AC 2011-2674: ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM) DEVELOPMENT AND TEACHING PLATFORM

Antonio Francisco Mondragon-Torres, Rochester Institute of Technology

Antonio F. Mondragon-Torres received the B.Sc. degree with honors from Universidad Iberoamericana, Mexico City, Mexico, the M.Sc. degree from Universidad Nacional Autnoma de Mexico, Mexico City, Mexico, and the Ph.D. degree (as a Fullbright-CONACYT scholarship recipient) from Texas A&M University, College Station; all degrees in Electrical Engineering in 1990, 1996, and 2002, respectively. From 1988 to 1995, he worked in a telecommunications company TVSCOM, Mexico City, Mexico, designing teletext products, first as a Design Engineer and later as a Design Manager. In 1995, he joined the Mechanical and Electrical Department, Universidad Iberoamericana as an Associate Professor. From 2002 through 2008 he was with the DSPS R&D Center's Mobile Wireless Communications Technology branch, Texas Instruments Dallas, TX and in 2008 he moved to the nanoMeter Analog Integration Wireless branch where he worked as Analog IP verification technical lead. In 2009 he worked for Intel Guadalajara, Design Center in Mexico as Front-End/Back-End technical lead. In 2009 he joined the Electrical, Computer and Telecommunications Engineering Technology Department at the Rochester Institute of technology where he currently is a tenured track assistant professor. His research interests are analog and digital integrated circuit implementation of communications systems, and System-on-a-Chip methodologies.

Mahesh Nandan Kommi, Rochester Institute of Technology

M.S in Telecommunication Engineering Technology from Rochester Institute of Technology, NY, USA
B.E in Instrumentation Engineering from Muffakham Jah College of Engineering and Technology, AP, India

Tamoghna Bhattacharya, Rochester Institute of Technology

Graduate Student in Telecommunications Engineering Technology

Orthogonal Frequency Division Multiplexing (OFDM)
Development and Teaching Platform

Abstract

This paper describes a work in progress on a digital baseband communications development and teaching platform based on Orthogonal Frequency Division Multiplexing (OFDM) modulation. The objective of this platform is to allow undergraduate and graduate students at different levels in their engineering technology programs to interact with a complete digital baseband communications system to understand explore and analyze the different components required. For senior and graduate level courses, students will have the opportunity to modify the platform to: try their own implementations; perform hardware acceleration; analyze finite precision implementation effects. While this system is generic and it is not currently tied to any particular OFDM communications standard, it will be open to the academic community to allow modifications and improvements.

Introduction

The computer engineering technology program has a very robust course sequence in digital and embedded systems design with a strong emphasis on hands-on experience for the students. The students in our program are currently being exposed to digital design using hardware description languages (HDL) such as VHDL, as well as software and firmware code design using high-level languages such as C and C++.

One industry trend is to use Field Programmable Gate Arrays (FPGA) as hardware verification platforms for digital communications and digital signal processing systems. In commercial products, the final implementation of these systems will be most likely targeted to an Application Specific Integrated Circuit (ASIC) due to final cost under volume production. For architecture exploration, design tradeoffs and concept feasibility demonstration, an FPGA is the ideal platform for proof of concept and hardware acceleration.

Orthogonal Frequency Division Multiplexing (OFDM) [1-3] is a modulation technique that is not new, but the technology required for its implementation has evolved over the last ten years to make it now practical. Nowadays a large number of communication standards (e.g. DSL, IEEE 802.16e (WiMax), 3GPP LTE, DVB-H, 802.11a and 802.11n) have adopted this modulation technique due to its robustness and performance advantage over other communication techniques.

The heart of the OFDM modulation technique lies in the use of the Fast Fourier Transform (FFT), which is a very structured algorithm to convert a time domain signal into the frequency domain and by taking the inverse FFT (IFFT) can be transformed back into the time domain. In Figure 1, a complete digital communication system that employs OFDM modulation is shown [1].

