SDC Testbed: Software Defined Communications Testbed for Wireless Radio and Optical Networking

Boris Shishkin, Doug Pfeil, Danh Nguyen, Kevin Wanuga, James Chacko,
Jeremy Johnson*, Nagarajan Kandasamy, Timothy P. Kurzweg, Kapil R. Dandekar
Department of Electrical and Computer Engineering
Department of Computer Science*
Drexel University
Philadelphia, PA 19104
Email: {bs44,dsp36,dhn24,ktw25,jjc652,jjohnson,kandasamy,kurzweg,dandekar}@drexel.edu

***Abstract*—This paper describes the development of a new Software Defined Communications (SDC) testbed architecture. SDC aims to generalize the area of software defined radio to include propagation media not exclusively limited to radio frequencies (optical, ultrasonic, etc.). This SDC platform leverages existing and custom hardware in combination with reference software applications in order to provide a complete research and development platform. This platform can be used to implement current and future standards that make use of highly demanding communications techniques, including ultrawideband (UWB) radio and free-space optical communications. This paper describes the commercial and custom hardware that is being integrated into the platform, including the baseband hardware and the modular transceiver frontends. Furthermore, the paper describes the software development currently in progress with this platform, including the integration of available open source designs into the platform, and the development of custom IP for scalable OFDM PHY implementations in radio and optical communications. We seek to create a complete research platform for the commercial and academic wireless communities, capable of delivering the highest possible performance and flexibility while providing the necessary development tools and reference designs in order to minimize system learning curve and development cost.**

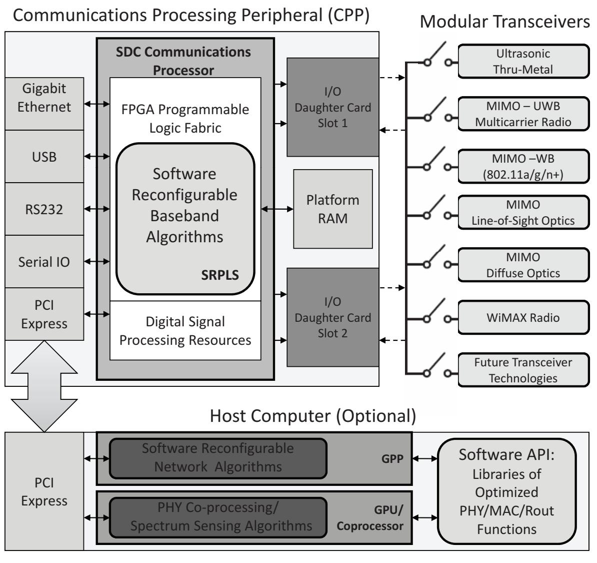
Communications Processing Peripheral (CPP)
Modular Transceivers
SDC Communications Processor
Gigabit Ethernet
USB
RS232
Serial IO
PCI Express
FPGA Programmable Logic Fabric
Software Reconfigurable Baseband Algorithms
SRPLS
Digital Signal Processing Resources
I/O Daughter Card Slot 1
Platform RAM
I/O Daughter Card Slot 2
Ultrasonic Thru-Metal
MIMO – UWB Multicarrier Radio
MIMO –WB (802.11a/g/n+)
MIMO Line-of-Sight Optics
MIMO Diffuse Optics
WiMAX Radio
Future Transceiver Technologies
Host Computer (Optional)
PCI Express
Software Reconfigurable Network Algorithms
GPP
PHY Co-processing/ Spectrum Sensing Algorithms
GPU/ Coprocessor
Software API: Libraries of Optimized PHY/MAC/Rout Functions
Software Defined Communications (SDC) Testbed overview

Fig. 1. Software Communications Testbed Conceptual Diagram

I. Introduction

The Software Defined Communications (SDC) Testbed seeks to provide a research and development platform capable of designing and prototyping next generation wireless communication standards which make use of radio, optical, and other communications modalities. The vision for this platform builds on the current principles of Software Defined Radio (SDR). Specifically, it extends this concept through utilization of propagation media beyond the RF spectrum while meeting the high bandwidth demands of emerging communications standards. The project will provide an open source tool for both the commercial and academic wireless research communities. Furthermore, it is designed to provide a cohesive and affordable hardware/software infrastructure for fast and flexible algorithm development across multiple layers of the communication stack.

