MAC and Baseband Hardware Platforms for RF-MIMO WLAN

Z. Stamenković[*], K. Tittelbach-Helmrich[*], M. Krstić[*], J. Ibanez[#], V. Elvira[#], and I. Santamaria[#]

[*]*Systems Department, IHP GmbH*
Im Technologiepark 25, 15236 Frankfurt (Oder), Germany
stamenkovic,tittelbach,krstic@ihp-microelectronics.com

[#]*Communications Engineering Department, University of Cantabria*
Plaza de la Ciencia s/n, 39005 Santander, Spain
jesus,victorea,nacho@gtas.dicom.unican.es

***Abstract* — The paper describes hardware solutions for the IEEE 802.11 MAC (Medium Access Control) layer and IEEE 802.11a digital baseband in an RF-MIMO WLAN transceiver that performs the signal combining in the analogue domain. Architecture and implementation details of the MAC processor including a hardware accelerator and a 16-bit MAC-PHY interface are presented. The proposed hardware solution is tested and verified using a PHY link emulator. Architecture, design, implementation, and test of a reconfigurable digital baseband processor are described too. Description includes the baseband algorithms (the main blocks being MIMO channel estimation and Tx-Rx analog beamforming), their FPGA-based implementation, baseband printed-circuit-board, and real-time tests.**

***Index Terms* — Baseband, MAC, MIMO, processor**

I. INTRODUCTION

Current multiple-input multiple-output (MIMO) wireless systems perform the combining and processing of the complex antenna signal in the digital baseband. Since complete transmitter and receiver are required for each path, the resulting power consumption and costs of the conventional MIMO approaches [1] limit applications for ubiquitous networks. A low-power and low-cost RF-MIMO (MIMAX) system for maximum reliability and performance (Figure 1) compliant to the IEEE Standard 802.11a [2] has recently been proposed [2], [4]. It significantly decreases the hardware complexity by performing the adaptive weighting and combining of the antenna signals in the RF front-end [5]-[8].

Multiple antennas are used to increase the transmission reliability through spatial diversity. Redesigns have mostly been done in the physical medium dependent (PMD) layer. They demand for changes in the physical layer convergence (PLC) and medium access control (MAC) protocols to optimally exploit the benefits of the new RF front-end [9]-[14]. The PLCP pursues mapping MAC protocol data units in PMD layer compliant frame formats. This task is common for all communication schemes defined by the IEEE Standard 802.11. Furthermore, the spatial diversity must be exploited, possible impairments in the RF spatial processing have to be compensated and the MIMO channel has to be estimated. Particularly, these tasks are not needed in the IEEE802.11a scheme, which is specified for SISO communication.

There are several differences between the MIMAX approach and the full multiplexing MIMO approach. In MIMAX, the same weight is used for all subcarriers in OFDM transmissions whereas it is possible to weight each subcarrier independently from the others in the full MIMO transmission scheme.

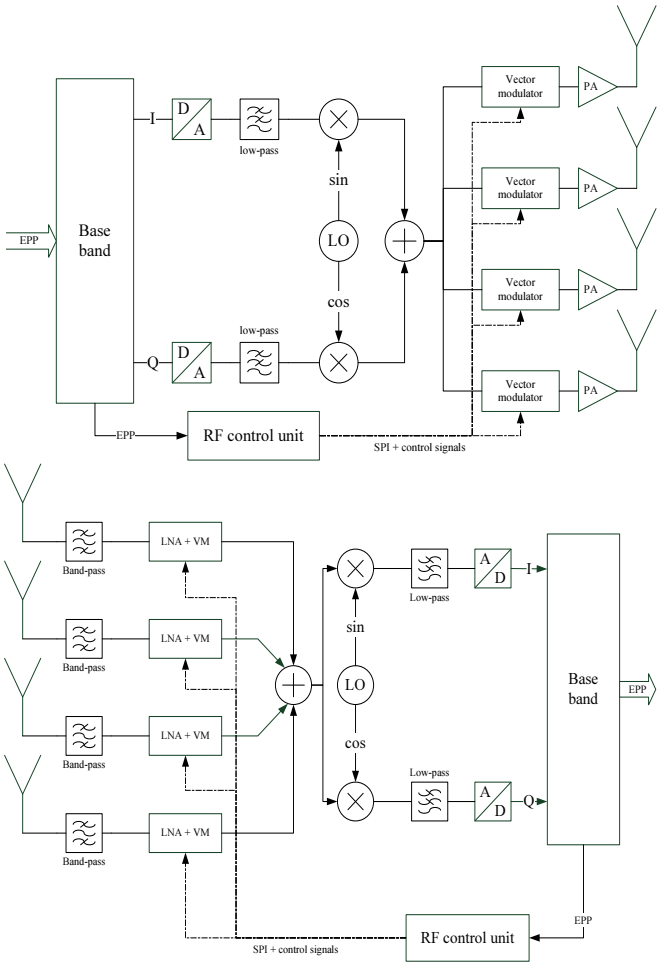

Figure 1: MIMAX transmitter and receiver

Integrating the signal processing in analogue circuits is limited in the maximum achievable resolution because of noise processes, process variations or nonlinear behavior of the devices. Therefore, the signal processing has to be calibrated by the baseband to adapt to the RF impairments. This mainly considers the correlation between real and imaginary parts of the vector modulator approach. Compensation is achieved by a calibration performed by the RF control unit in Figure 1. The characteristics of the vector modulator are analyzed by this module and stored in an internal memory. The weights provided by the baseband are then transferred into corresponding values of the vector modulator using the previously determined relationship and these new weights control the vector modulator. Integrating additional calibration options in the RF front-end and the RF control unit allow an internal adaptation to impairments of the fabrication process and a feedback to the baseband processing. These techniques are based on look-up tables or neural network approaches. The vector modulator is connected to the RF control unit by a serial peripheral interface (SPI).