The approach in OFDM systems is to have digital information encoded by traditional phase modulation technique such as Quadrature Amplitude Modulation (QAM). This modulation

technique maps a series of bits into symbols. The number of symbols used for each OFDM frame is traditionally a power of two. Then the IFFT of a block is performed on the frame to convert it back into a time domain representation that can be further processed and sent through the transmitter chain and through the antenna. On the receiver side the process is reversed after frame synchronization by taking the FFT of the received block and obtaining an estimate of the QAM symbols which are mapped back into a series of bits. This sounds pretty straightforward but there are many subtle details that could be investigated in terms of the effects of: distortion, channel noise, multipath propagation, fading, Doppler shift, synchronization, etc.

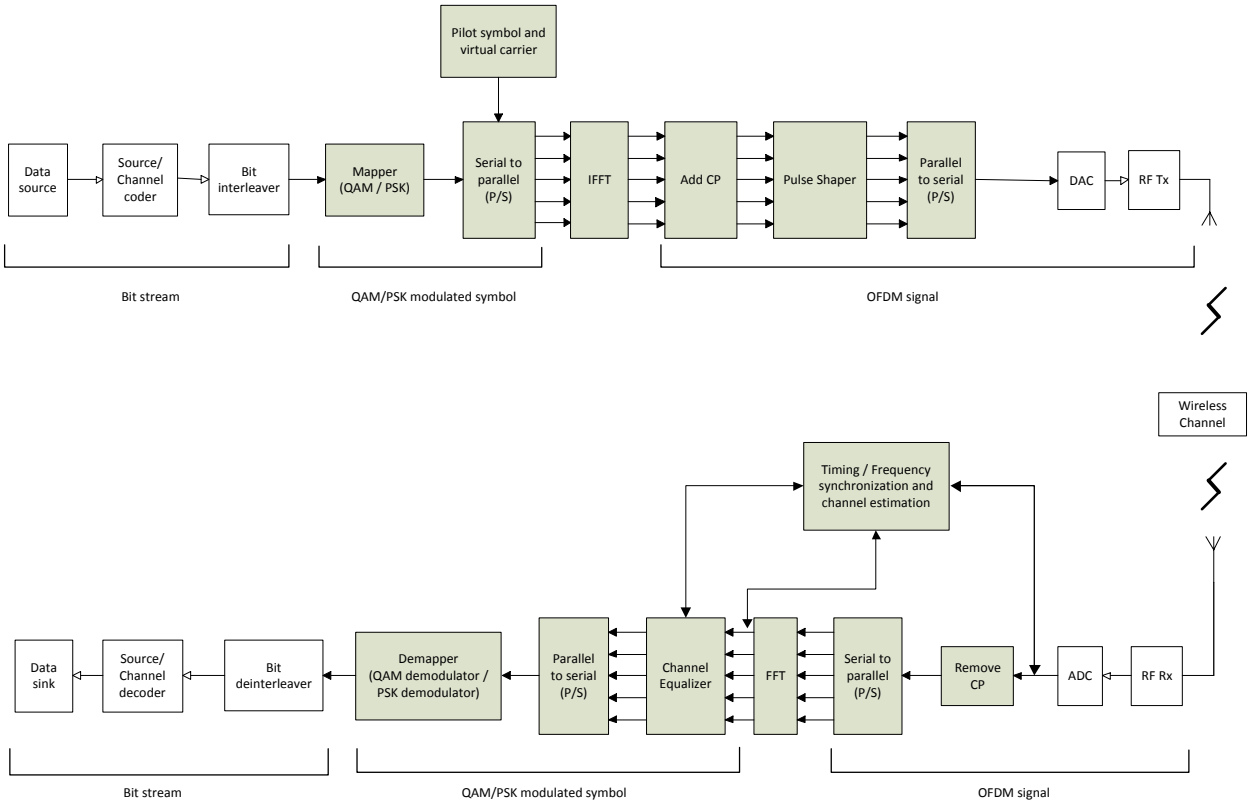

Figure 1. Digital communications system using OFDM modulation.

