A novel system on chip for software-defined, high-speed OFDM signal processing

Joachim Meyer*, Michael Dreschmann*, Djorn Karnick†, Philipp C. Schindler†,
Wolfgang Freude†, Juerg Leuthold†‡ and Jürgen Becker*

*Institute for Information Processing Technologies (ITIV), Karlsruhe Institute of Technology (KIT)
†Institute of Photonics and Quantum Electronics (IPQ), Karlsruhe Institute of Technology (KIT)
Email: {joachim.meyer, michael.dreschmann, djorn.karnick, philipp.schindler, wolfgang.freude, becker}@kit.edu
‡now at Electromagnetic Fields and Microwave Electronics Laboratory (IFH),
Swiss Federal Institute of Technology in Zurich (ETH), Email: leuthold@ethz.ch

***Abstract*—In this paper the authors describe a novel system on chip (SoC) that is especially developed for digital signal processing of high-speed orthogonal frequency division multiplexing (OFDM) signals with data rates up to gigabits per second. Besides offering a new degree of freedom for the tradeoff between flexibility and performance during runtime, the modular concept of the SoC also allows a tradeoff between performance and costs during design time. The flexibility to adapt the OFDM system parameters by software enables even system designers without a good knowledge of hardware design to implement high-speed OFDM systems. An example configuration of the architecture was implemented on a Virtex-6 FPGA in order to set up a software-defined OFDM transmitter, achieving data rates of several gigabits per second. The paper closes with implementation and performance results of experiments using the developed transmitter and an optical transmission of the generated OFDM signals.**

***Index Terms*—SoC; MPSoC; FPGA; OFDM; high-speed; fibre**

I. Introduction

Nowadays orthogonal frequency division multiplexing (OFDM) is deployed in many of today's communication systems, especially in the wireless domain. However, in the recent years OFDM has also become applicable in high-speed optical communication systems, compare [1]. The rising request for higher data rates caused by this development leads to increasing requirements for the digital signal processing of these applications. Until today, when targeting data rates of one giga-sample per second and more, usually a dedicated circuit implemented in an FPGA or in an ASIC is inevitable.

While taking performance to the limit, dedicated circuits have two major drawbacks. On the one hand the development of such circuits is very time consuming and costly, requiring hardware design specialists. On the other hand, the parameters of such circuits are tailored to one special application (e.g. FFT size, compare [2]). This inflexibility inhibits the reuse of such circuits for other applications and enforces a redesign, most likely from scratch.

In order to tackle this problem, a multiprocessor system on chip (MPSoC) was developed to enable a better tradeoff between performance, costs and flexibility. The implied tradeoff between performance and flexibility which results from the usage of multiple dedicated application processors is supplemented by the tradeoff between performance and cost (e.g. resources) when determine the number of implemented application processors during design time. Once implemented, due to the programmability of the SoC, hardware designers are not needed anymore to build up high-speed OFDM systems.

The rest of the paper is organized as follows: Section II presents related work. The next section explains the concept of the SoC and introduces the developed processors. Section IV describes the prototype transmitter implementation and presents the results of the measurements. The last section, section V, concludes this paper.

II. Related Work

One method to achieve high data rates for OFDM systems is the usage of modern digital signal processors (DSPs). By featuring wide data busses and optimizations for signal processing operations (e.g. multiply-accumulate) such devices achieve a higher performance than general purpose processors, but still offer very high flexibility since they are fully programmable. Implementations as presented in [3] or [4] show how the system parameters can be easily varied by exchanging the assembler code. The price for this high flexibility is the rather slow performance of usually several kilo-samples or at most mega-samples per second.

The other extreme is given by dedicated circuit implementations as seen in [5] and [6], where sample rates up to about 25 giga-samples per second were achieved, but parameter variations are not possible at all. Such systems are usually tailored to one specific application, taking advantage of optimizations which are only valid in this specific scenario. Therefore, even the slightest changes (e.g. different cyclic prefix size) require major redesigns.