The platform targets a variety of application scenarios. These scenarios can be simple one or two node applications such as channel sounding, localization, and point to point communication. Alternatively, multi-node applications such as ad-hoc, sensor, and cognitive networks are also supported. Furthermore, as multiple input multiple output (MIMO) becomes a ubiquitous technique in wideband wireless standards, the need for wireless testbeds to support it becomes apparent. Multiple parallel frontends are incorporated into the SDC platform to provide the first testbed that can perform MIMO ultrawideband (UWB) and MIMO diffuse optical communications. We envision that the testbed can leverage these capabilities towards the development and evaluation of new standards and protocols.

A high speed host interface and the latest processing technology provides a new level of integration between the processing resources on the platform and those available on the host, tightly coupling processing technologies that were previously underutilized and independent. Thus, we envision that the SDC platform can be used to implement algorithms (e.g., cognitive dynamic spectrum access) not only on the resident FPGA, but also on CPU and GPU resources available on a host computer. Modular design, upgradability, and utilization of industry standard interfaces are also key requirements in the design of the platform. These factors allow the SDC testbed to take full advantage of hardware advancements and scale with the financial constraints and requirements of the end user.

Figure 1 shows the major components of the platform, including interchangeable frontends, the Communications Processing Peripheral (CPP), and a host platform. The host platform is connected to the SDC testbed via a high bandwidth serial interface, which can be used to extend the capabilities of the SDC by providing additional processing resources and higher layer management functions at the discretion of the user. A variety of processing technologies are available to the user through a combination of FPGA based resources on the CPP and GPP, or GPU and CPU based resources on the host. Through use of the appropriate transceiver frontends, the platform can access a variety of communications mediums, including RF, optical, and ultrasonic. MIMO capabilities on the platform are ensured by providing at least two communication interfaces per CPP.

II. Platform Design Strategy

The platform is based on a culmination of experience gathered from previous SDR platforms (i.e., HYDRA [1], gnuRadio USRP/USRP2 [2], and WARP [3]). While these platforms all leverage open source design and significant community involvement, some common limitations include host interface bottlenecks, insufficient processing resources on the platform, and custom peripheral interfaces which cannot evolve with advances in hardware. Current SDR platforms also show a relative disparity between available processor technology and what technology is actually implemented in the SDR. Hardware advancements in processor technologies and manufacturing processes are rapid and platforms that are hardware specific tend to become outdated quickly, sometimes becoming obsolete before they enter the market. Emerging SDR platforms such as Microsoft SORA [4] attempt to remedy some of these shortcomings, however they tend to employ proprietary IP or interfaces. This significantly limits their flexibility to the end user and makes them less appealing for resource intensive applications.

The SDC testbed is being designed with these lessons in mind. It utilizes early release reference hardware for processing, and is able to migrate to the newest hardware as it becomes available. The platform incorporates host and radio peripheral interfaces that are backed by commercial standards and ensure longevity and interoperability for the foreseeable future. Finally the platform builds on a strong software infrastructure pioneered by other open source SDR platforms, utilizing market standard tools, and IP provided directly by the hardware manufacturer. This minimizes licensing constraints and opens the platform to the largest possible user base.

Figure 2 shows both the commercial hardware that is currently being used for this project as well as the custom hardware that is being developed to expand the capabilities of the platform. The details of the individual hardware elements are described in the following two sections.

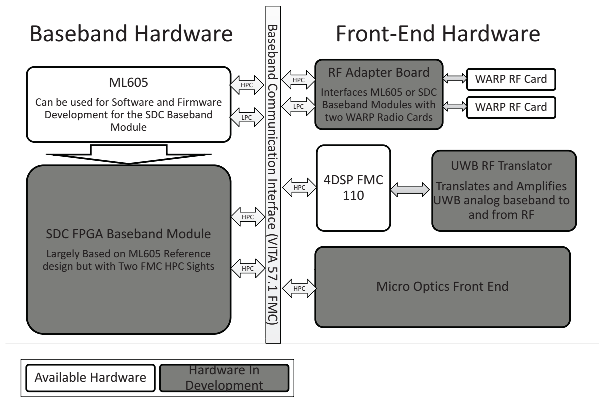
Baseband Hardware
Front-End Hardware
Baseband Communication Interface (VITA 57.1 FMC)
ML605
Can be used for Software and Firmware Development for the SDC Baseband Module
HPC
LPC
SDC FPGA Baseband Module
Largely Based on ML605 Reference design but with Two FMC HPC Sights
RF Adapter Board
Interfaces ML605 or SDC Baseband Modules with two WARP Radio Cards
WARP RF Card
WARP RF Card
4DSP FMC 110
UWB RF Translator
Translates and Amplifies UWB analog baseband to and from RF
Micro Optics Front End
Available Hardware
Hardware In Development

