A 50MHZ+ BANDWIDTH REAL-TIME FPGA IMPLEMENTATION OF OFDM-BASED PHY TRANSCEIVER FOR 5G

Carlos Ribeiro (Instituto de Telecomunicações/Instituto Politécnico de Leiria, Leiria, Portugal; carlos.ribeiro@ipleiria.pt)
Atílio Gameiro (Instituto de Telecomunicações/Universidade de Aveiro, Aveiro, Portugal; amg@ua.pt)

ABSTRACT

This paper introduces a fully pipelined implementation architecture for OFDM-based transceivers that moves one step closer to fulfilling the envisioned 5G bandwidth demands. The implementation uses high level abstraction tools to develop and test the algorithms, significantly increasing the productivity and reducing costs and time-to-market. The proposed architecture defines a common interface between blocks, with FIFOs interconnecting the blocks and a soft-handshake, which presents significant advantages over the other reported architectures.
The proposed architecture is implemented in a fully working real-time platform, using COTS FPGA and RF development boards. The demonstrator has a real-time scalable bandwidth from 20MHz to 54MHz. Using 64-QAM modulation and 1024 carriers, the implementation attains over 300Mbits/s of raw bitrate in each direction, using less than one quarter of the FPGA resources.

1. INTRODUCTION

The development of 5G standards is currently under way. In several organizations, the interested parties all try to define the challenges and requirements of the next generation of cellular networks. All forums seem to agree that the 5G will need to provide a massive capacity increase to tens of Gbps, to support the evolution of the demand for wireless services. A global consensus is growing that 5G will require 500MHz to 1GHz of bandwidth [1].

The air interface for 5G is under discussion by different initiatives. The 5GNOW project [2] is one such initiative focused on the definition of new modulation schemes. The initial results of this project, and many others, point to the adoption of OFDM-based modulation schemes.

Significant breakthroughs in the baseband architectures and supporting hardware platforms must be proposed to support the computationally intensive new adaptive air interfaces and solutions like massive MIMO [3].

Several publications describing implementations and platforms that offer the possibility to prototype OFDM systems can be found in the literature. However, few address 4G systems [4-7] and even fewer show the potential to process post-4G systems. The prototype presented in [4] is partially implemented on the Wireless Open Access Research Platform (WARP) [8]. It reports the partial implementation of a 20MHz bandwidth LTE uplink receiver.

The WARP is a wireless platform built to prototype advanced wireless systems. The open access to its reference designs boosted its success and it has been successfully adopted by a significant number of OFDM-based implementations, mainly focusing on 802.11-related prototypes. The platform adopts a high level design flow resorting to Xilinx's System Generator library and MatLab's Simulink for the design and functional validation of the algorithms. Low level hardware description languages, VHDL and Verilog, are used to link the System Generator netlists with the custom-designed FPGA and RF boards from Mango Communications [9]. The latest WARP digital processing board is based on Xilinx Virtex-6 high performance FPGAs. Its highest performance RF board has a maximum bandwidth of 40MHz that limits its use for post-4G prototyping. The OFDM reference design adopts a traditional synchronous architecture that leads to long critical paths. This architecture, therefore, limits the system performance. The adoption of custom hardware may also prove to be a limiting factor, forcing the project promoters to keep the hardware up-to-date. This implies a constant and costly development of expensive baseband and RF boards.

The LTE prototyping platform in [5] is based on National Instrument (NI) proprietary PXI platform and USRP-RIO hardware. NI graphical system design LabVIEW is used to design the algorithms. It reports the implementation of an LTE network where the nodes are able to process LTE signals up to 20MHz bandwidth. The USRP-RIO hardware limits the platform bandwidth to 40MHz, not enough for post-4G prototyping.

This paper introduces a fully pipelined implementation architecture for OFDM-based transceivers that moves one step closer to fulfilling the envisioned 5G bandwidth demands. The prototype is implemented in COTS Xilinx

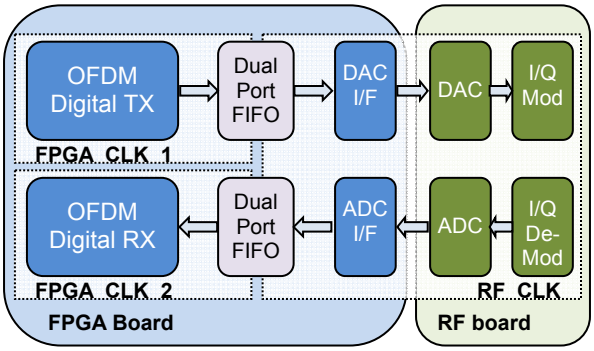


Figure 1 – "Flatten" block diagram of the platform's main functional blocks.