The objective in the development of this platform is to allow the students to work in related courses on particular signal processing blocks at different level of abstraction. These could be divided into:

- ***Algorithmic:*** Use of Matlab/Simulink and Altera-DSP Builder or Xilinx-System Generator to create a system that can be selectively run on the hardware platform. This could be the objective of a digital communications or digital signal-processing course.
- ***Electronic System Level:*** Convert an architectural untimed C coded algorithm to Register Transfer Logic (RTL) using Mentor Graphics CatapultC, or equivalent tool. This could be the objective of an advanced digital systems design course that could allow the student to explore different architectures and trade-offs in terms of area, power and speed.
- ***RTL Level:*** Design an efficient architectural level description of the algorithm implemented in an HDL language. This could be the objective of a regular course where

different blocks in the transceiver chain could be specified, designed and their performance be measured against a reference model.

The outcome of the implementation of this development platform should be aa stable OFDM reference system that will allow the students to interact at different levels of abstraction b; to obtain a vision of a whole modern digital communications system and c: to understand how the changes made by them could have an impact on the overall system performance. Further reports and assessments will be generated once the platform is completed and under assignment to different courses. The experience of a real implementation will allow the students to separate the theory from the implementation and to evaluate the real effects on an algorithm's hardware implementation.

Advantages of the OFDM scheme

As mentioned before, OFDM modulation is not a new technique but advances on the technology had just recently made it possible to implement it in commercial products. OFDM brings a unified concept where a very important concept such as Fourier decomposition is being used as the media for transmitting and receiving information. A discrete form of the Fourier transform (DFT) has been used in digital signal processing algorithms for many years as a foundation to perform diverse algorithms and methods. A fast implementation relies on the structure of the DFT to drastically improve its computational complexity; this is called the Fast Fourier Transform and has generated large momentum to create efficient architectures for its computation and to find applications to its use. On top of that, many wired and wireless fixed and mobile communications systems have chosen OFDM as their modulation and multiplexing standard. Most fourth generation cellular systems and wireless local area networks have adopted this scheme.

OFDM is simple enough to understand, but at the same time is one of the most reliable modulation methods. In addition, OFDM is robust to many of the unwanted effects in communication systems. An example of this modulation scheme can be understood and implemented using platforms such as Matlab and SystemVue where the students can explore and understand all the processes involved in its implementation with simple commands. This is contrary to other digital communications methods that have evolved such as CDMA which is not easy to understand without a strong background in digital communications.

Scope of the development and teaching platform

What this work intends to carry out, is to develop a platform where students could explore, experiment and implement their own ideas. Of course any text book or article can explain how OFDM works and how it is being implemented, but here we want to go one step beyond and offer the student a unified way to interface with a complete system.

The main focus of our research, tries to answer the following question: How can we provide a generic platform in which students can explore hardware implementation of digital signal and communications algorithms? Including: data movement in interleavers; digital filtering with finite and infinite impulse response filters; signal decomposition using transforms; frame synchronization using autocorrelations; phase rotations using complex multiplications, etc.

In this platform we have designed a Simulink model of the entire OFDM communication system. Now it is possible for a student to pick a block or multiple blocks from the OFDM Simulink model, implement it in hardware and loop it back to the Simulink model. The hardware implementation could be done for example either by using DSP builder or Catapult C. Working on DSP builder would expose students to challenges faced in designing and developing systems in Simulink and performing hardware in loop implementation. Extending their design to implement on Catapult C would expose them to challenges faced in developing hardware implementation in higher level language, a thorough understanding about the architectural constraints, make them understand how different architectures like pipelining make a difference to the design and also give a thorough understanding about different tradeoffs in designing the system. While this could be done on any system, integrating the design developed by the students to the actual OFDM Simulink model would give a chance to the students to get a feel of how a particular design of a particular block affects the performance of the entire system. Developing a hardware implementation of the entire system would give the students a deep insight of the OFDM concepts and would allow them to extend their research to various other application of the OFDM communication system.