The RF-MIMO analogue front-end needs new algorithms to exploit the available spatial diversity of the MIMO channel. Several challenges are addressed in the physical layer convergence protocol. First, the impairments of the RF front-end are considered in the baseband processor. The algorithms must operate reliably and robustly with respect to the limited resolution of the RF front-end. Moreover, these algorithms must determine the optimal complex weights to be applied at each antenna (implemented by means of vector modulators). The MIMO beamforming algorithms need channel state information at both sides of the link, which is obtained by a specific training procedure. Different optimization goals can be used when determining the optimal Tx/Rx weights [6]. Because of its simplicity, the maximization of the signal-to-noise-ratio (SNR) is the criterion chosen for implementation.

In order to test the modifications in the IEEE802.11 MAC layer [2], a simulation model of the IEEE802.11 WLAN has been developed in the Specification and Description Language (SDL) [15]. It is composed of simplified models for the 5 GHz OFDM physical layer (PHY), and a detailed model for the medium access control (MAC) layer. The model is used to verify the functional correctness of the MAC design and to investigate the performance.

The MAC processor architecture is presented in Section II. The hardware accelerator that performs the most time critical MAC functions is described in Section III. The baseband architecture is presented in Section IV. Functional modules of the baseband processor are described in Section V, VI, and VII. The implementation details are presented in Section VIII and test details in Section IX. The conclusions are drawn in Section X.

II. MAC ARCHITECTURE

The MAC protocol complies with the IEEE Standard 802.11 and accounts for the following extra requirements due to RF-MIMO technology:

1. Maintenance of a database of active and available users (MAC address, number of antennas at the user, last optimum weights, etc.).
2. Configuration of the transceiver's MIMO front end, i.e. the antenna weight coefficients, before sending or receiving WLAN frames.
3. Measurement of the channel parameters to determine the optimal weights for every WLAN connection.

Using the SDL simulation results, a sophisticated hardware/software partitioning of the MAC layer design is carried out to eliminate performance bottlenecks. Finally, the functionalities of transmitting and receiving paths (Figure 2) are assigned to a MAC processor that consists of a general purpose processor (MAC software) and an additional hardware accelerator (MAC hardware).

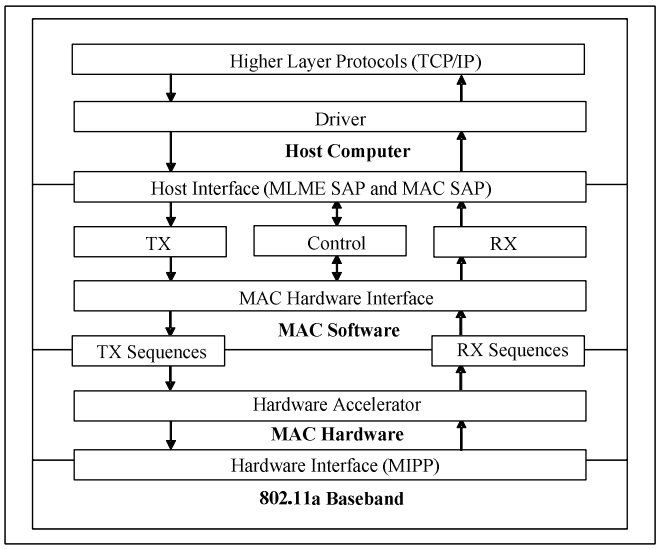

Figure 2. Hardware/software partitioning of the MAC layer

In order to develop a universal RF-MIMO WLAN board independent of any host computer system, we have implemented the complete IEEE 802.11 compliant MAC protocol on the WLAN module. No parts of the MAC need to be integrated into the host driver, which greatly relaxes timing demands within the host computer's operating system. The MAC layer is implemented as hardware/ software co-design for a 32-bit general purpose processor and the RF-MIMO specific hardware accelerator.

The software part of the MAC layer generally covers all functionality which is not timing critical or which benefits from great flexibility. This includes maintaining the queue of frames to be transmitted, deferring frame transmissions to stations in power-save mode, frame fragmentation in the transmitter (if desired) as well as de-fragmentation and duplicate detection at the receiver. Also, all the MAC management procedures like scanning, joining, authentication, association, etc. have been programmed in software.

The hardware accelerator functionality for the transmit direction includes a buffer for the next frame, the generation of cyclic redundancy checks (CRC) and an encrypt option. After having sent off the frame, the hardware accelerator waits

for the acknowledgement and signals the success or failure (timeout) of the frame transfer to the software. In the receive direction, a CRC checker, a frame address filter, the generation of acknowledgements and CTS frames and a decryption module are integrated in hardware. Tracking channel state (busy/idle) including back-off for sending frames, 6 timers (32 bit, timer tick 1 µs) and the system time (64 bit) are also provided as hardware modules.

A simplified functional architecture diagram of the MAC processor is shown in Figure 3. The blocks shown in the left part represent the MAC functions executed in software on a 32-bit General Purpose Processor (GPP). The right part sketches the functional scope of the hardware accelerator including an interface between the MAC and PHY layers called MIPP interface [15]. This parallel port interface is a combination of a 16-bit parallel bidirectional data bus and some control and handshake signals.