ML605 development board and Analog Devices AD-FMCOMMS1-EBZ RF boards. The platform uses high level abstraction tools to develop and test the algorithms, significantly increasing the productivity, reducing costs and time-to-market. The proposed architecture defines a common interface between blocks, with FIFOs interconnecting the blocks and a soft-handshake, which presents significant advantages over the other reported architectures.

2. FULLY PIPELINED OFDM TRANSCEIVER

The platform was developed with the goal of providing a high performance prototyping tool for OFDM-based systems with high levels of portability and flexibility, envisioning the test of post-4G baseband algorithms. Figure 1 depicts the main functional blocks of the OFDM transceiver depicted on an abstracted single layer.

To make the platform portable and not dependent on specific hardware, the OFDM baseband transmitter and receiver blocks interface with the remaining hardware through dual port FIFO SRAMs, with independent clocks on each side.

Realizing that the strict synchronicity of the OFDM transmitter and receiver can be limited to the ADC and DAC, in the present platform the timing constraints can be significantly relaxed. The baseband transmitter must only ensure that the DAC interfacing FIFO never gets empty and the baseband receiver that its ADC interfacing FIFO never gets full. Meeting these 2 simple conditions is enough for the transceiver to work perfectly.

The simpler implementation will have 3 independent clock domains: transmitter, receiver and RF board interface (in figure 1 denoted FPGA_CLK_1, FPGA_CLK_2 and RF_CLK, respectively). This clock domain splitting will allow a greater control of the clock trees and limits the potentially problematic RF_CLK domain strictly to the area within the FPGA that holds the RF board interface block.

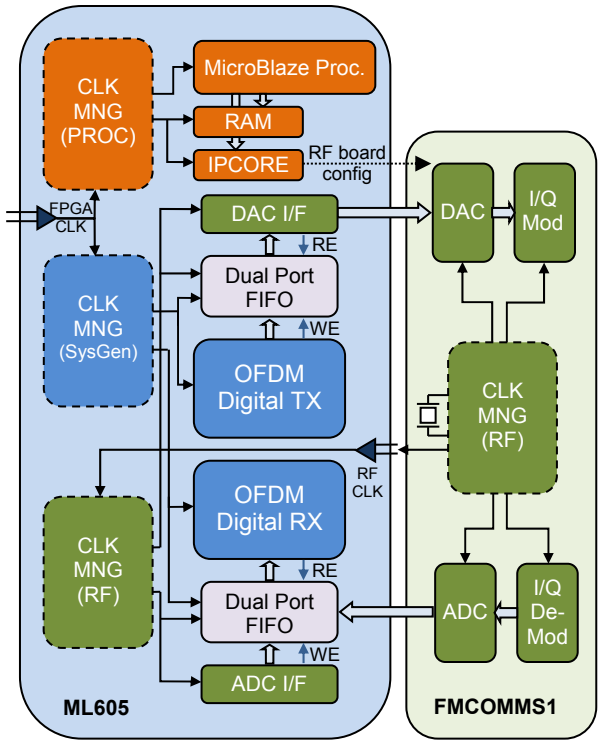


Figure 2 - Implementation main block diagram with clock distribution network.

This architecture considerably simplifies its implementation in any FPGA-based digital signal processing board. The transmitter and receiver are self-contained and their performance is mainly independent of the adopted RF board and its (more and more complex) clocking structure.

Figure 2 depicts the block diagram of the implementation of the platform using Xilinx ML605 development board and Analog Devices AD-FMCOMMS1-EBZ RF board. The use of COTS development boards significantly lowers the cost when compared with platforms that use custom hardware. Besides the functional blocks, the figure also depicts the clocking structure of the implementation. In this example there are 5 clock domains on the FPGA: baseband transmitter, baseband receiver, ADC interfacing, DAC interfacing and processor. The clock domains are controlled by 3 clock management entities (CLK_MNG_PROC, CLK_MNG_SysGen and CLK_MNG_RF in figure 2). CLK_MNG_PROC clock management entity receives a reference clock from the ML605 board and generates all the clocks for the soft-core MicroBlaze processor (and its peripherals), needed to configure and control the AD-FMCOMMS1-EBZ board. CLK_MNG_SysGen clock management entity receives the same reference clock from the ML605 board and generates 2 independent clocks for the baseband transmitter and baseband receiver domains. CLK_MNG_RF clock

