MIMO OFDM Transceiver for a Many-Core Computing Fabric - A Nucleus based Implementation

T. Kempf, D. Guenther, A. Ishaque, G. Ascheid
Institute for Communication Technologies and Embedded Systems, RWTH Aachen University, Germany
Torsten.Kempf@ice.rwth-aachen.de

Abstract - In this paper we analyze the potential as well as the limitations of Many-Core Computing Fabrics (MCCFs) when implementing Software Defined Radio (SDR) applications. These MCCFs consist of multi-core computing clusters that include heterogeneous processor cores and can be equipped with application specific accelerators. We focus on the computationally intensive baseband processing of modern wireless communication standards. Investigations are carried out for an implementation of a MIMO OFDM transceiver for which the IEEE 802.11n standard serves as reference regarding frame structure and timing requirements. To cope with the computational complexity and tight real-time constraints while maintaining easy porting of the investigated transceiver, it has been developed according to the Nucleus concept [1]. Following this, a thorough analysis of the application is conducted to determine the critical algorithmic kernels (*Nuclei*) contained within the transceiver. Efficient implementations (*Flavors*) of these Nuclei for the MCCF have been developed. Resultant algorithmic performance (e.g. frame-error-rate) as well as the system performance (e.g. latency and throughput) are discussed.

I. Introduction

The increasing demand for high data rate access is commonly answered by exploiting MIMO OFDM techniques in modern wireless communication standards [2][3]. The use of multiple input and output (MIMO) antennas increases the achievable data rate, while orthogonal frequency division multiplexing (OFDM) is utilized to transmit data in parallel frequency bins. As a consequence, the transceiver complexity increases significantly compared to simple single-input single-output transceivers.

Previous investigations of the computational intensive baseband processing [4] [5] have identified several key features for efficient SDR platforms. The overall computational demands are in the area of several hundred GOPS, while the power budget is limited to a few hundred mW when targeting consumer markets [6]. To cope with these requirements multiple processing elements are mandatory. In addition, each processing element has to be optimized for its intended task to increase energy efficiency. Accordingly, potential SDR architectures have to be Multiprocessor System-on-Chip (MPSoC) architectures including specialized processing elements with e.g. SIMD (single-instruction multiple-data) operations.

This work has been supported by the UMIC Research Centre, RWTH Aachen University and by the EC under grant 2PARMA FP7-248716.

Processor cores supporting SIMD operations are the fundamental building blocks of SDR platforms in academia (IMEC Cobra [7] and the SODA [8]) as well as in commercial products like the TMS320TCI6618 [9] SoC by Texas Instruments or Freescale MSC8156 DSP [10]. Besides this key component, most platforms include dedicated HW accelerators for improved performance, e.g. executing the channel decoding. In TI platforms, a specific coprocessor performs this task, while IMEC's Cobra platform contains a FlexFEC processor template for LDPC and turbo-decoding.

In contrast, to the above mentioned architectures that are optimized for baseband processing, tiled many-core architectures are available in the general purpose computing domain. Examples are Intel's Tera-scale Computing Research Program [11] and the Shapes [12] architecture. Their regular structure ease software and hardware development. Looking solely on the provided GOPS, these architectures might be able to execute baseband processing applications as well. However, this number is typically misleading since real-time requirements cannot be achieved due to overheads when accessing the memories and synchronizing the different processor cores. Furthermore, energy consumption is far too high for mobile handsets. This makes these architectures typically not suitable for SDR applications.

Recently, a new family of platforms called Many-Core Computing Fabrics (MCCFs) [13] have been announced. These are an interesting option to cope with the flexibility and performance requirements of future SDRs. They combine the regular structure of tiled architectures as well as the heterogeneous characteristic of current SDR platforms.

MCCFs consist of several multi-core computing clusters that are connected via a regular Network-on-Chip infrastructure. To obtain high performance and low energy consumption while keeping the architecture modular and cost effective, application specific optimizations can be applied on the cluster and processor core level. Processor cores can be customized

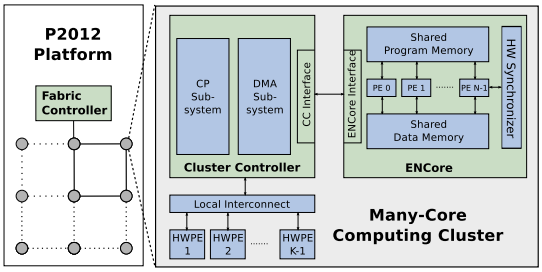


Fig. 1. P2012 Platform (based on [2])

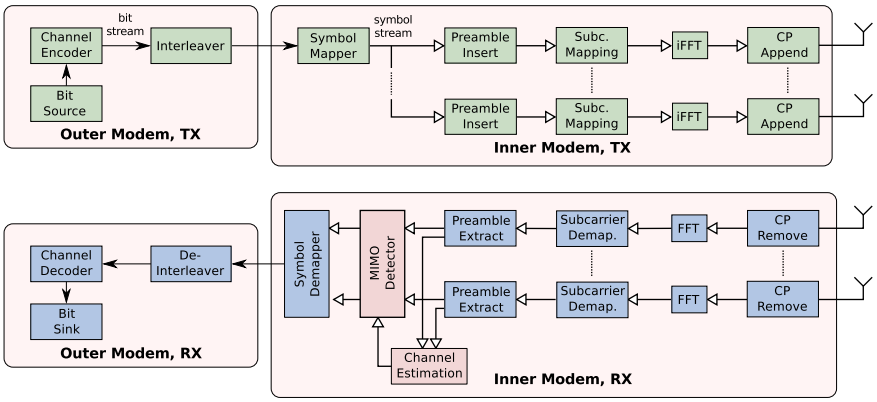


Fig. 2. MIMO OFDM Transceiver Overview

with extensions, e.g. vector or bit operations. Therefore, each cluster can include multiple heterogeneous processor cores. In addition, hardwired accelerators can be added to enhance the processing capabilities for a targeted application class.

Figure 1 depicts the P2012 platform of ST Microelectronics [13] that is currently under development. The platform consists of a scalable number of multi-core computing clusters each comprising an ENcore cluster. A single ENcore cluster can be customized to contain up to 16 STxP70-V4 processor cores whereas each core can be extended with application specific instructions, e.g. supporting efficient vector computations or bit manipulations. Additional HW accelerators can be connected via a local interconnect network to support particular functions. For external data communication each cluster contains a DMA subsystem and a streaming interface. Additionally, a HW synchronizer is attached to support fast synchronization of the processor cores.