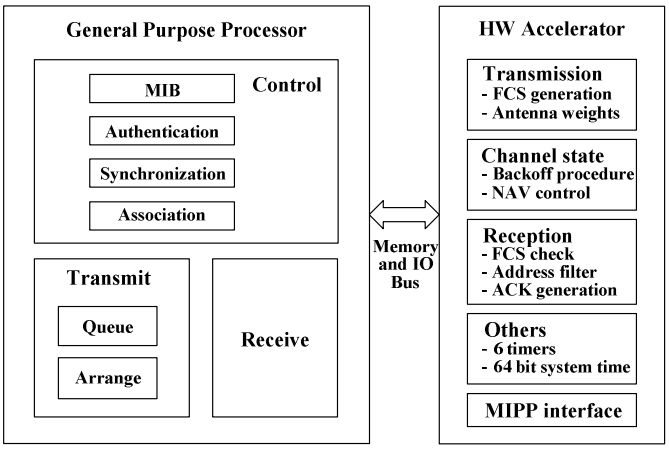

Figure 3. Functional block diagram of the MAC processor

The general purpose processor (Figure 4) is based on a MIPS32 4KEp core with instruction and data caches. All external interfaces including the MAC hardware accelerator are attached to the MIPS processor's memory bus as memory-mapped I/O components. The processor interfaces comprise a CardBus interface to a host PC, a serial RS232 interface for firmware download, an EJTAG interface with Test Access Port (TAP) acting as a hardware debugger, and general purpose I/Os (GPIO).

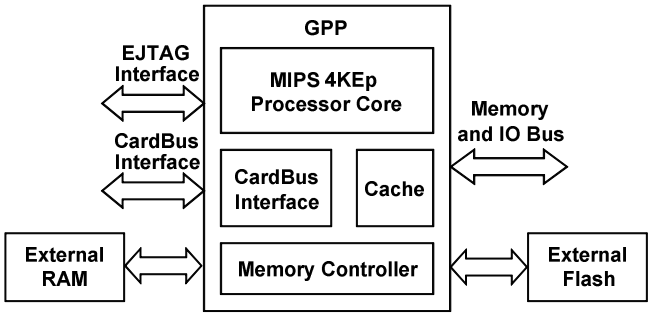

Figure 4. Hardware architecture of the general purpose processor

III. MAC Hardware Accelerator

Figure 5 represents architecture of hardware accelerator itself. The MAC interface consists of data bus, address buss and some control signals. There is set of instructions for the hardware accelerator implemented in MAC software. Access to specific modules is provided by the address decoder. The status register collects any relevant information about processes in other modules and thus allows communication with MAC software. The transmitter module provides functionality for the transmit direction and collision avoidance. The receiver fulfils its natural functionality described earlier. The control component is a broker between MAC and PHY.

All components accessing PHY via the MIPP interface are under the authority of an arbiter block. In order to increase the attainable system throughput, the authors have replaced the standard 8-bit EPP interface with a 16-bit interface.

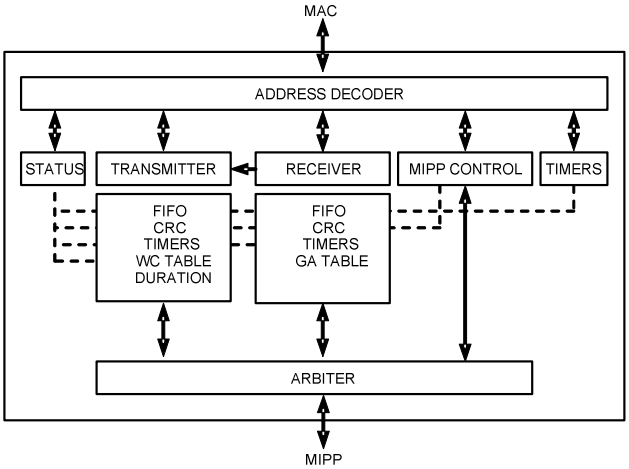

Figure 5. Block diagram of the hardware accelerator

This section describes details of the most time critical MAC functions and their implementation in hardware. The functionality of the hardware accelerator is defined and verified by simulation within the MAC SDL model. Finally, the hardware accelerator is designed in VHDL and implemented on an FPGA.

The transmitter tracks the channel state (idle or busy). It buffers the next frame and sends it after performing the back-off procedure. In parallel, it generates the cyclic redundancy checks (CRC). For frames, for which an acknowledgement is expected, it sets a respective timeout and checks for successful delivery. The transmitter block also contains a unit managing the IEEE802.11 Network Allocation Vector (NAV) which is a mechanism for channel time reservation in the case of frame fragmentation or to solve the hidden node problem in conjunction with RTS/CTS frames.

As a MIMO extension, the transmitter contains a table of antenna weight coefficients for distinct connections. It transfers the respective weight coefficient to the PHY layer before sending a frame. When a frame exchange sequence is finished, it sets some configurable default weight coefficients which should be good enough to receive a short RTS frame from any station. From the source address contained in the RTS frame, the optimal weight coefficients for that connection can be deduced and set in the PHY layer before receiving the (possibly long) frame itself.

The receiver comprises a CRC checker, a frame address filter, and the generation of acknowledgements and CTS frames. The control component, as a broker between MAC and PHY, sets and reads the PHY parameters, controls the timers for handshake of the MIPP interface, and stores the received data from PHY after any set/write command from MAC.