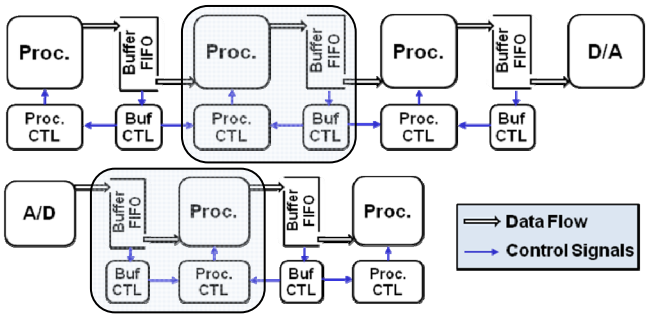


Figure 3 – Conceptual transmitter (top) and receiver (bottom) chains.

management entity receives the reference clock from the AD-FMCOMMS1-EBZ board and generates 2 independent clocks for the ADC interfacing and DAC interfacing domains. These clocks are synchronous with the RF receiving and transmitting path and define the transmitter and receiver sampling rates and respective bandwidths.

2.1. Asynchronous Modular Architecture

The advantages of the lack of strict synchronicity discussed previously can be extended to every processing block in the OFDM transceiver. If FIFO buffers are placed at the interface of each processing element and underflow and overflow is prevented, the processing in each block can be performed asynchronously, independent of the remaining elements. The condition that must be met by the each processing block is that it has enough processing to keep adequate buffer levels, i.e., that the average throughput of the block is higher than the consumer/generator (transmitter/receiver) rate. Each processing element is part of a module that includes 1 FIFO and 2 finite state machines (FSM) that control the processing and monitor the FIFO levels. Figure 3 presents a conceptual transceiver chain.

The algorithm/function is implemented in the processing element (Proc). The processing control element (Proc_CTL) is the main FSM in the block and controls the processing element according to its state and the input signals. The FIFO buffer interfaces adjacent blocks and provides the fill level indication to the buffer level monitoring FSM element (Buf_CTL). This element is similar in all blocks, changing only the buffer size. The Buf_CTL implements a soft handshake with the interfacing block, signaling that is reaching the critical fill level and that it must stop consuming/generating elements within the time period negotiated in the design phase.

The DAC interface drives the pressure for all the transmission chain, imposing the consumption rate. The ADC interface drives the pressure for all receiving chain, imposing the generation rate. When moving away from the analog-to-digital interfaces, the rate of the consumer is no longer constant. Thus, at each interface a simple area vs. performance tradeoff must be negotiated in the design phase involving generator peak throughput, buffer size and consumer peak throughput. Higher performance dictates bigger buffers to accommodate larger generation/consumption peaks and smaller foot-prints imposes smaller buffers and lower performance (higher percentage of processing devoted to starting and ending each processing run). The soft handshake eases the design of control FSMs. Unlike hard handshake, where the processing must be immediately stopped on the de-assertion of enabling signals, this soft handshake provides some flexibility for the interfacing block to end its current task (established during design phase). Control FSMs do not need to account for the de-assertion of enabling signals in every state (with everything that comes with it: disabling processes, safe-guarding of temporary values, restoring state conditions, re-enabling of processes, …).

This modular architecture, fully pipelined, with distributed control and parallel processing is the key feature that sets this platform apart from other FPGA-based platforms.

The distributed control model relaxes the synchronization requirements associated with the centralized control and enables the simultaneous processing of multiple actors, exploiting the parallelism of the architecture and significantly increasing the architecture throughput.

The inter-block communication over dedicated resources, free of contention, also contributes to the full exploitation of the architecture's inherent parallelism, enhancing its throughput.

When porting the implementation to a specific hardware, the presence of FIFOs considerable reduces the length of clock paths, leading to increased overall performance/throughput of the system, by pushing the systems maximum clock rate. If ported to ASIC, it would significantly lower energy consumption (much simpler power-hungry clock distribution structures) and reduce chip area (simpler routing of clock signals lowers the gate count significantly; limited number of hardware resources wasted to meet the clock constraints).

The decoupling of the processing in each block, leads to different processing duty-cycles for the different blocks, opening way to the use of energy efficient techniques like clock gating to drastically reduce the system power consumption.

2.2. Platform Development Flow

The suitable choice of the system-level design flow (and associated tools) can dictate the success or failure of the platform. The adoption of low level development tools may enable a high performance and resource saving solution for a simple module, but can become a designer's nightmare when verifying a complex system.