In the following of this paper, we will discuss the implementation of the inner modem of an MIMO OFDM transceiver according to the Nucleus methodology onto platform P2012. First, state-of-the-art transceiver algorithms have been inspected and several *Nuclei* have been identified. In the next step, efficient implementations (*Flavors*) of these Nuclei have been implemented on the target platform. They account for more than 90% of the execution time of the complete application. For later tool usage these Flavors have been precisely characterized with respect to execution time and algorithmic performance. An application-to-architecture mapping has been computed, that achieves real-time performance when the system operates at a coded data rate of 192 Mbit/sec. This mapping has been carried out manually and the implementation has been verified using the Transaction Level Model (TLM) of the platform.

Before going into the detail of the implementation challenges, the inspected MIMO OFDM transceiver and the targeted hardware platform are briefly highlighted.

II. MIMO OFDM Transceiver and HW Platform

Mostly all modern communication standards, e.g. LTE, WLAN 802.11n and WIMAX, incorporate MIMO OFDM transmission techniques for high data rate access. In this paper only a brief overview will be given, whereas a broad overview of the theory behind OFDM systems can be found in [14].

Figure 2 illustrates the investigated physical layer application for a single communication link with baseband MIMO-OFDM configuration. An open-loop MIMO OFDM system with configurable transmit (N_t) and receive (N_r) antennas ($N_t = N_r \in \{2, 4\}$) is considered. In the following, the main components of the transceiver are described.

The incoming input bits from the Medium Access Control (MAC) layer are encoded and interleaved to cope with channel impairments and to protect the transmission against burst errors. Afterwards, these coded bits are mapped onto complex symbols and distributed among the different spatial streams. This separates the input stream into multiple parallel data streams that are sent through the different transmit antennas. Each of the parallel transmit paths consists of a complete OFDM transmitter. First the symbols are mapped to their subcarriers and the pilots as well as the frame preamble are added. The OFDM modulator derives a time signal by applying the inverse discrete Fourier transform (iDFT/iFFT) and the cyclic prefix is appended to guard the OFDM symbol from inter-symbol interference. Finally, the transmitter windows the signal and hands it to the analogue domain by D/A conversion. The structure and timing requirements of the complete physical layer frame are depicted in Fig. 3.

The receiver (lower part of Fig. 2) largely performs the functionality of the transmitter in reverse. The received signal, that has undergone channel effects, is represented in digital state after A/D conversion. After the elimination of inter-symbol-interference and synchronization using the preamble sequence, the channel estimator uses this preamble sequence to extract the multi-path fading effects of the channel characteristic. After de-multiplexing the data subcarriers, MIMO detection and symbol demapping are performed over all receive paths to recover the combined binary stream. After de-interleaving and channel decoding the final output bit stream is delivered to the MAC-layer.

Within the $4\mu s$ of an OFDM slot, N_t-data streams need to be processed in parallel (Fig. 3). This requires processing the complete frame within $(N_P + N_D) \cdot 4\mu s$, with N_D the number of OFDM data symbols per data stream and N_P the number of OFDM preamble symbols per data stream. This leaves only a few thousand cycles for executing a given functionality within the transceiver chain. Obviously, this

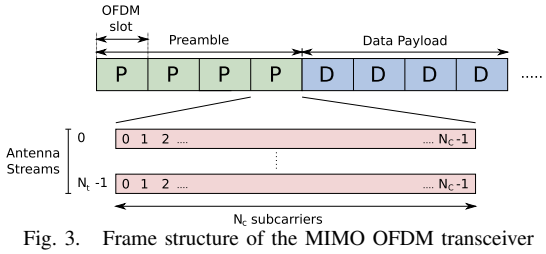


Fig. 3. Frame structure of the MIMO OFDM transceiver

makes it extremely challenging to implement even a single wireless communication standard onto such SDR platform.

In the following of this paper, the potential but also limitations of many-core computing fabrics, in particular the P2012 platform of ST Microelectronics [13], shall be investigated when targeting SDR applications. The platform, illustrated in Figure 1, contains a scalable number of computing clusters connected via a regular interconnect architecture. Each computing cluster comprises a controller subsystem, a DMA subsystem and a scalable number of hardware accelerators as well as up to 16 STxP70-V4 processor cores (ENCore cluster). Each STxP70-V4 processor core is a 7-stage, dual-instruction issue RISC processor that can be extended with application specific instructions, e.g. supporting efficient bit manipulations or vector computations. Especially, the latter vector unit is of vital importance for implementing the given SDR application. The HW accelerators can be connected via a local interconnect network to support critical functions of the application. However, the configuration used within this paper contains no HW accelerator.

All processor cores within one cluster are connected to a shared data memory via two scalable memory ports. Thanks to efficient memory banks, each processor core can access one data word and fetch one instruction in every cycle. In order to access memory outside the cluster, a DMA or a streaming interface exist. Finally, each cluster contains a HW synchronizer that supports together with the memory architecture low-latency synchronization and high-throughput data exchange between the various processor cores.

Within this paper, the P2012 platform serves as a reference architecture for the mapping of an SDR application onto a MCCF. It should be directly mentioned that due to the high latency for data communication between the different clusters, the target is to implement the transceiver on a single cluster to achieve real-time constraints. Additionally, the P2012 platform does not include a radio frontend in general, therefore it is assumed that a radio frontend is connected via a HW accelerator to the cluster.

Together with the hardware platform, a set of high level programming models and tools is available [13]. The native programming layer provides a low-level C-based API to make use of platform specific features, e.g. the hardware synchronizer. Standard-based programming models support OpenCL and OpenMP. Other more advanced programming models support multi-threading and data-flow models. Unfortunately, the tight latency and throughput constraints of the application prevent us from using these convenient programming models. Therefore, only low-level features from the native programming layer will be utilized throughout the implementation.

Before going into details of our case study, the related work will be sketched.

III. Related Work

Today, SDR platforms can be found in many wireless communication devices ranging from wireless handsets to infrastructure systems like basestations. The various benefits of SDR platforms have led to a fast adoption of the technology. For example, low volume markets, like satellite communication or systems for military communication, can benefit from the reduced development costs due to a common hardware platform that supports different wireless communication standards. For basestations the demand for multi-standard support and reduced maintenance costs have led to the employment of common SDR platforms. Examples are the System-on-Chip platforms of Freescale (MSC8156 DSP [10]) and Texas Instruments (TMS320TCI6618 [9]) that are intended for the use in femtocells. Both support a wide range of wireless communication standards like WCDMA/HSPA/HSPA+, TD-SCDMA, GSM, LTE and WiMAX.