The arbiter controls the MIPP handshake and the access to bi-directional data bus. A special priority mechanism has been developed to prevent undesired delays in the data flow and raise the data reliability. The priority mechanism is implemented as a state machine driven by signals responsible for:

- reset,
- sending the frame data,
- sending and receiving the control data, and
- receiving the frame data.

Transmitted data have the highest priority. Then the control data come. After writing to the MIPP interface, the arbiter automatically will read one word from PHY. This atomic set of instructions prevents from unexpected data loss. Reading of the frame data from PHY has the lowest priority. Of course, when the reset occurs the state machine will stop for given number of clock cycles and go to idle state.

IV. Baseband Architecture

The architecture of the baseband processor is shown in Figure 6. It is composed of two main parts: the baseband processor implementing the IEEE Standard 802.11a and new MIMAX baseband modules implementing new functionalities required by the MIMAX RF front-end architecture.

The new functionalities are grouped into two main modules: channel estimator and MIMAX RF weights (or beamforming) block. These MIMAX modules will be active only when a MIMAX training frame is detected by the Tx/Rx control block, which transfers the MIMAX signal field data to the MIMAX control block in order to start the procedure (i.e. the MIMAX channel estimation and beamforming).

More precisely, the architecture of the baseband processor integrates the following modules:

- MIMAX channel estimation: This module estimates the $n_T n_R$ MIMO channel. The estimation is based on the FFT analysis of the $n_T n_R$ training OFDM symbols of the received training frame. The n_T and n_R parameters denote the numbers of transmit and receive antennas. It works in the frequency domain taking the FFT signal provided by the IEEE802.11a processor as input and uses a least squares estimation method (Section V).
- MIMAX RF weights: It takes the estimated MIMO channel as input and computes the optimal Tx/Rx beamforming weights using the Max-SNR algorithm described in Section VI. It is the most important block in terms of complexity and FPGA resources.
- Frequency offset estimation: Due to the residual frequency error at the output of the conventional IEEE802.11a synchronizer, it might be necessary to include a frequency offset estimator working in parallel with the MIMAX channel estimation and RF weights modules (Section VII). To estimate the frequency offset, it is necessary to transmit an additional training symbol, resulting in a training frame of $n_T n_R+1$ training symbols.
- Weight correction: This module multiplies the weights by a unitary (e.g. rotation) matrix in order to compensate the effects of the residual frequency offset and specific Tx/Rx beamformers used during training.

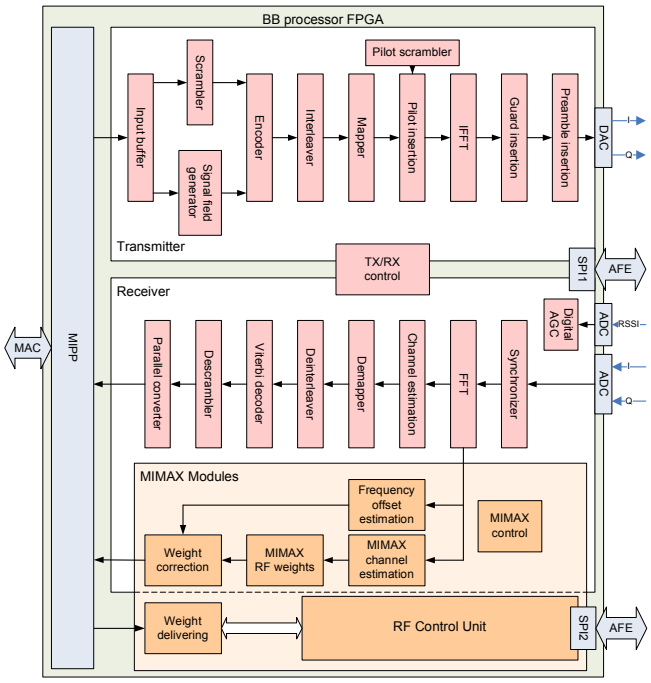

Figure 6. Architecture of the MIMAX baseband processor

- Weight delivery: It transfers the calculated optimal weights to the MAC processor (the weight updating). In addition, it allows applying (from the baseband) the predefined set of weights during training (the weight setting) and transferring (from MAC) the optimal or default weights during data transmission or reception (the weight uploading).
- MIMAX control: This module controls the signal and data flow among all MIMAX blocks. It receives from the Tx/Rx control block information included in the training frame signal field (the number of Tx/Rx antennas, the number of training symbols), as well as some activation and synchronization signals.
- RF control unit: This is a control interface between the baseband processor and analogue front-end (AFE). It is an integrated part of the baseband processor.

All the MIMAX blocks are activated only when a training frame is received. Therefore they can be powered down while either processing conventional data frames or transmitting training frames. Only the MIMAX control block, the weight delivery block, and the RF control unit remain active at any time because it must transfer and set the weights from the MAC processor to the RF control unit.

The complete baseband processor was initially designed using a Matlab model that uses floating-point operations to implement all processing stages. This floating-point model is useful to obtain an upper bound on the expected performance of the baseband processor, but cannot be used for FPGA implementation. A fixed-point Matlab model was then developed that allowed us to take design decisions with regard to the required precision (e.g., number of bits, number of iterations to be applied in the algorithms, etc.)

V. Channel Estimation

The MIMAX channel estimator uses the $n_T n_R$ training OFDM symbols included in a training frame. Each training symbol is affected by a specific pair of Tx and Rx beamformers. A conventional least squares algorithm is used to estimate the $n_T n_R$ equivalent SISO channels at the 52 active subcarriers.

