5. The UDP data packets coming/going from/towards USRP2 is handled by the CoreConnect™ Proessor Local Bus (PLB) between the PowerPC™ and FPGA circuit.

The interfacing using LwIP protocol has severe bottle necks in terms of data rate. The maximum data rate it can achieve is 200 Kb/s. This limits the data rate that can be realized in IEEE 802.11a application. In order to overcome this, the nest step towards this work is focussed on getting the serial connector port present in the USRP2 to work with the FPGA board. Any FPGA board with serial connector port can bse used to target high bandwidth MIMO application. The serial connector can achieve a data rate of 2 Gb/s.

5. RESULTS

This section is going to talk in brief about the Table1 through Table 5. These tables' presents the profiling summary of each individual module designed using MBD tools in this work. The values present in the table are significant from the system design perspective. The information contained in the table forms as a basis for any FPGA based system design. The logic resources present in the FPGA chip determines the kind of application it can carry. The kind of logic resources present in a FPGA chip varies greatly depending on the type and manufacturer. In this work, two Xilinx FPGA devices from Virtex™ family are chosen and the models created can be ported between the two. The table indicates values for modules that played a

key role in the final design of the system. Other modules like BPSK, QPSK, QAM modulation/demodulation and channel estimation have been created, and not shown here, for the final realization of the system.

The profiling results were directly obtained from the Xilinx ISE tools. The logic resources consumption and timing summary indicates that the blocks designed using MBD tools can achieve the timing specification of the IEEE 802.11a standard. The same can be extended for IEEE 802.16e standard. The fields whose values are indicated as hyphen is not displayed by the tool. The timing summary is important to realize the maximum clock rate at which the FPGA circuit can operate. This clock rate determines the rate at which the data can be consumed and produced by the FPGA circuit.

6. CONCLUSION

The workflow presented in this paper acts as a prototyping model for SDR applications on FPGA. Most of the SDR platforms like GNURadio and OSSIE rely on the GPP for baseband processing applications. The MBD design flow presented here can be used to create a open source IP cores for deploying signal processing components on the FPGA with a USRP2 acting as an RF front end. The productivity of the MBD tools is demonstrated in this work. Creating the signal processing blocks for MIMO-OFDM transceiver took 3-4 months with a single person effort. The saving in time is substantial compared to traditional design flow. Moreover, there is no compromise on the expected system design. Additionally, the creation of PIM is very helpful in migrating to different platforms of multiple vendors. Thus, the goal of scalability, flexibility and adaptability is achieved through the device like FPGA and design flow like MBD.

7. REFERENCES

[1] J. H. Reed, *Software Radio- A Modern approach to Radio engineering*: Perason Education, 2006.

[2] R. Woods, *et al.*, *FPGA-based implementation of Signal Processing systems*: Wiley Publications, 2008.

[3] (2009). *What is JPEO JTRS?* Available: http://jpeojtrs.mil/files/org_info/JPEO_JTRS_Org_Overview.pdf

[4] (2005). *Xilinx ML40x Getting Started Tutorial*. Available: http://www.xilinx.com/support/documentation/boards_and_kits/ug083.pdf

[5] S. M. Alamouti, "A simple Transmit Diversity Technique for Wireless Communication," *IEEE Journal Selected areas of communication*, vol. 16, pp. 1451-1458, 1998.

[6] IEEE, "IEEE 802.11a-1999 Wireless LAN Medium Access Control (MAC) and Physical layer (PHY) specifications-High speed physical layer in the 5 GHz band," ed, 1999.

[7] M. Janakiraman, *Space-time codes and MIMO systems*: Artech house universal personal communications, 2004.

[8] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*: Sams Publishing, 2002.

[9] T. M. Schimdl and D. C. Cox, "Robust Frequency and Timing Synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, pp. 1613-1621, 1997.

[10] T. Keller, *et al.*, "Orthogonal Frequency Division Multiplex Synchronization Techniques for Frequency-selective Fading Channels," *IEEE Journal on selected areas of wireless communication*, vol. 19, pp. 999-1008, 2001.

[11] T. D. Chiueh and P. Y. Tsai, *OFDM Baseband Receiver Design for Wireless Communication*: John Wiley and Sons, 2007.

[12] S. Coleri, *et al.*, "Channel Estimation Techniques based on Pilot Arrangement in OFDM systems," *IEEE Transactions on Broadcasting*, vol. 48, pp. 223-229, 2002.

[13] *OSSIE:Open Source SCA Implementation Embedded*. Available: http://www.ossie.wireless.vt.edu

[14] M. Carrick, "Logical representation of FPGAs and FPGA circuits within SCA," Master of Science, Virginia Polytechnic and State Unversity, Blacksburg, VA 2009.