Recent SoC implementations which target OFDM communication (e.g. [7] or [8]) rather concentrate on power consumption and integration than performance. Hence, the maximum sample rates are in the range of several mega-samples per

978-1-4799-1132-5/13/$31.00

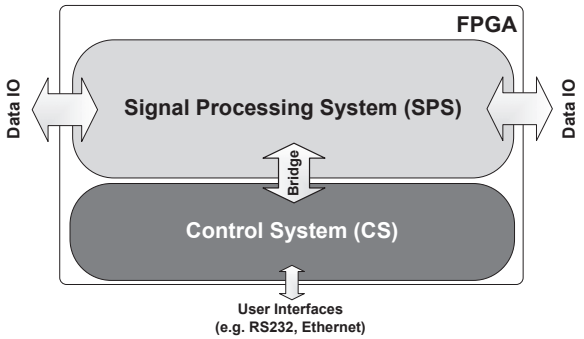

Fig. 1. On a high level, the developed system on chip can be divided in a control system (CS) and a signal processing system (SPS), connected via an asynchronous bus bridge.

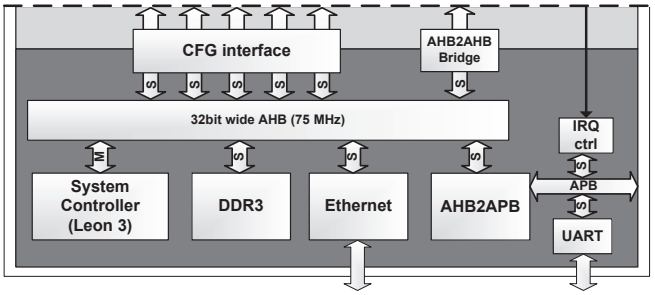

Fig. 2. The main task of the control system is to keep the processors of the SPS busy. Although for our prototype we choose a Leon-3 based system, in general any processor subsystem can do the job.

second. To our knowledge there is no work available which is qualifying for a detailed comparison with our novel MPSoC by offering a similar tradeoff in performance and flexibility.

III. Implementation

The composition of the developed OFDM MPSoC is shown in figure 1. On a high level, the architecture can be divided into two subsystems, the control system (CS) and the signal processing system (SPS). The two subsystems are connected by an asynchronous bus bridge which is able to copy data from the SPS into a bridge memory, directly accessible by the CS, or vice versa. By doing this, the bus bridge also converts the clock frequency and the bus width.

A. Control System (CS)

The control systems main tasks are to provide user interfaces, configure the processors of the SPS and to load the SPS processors program code into their command memory. It can optionally perform signal processing or data analyses by accessing the data of the SPS using the bus bridge or react on interrupts coming from any of the SPS processors (e.g. replace a processors code with new commands). However, in order to achieve high data rates, the signal processing should be handled inside the SPS only.

The SPS processor configuration interfaces are implemented as a 32-bit wide Advanced Microcontroller Bus Architecture (AMBA) Advanced High-performance Bus (AHB) architecture. Since the CS and SPS support different clocks, the configuration interfaces as well as the bus bridge include clock domain bridges. The only requirement to the CS system is to support the AHB bus system to access these interfaces. Besides that, any processor subsystem is applicable.

Figure 2 illustrates the CS-subsystem as used in our prototype. It mainly consists of components of the Gaisler Research Library (GRLIB, see [9]) which is a collection of IP-cores basing on a SPARC-V8 compatible processor, the Leon 3. We chose this system because it is free for a non-commercial usage and it offers comfortable programming and debugging features like c-compilers and an eclipse plugin.

B. Signal Processing System (SPS)

The signal processing subsystem, which is the actual multiprocessor system on chip, consists of several types of processors connected by a 64-bit wide AHB bus system and by a configurable interrupt bus. The system uses a 16-bit fixed point numeric notation; therefore two complex samples can be transported with one bus access. Each type of processor takes care of a special task in the OFDM signal processing chain. Figure 3 shows an example for an SPS in an OFDM receiver, which is able to demodulate an OFDM signal in order to compare the received data with a locally generated pseudo-random bit stream sequence (PRBS).

To program an SPS processor, the CS system configures all OFDM parameters of the processor and writes a sequence of commands into the command memory. Most of the processors have only one command type, consisting of the read address and the write address of the operation. Besides stopping after executing all commands, a processor can also operate in different kind of loop modes. In those modes, a new iteration of the commands is triggered at specific events, e.g. when receiving an interrupt from another processor.