The platform is being developed in Simulink which is a block level system design framework that has time awareness. So what are the possibilities opened by using Simulink? First, we have a platform that has time dependency as real systems do, at the same time the Simulink blocks can be designed using Matlab code that is more algorithmic focused and do not have an embedded time relationship. Since the focus of the project is on hardware design, exploration and implementation, the Simulink blocks can be implemented by FPGA vendor libraries, C or HDL language all which could run natively on hardware platforms.

This platform could also be developed on an ESL tool called Catapult C. Using an ESL tool a student would code the system in a high level language either in C or C++ and change the architectural constraints depending on the necessity from the features available in the tool. This tool is easy to implement and very efficient in comparing the performance of different architectural models of a system [4].

This platform would allow us to implement some of the ideas proposed in an earlier work on a wireless communications systems course as detailed in [5].

Hardware levels of abstraction

System level design is an area which is encouraged when new systems are designed. This allows making high level and architectural decisions at early stages in the development. Hardware development is somewhat similar to system level design since HDL languages allow different levels of abstraction to describe the hardware components. These levels are: behavioral level, register transfer level (RTL) and structural level. Where there is a coarse functional description at behavioral, an architectural cycle accurate description at RTL and a hardware resource dependent at the structural.

While all these are related to hardware development, we can find similar concepts on the system level platform, where the hardware level of granularity or detail can be refined by different levels of Implementation.

Hardware levels of implementation

As mentioned before in the introduction, the platform purpose is threefold. Generate automatic HDL code from FPGA vendor libraries, with possibility to run as hardware accelerators; use ESL tools to generate a more controlled hardware architecture from a C/C++ untimed description; and finally to architect and write the HDL code. All these three could be integrated into the signal flow to validate processing speed as well as interfacing to a complete system

Automatic HDL code generation

Let's take a closer look at the first level of implementation which is generating automatic HDL code from a Simulink model. Each block or a set of few blocks of the entire communication system can be implemented on hardware. So far, we have used an Altera Stratix III FPGA to do system level hardware testing of the Fast Fourier Transform block in the OFDM communication model. To do this we have used Hardware in Loop (HIL) block provided by the DSP builder Altera library. This block acts as a link between Simulink and the actual hardware we want to configure.

In modern digital communication systems, the current trend is to implement a pipelined FFT to generate orthogonal sub-carriers. A pipelined FFT generate an output every clock cycle which helps in real-time applications like digital communication systems where data is being continuously fed. We have designed Simulink models to implement FFT using butterfly diagrams which use simple Simulink blocks as well as pipelined FFT which use the advanced block set from DSP Builder. In this paper we are going to talk more about the pipelined FFT for the above mentioned reasons. For more information on the architecture of the pipelined FFT implemented refer to [6].

The hardware implementation was done using the Altera's Quartus II version 10.1 and DSP Builder version 10.1. Care must be taken to properly design a Simulink model which would involve block sets from both advanced and standard block sets of DSP Builder. We created this model in layers. The lower level consists of the device block which has the information about the FPGA available in the hardware platform (Stratix III) and the functional blocks that essentially form the FFT. However, on the top level we could only use the signal and control blocks from the advanced block set and other blocks have to be at the lowest level in the design hierarchy. We make use of the signal compiler and testbench from the standard block set on the top level. The signal compiler is used for creating a Quartus II project, start synthesis, to launch place and route after generating the HDL code. The testbench is used to compare the block level simulations in Simulink and the HDL simulations using Modelsim. Input and output blocks are inserted before and after the subsystem that contains the advanced block set. These blocks have external type parameters to convert from floating or other format handled by Simulink to fixed point as FPGA implementations can only be configured for fixed point. These blocks act as boundaries to the advanced and basic block sets. The procedure to convert the FFT model to HDL, configure the FPGA with the HDL code, and running it from Simulink is detailed below.