The platforms, in general, incorporate the concept of heterogeneous Multiprocessor System-on-Chips (MPSoC). In this context, heterogeneous means that the processor elements, in particular the processor cores, are different. The TMS320TCI6618 platform, as example, contains four TMS320C66x DSPs and a bunch of coprocessors for various tasks of baseband processing such as channel decoding (Viterbi- and Turbo-decoding) as well as for Fast Fourier Transformation.

For efficient usage in the domain of baseband processing, the individual processor cores are equipped with SIMD (single-instruction multiple-data) operations that are commonly available in vector processors such as the Tensilica ConnX [15] and EVP [16], but also within VLIW architectures like TI's C64x+ and C66x DSP [9]. Other processor cores like Tensilica's Xtensa [15] and the STxP70 [17] can be easily extended with additional vector units.

Compared to architectures used in the infrastructure, hardware platforms targeting mobile handsets focus on low energy consumption due to the limited capacity of the battery. Examples of such are the commercially available MUSIC platform [5] as well as IMEC's Cobra platform [7] and the SODA [8] architecture. However, the fundamental architecture still remains a heterogeneous MPSoC with efficiently supported SIMD operations. When implementing baseband applications, these achieve superior performance compared to general purpose computing architectures, like Intel's Terascale Computing Research Program [11].

Yet they have a common drawback. Software development on a homogeneous MPSoC architecture is difficult, but is even more complex on a heterogeneous one. For example, when porting the application from one platform to the other, functions might be mapped to different processor cores. In a homogeneous environment this requires basically no effort, whereas in a heterogeneous architecture the software needs to be ported. Assuming a RISC type of processor, the effort might be limited. But for vector or VLIW processor cores the effort might be huge, since software developers typically

make extensive use of compiler-known-functions or inline assembly. For reduced time-to-market, it is of practical interest to minimize this effort by hardware and/or software design solutions.

The P2012 platform [13] provides an effective trade-off between the highly optimized application-specific SDR platforms and a general tiled many-core architecture. Each cluster provides enough computational performance to cope with the complex baseband processing of modern communication standards and can be equipped with special hardware accelerators if necessary.

Despite the extensive programming models [13] that are available for platform P2012, efficient usage of the vector processing unit requires low-level software development. Targeting the physical layer application, data exchange and synchronization has to make use of low-level mutexes and semaphores (less than 100 cycles). Therefore, programming models operating on a high-level with convenient data structures [18][19] *cannot* be used due to the tight real-time requirements. Accordingly, software implementations tend to be hardware specific which typically impedes porting the application to other SDR platforms.

To cope with this issue, the Nucleus methodology [1] and tooling [20] supports an effective design approach that guarantees simple porting while maintaining efficiency. While the Nucleus methodology aims at the baseband processing, other available tools focus mostly on the deployment and configuration aspect of SDR platforms [21][22][23]. Other design tools are developed and hence limited to a specific SDR development platform like Lyrtech [24] or Coherent Logix [25]. Following the Nucleus concept, the transceiver is analyzed next.

IV. Nuclei Identification

First, the overall **system model**[1] shall be introduced. The transmission over the channel is modeled by multiplying the transmitted symbol vector $\mathbf{x}$ of dimension N_t with the channel matrix $\mathbf{H}$ of dimension $N_r \times N_t$ and a complex circularly symmetric Gaussian noise $\mathbf{n}$ of dimension N_r is added. The obtained result is the received symbol vector $\mathbf{y}$.

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{1}$$

A. Channel Estimation

One of the major tasks within a receiver is to revert the impact of the channel. In order to do so, the channel characteristics need to be estimated, which is carried out by making use of the known preamble. This so-called channel estimation is performed per subcarrier. Within the investigated transceiver, the preamble structure of the WLAN 802.11n standard was chosen. Since a minimum of N_t preamble symbol vectors p_i of dimension $N_t \times 1$ are required, the preamble is described as a matrix $\mathbf{P} = [\ p_0\ ..\ p_{N_t-1}\ ]$ and the corresponding received symbol vectors s_i of dimension $N_r \times 1$ are combined into a matrix $\mathbf{S} = [\ s_0\ ..\ s_{N_t-1}\ ]$. Similar the interfering noise can be written as a noise matrix $\mathbf{N} = [n_0\ ..\ n_{N_t-1}]$, so that

$$\mathbf{S} = \mathbf{H}\mathbf{P} + \mathbf{N} \tag{2}$$

The task of the channel estimation is to provide the later MIMO detection an estimate $\hat{\mathbf{H}}$ of the channel matrix $\mathbf{H}$ per subcarrier.

A.1 Least Squares (LS) method

This rather simple method neglects the noise summand of equation (2) so that

$$\hat{\mathbf{H}}_{\mathbf{ls}} = \mathbf{S}\mathbf{P}^{\mathbf{H}} \left(\mathbf{P}\mathbf{P}^{\mathbf{H}}\right)^{-1} \tag{3}$$

Matrix $\mathbf{P}$ is selected as an orthogonal matrix for most of today's standards including IEEE 802.11n. Due to this property, channel estimation can be reduced to:

$$\hat{\mathbf{H}}_{\mathbf{ls}} = \frac{1}{N_t} \cdot \mathbf{S}\mathbf{P}^{\mathbf{H}} \text{ with } \mathbf{P}\mathbf{P}^{\mathbf{H}} = \mathbf{P}^{\mathbf{H}}\mathbf{P} = N_t \mathbf{I}_{\mathbf{N_t}} \tag{4}$$

The matrix $\mathbf{I}_{\mathbf{N_t}}$ denotes an identity matrix of dimension N_t.

A.2 Minimum Mean Square Error (MMSE) method

The MMSE estimation method derives the channel matrix estimate $\hat{\mathbf{H}}_{\mathbf{mmse}}$ by minimizing the mean square error between the channel matrix $\mathbf{H}$ and the estimate $\hat{\mathbf{H}}_{\mathbf{mmse}}$. Under the assumption of additive white gaussian noise with variance σ_n^2 and E_s being the total transmit power per antenna, the result is

$$\hat{\mathbf{H}}_{\mathbf{mmse}} = \mathbf{S}\left(\mathbf{P}^{\mathbf{H}}\mathbf{P} + \frac{\sigma_n^2}{E_s}\mathbf{I}_{\mathbf{N_t}}\right)^{-1} \mathbf{P}^{\mathbf{H}} \tag{5}$$

.