Almost all OFDM parameters of the processors in the SPS can be configured by the CS, including but not limited to: FFT size, modulation format of each subcarrier, number and position of pilot tones, value of pilot tones, size of cyclic prefix, delay widths, summation widths and different thresholds of an extended version of the synchronization mechanism given by [10], number of bits to be clipped or rounded, preamble, number of preamble symbols and the number of data symbols.

1) SPS AHB Interconnect: The high-speed interconnect between the processors is one of the most important components. We developed a 64-bit wide multiplexer based multilayer AHB bus matrix similar to [11], using fix priorities for arbitration. The amount of master ports and the amount of slave ports can be set independently by the designer. With such a bus matrix, many bus masters can perform parallel transmissions, compare figure 4. Therefore, such an interconnect provides a very high performance but also requires a high amount of resources.

2) OFDM Processors: There are several types of processors each of them is developed for one specific task, like the calculation of a Fast Fourier Transform (FFT) or the

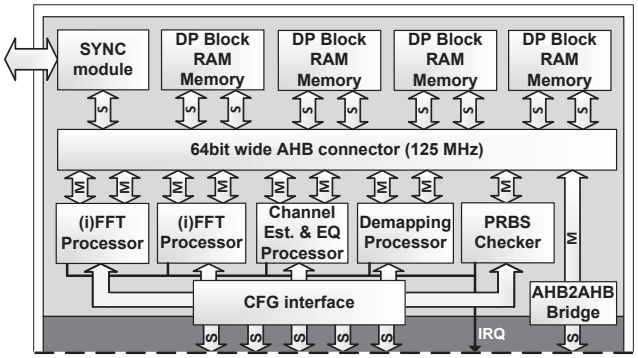

Fig. 3. The Signal Processing System consists of several specialized processors and several dual port Block RAM memories which are connected by a 64-bit wide AHB bus system.

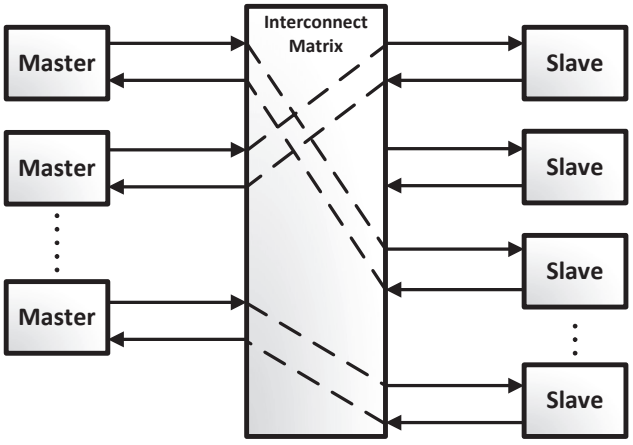

Fig. 4. The 64-bit wide multiplexer based multilayer AHB bus matrix allows parallel transactions in order to enable a full utilization of the SPS processors

mapping of binary data to the corresponding constellation points of the actual modulation scheme. For configuration and programming, all processors have an AHB slave interface.

One design aspect for all ordinary processors is to ensure a constant data throughput of two samples per clock cycle. Therefore, each of them is fully pipelined and most of them have two master interfaces, one to fetch data, one to write back the results. Table I of section IV lists all developed SPS processors.

The (i)FFT processor type is an exception since this processors can only provide a throughput of two samples per clock if the FFT size is smaller than 64. If it is bigger, either more processors are needed in order to perform a joint calculation of the transform or one processor needs to process the data several times and therefore will have no constant throughput of two samples per clock cycle anymore. When used in joint mode, x processors can calculate FFTs with a maximum size as given in equation 1.

$$\text{Max. FFT size of x processors} = 2^{(\mathrm{X}\times 4+1)} \qquad (1)$$

3) Dual Port Block RAM Memory: Another very critical component for a high-speed OFDM signal processing system is the main memory. It is essential to provide enough bandwidth for all processors working in parallel. To avoid a very complex memory controller as well as for flexibility and scalability reasons, we decided to use the on-chip memory components of the FPGA (Block RAM) for this job.