Some design decisions has been taken in order to simplify the implementation of the MIMAX channel estimator. First, the identity matrix has been selected for the Tx and Rx beamforming matrices used during the training stage. Second, the MIMAX training symbols will be the same as the IEEE802.11a long training symbols composed of 52 subcarriers modulated by BPSK values.

As Figure 7 shows, the MIMAX channel estimator works in the frequency domain (i.e., after FFT) and could include an optional post filtering procedure to smooth the resulting frequency responses. From an implementation point of view, the LS estimator requires very few FPGA resources (just sign inverters and control logic), but the post filtering process could be expensive in terms of memory and MACs (while providing marginal BER improvement). For this reason, we have initially designed only the LS version of the MIMAX channel estimator block.

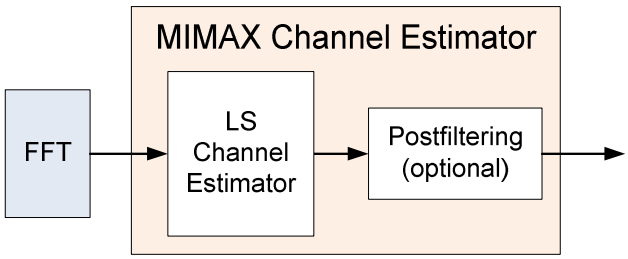

Figure 7. MIMAX channel estimation

VI. Beamforming Weights Calculation and Delivery

We have focused on the implementation of the Max-SNR beamforming algorithm. This initial algorithm has been chosen because other criteria proposed in [6] use the Max-SNR solution as a starting point.

Furthermore, the choice of the Max-SNR algorithm for implementation simplifies the architecture of this block without significant deterioration of the performance of the whole system. The proposed algorithm reduces to the maximization of the energy of the equivalent SISO channel or, in other words, to the maximization of the received SNR:

$$\arg\max_{\boldsymbol{w}_T,\boldsymbol{w}_R} = \sum_{k=1}^{N_c} \left| \boldsymbol{w}_R^H \boldsymbol{H}_k \boldsymbol{w}_T \right|^2, \text{ s.t.} \qquad \left\| \boldsymbol{w}_T \right\|^2 = \left\| \boldsymbol{w}_R \right\|^2 = 1,$$

where the $n_T n_R$ matrix H_k is the MIMO channel response at the k-th subcarrier, and $\boldsymbol{w_T}$ and $\boldsymbol{w_R}$ are the beamformers. These are complex vectors containing the RF weights to be applied by the AFE.

The input signals of the MIMAX RF weights block come from the channel estimator whose outputs are the 52 subcarrier samples for each one of the 16 (considering a MIMAX link with four antennas at the transmitter and receiver sides) equivalent SISO channels. Notice also that all operations are carried out with complex numbers. Specifically, the pseudocode for implementing this algorithm can be summarized in the following steps:

- **Step A**: Create 52 column vectors $\boldsymbol{x_k}$ (dimensions 16x1) where the i-th element of $\boldsymbol{x_k}$ is the sample of the k-th subcarrier for the i-th equivalent SISO channel. Create 52 16x16 matrices $\boldsymbol{X_k} = \boldsymbol{x_k}*\boldsymbol{x_k}'$. Add the 52 matrices $\rightarrow \boldsymbol{Y} = \Sigma \boldsymbol{X_k}$
- **Step B**: Calculate the dominant eigenvector $\boldsymbol{z}$ of the matrix $\boldsymbol{Y}$ using a fixed number of iterations of a power method.
- **Step C**: Construct $\boldsymbol{Z}$ as the 4x4 matrix resized from the 16x1 vector $\boldsymbol{z}$. The Max-SNR Rx beamformer $\boldsymbol{w_R}$ is the left singular vector of $\boldsymbol{Z}$, which is obtained applying again a fixed number of iterations of a power method.

A schematic diagram of the Max-SNR implementation steps is shown in Figure 8. Step A is creation of the 52 column vectors $\boldsymbol{x_k}$ where the i-th element of $\boldsymbol{x_k}$ is the sample of the k-th subcarrier for the i-th equivalent SISO channel. The size of $\boldsymbol{x_k}$ is $n_T n_R$ (16 in this case). It also creates the 52 rank-one matrices $\boldsymbol{X_k}=\boldsymbol{x_k}\boldsymbol{x_k}^H$ of 16x16 dimension and adds these 52 matrices in a sum $\boldsymbol{Y}$. Step B calculates the $\boldsymbol{z}$ dominant eigenvector of the sum matrix. The common way to calculate this dominant eigenvector is to perform the singular value decomposition (SVD). However, the implementation of a complete SVD is not needed as it would use too many resources. The alternative solution is the power method which was finally implemented. This method is probably the simplest one for finding the largest eigenvector of a matrix. From the $\boldsymbol{z}$ vector of 16x1 dimension obtained by Step B, we construct the $\boldsymbol{Z}$ matrix of 4x4 dimension resized by columns. Step C calculates the SVD maximum eigenvector of $\boldsymbol{Z}$ in order to extract the first row of the $\boldsymbol{U}$ matrix. Again, it is not necessary to perform the complete SVD. A beamforming weight coefficient can be calculated as the dominant eigenvector of the product $\boldsymbol{Z}\boldsymbol{Z}^H$ where $\boldsymbol{Z}^H$ is the Hermitian of matrix $\boldsymbol{Z}$. Thus Step C can be split in two substeps: the first one is a matrix multiplication and the second is a 4x4 power method. The resultant vector of this last power method is the $\boldsymbol{w_R}$ beamforming weight under the Max-SNR criterion.

