Rapid Prototyping System for the Evaluation of MIMO Receive Algorithms

Luis G. Barbero, *Student Member, IEEE,* and John S. Thompson, *Member, IEEE*

***Abstract*— A rapid prototyping system for the analysis of multiple input-multiple output (MIMO) algorithms is presented in this paper. It combines the flexibility of MATLAB with the processing power of existing field-programmable gate arrays (FPGAs) to speed up the implementation and testing of real-time multiple-antenna receiver algorithms, suitable for integration into current wireless communication systems. As an application example, the sphere decoder (SD) for uncoded MIMO systems has been implemented and tested on this rapid prototyping system.**

***Index Terms*— Field-programmable gate array (FPGA), multiple input-multiple output (MIMO), rapid prototyping, spatial multiplexing, sphere decoder, wireless communications.**

I. Introduction

THE use of multiple input-multiple output (MIMO) technology has become the new frontier of wireless communications. It enables high-rate data transfers and improved link quality through the use of multiple antennas at both transmitter and receiver [1]. In recent years, prototyping multiple-antenna systems has become increasingly important [2], [3]. These prototypes are used as a means of validating results anticipated by theoretical research, as product demonstrators or as real-time testing platforms. A wide variety of MIMO prototypes exist that can be classified into three different categories according to their features and their main research interest:

1) Complete real-time MIMO system with a wireless link to validate the integration of the technology into wireless communication systems [4], [5].
2) Real-time MIMO system using channel emulators to test different channel conditions [6].
3) Offline MIMO processing with an RF front-end to concentrate on channel characterization [7]-[9].

The prototyping alternatives described above pay special attention to the different aspects of system integration and real wireless channel testing. As a result, the MIMO algorithm block is normally reduced to implementing a well-known existing algorithm, selected according to computer simulations or results found in the literature. Therefore, these prototypes are not suitable for implementing and testing novel MIMO algorithms to expand initial simulation results. To overcome this problem, the prototyping system proposed here concentrates only on the signal processing aspects of the MIMO algorithm using a field-programmable gate array (FPGA), with the rest of the MIMO system being computer-simulated. With this approach, the real-time implementation of the MIMO algorithm is embedded in a computer-based system providing a true hardware-in-the-loop platform.

This work was partially supported by Alpha Data Ltd. (www.alpha-data.com).

The authors are with the Institute for Digital Communications at the University of Edinburgh, EH9 3JL Edinburgh, UK (e-mail: l.barbero@ed.ac.uk; john.thompson@ed.ac.uk)

II. MIMO System

The uncoded MIMO system considered has M transmit and N receive antennas, with $N \geq M$, denoted as MxN. The signals to be transmitted are spatially multiplexed and sent in parallel, as shown in Fig. 1, resulting in a data rate increase compared to an equivalent single-antenna system. The wireless channel adds noise to the received signals and provides a propagation path from each transmit to each receive antenna resulting in interference between the transmitted signals. The 'MIMO Algorithm' block in Fig. 1, that compensates for the effect of the wireless channel, is targeted for a hardware implementation.

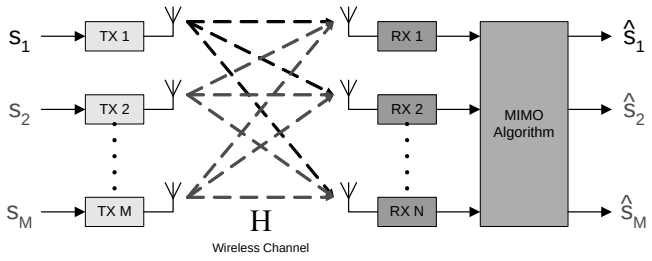

Fig. 1. MIMO system block diagram

A. MIMO Receive Algorithms

The algorithms for the detection of spatially multiplexed MIMO signals can be divided into four categories [10], [11]:

1) *Linear detector*: it inverts the channel effect using zero forcing (ZF) or minimum mean-square error (MMSE) criterion.
2) *Successive interference cancellation detector*: iterative version of the previous detector using the vertical Bell Labs layered space time (VBLAST) algorithm.
3) *Maximum-likelihood detector (MLD)*: it provides optimum maximum likelihood (ML) performance at the expense of an exponential complexity with M.
4) *Sphere decoder (SD)*: it reduces the complexity of the MLD while still providing ML performance.