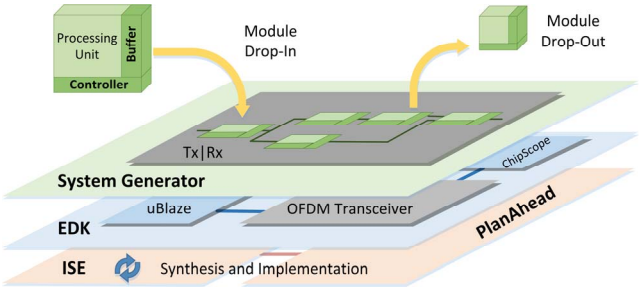


Figure 4 – Platform development tools.

The present platform uses one of engineers' favorite development flows, where the algorithms are designed and tested in Simulink environment and a nearly automated process returns the FPGA configuration files. Figure 4 depicts the distinct development tools of the platform. Processing units are designed, tested and evaluated in Simulink/System Generator; System Generator exports the transceiver netlists to be integrated with MicroBlaze and ChipScope modules in EDK; synthesis and implementation steps are performed in ISE and, if needed, tuned with PlanAhead.

The common block interface with FIFOs results in a simulator-like behavior, with "drag-and-drop" of blocks, (almost) without timing constraints, easing the collaboration between different teams.

The presence of FIFOs in the interface between blocks results in the lack of strict timing constraints between adjacent blocks, leading to a much simpler and faster debugging of the system. Each individual block can be tested in a standalone Simulink model, where the input sequences are extracted from previous block(s) to variables or structures and fed to the block under test. Pairs of blocks (corresponding transmitter and receiver operations) can be tested back-to-back due to the common block interface, easing the validation of its functionality.

The common block interface also means that each block can be reused in the system or ported to other systems. Blocks can be added or removed from any place in the system as the interface is common. The only precaution is to check if the average throughput of the block is enough to prevent underflow and overflow of the interfacing FIFOs.

To better illustrate how these tools are integrated in the design flow, figure 5 depicts the implementation flow chart. The transceiver is designed in System Generator and its performance evaluated using the bit and clock cycle accurate built-in simulator. At this point, the design is independent of the target FPGA, although the designer should be aware of its architecture in order to implement efficient modules.

Once the transceiver is functional and its performance meets the requirements, it must be connected to the external hardware interface block (in the current implementation the

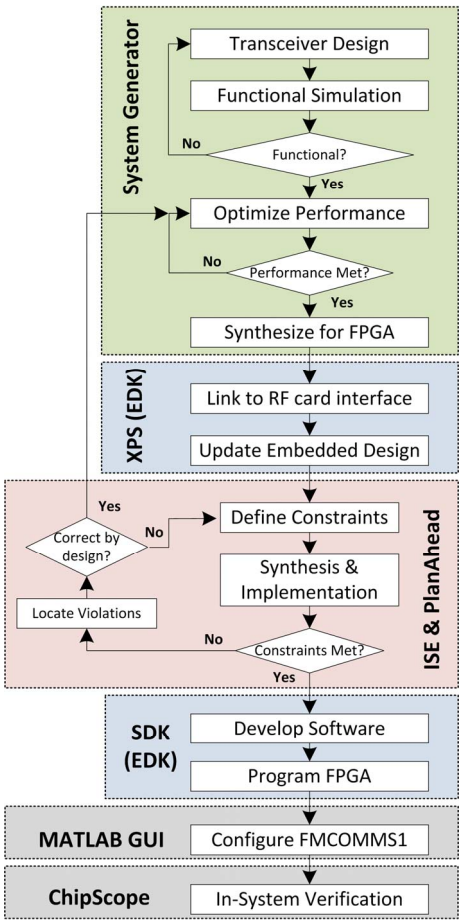


Figure 5 – Platform implementation flow chart.

AD-FMCOMMS1-EBZ embedded reference design) and the real-time hardware debugging tool ChipScope. This step is performed with the EDK software.

Synthesis and implementation steps are performed in ISE, after defining the necessary pinout, area and timing constraints. PlanAhead is used to identify and correct critical paths when ISE failed to meet the timing constraints. Once the system is correctly implemented, the software is merged with the FPGA configuration file with the SDK tool. System validation, debugging and performance evaluation is performed with the real-time Xilinx ChipScope Analyzer that enables the real-time analysis and capture of the transceiver's signals.

2.3. OFDM Transceiver

The architecture described in section 2.1 was used to design the OFDM transceiver; its block diagram is presented in figure 6. It is an actual screenshot of the top-level block diagram of the transceiver implemented in Simulink. Inside each block we can find the architecture defined in section 2.1. The current transceiver has one transmitting and one receiving path. Adding additional paths is straightforward and would re-use the blocks of the current implementation.