We first run the signal compiler block on the top level to generate HDL code and create a Quartus II project. Then compile the design with Quartus II using the compile option in the signal compiler block. We have now created a Quartus II project for the model and synthesized the HDL code for the same. Now save a copy of this model and instantiate a HIL block on the

top layer of the new model from the Altera DSP Builder library found in the standard block set. Open the HIL block and copy the Quartus II project that was earlier created into the file path. This would generate proper ports for the HIL block. Connect these ports to the appropriate signals. Configure the simulation in burst mode to observe high speed of simulation. In the next menu entry of the HIL block, compile the Quartus II project again, scan JTAG in order to recognize the FPGA device and program it. If we simulate this model it runs at a remarkable speed when compared with the native Simulink simulation. Figure 2 below shows the model which has the advanced block set replaced with a HIL block. This example was modified from the one supplied by Altera to run the FFT on the FPGA platform and to be controlled by the Simulink simulation. We are in the process of converting some other algorithms into hardware following the same methodology to be able to create custom hardware acceleration blocks [7].

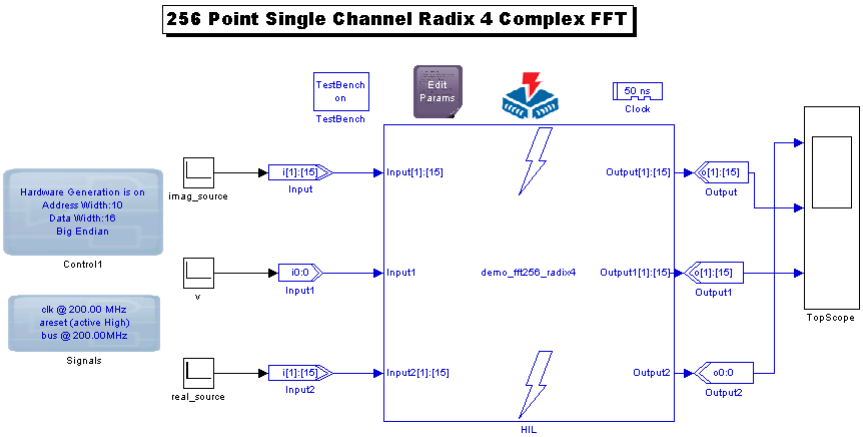

Figure 2. Hardware In the Loop (HIL) Simulink simulation, actual code runs on the FPGA.

Electronic system level (ESL) tools

While the previous approach gets the job done in terms of hardware generation and acceleration, it does not contain the means to explore different architectures using a single source code, all synchronization and data flow has to be embedded into the Simulink block, and this means that the architecture gets fixed by the implementation, but the actual hardware is library dependent. Another approach which has been gaining popularity among hardware designers and companies in the design and implementation of digital communications and signal processing applications is the use of Electronic System Level tools or ESL.

So what is the main difference between the first approach using a library of signal processing code generator blocks and ESL? The short answer is QoR or Quality of Results! ESL tools allow the designer or architect to explore different tradeoffs in the design such as: area, power, timing and resource allocation and sharing. In Figure 3 we show the Gantt diagram or the equivalent of a hardware schedule for the blocks used in a FFT. It can be appreciated that memory access, add and multiply operations are detailed as well as the sequence of operations. In addition, this is an example of the first solution explored.