Among those algorithms, only the first three have received substantial attention from a rapid prototyping point of view.

III. Rapid Prototyping System

The rapid prototyping system must have the simplicity and, at the same time, the flexibility required to quickly move from a computer-based implementation of an algorithm to its real-time implementation. As opposed to previous prototyping approaches, the focus of our approach is on the analysis of the MIMO algorithm. It is possible, then, to analyze novel MIMO algorithms while, at the same time, looking at their hardware implications. The main features to facilitate that process are the following:

- A reconfigurable hardware platform to implement and analyze different MIMO algorithms.
- A prototyping methodology that does not require detailed knowledge of the underlying hardware to allow a rapid implementation of the algorithms.
- A uniform testing environment to compare computer-based simulations and hardware execution.
- Possibility of running the algorithms in real-time to characterize their throughput (i.e. number of bits that can be detected per second).
- A simple and flexible interface to synchronize the hardware platform and the computer-based MIMO system.

A. Hardware Platform

An FPGA board has been selected as the hardware platform for the study and prototyping of MIMO algorithms, given the high level of parallelism of FPGAs and their evolution, with higher densities and the addition of embedded multipliers. The platform for this work has been provided by Alpha Data Ltd. and consists of an ADC-PMC peripheral component interconnect (PCI) adapter board that hosts two FPGA boards: an ADM-XRC-II with a Xilinx Virtex-II (XC2V4000) and an ADM-XP with a Xilinx Virtex-II-Pro (XC2VP70), both with external SRAM memory for data storage. Fig. 2 shows the block diagram of the ADM-XP board.

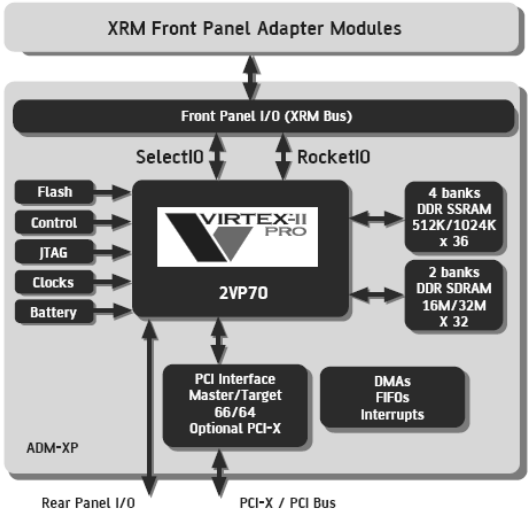

Fig. 2. ADM-XP block diagram (after www.alpha-data.com)

The platform provides the required degree of reconfigurability. In addition, the PCI interface and the board drivers guarantee a seamless synchronization and data transfer between the MIMO algorithm implemented on the FPGA and the rest of the MIMO system running on the host PC.

B. Rapid Prototyping Methodology

The rapid prototyping methodology selected is based on MATLAB, Simulink and the available Xilinx tools for the implementation of FPGA designs tailored to the Alpha Data FPGA boards. Fig. 3 shows the methodology that has been used for the rapid prototyping of the MIMO receive algorithms.

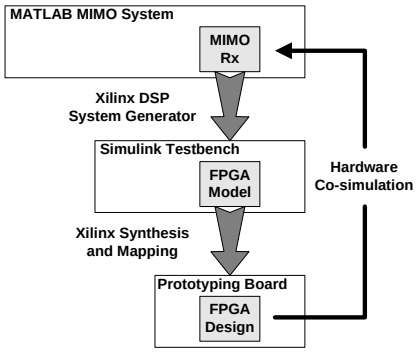

Fig. 3. Rapid prototyping methodology

Initially, MATLAB is used to implement a complete MIMO system including transmitter, channel simulator and receiver. This system provides us with the flexibility required to combine transmit/receive algorithms with different channel conditions. Although the modelling of a practical hardware system is limited with MATLAB, it allows us to quickly identify receive algorithms that could be targeted for the FPGA.