Fig. 2. SDC Hardware Diagram

III. Baseband Development

While other processing technologies such as GPP and DSP processors provide flexibility, targeting FPGA technology for communications algorithms allows for the highest level of parallelization and pipelining, providing the most efficient path for a reconfigurable communications platform with high bandwidth demands. Targeting FPGA technology for the CPP portion of the SDC however does not preclude the use of other processing resources, and tight coupling between the CPP and the host computer ensures that the end user can dictate which portions of the protocol stack are implemented in the FPGA and which are performed using GPP or other co-processors on the host. The open nature of the platform and this added flexibility would allow the user to implement established software APIs, and GPP optimized functions used in other open source projects. It would be possible, for instance, to leverage the optimized functions of the GNU Radio project instead of developing functions specific to the SDC. Other host APIs (e.g., SORA [4]) could also be easily leveraged if their licensing allows for implementation on third party hardware.

To implement the CPP we have been targeting the most current FPGA technology, which reduces migration time to next generation FPGA technology as it becomes available. For this purpose we selected the Xilinx ML605 FPGA based PCIe carrier [5] as the initial CPP. Notable characteristics of the ML605 include a PCIe 8x interface, two FMC module sites that have sufficient IO to facilitate integration of communication modules, a DDR3 memory interface, and a large Virtex-6 240 FPGA which dwarfs the resources provided by most existing SDR platforms. The availability of this piece of hardware signaled a level of maturity in FPGA development boards that allowed them to directly compete with the capabilities of established SDRs, and outperform them in terms of processing resources.

The ML605 FPGA has the necessary fabric to implement high order modalities such as 4x4 MIMO, and real time highly demanding modulations such as multicarrier UWB. Moreover, it provides an excess of FPGA space that is integral to new algorithm development (e.g., spectrum sensing, cognitive radio, and real time closed loop techniques using SVD [6]). An abundance of serial IO on the FPGA allows for the

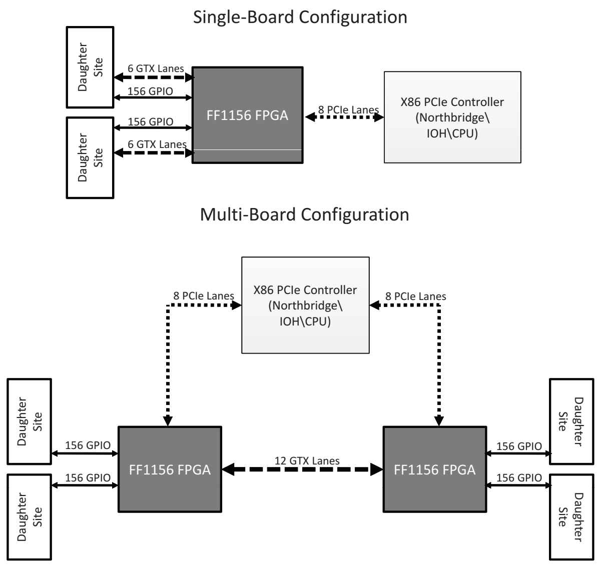

Fig. 3. SDC Single and Multi Board Configuration Diagram

implementation of simultaneous PCIe interfaces and FPGA to FPGA serial buses, enabling multiple platforms to be directly linked while being housed in the same PC chassis. Figure 3 illustrates how these links can be used for communicating with peripherals, or to allow multi-board configurations. Finally, the FPGA in the ML605 is pin for pin compatible with 4 other Virtex-6 models, giving the end user the flexibility to choose a size that meets their targeted applications, and allows for the potential to double the resources available on the current ML605.

While the industry standard PCI Express interface allows the platform to closely integrate with the the host, an industry standard VITA 57.1 peripheral interface allows interoperability of communications peripherals such as commercial and custom frontends. The current version of the ML605 has two such interfaces, one of which allows for the integration of high pin count frontends, while the second allows for low pin count frontends. However with minor hardware modifications the second interface can be expanded to facilitate a second high pin count frontend. Xilinx provides all necessary design files in order to facilitate this change and IO enumeration as well as routing estimates show that this modification can be performed without changing the stackup of the ML605. This board would be largely identical to the ML605 for SDC applications but would allow for additional development in MIMO configurations. Drexel University is working with Xilinx to make this modification possible as part of the SDC project.