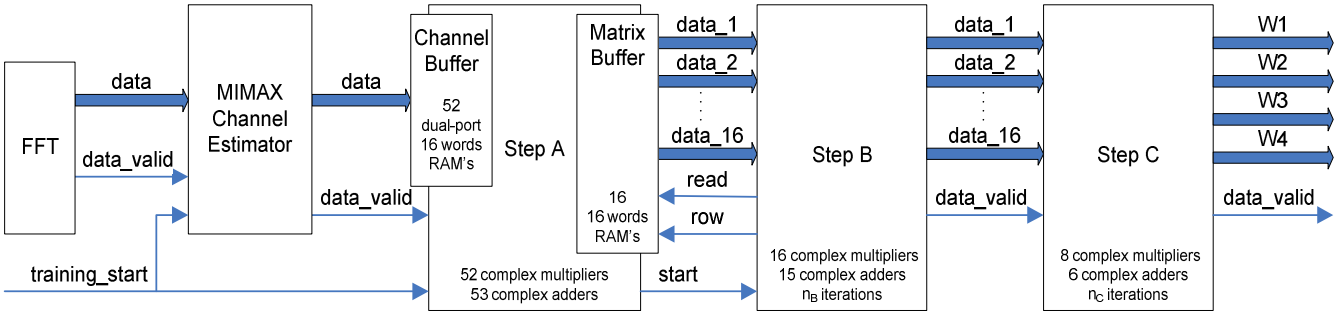

Figure 8. Max-SNR beamforming weights calculation

The first task of the weight delivery block consists of transferring the calculated optimal weights to the MAC processor after a training frame has been received. This is so-called weight updating and it is a straightforward procedure (Figure 9). The beamforming weights are provided directly by the MIMAX RF weights block (or by the weight correction block if finally needed).

The next task is to transfer the optimal or default weights from MAC to radio-frequency control unit (RFCU) during the transmission or reception of data frames. This procedure, called weight uploading, has been easily implemented by allowing a direct connection between the MAC processor and the RFCU as shown in Figure 10. Finally, the last task is to apply the predefined set of weights during transmission or reception of a training frame: this procedure is denoted as weight setting.

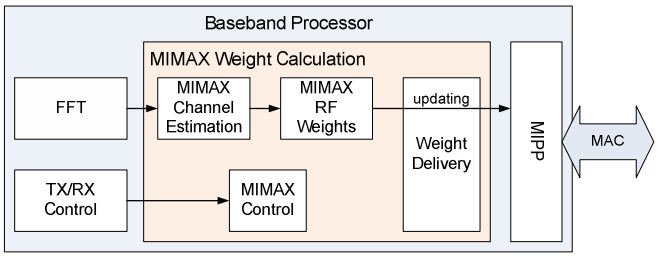

Figure 9. Illustration of the weight updating

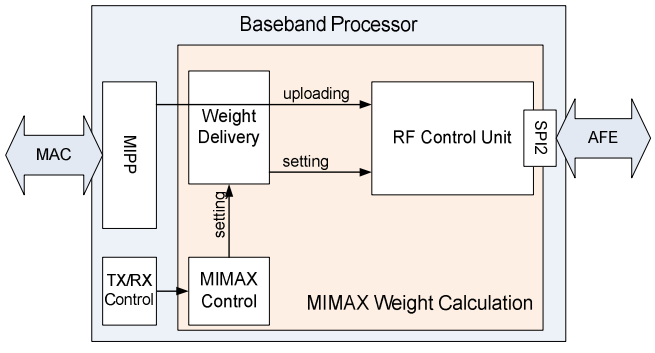

Figure 10. Illustration of the weight delivery

VII. Frequency Offset Estimation

Any residual frequency offset that occurs after the synchronizer stage of the conventional IEEE802.11a receiver distorts the weight calculations during training. Therefore, it could be necessary to estimate and compensate that residual frequency offset by transmitting two training symbols using the same pair of Tx and Rx beamformers. Under assumption that the residual frequency offset is lower than the subcarrier spacing, the maximum likelihood frequency offset estimator is given by

$$\hat{\Delta f}_{ML} = \frac{1}{2\pi\Delta t}\,\mathrm{angle}\left(\sum_{k=1}^{Nc} s_1[k]s_2^*[k]\right),$$

where N_c is the number of active subcarriers; s_1 and s_2 are the OFDM training symbols used for frequency estimation; and Δt means the time between symbols s_1 and s_2.

VIII. Implementation

In this section, the implementation process of the MAC and baseband processors is briefly described.

The MAC hardware accelerator has been designed and thoroughly simulated in VHDL. Afterwards, the VHDL model has been implemented on a Virtex5 LX50 FPGA using the Xilinx ISE tool. It is attached to an ASIC that contains the MIPS processor. This FPGA/ASIC solution allows for easy debugging and bug fixing under real-time conditions. The ASIC silicon chip of 50 mm^2 is fabricated in IHP's 0.25 µm CMOS technology [16]. A standalone MAC module in a CardBus form factor with the PCMCIA interface to the host computer and the MIPP interface to PHY is shown in Figure 11. It consumes the power of 1 W at the operating frequency of 80 MHz.