Once the algorithms of interest have been identified, they need to be implemented on the FPGA using a prototyping tool as similar as possible to a symbolic or mathematical programming language. For that purpose, the Xilinx DSP System Generator is used to implement the MIMO algorithms on the FPGA [12]. The tool is embedded in a Simulink environment and provides different blocks to perform basic mathematical and bit operations that can be directly mapped on the FPGA for real-time execution. This high level of abstraction provides two main advantages:

- The use of hardware description languages is not required to translate the signal processing algorithms from MATLAB to System Generator.
- The debugging of the FPGA design is simplified because mathematical operations of the algorithms can be easily identified in the FPGA model.

At the same time, this high level of abstraction also allows the FPGA design to be optimized. This is achieved by selecting the appropriate precision in the datapath operations and scheduling the different parts of the algorithm to make an optimum use of the processing power of the FPGA.

The development of the FPGA model is embedded in a Simulink testbench that has the basic transmit, receive and wireless channel functionalities of the complete MATLAB

MIMO system to allow the transmission and reception of data frames using the FPGA model of the MIMO receive algorithm. This testbench facilitates the debugging of the design in the development stage, with the possibility of monitoring every signal in the FPGA model.

The FPGA model is then synthesized, mapped and routed using Xilinx synthesis tools. The output data generated by those processes can be used to analyze the resource use and the timing constraints of the design, identifying where the bottlenecks of the design are in order to improve it.

After the final version of the model has been synthesized, a bitstream is generated for the hardware co-simulation of the FPGA design. This hardware co-simulation of the algorithm uses a memory interface embedded in a Simulink environment to synchronize and transfer data to and from the host PC. The communication between the Simulink environment and the FPGA is handled internally using shared memory blocks implemented in the FPGA, and externally using SRAM devices on the board.

Finally, the hardware design on the FPGA and the Simulink interface are embedded in the complete MATLAB model to evaluate the performance of the hardware implementation and compare it with MATLAB. In order to have a seamless integration in the MATLAB model, special routines have been implemented to adapt the data format to the FPGA memories used for synchronization.

The rapid prototyping methodology presented allows us to have a fast path from the original algorithm, with no assumptions about the hardware platform, to the final implementation running on the FPGA. In addition, the MIMO system model can perform real-time hardware-in-the-loop testing of the MIMO receive algorithm.

IV. Application Example: Sphere Decoder (SD)

A SD for MIMO systems has been implemented using the rapid prototyping system described above. The increasing interest on the SD lies on the fact that it provides ML performance with reduced complexity compared to the MLD. Therefore, the SD can be integrated into actual systems where the number of antennas or the constellation size make the MLD unsuitable due to its high complexity.

The MIMO testbed consists of a 4x4 system where the transmitted symbols from each antenna are taken independently from a 16-quadrature amplitude modulation (QAM) constellation. Assuming symbol-synchronous receiver sampling and ideal timing, the equivalent baseband received 4-vector, using matrix notation, is given by

$$\mathbf{r} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1}$$

where $\mathbf{s} = (s_1, s_2, ..., s_4)^T$ denotes the vector of transmitted symbols with $\mathrm{E}[|s_i|^2] = 1/4$, $\mathbf{n} = (n_1, n_2, ..., n_4)^T$ is the vector of independent and identically distributed (i.i.d.) complex Gaussian noise samples with variance σ^2 and $\mathbf{r} = (r_1, r_2, ..., r_4)^T$ is the vector of received symbols. The 4x4 matrix $\mathbf{H}$ denotes the flat fading channel matrix where h_{ij} is the complex transfer function from the jth transmit antenna to the ith receive antenna. The entries of $\mathbf{H}$ are modelled as i.i.d. Rayleigh fading with $\mathrm{E}[|h_{ij}|^2] = 1$.

The main idea of the SD is to reduce the computational complexity of the MLD by searching over only the receive, noiseless constellation points that lie within a hypersphere of radius R around the received signal. This process is represented by

$$\hat{\mathbf{s}} = \arg\{\min_{\mathbf{s}} \|\mathbf{r} - \mathbf{H}\mathbf{s}\|^2 \leq R^2\} \tag{2}$$

where the initial radius R is selected according to the noise level [11].