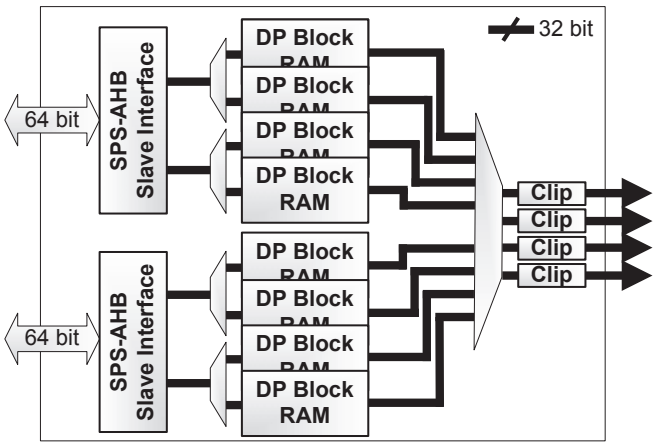

Fig. 5. This output data conversion module converts two 2-samples-per-clock data streams into one 4-samples-per-clock data stream.

Every of the dual port AHB Block RAM memories is 64-bit wide and provides two AHB slave ports which allow a parallel access of the memory content. The depth of one memory block and the total amount of memory blocks is configurable during design time of the SoC.

4) Data Converters: In order to be able to process more than two samples in parallel, special modules are required which convert one sequential stream of samples (e.g. one OFDM frame) of a high sample rate into several chunks of data (e.g. OFDM symbols) with a smaller sample rate, or vice versa. The smaller chunks can then be processed in parallel by multiple processors.

The mentioned conversion is done by using both ports of several Block RAM primitives of the FPGA. The concept is illustrated in figure 5 for a transmitter system which can process four samples per clock cycle. Here, the output data converter provides two 64-bit wide AHB slave interfaces to the SPS which are used to write two OFDM Symbols (two samples per symbol per clock cycle) in parallel into Block RAM memories. Afterwards, the data is read from the second ports of the Block RAMs, four Samples per clock cycle, one OFDM symbol after another.

The data conversion module of a transmitter includes additional functionality like clipping, functionality to add a cyclic prefix and an extra memory for an arbitrary preamble. The receiver includes the data conversion module in the synchronization processor. After timing synchronization to find the start of the frame and frequency synchronization to keep intersymbol interferences (ISI) low, the cyclic prefix of an OFDM symbol is removed and the symbols can be partitioned into several parallel data streams.