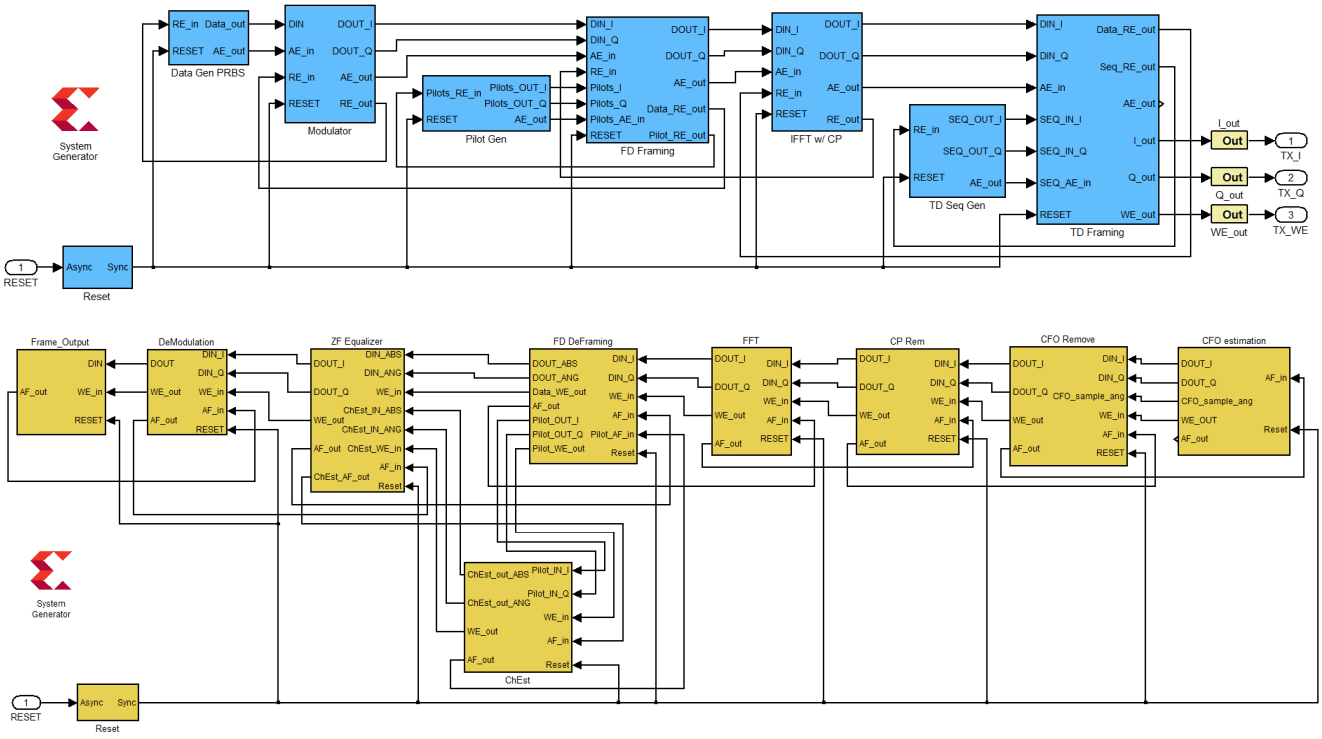


Figure 6 – OFDM transceiver block diagram; top – transmitter, bottom – receiver.

The transmitter includes:

- a pseudo-random binary sequence generator block that feeds the modulator with a known sequence of bits;
- a user configurable modulator that implements QPSK, 16QAM, 64QAM modulations;
- a pilot sequence generator block that outputs the values of the pilot symbols that will be used for the channel estimation in the receiver;
- a frequency domain (FD) framing block that builds the OFDM frame with the modulated data, pilots, and null carriers to shape the transmitted spectrum. The frame format is defined with a configurable ROM. Each symbol has 1024 carriers and a frame is made-up of an initial synchronization symbol, followed by 4 data-carrying symbols; the pilots are evenly spread in the odd data-carrying symbols, every 4 carriers;
- an 1024 points IFFT block transforms the frame into the time domain for transmission; it also adds a configurable length cyclic prefix (CP). The IFFT is implemented using Cooley-Tukey algorithm, with unscaled (full-precision) fixed-point arithmetic, Radix 4 decompositions for computing the DFT (the N-point FFT consists of $\log_4(N)$ stages, where each stage holds N/4 Radix-4 butterflies) and decimation-in-time [10]. This implementation takes advantage of the presence of DSP48 and BRAM blocks to lower the implementation area and keep the maximum path length low enough to enable the required bandwidth;
- a time-domain (TD) sequence generator that outputs a Constant Amplitude Zero Autocorrelation (CAZAC) Zadoff-Chu sequence similar to one found in LTE's primary synchronization signal;
- a TD framing unit that pre-appends the initial synchronization symbol to the remaining 4 data-carrying symbols in each OFDM frame.