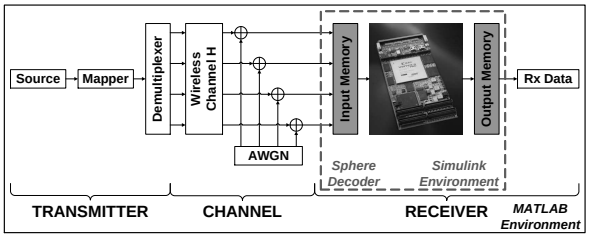

Fig. 4. Hardware-in-the-loop MIMO system diagram

The SD has been prototyped using the Xilinx DSP System Generator and integrated in the complete MIMO system. The communication between the MATLAB system and the FPGA is performed using internal RAM memory in the FPGA and memory models in Simulink that are directly accessible from MATLAB. The input memory of the FPGA contains the received symbols from all the receive antennas, the channel information and the initial radius required for the SD operation. The output memory contains the detected bits that are sent back to the MATLAB model. Fig. 4 shows the integration of the hardware implementation of the SD in the MATLAB system model.

TABLE I
FPGA Resource Use of 4-SDs

XC2VP70	Use	Percentage
Number of slices	21,467 / 33,088	64%
Number of flip-flops	17,691 / 66,176	26%
Number of 4-input LUT	36,249 / 66,176	54%
Number of embedded multipliers	156 / 328	47%
Number of block RAM	183 / 328	55%

Table I summarizes the resource use of the parallel implementation of 4 SDs on the Xilinx Virtex-II-Pro FPGA [13]. It can be seen that the 4 SDs use around half of the FPGA resources making intensive use of the RAM memory blocks. The number of memory blocks used is mostly due to the input and output buffers required for hardware co-simulation and communication with the MATLAB system model. The percentage of FPGA slices used is 64%, but it should be noted that each slice contains two flip-flops and two look-up tables (LUTs). Looking at their percentage of use, we can see that most of the slices are only partially used. This initial version of the SD could be optimized reusing the embedded multipliers when they are idle and also using approximations for the most computationally intensive operations like the Euclidean distance calculation.

Fig. 5 shows the fixed-point bit error ratio (BER) performance of the SD on the FPGA obtained over 20,000 channel realizations, compared with the floating-point MATLAB performance. The FPGA real-time execution matches the computer simulations except at high signal to noise ratio per bit due to the quantization noise (16 bits used per real and imaginary components).

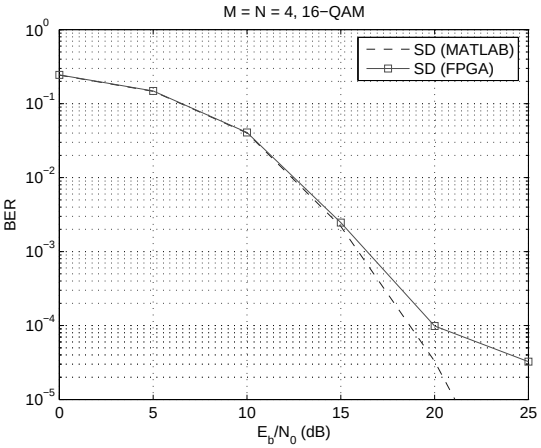

Fig. 5. BER performance of the FPGA-SD compared to the MATLAB-SD as a function of the signal to noise ratio per bit.

Table II compares the throughput of the parallel FPGA implementation of the SD with two application-specific integrated circuit (ASIC) implementations of the SD [14]. The FPGA implementation achieves a similar performance to that of ASIC 1, while using only half of the resources available on the FPGA. Therefore, the system could be improved adding more SDs in parallel on the same platform, increasing the throughput. In addition, the optimizations on ASIC 2 could also be integrated into the FPGA design.

TABLE II
COMPARISON OF REAL-TIME SD IMPLEMENTATIONS

SD	FPGA	ASIC 1 [14]	ASIC 2 [14]
MIMO configuration	4x4	4x4	4x4
Modulation	16-QAM	16-QAM	16-QAM
Granularity	4-SD	Single-SD	Single-SD
BER perfomance	ML	ML	close to ML
Clock Frequency (MHz)	50	51	71
Throughput at E_b/N_0 = 20dB (Mbps)	114.5	126	253

V. CONCLUSION AND FUTURE WORK

A rapid prototyping system has been proposed for the implementation, analysis and testing of MIMO receive algorithms in real-time. As opposed to previous prototyping approaches, the system here presented focuses on the study of advanced MIMO algorithms rather than on the integration of different components for complete system prototyping. By using existing FPGAs and the Xilinx DSP System Generator, the computer-simulated MIMO algorithms can be translated into an FPGA model without extensive hardware knowledge and reducing the design time. We believe that with this prototyping methodology we can close the gap between the software engineers who use a high level language for the modelling and evaluation of MIMO algorithms and the hardware engineers who concentrate on adapting a fixed MIMO algorithm for a given hardware platform.