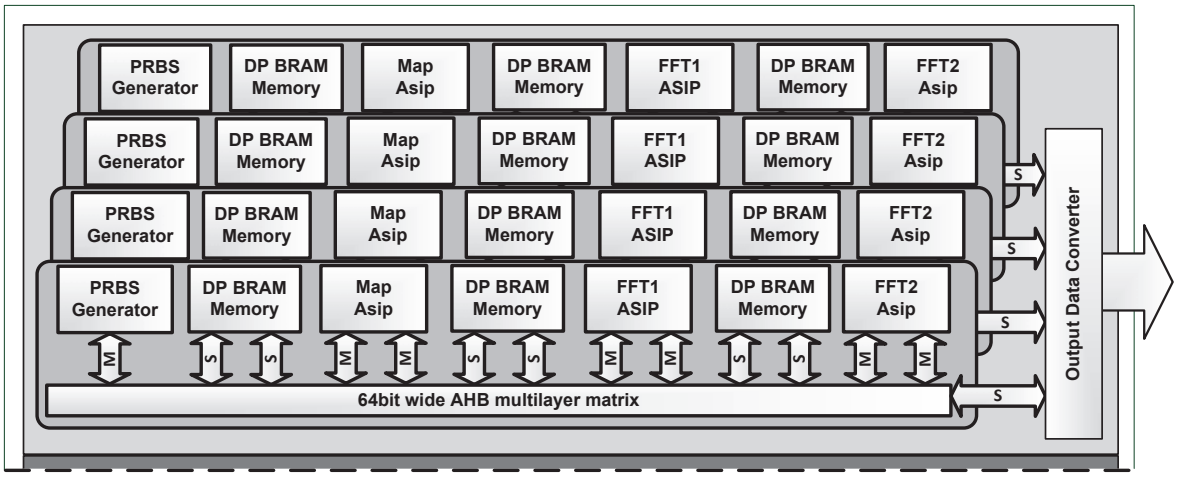

Fig. 6. SPS of a four-stream transmitter design as used in the experiments. In order to trim the system for performance, it is possible to implement several streams in parallel. In this case each stream generates OFDM symbols independently. The symbols are then concatenated by the output data converter. For data other than a PRBS, an additional component would have to partition the data in order to assign it to the multiple streams.

C. Performance vs. Cost Tradeoff

The modular concept of the MPSoC allows a tradeoff between the maximum performance and the resource consumption (performance vs. cost) of the system during design time. An example for optimizing the system for resources (cost) would be to include only (i)FFT processors, the output data converter and a synchronization module in the SPS. All calculation would be executed by the CS processor, except for the (i)FFT which would be outsourced to the SPS, highly increasing the performance when compared to a processing solely by the CS processor. By using the output data converter and the synchronization module, the system would still be able to send and receive OFDM Frames (e.g. bursts) with a high sample rate, but not continuously.

The other extreme, a complete optimization for performance, can be seen in figure 6. All OFDM processing in this transmitter takes place in the SPS. The SPS consists of several *streams* which each include all the necessary processors and memories to run constantly, processing two samples per clock. Each stream operates independent of each other, using an own bus matrix. The CS processor programs all the SPS processors one time, afterwards they will run autonomously, executing their program whenever they receive an interrupt from the preceding processor of the processing chain. This system configuration will allow sending one OFDM frame after another. Due to clarity the illustration in figure 6 does neither show the configuration interface nor additional logic like e.g. the bus bridge or analog interface to the digital to analog converters.

IV. Experiments and Results

A. System Configuration

In order to verify the functionality of the MPSoC we set up a transmitter experiment using a four-stream system configuration as illustrated in figure 6. With a SPS clock frequency of 125 MHz such an OFDM transmitter is able to generate continuous OFDM signals at one giga-sample per second. The CS is operating at 75 MHz and consists of Standard IP-cores from Gaisler Research [9], as seen in figure 2. The information sent was a pseudorandom binary sequence (PRBS) generated inside the MPSoC.

Each SPS stream consists of a PRBS generator, a modulation processor and two (i)FFT processors supporting (i)FFT sizes up to 512 subcarriers when working in joint mode. The output data converter is implemented only once but provides every stream with an own 64-bit wide AHB slave interface. The resource consumption for single modules as well as for the complete system is included in table I. Most processors vary in their resource consumption depending on several parameters the designer has to set at design time, e.g. size of memories, maximum size of preamble or maximum size of (i)FFT. The values in table I are valid for the standard settings we used for our four-stream transmitter design.

When comparing the resource consumption it gets obvious that a receiver system consumes a lot more resources due to the high resource consumption of processors which are only needed for receiver side algorithms (e.g. synchronization or equalization). However, keep in mind that there is only one synchronization processor needed whose resource consumption is increasing with the amount of parallel streams. The values of table table I is for a configuration with four streams. To get normalized values the values for the synchronization module should be divided by four.

B. Experimental Setup

We used a Xilinx Virtex-6 HXT 40G/100G Development Platform from Hitech Global deploying a Xilinx XC6V380HXT FPGA for a prototype transmitter design. The FPGA was connected to two 12-bit MAX5881 digital to analog converters (DACs) from Maxim. To assess the quality of the OFDM baseband signal, the in-phase and quadrature-

TABLE I
RESOURCE CONSUMPTION

Module	LUTs	Registers	BRAMs	DSPs
Processors				
PRBS Generator	591	529	0	0
Modulation	1 833	984	3	0
(i)FFT	4 610	3 344	14	20
O. Data Converter[1]	2 266	1 409	40	0
Synchronization[1]	49 051	39 460	320	54
Equalizer	20 608	18 099	10	60
Demodulation	813	759	1	0
PRBS Checker	570	412	0	0
Other Modules				
Busmatrix 8 x 7	2 804	952	0	0
DP BRAM Mem.	16	16	4	0
AHB2AHB Bridge	608	327	2	0
Control System	17 175	13 631	28	4
Complete Designs				
1 Gsps Transmitter	91 524	71 921	316	165
1 Gsps Receiver	205 389	173 239	640	459