Furthermore, the knowledge of the orthogonal preamble matrix $\mathbf{P}$ can be utilized to reduce the computational effort by simplifying the expression to

$$\hat{\mathbf{H}}_{\mathbf{mmse}} = \frac{1}{N_t + \frac{\sigma_n^2}{E_s}} \cdot \mathbf{S}\mathbf{P}^{\mathbf{H}} \tag{6}$$

B. MIMO Detection

Having an estimated version of the channel matrix, the MIMO detector has to revert the impact of the channel on the received data-payload to compute an estimate of the transmitted payload.

Linear detectors derive an equalizer matrix $\mathbf{G}$ of dimension $N_t \times N_r$ to mitigate the impact of the channel, according to

$$\hat{\mathbf{x}} = \mathbf{G}\mathbf{y} \text{ with } \mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{7}$$

[1]Independent fading among subcarriers is assumed, so that each subcarrier sees a flat channel and channel estimation and MIMO detection can be performed independently for each subcarrier.

B.1 Zero Forcing

The zero forcing detector neglects the noise term in eq. (4). Therefore, the estimated payload $\hat{\mathbf{x}}$ can be reconstructed as

$$\hat{\mathbf{x}} = \left(\hat{\mathbf{H}}^{\mathbf{H}}\hat{\mathbf{H}}\right)^{-1}\hat{\mathbf{H}}^{\mathbf{H}}\mathbf{y} \tag{8}$$

so the equalizer matrix $\mathbf{G}$ is the pseudo inverse of $\hat{\mathbf{H}}$.

$$\mathbf{G_{zf}} = \left(\hat{\mathbf{H}}^{\mathbf{H}}\hat{\mathbf{H}}\right)^{-1}\hat{\mathbf{H}}^{\mathbf{H}} = \mathbf{H}^{\dagger} \tag{9}$$

In the case of equal transmit and receive antennas ($N_t = N_r$) the matrix $\hat{\mathbf{H}}$ is a square matrix and the pseudo inverse can be replaced by the normal inverse $\mathbf{G_{zf}} = \hat{\mathbf{H}}^{-\mathbf{1}}$. In turn this strongly reduces the computational complexity.

B.2 Zero Forcing using QR decomposition

The above described zero forcing algorithm requires a matrix inversion, which suffers from mediocre numerical stability. To cope with this issue the zero forcing detector can be extended to use QR decomposition (ZF-QRD). $\mathbf{Q}$ is an orthogonal matrix of dimension $N_r \times N_t$ with $\mathbf{QQ^H} = \mathbf{I}$, while $\mathbf{R}$ is an upper triangular matrix of dimension $N_t \times N_t$. Accordingly, the detection is given by

$$\mathbf{y} = \mathbf{Hx} = \mathbf{QRx} \tag{10}$$

$$\tilde{\mathbf{y}} = \mathbf{Q^H y} = \mathbf{Q^H QRx} = \mathbf{Rx} \tag{11}$$

$$\hat{x}_i = \frac{\tilde{y}_i - \sum_{j=i+1}^{N_t} r_{ij}\hat{x}_j}{r_{ii}} \tag{12}$$

The triangular structure of $\mathbf{R}$ allows to solve the equation (11) in successive steps by back substitution [26] (eq. (12)). Please note that the enhanced numerical stability compared to a detection based on zero forcing with direct matrix inversion comes at the cost of increased computational complexity due to QR decomposition and back substitution.

B.3 Minimum Mean Square Error (MMSE)

The MMSE detector minimizes the expected mean square error between $\hat{\mathbf{x}}$ and the original payload data $\mathbf{x}$.

$$\mathbf{G} = \underset{\mathbf{G}}{\operatorname{argmin}}\, \mathrm{E}\left\{|\mathbf{x} - \hat{\mathbf{x}}|^2\right\} \tag{13}$$

Under the assumption of additive, uncorrelated white noise with variance σ_n^2 and mean transmit power E_s of one antenna, the following result is obtained

$$\mathbf{G_{mmse}} = \left(\hat{\mathbf{H}}^{\mathbf{H}}\hat{\mathbf{H}} + \frac{\sigma_n^2}{E_s}\mathbf{I_{N_t}}\right)^{-1}\hat{\mathbf{H}}^{\mathbf{H}} \tag{14}$$

B.4 MMSE using QR decomposition

Again the matrix inversion in eq. (14) can cause numerical instability. Similar to the zero forcing case the detector can be based on a regularized QR decomposition (MMSE-QRD) [26].

$$\tilde{\mathbf{H}} = \begin{pmatrix} \hat{\mathbf{H}} \\ \frac{\sigma_n}{\sqrt{E_s}}\mathbf{I_{N_t}} \end{pmatrix} = \begin{pmatrix} \mathbf{Q_a} \\ \mathbf{Q_b} \end{pmatrix}\mathbf{R} \tag{15}$$

MIMO detection can then be solved by using $\mathbf{Q_a}$ instead of $\mathbf{Q}$ with back substitution (see eq. (12)).

$$\mathbf{Q_a^H y} = \mathbf{R}\hat{\mathbf{x}} \tag{16}$$

B.5 MMSE-QRD with Dynamic Scaling

Using a QR decomposition based on the Gram-Schmidt method, the columns of the input matrix are projected onto each other and linear dependencies are eliminated by subtracting the results of the projections from the vectors in question. This subtraction becomes critical, once the result of the subtraction is out of the range where fixed-point arithmetic operates with the required precision. In [27], authors discuss how this problem can be relaxed by applying dynamic scaling during QR decomposition. The scaling factors used are either 2 or $\frac{1}{2}$ which can be efficiently implemented as shift operation. While the resulting $\mathbf{Q}$ matrix remains the same as this factor cancels out due to the normalization of the vector ($\mathbf{v_i}/\|\mathbf{v_i}\|$), the scaling modifies the resulting $\mathbf{R}$ matrix. Therefore, back substitution cannot be used and the MIMO detection is reformulated to a simple equalizer.

$$\hat{\mathbf{H}} = \mathbf{Q_a R} \quad \text{with} \quad \mathbf{R^{-1}} = \frac{\sqrt{E_s}}{\sigma_n}\mathbf{Q_b} \tag{17}$$

$$\mathbf{R}\hat{\mathbf{x}} = \mathbf{Q_a^H y} \quad \Rightarrow \quad \hat{\mathbf{x}} = \frac{\sqrt{E_s}}{\sigma_n}\mathbf{Q_b Q_a^H y} \tag{18}$$

Accordingly, this results in an equalizer matrix $\mathbf{G_{ds}} = \frac{\sqrt{E_s}}{\sigma_n}\mathbf{Q_b Q_a^H}$ that solely depends on $\mathbf{Q_a}$ and $\mathbf{Q_b}$ that are preserved during the QR decomposition including dynamic scaling. Therefore, the actual detection can be reduced to a simple matrix-vector multiplication of the equalizer matrices with the received symbol vectors.