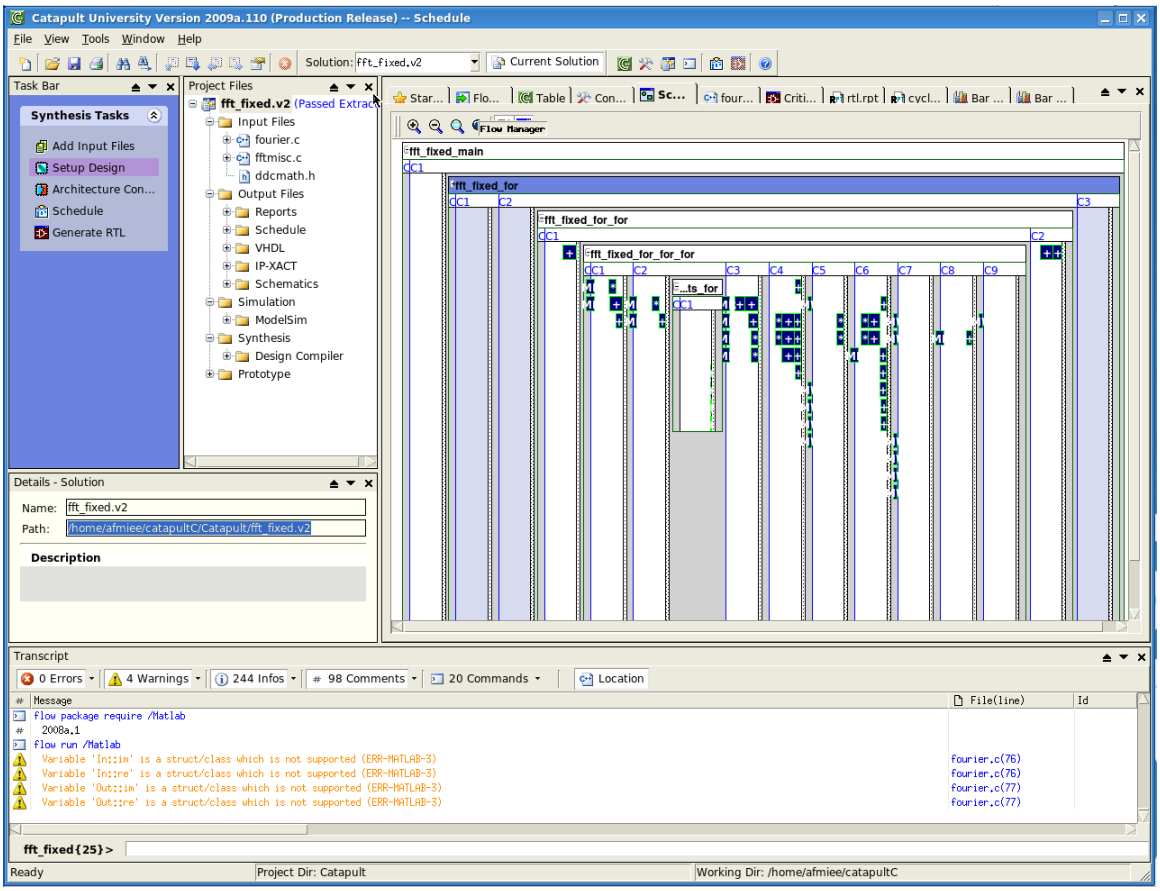

Figure 3. ESL Mentor Graphics CatapultC C++ to RTL tool.

There exist options to roll and unroll loops which will create a serial or a massive parallel implementation, allocate a fixed number of clock cycles to perform the operation, pipeline the operations, target area, and target performance among many other parameters, the designer can now compare different solutions and their architectural tradeoffs. An example of these parameters is shown in Figure 4.

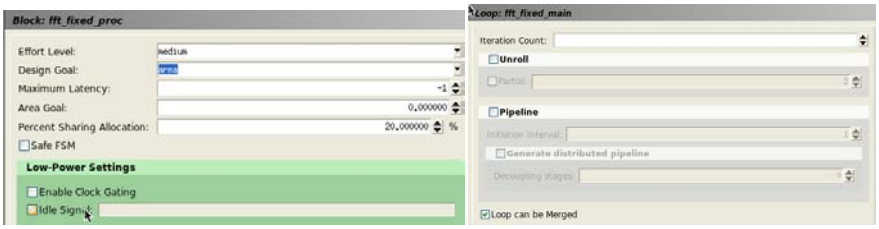

Figure 4. Examples of some architectural tradeoffs available on ESL tools.