New reference platforms are also in development at Xilinx in support of their next generation Virtex-7 FPGA technology. Because of the processor similarities between the current Virtex-6 and the next generation processor, the work currently in progress will be directly relevant in the next generation platform. Similarly, just as this project utilizes the ML605 for SDC applications, it will be able to do the same with the next generation development platform which will share the host and peripheral interfaces with the current platform.

IV. Frontend Transceiver Development

Current frontend transceiver development is based around the VITA 57.1 FMC module standard. As an industry accepted standard this interface allows for longevity, and system interoperability between different CPP modules and frontends, much more so than any custom peripheral interface. This is a critical aspect of this platform that differentiates it from custom SDR implementations.

Up to 160 GPIO links are possible through this interface, which can be used for single-ended or differential signaling. By augmenting the GPIO resources, up to 10 lanes of bidirectional high speed serial data are available for chip to chip communications through the interface. These can provide up to 50Gbps of bidirectional signaling bandwidth between the main system FPGA and peripheral devices housed in the communication site.

A series of ADC/DAC modules are commercially available for this interface and a number of companies such as 4DSP, Avnet, Lyrtech, Xilinx, and Curtiss Wright are developing and producing hardware that is fully compatible with the current reference hardware and future carrier boards. These include boards designed specifically for this project as well as future development boards employing next generation processor technology.

A. Conventional Wideband RF Frontend

For conventional wideband communications (e.g., 802.11) the testbed can use commercially available RF frontends that interface directly to the digital VITA 57.1 FMC interface, or do so through an adapter board. The WARP radio card, which is normally used with its own testbed [7], is an excellent candidate for integration into the SDC as a 802.11-style RF frontend. It is a flexible and well established RF transition platform that has a fully digital interface and onboard digital to analog conversion.

An adapter board is currently being developed to interface the ML605 and any future CPPs with the WARP radio frontends. The adapter board interfaces with both FMC connectors on the CPP and has sites to house two WARP radio cards and a WARP clock board. Voltage translation on the adapter board is performed between the 2.5V domain on which the ML605 interface operates, and the 3.3V domain of the WARP peripherals. A block diagram of the adapter is shown in Figure 4.

B. Ultra Wideband RF Frontend

The UWB implementation in the SDC is made possible through the combination of the CPP (ML605), with a commercially available FMC110 high speed ADC/DAC module developed by 4DSP [8], in conjunction with a custom-built UWB RF transceiver. The UWB RF transceiver is designed to generate an analog radio frequency signal from the digital waveform provided by the ML605. The RF signal is designed loosely around the European Computer Manufacturers Association (ECMA) standard [9] that better describes the necessary specifications of the radio frontend.

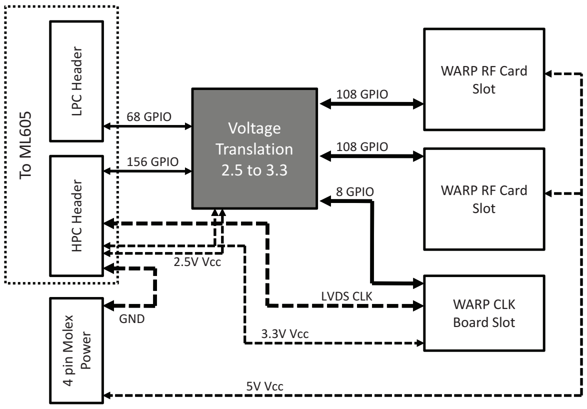

Fig. 4. Block Diagram of RF Adaptor board

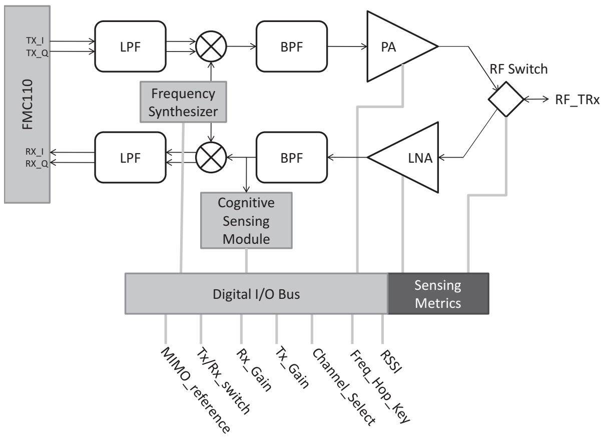

Fig. 5. UWB radio frequency transceiver module

The current system design separates the A/D and D/A conversion, and the radio frequency up-conversion into two separate systems. The analog and digital conversion is managed by the FMC 110 module which interfaces directly to the FMC-HPC interface on the ML605. The board has a 12-bit dual channel ADC and a 16-bit dual channel DAC. Both can process samples at a rate of 1Gsps. Additionally, support is provided for external reference clocks and triggers to synchronize multiple boards for MIMO communications.