B.6 Successive Interference Cancellation (SIC)

The non-linear SIC detection follows the basic principle at using QR decomposition and back substitution [26]. SIC can be implemented in two ways, either by using a normal (SIC-Zero Forcing) or a regularized (SIC-MMSE) QR decomposition. Compared to earlier mentioned detectors, the major difference is that within the successive backward solving, the results that are further reused are quantized (sliced) to constellation symbols. Accordingly, the SIC-ZF method starts from equation (11) and the SIC-MMSE approaches the detection by using equation (15). To compute the transmitted symbol vector the back substitution method (eq. (12)) is used within both approaches. Compared to linear ZF- and MMSE-QRD methods the only difference is that the reused symbols $\hat{\mathbf{x}}_\mathbf{j}$ are

Nucleus	Mathematical Equation	Used in	Eq. ref.
Matrix-Vector Multiplication	$\mathbf{y} = \mathbf{A}\mathbf{x}$	MIMO Detection with QRD	(11), (12)
Matrix-Matrix Multiplication	$\mathbf{X} \cdot \mathbf{Y}$	Channel Estimation	(4), (6)
Matrix-Hermitian Multiplication	$\mathbf{H} \cdot \mathbf{H^H}$	MIMO Detection	(8), (14)
Scalar-Matrix Multiplication	$s \cdot \mathbf{Y}$	Channel Estimation	(4), (6)
Matrix Inversion	$\mathbf{H^{-1}}$	MIMO Detection (ZF, MMSE)	(9), (14)
Upper Triangular Matrix Inversion	$\mathbf{H^{-1}}$ with H being a upper triangular matrix	Symbol Demapper Preprocessing	(22)
QR Decomposition	$\mathbf{H} = \mathbf{QR}$	ZF-QRD, SIC-ZF	(11)
QR Decomposition (regularized)	$\tilde{\mathbf{H}} = \begin{pmatrix} \hat{\mathbf{H}} \\ \frac{\sigma_n}{\sqrt{E_s}}\mathbf{I_{N_t}} \end{pmatrix} = \begin{pmatrix} \mathbf{Q_a} \\ \mathbf{Q_b} \end{pmatrix} \mathbf{R}$	MMSE-QRD, SIC-MMSE	(15)
Back Substitution (w/o slicing)	$\hat{x}_i = \frac{\tilde{y}_i - \sum_{j=i+1}^{N_t} r_{ij}\hat{x}_j}{r_{ii}}$	Linear MIMO Detection with QRD	(12)
Back Substitution (with slicing)	$\hat{x}_i = \frac{\tilde{y}_i - \sum_{j=i+1}^{N_t} r_{ij}\,\mathrm{Q}[\hat{x}_j]}{r_{ii}}$	Non-Linear MIMO Detection with QRD	(19)
FFT/iFFT	$X(k) = \sum_{n=0}^{N-1} x(n) \cdot W_N^{kn}$, $W_N = \exp\left(\pm j\frac{2\pi}{N}\right)$	OFDM Modulation/Demodulation	-
Soft-Output Symbol Demapper	$L(b_i) \approx \rho_k \left(\min_{s \in A_i^0} \lvert\mathbf{z}_k - s\rvert^2 - \min_{s \in A_i^1} \lvert\mathbf{z}_k - s\rvert^2 \right)$ with $\rho_k = \frac{1}{\sigma_n^2 \left[\left(\mathbf{H^H H} + \sigma_n^2 \mathbf{I_{N_t}}\right)^{-1} \right]_{k,k}} - 1$	Symbol Demapper	(20)

TABLE I. Nuclei of the MIMO OFDM Transceiver (datatype always complex)

replaced by the quantized (constellation) symbol stream $\mathrm{Q}\left[\hat{\mathbf{x}}_\mathbf{j}\right]$

$$\hat{x}_i = \frac{\tilde{y}_i - \sum_{j=i+1}^{N_t} r_{ij}\,\mathrm{Q}\left[\hat{x}_j\right]}{r_{ii}} \tag{19}$$

C. OFDM Modulation and Demodulation

OFDM modulation and demodulation can be efficiently realized by using the iFFT and FFT algorithm [28]. The iFFT and FFT algorithms, differ only in twiddle factors they use, whereas the computations are equal. FFTs are well-known algorithms and a large variety of efficient implementations exist. In this work we concentrate on the radix-2 algorithm since its requirements match the underlying hardware architecture.

D. Soft-Output Symbol Demapper

Within the transmitter the input bit stream has been converted by the symbol mapper to a stream of complex symbols based on a given constellation diagram like 4QAM or 16QAM. The symbol demapper reverts this mapping process and outputs the log-likelihood ratio for each received bit. Please note that due to superior performance of the channel decoders only soft-output symbol demappers are considered. In general, the log-likelihood ratio (LLR) L is computed as

$$L(b_i) = \ln \frac{\Pr\left(b_i = 1 | \mathbf{y}, \mathbf{H}\right)}{\Pr\left(b_i = 0 | \mathbf{y}, \mathbf{H}\right)} \tag{20}$$

Unfortunately, this computation is far too complex to be implemented on a MCCF, hence the *Max-Log Approximation* is applied. Furthermore, it is assumed that the distribution $p\left(\mathbf{y}|\mathbf{H}\mathbf{x}\right)$ is gaussian, the noise is white gaussian and the constellation has been Gray-coded. The computation of the LLR calculation for *stream k* with the detected symbol vector $\mathbf{z}$ can now be written as

$$L(b_i) \approx \rho_k \left(\min_{s \in A_i^0} \lvert\mathbf{z}_k - s\rvert^2 - \min_{s \in A_i^1} \lvert\mathbf{z}_k - s\rvert^2 \right) \text{ with } \tag{21}$$

$$\rho_k = \frac{1}{\sigma_n^2 \left[\left(\mathbf{H^H H} + \sigma_n^2 \mathbf{I_{N_t}}\right)^{-1} \right]_{k,k}} - 1 \text{ with} \tag{22}$$

$$\rho_k \approx \frac{1}{\left[\mathbf{Q_b Q_b^H}\right]_{k,k}} \tag{23}$$

A_i^0 denotes a subset of all constellation symbols that contain a zero at position i of their bitwise representation and A_i^1 at the same position respectively. The real valued scalar ρ_k denotes the signal to interference and noise ratio (SINR) for stream k [29].