This rapid prototyping system will be used for the analysis of new MIMO algorithms, now being evaluated in software, to increase the throughput of the SD without increasing the required processing power and keeping a BER performance as close as possible to the ML performance. For that purpose, new architectures suited to the high level of parallelism of FPGAs need to be investigated.

ACKNOWLEDGMENT

The authors would like to thank Alpha Data Ltd. for their support of this work and Dr. Andrew McCormick from Alpha Data for his continuous help in the integration of the prototyping platform.

REFERENCES

[1] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Technical Journal*, pp. 41–59, Oct. 1996.

[2] M. Rupp, A. Burg, and E. Beck, "Rapid prototyping for wireless designs: the five-ones approach," *Signal Processing*, vol. 83, pp. 1427–1444, 2003.

[3] T. Kaiser, A. Wilzeck, M. Berentsen, and M. Rupp, "Prototyping for MIMO systems: An overview," in *Proc. 12th European Signal Processing Conference (EUSIPCO '04)*, Viena, Austria, Sept. 2004.

[4] A. Adjoudani, E. C. Beck, A. P. Burg, G. M. Djuknic, T. G. Gvoth, D. Haessig, S. Manji, M. A. Milbrodt, M. Rupp, D. Samardzija, A. B. Siegel, T. Sizer, C. Tran, S. Walker, S. A. Wilkus, and P. W. Wolniansky, "Prototype experience for MIMO BLAST over third-generation wireless systems," *IEEE J. Select. Areas Commun.*, vol. 21, no. 3, pp. 440–451, Apr. 2003.

[5] R. M. Rao, W. Zhu, S. Lang, C. Oberli, D. Browne, J. Bhatia, J.-F. Frigon, J. Wang, P. Gupta, H. Lee, D. N. Liu, S. G. Wong, M. Fitz, B. Daneshrad, and O. Takeshita, "Multi-antenna testbeds for research and education in wireless communications," *IEEE Commun. Mag.*, vol. 42, no. 12, pp. 72–81, Dec. 2004.

[6] P. Murphy, F. Lou, A. Sabharwal, and J. P. Frantz, "An FPGA based rapid prototyping platform for MIMO systems," in *Proc. 37th Asilomar Conference on Signals, Systems and Computers*, vol. 1, Pacific Grove, CA, Nov. 2003, pp. 900–904.

[7] A. van Zelst and T. Schenk, "Implementation of a MIMO OFDM based wireless LAN system," *IEEE Trans. Signal Processing*, vol. 52, no. 2, pp. 483–494, Sept. 2004.

[8] W. Xiang, D. Waters, T. G. Pratt, J. Barry, and B. Walkenhorst, "Implementation and experimental results of a three-transmitter three-receiver OFDM/BLAST testbed," *IEEE Commun. Mag.*, vol. 42, no. 12, pp. 88–95, Dec. 2004.

[9] A. Gupta, A. Forenza, and R. W. Heath, "Rapid MIMO-OFDM software defined radio system prototyping," in *Proc. IEEE Workshop on Signal Processing Systems (SIPS '04)*, vol. 1, Austin, TX, Oct. 2004, pp. 182–187.

[10] M. Rupp, M. Guillaud, and S. Das, "On MIMO decoding algorithms for UMTS," in *Proc. 35th Asilomar Conference on Signals, Systems and Computers*, vol. 2, Monterey, CA, Nov. 2001, pp. 975–979.

[11] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Trans. Inform. Theory*, vol. 49, no. 10, pp. 2389–2402, Oct. 2003.

[12] Xilinx, Inc., "Xilinx DSP System Generator v6.3 user guide," http://www.xilinx.com.

[13] L. G. Barbero and J. S. Thompson, "Rapid prototyping of the sphere decoder for MIMO systems," in *Proc. IEE Conference on DSP Enabled Radio (DSPeR '05)*, vol. 1, Southampton, UK, Sept. 2005, pp. 41–47.

[14] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, July 2005.