HDL implementation

While previous approaches may be used to understand, explore and investigate different components used in digital communication systems. The most traditional approach is HDL coding where the students actually get their hands-on designing and coding different hardware blocks. Depending on their level as undergraduates or graduates, the challenge can be presented from designing a simple FIR filter using a straight architectural approach or some other configuration, to actually coding the FFT in a HDL.

Platform access

The student has access to Matlab, Quartus II, Modelsim and other tools available in the laboratories. To access the OFDM development platform, the student should use a version control system to checkout from the version control system the latest version of the OFDM platform into a working local copy. The student should be able to start simulations and interact with the system without worrying about disturbing the original setup, since he/she will be working on a local copy. Once the design is stable and ready for hardware implementation, he/she could locally or remotely connect to one of the FPGA platforms available to program their designs. The same platform could be offered in both Windows and Linux since the tools are available in both platforms.

Lessons learnt from developing this platform

Working on this platform has given us an opportunity to build a deep insight into what is happening in an OFDM communication system. While developing a system on various tools is nothing new, implementing the developed system on hardware has given us hands on experience on getting the designed model to work in real time. We have understood the merits and demerits in using a particular architecture in terms of performance, silicon area, and latency besides other factors. Overall it was an excellent platform to take a step beyond designing to implement the designed system in real time environment.

This platform has been evolving, maturing and being adapted to some of the EDA and ESL available technologies in the department of Electrical, Computer and Telecommunications Engineering Technology at the Rochester Institute of Technology. We have started to see this methodology to grow interest among other researchers at the institute and that is the reason this project could be a good startup to understand, explore and implement other algorithms. We will continue working on the project until the complete OFDM platform is completed and tested, after that it may be open to other communities willing to contribute to the platform.

Conclusions and future work

This work in progress described a digital baseband communications development and teaching platform based on the Orthogonal Frequency Division Multiplexing (OFDM) modulation. The objective of this platform is to allow students at different levels in their engineering technology programs to interact with a complete digital baseband communications system for exploration and analysis of the different components required.

A complete Simulink model is presented to the students. Students pick one or multiple functional blocks in the Simulink model of the OFDM system and implement it on hardware. This hardware implementation is now looped back to the Simulink model and various tests are done to evaluate the performance and correctness of the hardware implementation.

As mentioned before the heart of the OFDM modulation technique lies in the use of the Fast Fourier Transform. Hence as an example we have chosen to implement the FFT block of the OFDM communication system on hardware and loop it back to the Simulink model of the OFDM system. Students can extend this by picking any other block from the OFDM Simulink model and implementing it on hardware or follow this paper to implement the FFT block.

References

[1] Y. S. Cho, *MIMO-OFDM wireless communications with MATLAB*: Singapore ; Hoboken, NJ : IEEE Press : J. Wiley & Sons (Asia), c2010, 2010.

[2] A. R. S. Bahai, *Multi-carrier Digital Communications: Theory And Applications Of OFDM*: Springer, 2004.

[3] L. L. Hanzo, *OFDM and MC-CDMA: A Primer*: Wiley-IEEE Press, 2006.

[4] G. Martin, B. Bailey, and A. Piziali, *ESL Design and Verification*. San Francisco, CA: Morgan Kaufmann, 2007.

[5] S. Guzelgoz and H. Arslan, "A Wireless Communications Systems Laboratory Course," *Education, IEEE Transactions on*, vol. 53, pp. 532-541, 2010.

[6] H. Shousheng and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Custom Integrated Circuits Conference, 1998. Proceedings of the IEEE 1998*, 1998, pp. 131-134.

[7] A. Corporation, "An OFDM FFT Kernel for Wireless Applications," 2007.