E. Nuclei

Inspecting the above discussed algorithms, the Nuclei listed in Table I have been identified. In addition to them, the table highlights the mathematical equations, their usage within the transceiver and the equation references. At this point we should recall that until this point only the algorithms have been investigated and *no* implementation aspects have been taken into account. Next we will discuss the efficient implementations of the Nuclei, called *Flavors*.

V. Implementation Results

Table II highlights the execution time of the different flavors implemented on the STxP70-V4 processor core [17] with enabled vector unit extension called VECx. Especially, the computational power of this vector unit is of vital importance when targeting real time performance. The characteristics have been measured with the SDK version 2011.1 [13] and the STxP70 has been configured with 1 context, 32bit external

	System	2x2		4x4	
	Operation	cycles	time (μs)	cycles	time (μs)
Matrix/Vector Operations					
1	mat-mat add	13	0.022	23	0.038
2	mat hermitian	26	0.043	90	0.150
3	mat-rscal mul	26	0.043	45	0.075
4	mat-vec mul	44	0.073	70	0.117
5	mat-mat mul	102	0.170	301	0.502
6	mat-rmat mul	74	0.123	205	0.342
7	mat-mat mul 8vm2[1]	218	0.363	503	0.838
8	mat inv	385	0.642	1,328	2.213
9	tri mat inv	43	0.072	278	0.463
10	qrd	595	0.992	1,683	2.805
11	qrd regularized	702	1.170	1,622	2.703
12	ds-qrd-regularized	889	1.482	2,264	3.773
13	back subst.	954	1.590	2,106	3.510
14	back subst. slicing	1,170	1.950	2,538	4.230
OFDM slot wise operations					
15	bpsk soft demap	329	0.548	658	1.097
16	4qam soft demap	658	1.097	1,316	2.193
17	16qam soft demap	857	1.428	1,705	2.842
18	fft	1,774	2.957	3,548	5.913
19	fft mem realign	2,052	3.420	4,084	6.838
20	ifft	2,028	3.380	4,056	6.760

TABLE II. Flavor Implementations on the P2012 Platform (complex datatype, f=600MHz, STxP70 core config. 0x73B014BC)

memory interface, 128 bit EFU interface, and 2 hardware loops (core configuration set to 0x73B014BC 0x00000C00).

The measured performances include reading and writing of data, but no inter-core synchronization overhead. Please note that all Flavors have been implemented in 16bit fixed-point format and the source code has been developed in low-level C for exploiting the capabilities of the vector unit.

After introducing the characteristics of the individual Flavors, the implementation results of the complete transceiver are discussed.

A. MIMO OFDM Transceiver

In this paper not all algorithms can be discussed in detail. Accordingly, we will limit the number of implementation candidates by algorithmic considerations.

Considering channel estimation, the LS and MMSE method have been introduced. Since the MMSE channel estimation considers the additive noise when calculating the estimate, one would expect it to outperform the LS estimation. However, our floating-point simulations have shown only minor performance improvements which do not justify the increased computational complexity. Therefore, the later investigations focus on the simpler LS channel estimation.

Previously, six MIMO detection techniques have been introduced and implemented. Since most instructions of the STxP70 processor core can either operate on 8x 16bit values or 4x 32bit values, the fixed-point format has been limited to 16bit precision to minimize the required execution times. Due to this, methods using matrix inversions suffer from

[2]The abbreviation mat-mat mul 8vm2 means that the second matrix is not square but extended to eight column vectors.

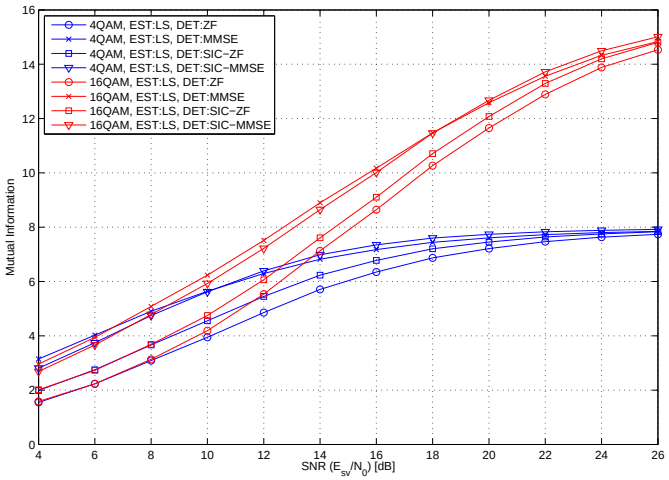


Fig. 4. Inner modem comparison - Mutual Information of 4×4 systems using floating point arithmetic

numerical instability and hence are not further discussed in this paper. Furthermore, we analyzed the maximum achievable performance of the inner modems of 4x4 transceiver systems using LS channel estimation and different MIMO detectors. To exclude the influence of channel coding, the mutual information [30] has been inspected. The mutual information $\mathrm{I}(C;\Lambda) = \mathrm{H}(C) - \mathrm{H}(C|\Lambda)$ of the codeword C and the soft output Λ of the soft demapper indicates how much information has been transmitted via the channel from C to Λ. $\mathrm{H}(C)$ is the entropy of the codeword C and $\mathrm{H}(C|\Lambda)$ is the entropy of the codeword C conditioned Λ. The mutual information is given by

$$\mathrm{I}(C;\Lambda) = 1 - \mathrm{E}\left\{\log_2\left(1 + e^{(2c-1)\lambda}\right)\right\}$$

with the bitwise elements c and λ of C and Λ respectively, whereas $\mathrm{E}\{\cdot\}$ denotes the expected value.

Figure 4 illustrates a comparison of various 4×4 systems using floating point arithmetic by using the mutual information for 4QAM and 16QAM. According to [28] the SNR region of interest is 10-15dB for 4QAM and 20-25dB for 16QAM when using a practical WLAN 802.11n transceiver. In these regions the MMSE and SIC-MMSE methods show only a minor deviation while the techniques neglecting the noise term show a decreased performance. To finally select the setup that is mapped to the P2012 platform, we have inspected the coded bit (BER) and frame error rate (FER).