[1] configured for 4 parallel streams

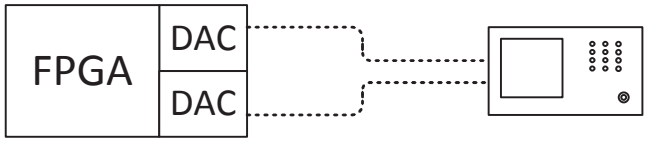

Fig. 7. To assess the OFDM baseband signal quality, the initial experimental setup recorded directly the outputs of the DACs. Afterwards, the signal was processed with Matlab.

phase component were recorded directly at the output of the DACs with an 80 giga-samples per second, real-time oscilloscope. The schematic of the initial experimental setup is illustrated in figure 7.

For this initial test we used an OFDM configuration with a 128-point inverse FFT to generate an OFDM-signal with 108 data carriers, 8 pilot tones and a cyclic prefix of 16 samples. One OFDM frame consisted of 244 data symbols and a 1312 sample preamble. For QPSK modulation of the subcarriers the resulting data rate was 1.44 gigabits per second while it was 2.88 gigabits per second for 16-QAM modulation.

The experimental setup for an optical transmission in a second experiment is depicted in figure 8. The baseband signal from the DAC outputs was up-converted to an electrical intermediate frequency by an analogue IQ-mixer. Subsequently the intensity of a semiconductor laser diode with a 3 dB-bandwidth of 16 GHz was modulated by a current carrying the passband signal.

The optical signal was detected by a photodiode. The modulated photocurrent was amplified and recorded with the real-time oscilloscope for offline demodulation. The transmitted signal consisted of 48 modulated subcarriers and 4 pilot tones

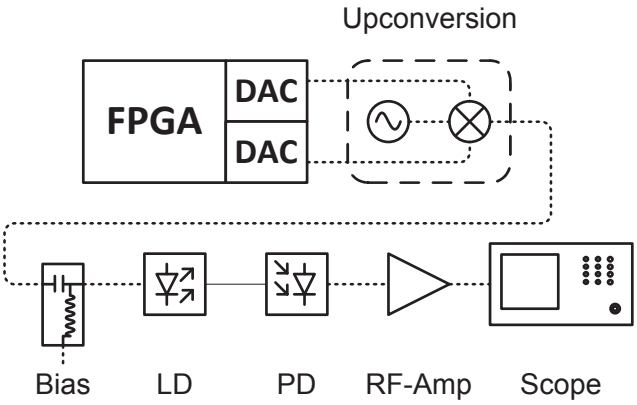

Fig. 8. The second experimental setup implemented the transmitter system in an optical OFDM communication setup. After mixing the generated signal on an intermediate frequency, it was used to directly modulate a laser. The receiver based on a photo diode and the similar offline processing as before.

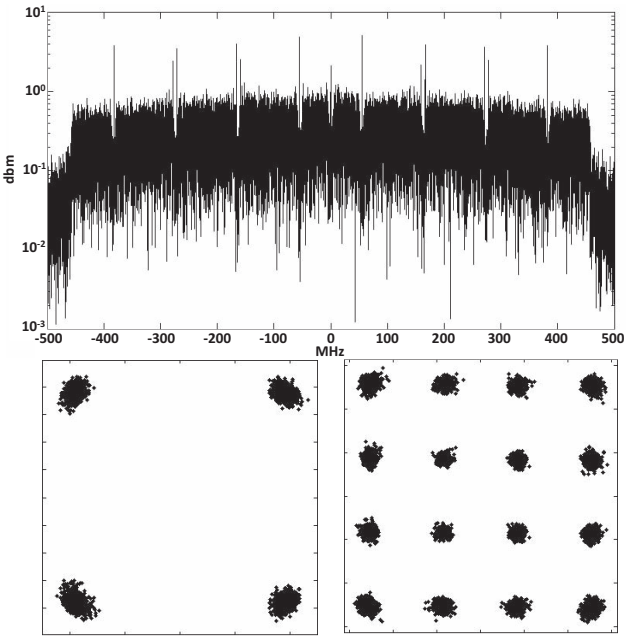

Fig. 9. Results of experiment one, recording directly the outputs of the Digital-to-Analog converters. On top is the electrical spectrum of the 128 subcarrier system. It is easy to identify the unmodulated outer carriers and the 8 pilot tones. The picture at the left bottom shows the constellation diagram for a QPSK configuration, the right bottom side is for a 16-QAM configuration.