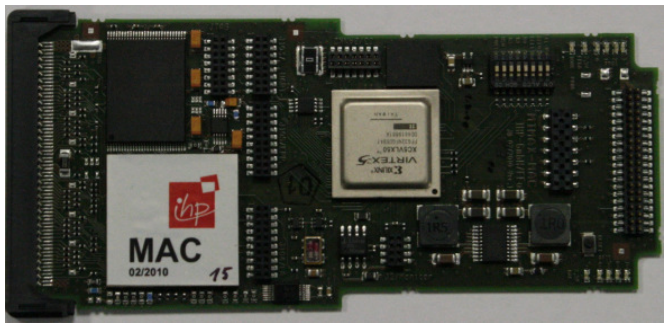

Figure 11. MAC hardware platform

For design and implementation of the baseband processor, we have used the Xilinx System Generator tool. This tool is a plug-in to the Matlab's Simulink that enables designers to

develop high-performance DSP systems to be implemented in FPGA technology. It can automatically translate designs into FPGA implementations that are faithful, synthesizable and efficient.

The chosen FPGA is a Virtex5 LX330 which has 34560 slices. Regarding the RF weights calculation block, some decisions have been taken to reach a good compromise between FPGA utilization and system performance: We used 5 iterations for each power method and 8 bits interfaces between the blocks shown in Figure 8. The conventional IEEE802.11a baseband processor occupies around 45 %, whereas the new MIMAX baseband modules occupy 33 % of the available slices. The operating clock frequency of the processor is 80 MHz.

The baseband modules are integrated in a dedicated baseband board featuring communication with the MAC processor and the analogue front-end. The baseband board incorporates, except a Virtex5 LX330 FPGA, all required interfaces, digital-to-analogue and analogue-to-digital converters for baseband signals, program flash, power and clock circuitries, and connectors. The photograph of the produced baseband board is shown in Figure 12.

Figure 12. Baseband hardware platform

IX. Test Setups

For testing the PHY and MAC components individually, we have developed two test setups. The first one is intended for PHY testing without MAC (MAC emulator). This will simplify many test operations like parameter settings since it is not required to "route" them through the complex MAC firmware. The setup consists of a data converter unit (MIPPToUSB in Figure 13) described in VHDL, some small USB hardware to directly connect the baseband board to the USB port of PC (bypassing MAC) and a terminal program on PC to send/receive commands directly to/from the baseband board.

The terminal program has several functionalities that are based on receiving and sending 32-bit words. The format of the words being sent corresponds to the one defined for the MIPPToUSB interface. When starting the program, a menu appears containing the list of all available options. By choosing the adequate command, it is possible to set and read any PHY parameter. In addition, there is a possibility to send a single beacon or training frame or to send frames periodically. Frame parameters, such as the length, data rate etc. can be selected. Received frames will be displayed and CRC checked. The program is written in C and supposed to be easily extendable for new features or adaptable to debugging problems.

The second test setup (Link Emulator) allows verifying the functionality and evaluating the performance of the MAC implementation including host drivers with an emulated PHY link. The setup provides communication between up to four MAC stations on two independent channels. The interface to the MAC board is generally the MIPP interface described above but, optionally, the MIPPToUSB component could be attached providing direct access to PC. The design has been implemented on a Virtex 1000E FPGA.

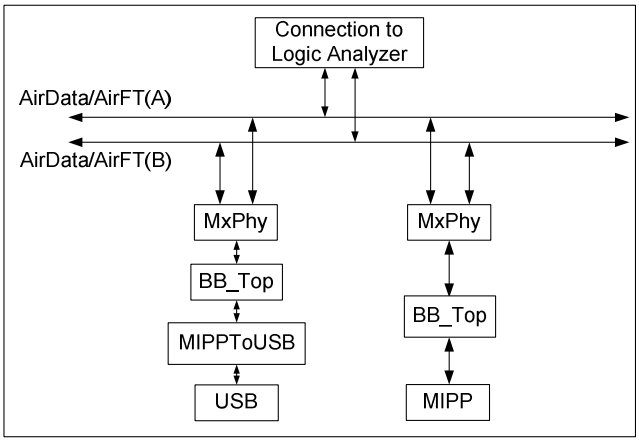

Figure 13. Block diagram of the PHY link emulator

The block diagram in Figure 13 shows the structure of the MIPP and USB parts of the Link Emulator. Additional connectors allow to monitor the frames transferred on both channels (AirData and AirFT signals) and some interface signals, e.g. for the USB port, on a logic analyzer for debug purposes.

The MIPP station in the Link Emulator consists of two main components. The first one is BB_Top which represents the external interface of the baseband processor. It is connected to the MxPhy component, which is responsible for receiving and sending data to the air link. It replaces the MIMAX baseband processor.

The USB station is the extension of a MIPP station with one extra component: MIPPToUSB. Besides that, there are no other changes in comparison to MIPP. Once the data frame is sent from one of the stations, the other stations recognize the incoming frame and receive it. Of course, it is possible to send frames from any of the stations, and it can be received by some or all stations. It is important to say that it is also possible to perform all relevant control and configuration commands for every station.

The baseband board was used for the real-time tests of the MIMAX baseband processor in several setups. First, we have verified the correct reading, changing, and re-reading of a few

configuration parameters. Then, using the USB terminal program a few beacon, data and training frames were transmitted and the generated I/Q signals at the DAC were analyzed to verify a correct transmission. Afterwards, some data frames were generated in Matlab and downloaded to the vector signal generator. The signals generated with the E4438C RF generator were used as I/Q inputs of the MIMAX baseband board and the correctness of the data was verified by the USB terminal program.