We will concentrate on the achieved FER in the following, however results obtained from the BER curves have shown the same behavior. Figure 5 depicts the FER for systems using MMSE-QRD, SIC-MMSE and MMSE-DS-QRD detection scheme. As a reference the FER of a floating-point implementation using MMSE detection is added. All investigated configurations use LS channel estimation and channel decoding is performed either by a BCJR decoder ($r = \frac{1}{2}$, $g_0 = (133)_8$, $g_1 = (171)_8$) or an LDPC decoder from the IEEE80211n standard (length$=$ 1944bit, $\mathrm{rate} = \frac{1}{2}$). Within all simulations the channel has been modeled as an AWGN channel including fading. The fading is modeled by a Rayleigh characteristic [31] with a power delay profile set to 150ns [32]

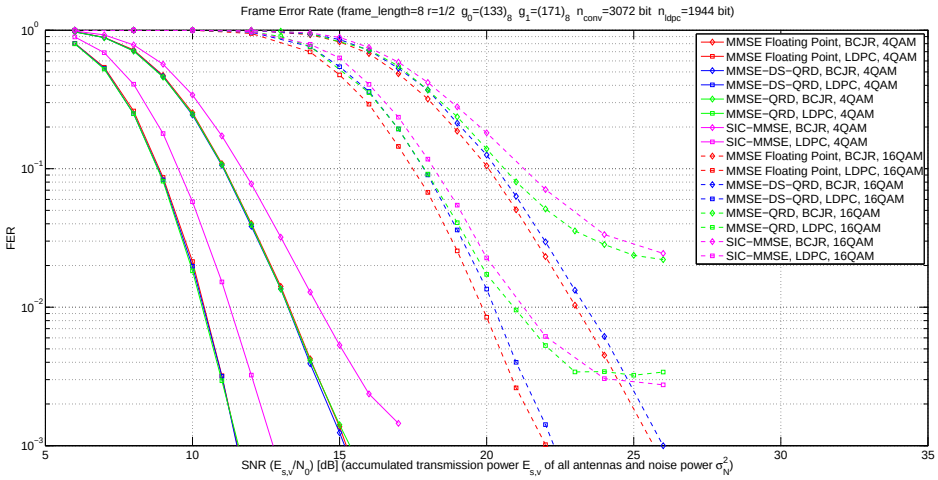


Fig. 5. FER comparison of 4x4 MIMO systems for short frames (8 OFDM data slots)

and an exponential drop of 20dB. Comparing only the fixed-point implementations for the 4QAM and 16QAM constellation, we assume that a FER of less than 1% at the receiver is acceptable (802.11n requires 10% FER). Therefore, we select the implementation that will be finally ported to the P2012 platform as the one that achieves this FER with lowest SNR requirements. Table III summarizes the measurements depicted in Fig. 5 for the best system configuration that uses a MMSE-DS-QRD detection method.

Please note that the implementations using SIC-MMSE and MMSE-QRD detection suffer from the 16bit fixed point precision. This effect is visible as a saturation in the FER and BER, e.g. as illustrated for 16QAM, but also at the SIC-MMSE curve with 4QAM constellation. In addition, the algorithmic performance of the SIC-MMSE can be enhanced when additional sorting is performed. However, the current implementation applies no sorting.

Using dynamic scaling together with MMSE-QRD detection we achieve the requirement of less than 1% FER with minimal SNR (Table III). Additionally, the floating point reference implementation shows that the performance degradation on the platform is marginal for 4QAM and for 16QAM approximately 0.4dB. Higher constellation orders have not yet been implemented but are under investigation.

Finally, to conclude the investigation of the algorithmic performance and to verify our implementation, the FER is measured for frames with variable length. Clearly, the expectation is that for longer frames a higher SNR value is required to achieve the same FER as for shorter frames. Accordingly, we measured three different use cases given by short frames having 8 data slots, medium-length frames having 24 data slots and long frames having 88 data slots. Based on the executed mode (constellation, code rate, etc.) these frames range from 1536 information bits (4QAM, $r = \frac{1}{2}$) up to 16896 information bits (16QAM, $r = \frac{1}{2}$). The results show the expected implementation behavior. It should be noted that the performance degradation of the BCJR is much more severe than using an LDPC decoder. This makes this decoder the preferred choice for implementation. Such behavior can be observed in the floating point reference as well.

	Short frame		Long frame	
	LDPC	**BCJR**	**LDPC**	**BCJR**
4QAM	10.4	13.2	11.0	14.9
16QAM	20.2	23.0	20.9	25.8

TABLE III. Minimum SNR for 1% FER for short and long frame for MMSE-DS-DRQ detection and LS channel estimation

B. Mapping of the Inner Receiver

Since the development of the receiver is much more complex than the transmitter, only the receiver will be investigated. The selected configuration to be mapped onto the P2012 platform uses LS channel estimation and a MIMO detection based on MMSE-DS-QRD. The constellation is set to 16QAM. Table IV highlights the execution time for each particular transceiver function executed on a single STxP70 processor core. In addition, the Flavors contained within each function (2nd column) is highlighted and the percentage of execution time that is spent within these Flavors is given (3rd column). The Flavors contained within the functionalities are responsible for more than 92% of the execution time.

The rather obvious data level parallelism can be used to map the application onto the P2012 platform. This parallelism is given by the number of data streams, the OFDM slots, and

Preprocessing (per OFDM frame)			
Task	**time** (μs)	**Flavors** (Tab. II)	**due to Flavors**
LS Channel Estimation	17.47	6	94%
Det. Preproc. & SINR calc.	215.31	12, 5	96%
Actual Processing (per OFDM slot)			
Task	**time** (μs)	**Flavors** (Tab. II)	**due to Flavors**
OFDM Demod. (mem. realign)	6.83	19	99%
Actual Detection	6.08	4	92%
Soft Demapping (16QAM)	2.84	17	99%

TABLE IV. Execution time measurements of inner receiver functionality with 4x4 antenna, 16QAM constellation, MMSE-DS-QRD detection and LS channel estimation (STxP70 core config. 0x73B014BC, f=600MHz)