generated by a 64-point iFFT. The number of OFDM-symbols per frame was 448 and the preamble had a size of 800 samples. Here the resulting data rates were 1.17 gigabits per second (QPSK) and 2.35 gigabits per second (16-QAM), respectively. The signal quality was evaluated by means of the error vector magnitude normalized to the longest ideal constellation vector (EVM_m), see [12].

C. Results

Figure 9 shows the electrical spectrum as well as the constellation diagrams of the measured baseband signal of

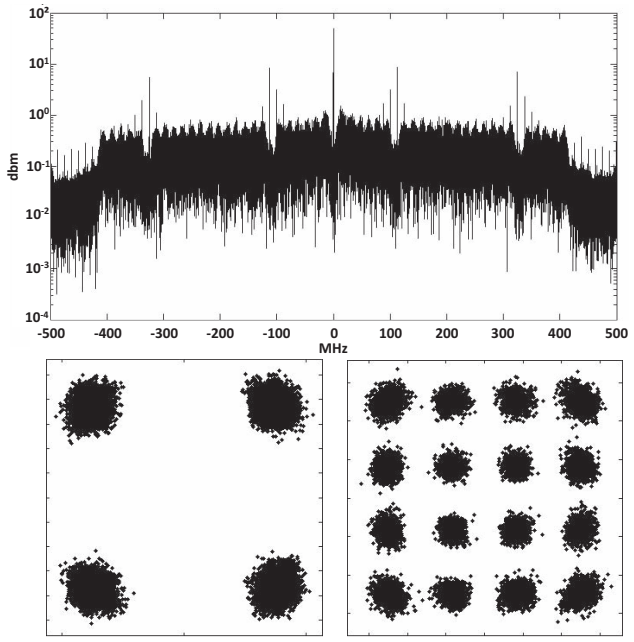

Fig. 10. The results above derive from the optical OFDM experiments with a 64 subcarriers configuration using 4 pilot tones. The electrical spectrum is on top, the constellation diagram on the left bottom side is for QPSK, the one on the right bottom side for 16-QAM.

the DAC outputs for the QPSK and 16-QAM configuration. At the DAC sample rate of 1 giga-sample per second the maximum signal bandwidth is 1 gigahertz. However, the outer 5 to 6 subcarriers remained unmodulated, thus narrowing the bandwidth. The EVM_m of the signal was 3.8% for the QPSK configuration and 5.1% using 16-QAM modulation.

Figure 10 shows the electrical spectrum and constellation diagrams of the optically transmitted signal showing a successful reception and demodulation. Due to the smaller number of subcarriers, the individual pilot tones are more distinct. The EVM_m of the received signal with QPSK-modulated subcarriers was 10.7% while it was 8.4% for the 16-QAM configuration. Both values remain well below the threshold values for error-free reception [12].

V. Conclusion

With the new system the authors introduce an OFDM transceiver framework which enables a new tradeoff between flexibility and performance. Due to the possibility of changing all the important OFDM parameters easily by software, a transceiver using this architecture can be deployed in many different OFDM applications and therefore offers a high reusability, almost similar to a digital signal processor based solution. At the same time the system is able to provide data rates of several gigabits per second, a performance which previously was reserved for customized dedicated circuit designs only.

Furthermore, due to its modular concept following industry standards, the novel transceiver system enables system designers to cover an even wider range of OFDM applications by allowing a tradeoff between performance and cost during design time. By following industry standards, the effort for custom extensions is reduced as well.