The radio frequency up/down converter can be seen in figure 5. As the diagram shows, the up-converter is a basic heterodyne direct up/down converter. Transmit and receive selection are manageable from external digital I/O. Variable gain amplifiers are included to allow for user control of both transmit power and receive gain. Channel selection is managed through programming of the dedicated frequency synthesizer. These boards can also make use of external reference clock signals to synchronize multiple frontends for MIMO communications. Filtering is added to minimize spurious and harmonic emissions.

The cognitive sensing module is being designed for testing and evaluation of cognitive dynamic spectrum access (DSA) algorithms. The goal of these algorithms is to make use of local sensing and adapt the communications link according to the information gained. The initial design of the sensing module is simply a heterodyne down-converter with a parallel filter bank and basic parallel processing architecture. This module would allow for multiple channels to be sensed in parallel. Repeated sensing will allow for intelligent selection based on statistical models of channel occupancy.

C. Free Space Optical Frontends

Initial optical frontend development has focused on the development of transceivers to implement optical Alamouti coding [10], [11]. These transceivers make use of MIMO techniques commonly used in RF communications in addition to space-time coding techniques to mitigate crosstalk, as well as misalignment for small scale optical systems, without the need for lenses or other cumbersome optical components.

Transmission is done using On-Off Keying, which is fed to the optical frontend via differential GPIO or through the Virtex-6 Gigabit Transceivers. At the receiver, the data received by each photodiode is passed through a transimpedance amplifier, which converts the received current signal into a voltage signal. Additional amplification is needed and the signal is then converted using a high speed analog to digital converter (ADC). The ADC at the receiver is a TI ADS5400EVM evaluation board. This board is an off the shelf ADC board that interfaces through an off the shelf adapter to the Xilinx ML605. The board has a 12-bit dual channel ADC which can process samples at a rate of 1Gsps. The ADC is needed so that the receiver can make use of channel information for the decoding process. The ADC has a COTS interface into the Virtex-6 which receives the digital data and decodes it.

Current work is being done to implement an optical OFDM transceiver for the SDC platform. The transmitter circuity for this design is a modified version of the transceiver for optical Alamouti coding, though the receiver circuitry remains the same. Since the transmitter employs OFDM modulation, the addition of a DAC at the transmitter to produce an analog modulation is required. In this case, the TI DAC5681ZEVM evaluation board is used to provide a 16-bit dual channel DAC which can process samples at a rate of 1Gsps.

V. Firmware/Software Development

A. Development Tools

We aim to develop an open source software framework that is flexible and modular. The goal is to produce a software system capable of supporting high data-rate processing, while flexible enough to facilitate rapid development of hardware and software co-design in communication systems. At all levels of software and hardware synthesis, we make extensive use of established industry-standard development tools to achieve a uniform design flow.

The software development process is partitioned into two portions: physical layer algorithms and upper layer protocol design. Xilinx System Generator is used for exploration of physical layer design space and generation of hardware platforms directly from the MATLAB Simulink environment. Such automatic process provides both flexibility in design and ease of implementation in hardware. Integrated in Simulink, System Generator provides a high level of abstraction to facilitate rapid prototyping and simulation of physical layer hardware design.

A flexible and efficient set of Simulink blocks and System Generator are obtained through automated generation using SPIRAL [12], [13] which automatically produces efficient hardware and software DSP implementations. Moreover, since the same high level description of SW and HW blocks are used, it is possible to systematically explore various HW/SW tradeoffs.

When the physical layer processing hardware is generated, the complete system with other peripheral hardware cores is assembled in Xilinx Platform Studio. Software drivers for these peripheral cores are also developed with Platform Studio. Together with the available platform support packages, the users can assemble a customized communication system in FPGA fabric within a short amount of time. Finally, upper layer protocols can also be implemented in software using languages such as C/C++ and MATLAB.