The receiver includes:

- a carrier frequency offset (CFO) estimation block that implements the required timing synchronization tasks: estimation of start of OFDM symbol; estimation of start of OFDM frame; estimation of the frequency offset. The 3 required estimations are performed by a joint Maximum Likelihood algorithm [11] that takes advantage of the presence of the higher-power synchronization symbol and the presence of CP in each symbol of the frame. The estimation of the start of the OFDM symbol and the estimation of start of OFDM frame are used to define the time boundaries of each received frame;

- a CFO compensation module that removes the estimated frequency offset affecting each received symbol, caused by the mismatch between TX and RX oscillators; the incremental phase rotation is implemented with an hardware-efficient CORDIC algorithm;
- a CP removal block is responsible for deleting the CP present in the beginning of each TD OFDM symbol;
- an FFT block transforms each symbol to FD for further processing. Its implementation is similar to the IFFT block in the transmitter;
- an FD deframing unit is used to separate pilots carriers, data carriers and null carriers. While the values in the data carriers are feed to the *Zero Forcing* (ZF) equalizer module, the values in the pilot carriers are feed to the Channel Estimator (ChEst) block. The null carriers are discarded;
- a ChEst block estimates the channel in the pilot positions using *Leasts-Squares* algorithm, and performs a FD linear interpolation to extend the channel estimation to the data carriers. The symbols that carry only data use the channel estimation of the previous pilot-carrying symbol;
- a ZF equalizer module compensates the channel distortion with a single complex division per carrier, restoring the transmitted constellation; To lower the implementation area, the equalizer is implemented in polar coordinates, built around a single real divider block that compensates the magnitude distortion of each carrier and a CORDIC-based phase rotation block that rotates the phase of the received value in each data carrier to the original phase;
- a Demodulator unit performs a hard decision to estimate the transmitted bitstream.

Table 1 – Transceiver design and implementation parameters.

Design Parameters		*Implementation Parameters*	
System carriers	1024	**Tx domain clk freq.**	150MHz
Loaded carriers	800	**Rx domain clk freq.**	150MHz
Full signal bandwidth	54MHz	**DAC domain clk freq.**	54MHz
Mod. signal bandwidth	42,2MHz	**ADC domain clk freq.**	54MHz
Carrier separation	52,7KHz	**Proc. domain clk freq.**	100MHz
FD pilot separation	4	**Digital data width**	32 bits
TD pilot separation	2	**DAC resolution**	16bits
Modulation	QPSK, 16/64 QAM	**DAC sample freq.**	108MHz
OFDM frame	5 symbols	**ADC resolution**	14 bits
Sync symbol	CAZAC Zadoff-Chu	**ADC sample freq.**	54MHz
CP length	256	**RF center freq.**	2,4GHz

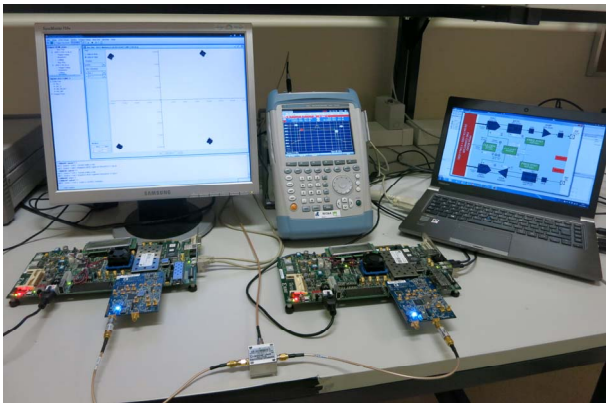


Figure 7 – Photo of the setup.

The key parameters of the transceivers are summarized in Table 1.

3. SETUP EVALUATION

A setup was assembled to evaluate the performance of the platform. Figure 7 shows a photo of the setup. The platform evaluation setup is made-up of a receiver, a transmitter, a power splitter, a real-time Xilinx ChipScope analyzer, a spectrum analyzer and the MATLAB setup configuration Graphical User Interface (GUI).

The transmitter and receiver use similar hardware: Xilinx ML605 development boards (with Xilinx Virtex6 FPGA) with an Analog Devices AD-FMCOMMS1-EBZ RF daughterboards. On the photo, the lower right set is the transmitter and the receiver is on the right. In the middle of both, we can see a power splitter that allows the transmitted signal spectrum to be monitored.