The most important test aimed at checking the correct real-time behavior of the developed MIMAX modules. For this test we generated training frames for a 4x4 MIMO system where each of the 16 training symbols was affected by a different SISO channel. These training frames were generated in Matlab and distorted by known MIMO channels. The training sequence was transmitted with the vector signal generator and the optimal weights calculated by the processor were provided to the USB terminal program. The beamforming weights obtained in simulation and those provided by the baseband board are compared in Figure 14. This test was repeated for different channel conditions: in all examples, a very good agreement between the weights obtained in simulation and those provided by the baseband board was observed.

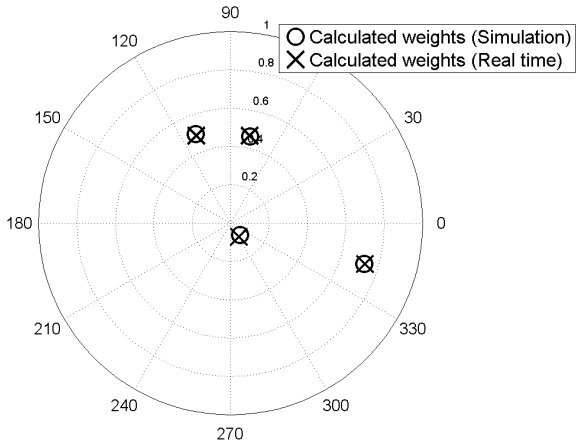

Figure 14. RF weights calculated in simulation and in real time

X. Conclusion

In this paper, we have described the architecture, design, implementation, and test of the new MAC and baseband processors of the RF-MIMO WLAN. These processors fulfill all the requirements of the new analogue front-end that exploits the available spatial diversity of the IEEE802.11a communication scheme.

Acknowledgement

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7 (2007 - 2013) under the grant agreement no. 213952 also referred as MIMAX.

References

[1] H. Boelcskei, D. Gesbert, C. B. Papadias, and A.-J. van der Veen, *Space-Time Wireless Systems: From Array Processing to MIMO Communications*, Cambridge University Press, Cambridge 2006.

[2] IEEE Standard for Information technology - Local and metropolitan area networks - Specific requirements: Wireless LAN MAC and PHY Specifications, IEEE Std 802.11, *IEEE Computer Society*, 2007.

[3] MIMAX: Advanced MIMO systems for maximum reliability and performance, http://www.ict-mimax.eu, 2008.

[4] Z. Stamenkovic, K. Tittelbach-Helmrich, M. Krstic, J. Perez, J. Via, and J. Ibanez, "Architecture of an Analog Combining MIMO System Compliant to IEEE802.11a," *Proc. ICT-MobileSummit 2009*, Santander (Spain) 2009, (pp. 1-8)

[5] F. Ellinger and W. Baechtold, "Adaptive Antenna Receiver Module for WLAN at C-Band with Low Power Consumption," *IEEE Microwave and Wireless Components Letters*, vol. 12, pp. 348-350, 2002.

[6] J. Via, I. Santamaria, V. Elvira and R. Eickhoff, "A General Criterion for Analog Tx-Rx Beamforming under OFDM Transmissions," *IEEE Trans. on Signal Processing,* vol. 58, pp. 2155-2167, 2010.

[7] R. Eickhoff, R. Kraemer, I. Santamaria, and L. Gonzalez, "Developing Energy-Efficient MIMO Radios," *IEEE Vehicular Technology Magazine*, vol. 4, pp. 34-41, 2009.

[8] A. Jahanian, F. Tzeng, and P. Heydari, "Code Modulated Path Sharing Multi-Antenna Receivers: Theory and Design," *IEEE Trans. Wireless Communications*, vol. 8, pp. 2193-2201, 2009.

[9] D.J. Dechene, K.A. Meerja, A. Shami, and S. Primak, "A Novel MIMO-Aware Distributed Media Access Control Scheme for IEEE 802.11 Wireless Local Area Networks," *Proc. 32nd IEEE Conference on Local Computer Networks*, Dublin (Ireland) 2007, (pp. 125-132)

[10] J. Mirkovic, G. Orfanos, H.-J. Reumerman, and D. Denteneer, "A MAC Protocol for MIMO Based IEEE 802.11 Wireless Local Area Networks," *Proc. IEEE Wireless Communications and Networking Conference*, Hong Kong (China) 2007, (pp. 2131-2136)

[11] A. Jahanian, F. Tzeng, and P. Heydari, "Code-Modulated Path-Sharing Multi-Antenna Receivers: Theory and Analysis," *IEEE Trans. Wireless Communications*, vol. 8, pp. 2193-2201, 2009.

[12] L. Yuxia and V.W.S. Wong, "Cross-Layer Design of MIMO-Enabled WLANs with Network Utility Maximization," *IEEE Trans. Vehicular Technology*, vol. 58, pp. 2443-2456, 2009.

[13] A. M. Ashtaiwi, *MIMO-Aware Medium Access Control in IEEE 802.11 Networks*, Queen's University, Ph.D. Thesis, Kingston 2009.

[14] H.-P. Loeb and C. Sauer, "A Modular Reference Application for IEEE 802.11n Wireless LAN MACs," *Proc. IEEE International Conference on Communications*, Dresden (Germany) 2009, (pp. 1-5)

[15] Z. Stamenkovic, E. Miletic, M. Obrknezev, and K. Tittelbach-Helmrich, "MAC Protocol Implementation in RF-MIMO WLAN," *Proc. 16th IEEE International Conference on Electronics, Circuits, and Systems*, Yasmine Hammamet (Tunisia) 2009, (pp. 303-306)

[16] Innovations for High Performance microelectronics, http://www.ihp-microelectronics.com