B. Host Based Processing

One of the major setbacks to current SDR platforms is the massive bottleneck between system and host. This bottleneck forces platforms to perform higher layers of the protocol stack on limited processing cores within the FPGA or abandon real time processing altogether. The SDC design aims to solve this problem by providing a PCIe interface between FPGA and host.

The necessary firmware and software stack for fast serial communication between host and SDC is currently being implemented with an 8x PCIe reference design provided by Xilinx. The stack leverages the Connectivity Targeted Reference Design for Virtex-6 platforms [14] to develop the necessary host framework and drivers to interface with a CPP PICe board. This framework allows user applications on the host to access the CPP using DMA. A PCIe end-device implementation on the FPGA allows the DMA calls from the user applications to configure registers and memory space within the CPP. Access is provided to both internal FPGA memory and external DDR3 memory interfaces on board the CPP.

On the host side driver development is concentrated on allowing user applications to quickly and efficiently use the PCIe interface. Reference Xilinx drivers have been modified to allow data streaming from a host disk or application directly to predefined memory locations on board the CPP. This allows for the population of CPP buffers and registers using MATLAB variables and a custom MEX-file implementation.

On the FPGA side, the Connectivity Targeted Reference Design features external memory interfaces, FIFOs, and DMA IP framework for streaming data to peripherals and for accessing onboard RAM. The design can interface with other custom IP (e.g., WARPLab) or the scalable OFDM PHY (described in the next section) through User Space Registers. A simple buffer-and-transmit design is being integrated into the PCIe framework as a proof of concept.

The objective of the ongoing PCIe development, however, is to automate the task of integrating custom FPGA implementations with the current PCIe design in order to provide a universal, interchangeable interface between the firmware designs implemented in the FPGA and the resources and management infrastructure available on the host. This would allow a level of transparency to the end user that would facilitate new design development while minimizing concerns about system integration.

For immediate user application, we are integrating existing open source designs with the SDC Testbed. In particular, we are implementing an Ethernet interface between the host and the Virtex-6 ML605 development platform. This framework brings together MATLAB and the ML605 to perform wireless communication.

All physical layer processing will be handled offline by MATLAB on a host PC. The data samples are then transferred via Ethernet to the ML605 hardware for buffering and transmitting over the air in real-time using high speed frontends. At the receiver, the samples are captured, buffered, and then transferred back to MATLAB via Ethernet for further offline processing. In this particular setup, the ML605 board is used solely for wireless transmission and does not participate in physical layer processing. This design flow allows users to rapidly prototype new physical layer processing schemes in the familiar MATLAB programming environment. This design provides similar functionality to the WARP platform using the WARPLab design. This design also allows users without a PCIe host to use the Ethernet for connectivity between the SDC platform and host.

FPGA resource utilization is fairly low for this mode of operation. Since processing is done mainly on the host PC, the FPGA is effectively only utilized for storing-and-forwarding of data. Resource utilization is approximately 20% of the FPGA capacity because use of the PCIe bus requires complex control mechanisms and a DMA controller. However, this design essentially only uses logical units and block RAM units. For the design using Ethernet instead of PCIe, resource utilization is much lower, at only 10% of the FPGA capacity. In both cases, the majority of the resources are dedicated to the chosen communication mechanism between the host and FPGA.

C. FPGA Based Processing

One of the major design considerations for this platform is the ability to easily test various PHY layer implementations. While current platforms allow the users to change the FPGA code via open source firmware and low level tutorials, making changes to the PHY layer is still out of reach for many communication protocol researchers. The lack of flexibility due to system complexity hinders the ability of current available testbeds to be a true software defined radio. The scalable OFDM physical layer software being developed for this testbed creates a layer of abstraction which enables the user to change various PHY operations easily without needing to know how to develop code for an FPGA platform.

In addition, various levels of adaptation can be done on-the-fly (limited by FPGA resources) allowing for even more flexibility in system design.

While various frontends are being developed for this platform, the base PHY layer firmware design will be loosely based on various UWB standards. This ensures high bandwidth compatibility, while maintaining the ability to choose lower bandwidth options for simpler designs. In addition to radio frequency applications, this PHY layer will be designed to support applications in free space optical communications as well. Through the inclusion of an optional conjugate symmetric block (used for low speed UWB applications), real-valued data is generated for the optical frontends.