On the top row, from left to right, we can see the Xilinx ChipScope analyzer window, the spectrum analyzer and a laptop with the configuration GUI.

For debugging and performance evaluation purposes, Xilinx ChipScope Integrated Logic Analyzers (ILA) modules were attached to the transmitter, ADC and receiver domains, at different locations. The ILA units capture real-time samples of the signals being monitored. The receiver domain ILA unit was placed at the output of the equalizer, enabling the real-time monitoring of the equalized constellations for the different modulations. In the photo, the Xilinx ChipScope analyzer window is showing the overlapping of a set of received QPSK constellations.

The spectrum of the transmitted signal, captured by the spectrum analyzer, can be seen in figure 8. The capture

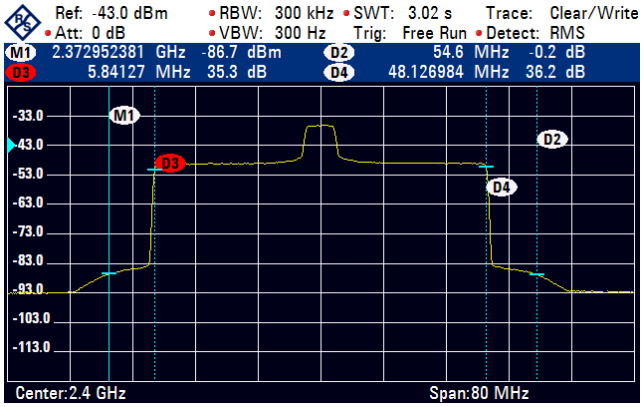


Figure 8 – Measured transmitted spectrum.

shows 4 markers: M1, D2, D3 and D4. Marker M1 is the reference for the values of the remaining markers. Marker D2 shows the signal full bandwidth of 54MHz. This bandwidth can be real-time configured to any value bellow 54MHz via the DAC/ADC domain clocks.

In the present setup, the lower bandwidth is limited by the ADC lowest sampling frequency of 20MHz (this limit can be easily removed with a decimation filter at the output of the ADC). Markers D3 and D4 show the modulated bandwidth of 42MHz. The modulated bandwidth can be configured in the FD (de-) framing blocks to any value up to the full bandwidth.

The MATLAB configuration GUI communicates with the transmitter/receiver embedded MicroBlaze processor via USB and configures in real-time the most relevant parameters in the RF boards (RF center frequencies, ADC/DAC sampling rates, clock manager output frequencies, amplifier gains, etcetera).

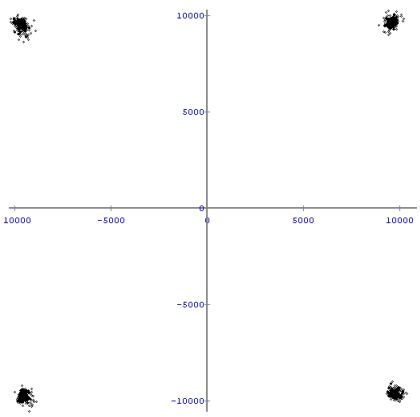
Transmitter and receiver were connected back-to-back and the received constellations for all modulations where captured by the ILA unit. Figure 9 shows the real-time ILA captures. The low distortion of the received constellations with an EVM of -36dB provides a strong indication on the accuracy of the implemented algorithms. BER tests indicated that the decoded bitstream had no errors.

The performance achieved in the real-time RF back-to-back testing highlights the potential of the proposed platform: the transceiver attained 54MHz bandwidth, leading to over 300Mbps of raw bit rate per OFDM path, when using 64QAM modulation.

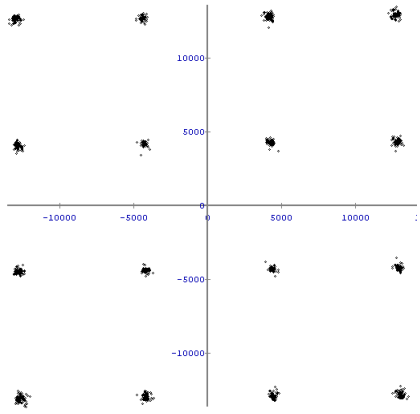
The FPGA resource utilization of the full transceiver (the ML605 block diagram depicted in figure 2) is summarized in table 2. The reported number of used BRAMs includes the ones used by the embedded MicroBlaze processor and ILA units. Analyzing the percentage of resource utilization, we can conclude that up to 4 full OFDM transceivers can be simultaneously implemented in the ML605, boosting the overall throughput to over 1Gpbs in each direction.