The developed system demonstrates how powerful a modern specialized multiprocessor system on chip can be. Even system designers with no knowledge of hardware design can set up a wide range of high speed OFDM systems using a high-level programming language. Depending on the configuration and the number of processors, the proposed transceiver system achieves data rates of several gigabits per second. To our knowledge, this architecture is the fastest *software-defined* OFDM transceiver system to date.

Acknowledgment

The authors acknowledge support from the BMBF joint project CONDOR, the EU research projects ACCORDANCE and OTONES, the Xilinx University Program (XUP) and Maxim Integrated.

References

[1] R. Schmogrow, M. Winter, D. Hillerkuss, B. Nebendahl, S. Ben-Ezra, J. Meyer, M. Dreschmann, M. Huebner, J. Becker, C. Koos, W. Freude, and J. Leuthold, "Real-time OFDM transmitter beyond 100 Gbit/s," *Opt. Express*, vol. 19, no. 13, pp. 12740–12749, Jun 2011.

[2] M. Dreschmann, J. Meyer, M. Hubner, R. Schmogrow, D. Hillerkuss, J. Becker, J. Leuthold, and W. Freude, "Implementation of an ultra-high speed 256-point FFT for Xilinx Virtex-6 devices," in *Industrial Informatics (INDIN), 2011 9th IEEE International Conference on*, 2011, pp. 829–834.

[3] Hen-Geul Yeh and V.R. Ramirez, "Implementation and Performance of a M-ary PSK and QAM-OFDM System in a TMS320VC5416 Digital Signal Processor," in *Digital Telecommunications, 2007. ICDT '07. Second International Conference on*, 2007, pp. 19–19.

[4] Hai Yan, Shengli Zhou, Zhijie Shi, Jun-Hong Cui, Lei Wan, Jie Huang, and Hao Zhou, "DSP implementation of SISO and MIMO OFDM acoustic modems," in *OCEANS 2010 IEEE - Sydney*, 2010, pp. 1–6.

[5] C. Toal, S. Sezer, D. Burns, P. Xaio, and V. Fusco, "A 1Gbps FPGA-based wireless baseband MIMO transceiver," in *SOC Conference (SOCC), 2012 IEEE International*, 2012, pp. 202–207.

[6] Rene Schmogrow, Marcus Winter, Bernd Nebendahl, David Hillerkuss, Joachim Meyer, Michael Dreschmann, Michael Huebner, Juergen Becker, Christian Koos, Wolfgang Freude, and Juerg Leuthold, "101.5 Gbit/s Real-Time OFDM Transmitter with 16QAM Modulated Subcarriers," in *Optical Fiber Communication Conference*. 2011, p. OWE5, Optical Society of America.

[7] Wei-Hsiang Tseng, Ching-Chi Chang, and Chorng-Kuang Wang, "Digital VLSI OFDM transceiver architecture for wireless SoC design," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, 2005, pp. 5794–5797 Vol. 6.

[8] P. Gruijters and B. Vandewiele, "Algorithm Partitioning and SoC Design for OFDM Communication Systems Using Multiple Application Specific Processors," in *Multimedia, 2006. ISM'06. Eighth IEEE International Symposium on*, 2006, pp. 481–487.

[9] Gaisler Research, *GRLIB IP Library version 1100, build 4108*, Aeroflex (http://www.gaisler.com/index.php/products/ipcores/soclibrary).

[10] T.M. Schmidl and D.C. Cox, "Robust frequency and timing synchronization for OFDM," *Communications, IEEE Transactions on*, vol. 45, no. 12, pp. 1613 –1621, Dec. 1997.

[11] E Raja and KV Ramana, "Implementation of Multilayer AHB Busmatrix for ARM," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 1, no. 5, pp. 27–30, November 2011.

[12] R. Schmogrow, B. Nebendahl, M. Winter, A. Josten, D. Hillerkuss, S. Koenig, J. Meyer, M. Dreschmann, M. Huebner, C. Koos, J. Becker, W. Freude, and J. Leuthold, "Error vector magnitude as a performance measure for advanced modulation formats," *Photonics Technology Letters, IEEE*, vol. 24, no. 1, pp. 61–63, 2012, Corrections in: IEEE Photon. Technol. Lett. 24 (2012) 2198.