The additional degrees of freedom provided by the abstracted scalable OFDM design are enabled through a central control design which can be programmed via a MATLAB GUI or M-file. Currently, settings can be selected for modulation rate, coding rate and number of subcarriers. The current supported variations are:

- Modulation Rate: QPSK, 4QAM, 16QAM, and 64QAM
- Coding Rate: 1/3, 1/2, 11/32, 5/8, and 3/4
- Subcarriers: 32, 64, 128, and 256
- Other options: Conjugate Symmetric and Interleaver

From experience using other SDR platforms such as WARP, we have determined that using System Generator is the most practical way to build the PHY layer implementation. Each functional block of the scalable OFDM design is tied into a controller block which provides the necessary data to the functional blocks based on the selected variables. While these predefined variables provide a level of user modification, it does not give the user complete freedom if they want to do something that is not predefined in the system. In order to allow for greater user customization, the functional blocks have been designed into standalone functions which operate as independently of each other as possible. Since it is important for certain information to be passed from one block to the next, complete independence is not an option. However, to make modification the system easier, the proposed system design includes a meta-description language which defines each block in a context which can be understood by users. This allows users to either remove blocks, or modify blocks and maintain the data formatting and scheduling needed for the system to operate as desired.

Resource utilization on the FPGA is much higher in the case when processing is done on the FPGA. While the same resource utilization is required for the communication between the host and FPGA, additional resources are needed to provide the necessary processing of data. MIMO OFDM processing for UWB would take up approximately 60% of the FPGA resources, including the PCIe resources needed to communicate with the host.

VI. Initial Measurement Results

Initial measurement results have been gathered using a prototype of the system architecture. The design uses the Xilinx ML605 connected to the 4DSP FMC110 analog conversion frontend. Preprocessing of OFDM symbols is performed in MATLAB, and then the raw samples are sent to the ML605. The ML605 buffers and sends this data to the FMC110 board. The signal is then looped back and received by the system and raw samples are sent back to the host PC running MATLAB for post processing. The transmitted signal is comprised of 40 OFDM symbols. Each OFDM symbol has 64 subcarriers, 48 of which are used to carry data. Each subcarrier uses 8 bits to perform 256 QAM. As expected, the signal constellation (Figure 6) shows good agreement between transmitted and received symbols, given the wire feedback channel.

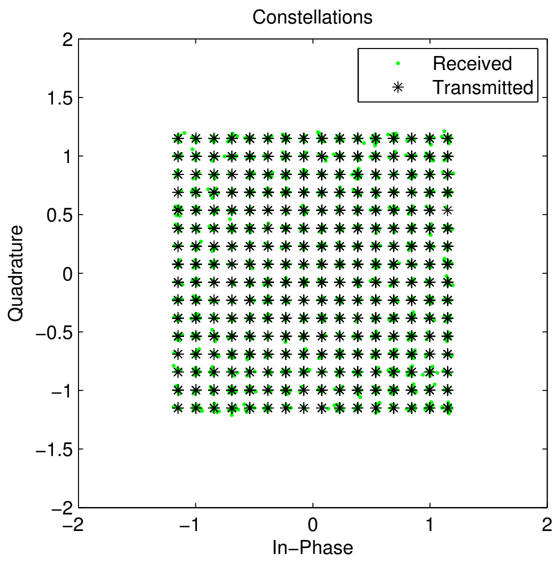

Fig. 6. Signal constellation for transmitted and received signal

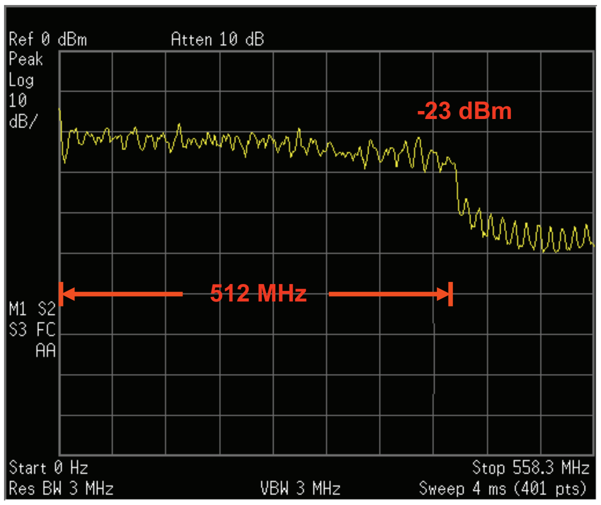

Fig. 7. Power spectral desnsity of the baseband signal

Spectral analysis was performed on the baseband signal as well. The power spectral density of the output of the D/A converter is provided in Figure 7. The bandwidth of the baseband signal is roughly 512 MHz with a power of -23 dBm.