Table 2 – ML605 resource utilization.

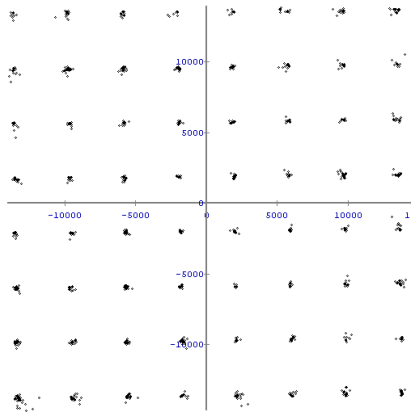
ML605 resource	*Number*	*Percentage*
Registers	31168	10%
LUTs	29705	19%
BRAMs	127	30%
DSP48	166	21%



a) QPSK



b) 16QAM



c) 64 QAM

Figure 9 – Received constellations.

4. MAIN CONCLUSIONS

This paper introduces a fully pipelined platform for OFDM-based transceivers that goes significantly beyond the state-of-the-art, moving one step closer to fulfilling the envisioned 5G bandwidth demands.

The platform is not hardware dependent and we describe an implementation using COTS Xilinx ML605 FPGA-based digital signal processing development boards and Analog Devices AD-FMCOMMS1-EBZ RF cards.

The use of dual port FIFO SRAMs that split the baseband digital signal processing from the RF interface (and their associated clock domains), makes migrating to other hardware substantially simpler.

The use of COTS hardware significantly lowers the cost when compared with platforms that use custom hardware and enables its implementation on the latest hardware available in the market.

The higher complexity of each individual block of the proposed architecture, that, apart from algorithm processing hardware, must also include 2 additional FSMs a one FIFO, is clearly compensated by its advantages. The block's interfacing FIFOs and normalized interface enables the individual design and test of each block, the simple add/remove of blocks and its re-use. The performance evaluation showed it substantially outperforms the existing prototyping platforms.

5. ACKNOWLEGMENTS

This work was supported by the Portuguese Fundação para a Ciencia e Tecnologia (FCT) COPWIN (PTDC/EEI-TEL/1417/2012) and HETCOP (PEst-OE/EEI/LA0008/2013) projects. The work of Carlos Ribeiro was financially supported by FCT/MEC and its funding program under the postdoctoral grant SFRH/BPD/104212/2014.

6. REFERENCES

[1] "White Paper for Research Beyond 5G", draft white paper, http://networld2020.eu/, October 2015.

[2] "5GNOW – 5th Generation Non-Orthogonal Waveforms for Asynchronous Signalling", http://www.5gnow.eu, 2015.

[3] Rusek, F.; Persson, D.; Buon Kiong Lau; Larsson, E.G.; Marzetta, T.L.; Edfors, O.; Tufvesson, F., "Scaling Up MIMO: Opportunities and Challenges with Very Large Arrays," *IEEE Signal Processing Magazine*, vol.30, no.1, pp.40-60, Jan. 2013.

[4] Guohui Wang; Bei Yin; Amiri, K.; Yang Sun; Wu, M.; Cavallaro, J.R., "FPGA prototyping of a high data rate LTE uplink baseband receiver," Asilomar Conf. Signals, Systems and Comp., pp.248-252, Nov. 2009.

[5] Gupta, R.; Vogel, T.; Kundargi, N.; Ekbal, A.; Morelli, A.; Mancuso, V.; Sciancalepore, V.; Ford, R.; Rangan, S., "LabVIEW based platform for prototyping dense LTE networks in CROWD project," European Conference on Networks and Communications, pp.1-5, June 2014.

[6] Bates, D.; Henriksen, S.; Ninness, B.; Weller, S. "A 4x4 FPGA-based wireless testbed for LTE application," Personal, Indoor and Mobile Radio Communications Conf., pp. 1 – 5, Sept. 2008.

[7] López-Martínez, F. J.; Castillo-Sánchez E.; Martos-Naya, E.; Entrambasaguas, J. T., "Design and FPGA implementation of an OFDMA baseband modem for 3GPP-LTE physical layer," International Conference on Consumer Electronics, pp.1-2, Jan. 2009.

[8] "WARP: Wireless Open Access Research Platform", warpproject.org.

[9] Mango Communications, https://www.mangocomm.com/, 2015.

[10] LogiCORE IP Fast Fourier Transform v7.1, Xilinx Inc,March 2011.

[11] J. J. van de Beek, M. Sandell and P. O. Borjesson, "ML estimation of timing and frequency offset in OFDM systems," *IEEE Transacions on Signal Processing*, page 1800-1805, July 1997.