VII. Conclusion and Future Work

The current SDC platform is in its early stages of development, however, by utilizing existing hardware and software infrastructures, it has been possible to make progress in platform software infrastructure and algorithm design in a very short period of time. By adopting the commercially available ML605 we have been able to implement open source designs on a completely new hardware architecture, and leverage commercial frontends like the 4DSP FMC110 to generate UWB waveforms and transmit as well as receive their baseband signals.

At the baseband, with the support of Xilinx reference designs we are able to produce a host to CPP wideband interface that can be combined with a variety of FPGA communications design instantiations. This allows the CPP to be used as a buffering and transceiver platform, or as a peripheral baseband processor. For the modular frontend transceivers, hardware design is proceeding on several fronts to provide the SDC with conventional wideband, UWB, and optical communications frontends. In short order the platform will use these modalities to transmit and receive signals thought a wireless medium. With support from Xilinx and the wireless communications community it will be possible to manufacture CPP peripherals which are interchangeable with the current development boards but provide additional resources for MIMO applications.

Current software development is ongoing in several key areas of the SDC platform. Open source designs are being optimized for this platform, a custom reconfigurable OFDM implementation will allow for rapid prototyping of UWB and wideband physical layers, and a robust PCIe host interface is being optimized for fast and simple integration with user developed FPGA IP cores. Furthermore, the platform is being designed to support the potential use of computing resources not necessarily limited to an FPGA.

The project's overall aim of providing an open source framework for advanced algorithm development is being met by a combination of hardware and software advances leading to a cohesive unified platform. As the project continues, we hope to provide a unique platform to the research community for open source development of high-throughput communication and networking architectures algorithms.

Acknowledgment

This project is funded by NSF grants CNS-0854946 and CNS-0923003.

References

[1] K. Mandke, S.-H. Choi, G. Kim, R. Grant, R. Daniels, W. Kim, R. Heath, and S. Nettles, "Early results on hydra: A flexible MAC/PHY multihop testbed," in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, 2007, pp. 1896 –1900.

[2] GNU radio project. [Online]. Available: http://www.gnu.org/software/gnuradio/

[3] K. Amiri, Y. Sun, P. Murphy, C. Hunter, J. Cavallaro, and A. Sabharwal, "WARP, a unified wireless network testbed for education and research," *Microelectronic Systems Education, 2007. MSE '07. IEEE International Conference on*, pp. 53–54, June 2007.

[4] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. Voelker, "SORA: High performance software radio using general purpose multi-core processors," *Networked Systems Design and Implementation, 2009. NSDI '09. 6th USENIX Symposium on*, April 2009.

[5] Xilinx Vitex-6 ML605 hardware user guide. [Online]. Available: http://www.xilinx.com/support/documentation/boards_and_kits/ug534.pdf

[6] K. Cunningham, Y. Wang, P. Nagvajara, and J. Johnson, "Singular value decomposition hardware for MIMO: State of the art and custom design," *Proc. 2010 ReConFigurable Computing and FPGAs, 2010, International Conference on*, December 2010.

[7] WARP: Wireless open access research platform. [Online]. Available: http://warp.rice.edu/trac/

[8] 4DSP FMC110 specifications. [Online]. Available: http://www.4dsp.com/FMC110.php

[9] ECMA-368: European computer manufacturers association standard 368. [Online]. Available: http://www.ecma-international.org/publications/standards/Ecma-368.htm

[10] M. Simon and V. Vilnrotter, "Alamouti-type space-time coding for free-space optical communication with direct detection," *IEEE Transactions on Wireless Communications*, vol. 4, no. 1, pp. 35–39, Jan. 2005.

[11] M.-E. Dumitru, "FPGA implementation of diversity and spatial multiplexing for MIMO free space optical interconnects," Master's thesis, Drexel University, Philadelphia, PA, September 2010.

[12] Spiral project. [Online]. Available: www.spiral.net

[13] M. Puschel, J. Moura, J. Johnson, D. Padua, M. Veloso, B. Singer, J. Xiong, F. Franchetti, A. Gacic, Y. Voronenko, K. Chen, R. Johnson, and N. Rizzolo, "Spiral: Code generation for DSP transforms," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 232 –275, 2005.

[14] Xilinx Virtex-6 FPGA connectivity kit reference design user guide. [Online]. Available: http://www.xilinx.com/support/documentation/boards_and_kits/ug664.pdf