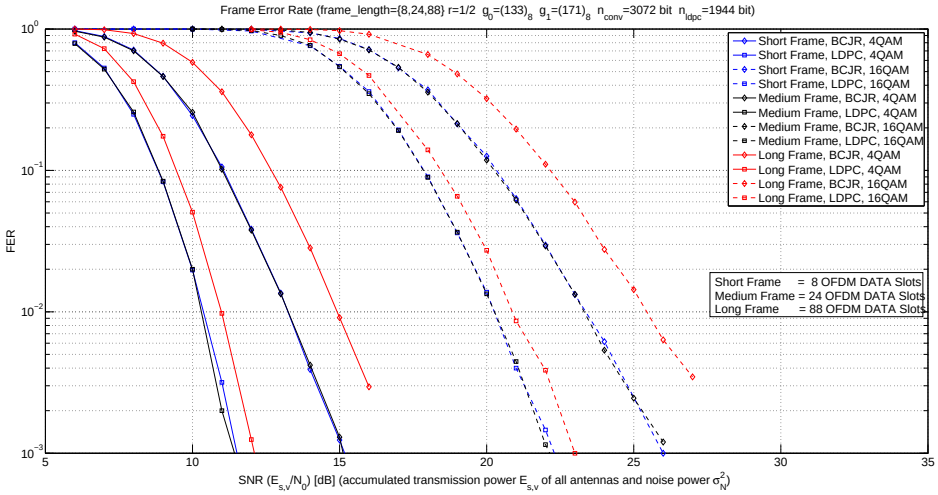


Fig. 6. Investigation of frame size effect for MMSE-DS-QRD detection

subcarriers (Figure 7).

Based on the measured execution times, three groups of processor cores have been defined. All processors of one group execute the same function, hence have the same executable. The first group executes OFDM demodulation and contains $N_{mod} = 2$ processor cores. Another group executes channel estimation, preprocessing of the MIMO detection and the actual detection. This group contains $N_{ppd} = 4$ processors, of which 2 can be switched off after 27 OFDM data slots (backlog phase) have been processed. Finally, $N_{sd} = 2$ processors are assigned to execute soft symbol demapping. Again, after processing 27 OFDM data slots of one frame, 1 of the 2 processor cores can be switched off. For this mapping a frame length of 27 OFDM data slots is the minimum for achieving throughput constraints and minimizing latency ($8.2\mu s$).

Figure 8(a) illustrates the computed execution characteristic of the complete receiver for the minimum supported frame length (27 OFDM data slots). The calculation has been solely based on the conducted timing measurements of Flavors and the selected mapping. This configuration has been implemented and the correct timing behavior has been verified (Figure 8(b)) using the Transaction Level Model (TLM) of the platform P2012 (SDK 2011.1). To minimize synchronization overhead, low-level platform features that make explicit use of

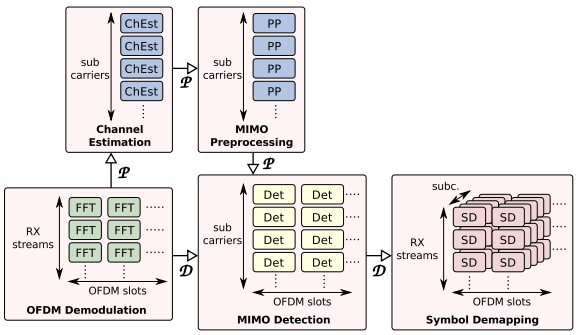


Fig. 7. Parallelism contained within the MIMO OFDM receiver

the hardware synchronizer of platform P2012 have been used.

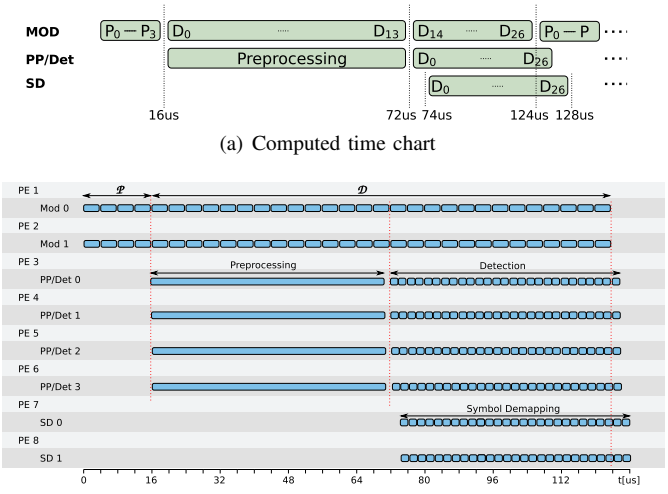
With the configuration described above, the inner modem part of the MIMO OFDM receiver has been implemented with real-time behavior and it supports a coded data rate of 192 Mbit/s with low frame error rate (less than 1%).

C. Summary of Receiver Implementation

These final measurements have demonstrated that the inner receiver of the targeted MIMO OFDM receiver can be implemented on the target platform in real time, although the platform's target is not primary in the field of baseband processing. Based on our algorithmic analysis and the implementation of the inner modem, we identified the vector unit as of vital importance to achieve reasonable performance. To fully exploit these capabilities most of the code makes extensive use of low-level C-Macros for using the vector unit and only a few control statements use the general purpose instruction set.

For implementing the application on the overall platform, the hardware synchronizer with low-level event signaling has been an excellent choice for synchronizing the different functions executed on the various processor cores. Similarly, the shared memory architecture was simple to use and supported low-latency communication between the processor cores. Apart from these low-level platform features, other programming models on higher software levels could not be used, since the encountered overhead has been too large for targeted SDR application.

Besides these benefits of the platform, the limitations should not be concealed. Throughout the transceiver development, only a small set of vector unit instructions has been used (approx. 20% of 214 instruction have been used). Other instructions that would significantly enhance the performance have been missing, e.g. a multiply-shift instruction. Especially, such native fixed-point support would have tremendous impact on the required cycle counts. However, in turn this would limit the flexibility to a given fixed-point format. Clearly, this might effect the mapping of the investigated transceiver since each implemented function has its own optimized fixed point format. Nevertheless, removing not required instructions and



(a) Computed time chart

(b) Measured time chart

Fig. 8. Execution characteristic of mapped MIMO OFDM receiver (frame length = 27 OFDM data slots)

adding some others could enhance the execution behavior and energy consumption might be reduced. Furthermore, adding application specific instruction set processors or HW accelerators could increase the performance and energy efficiency.

VI. Conclusion and Future Work

In this paper we successfully implemented a MIMO OFDM transceiver with 4x4 antenna configuration and timing requirements taken from the 802.11n standard following the Nucleus concept. First, a thorough analysis of the algorithms has been conducted and a set of *Nuclei* has been extracted. Second, efficient implementations called *Flavors* have been developed for each Nucleus targeting the P2012 platform. The Flavors have been characterized in terms of execution behavior but also regarding their algorithmic performance.

To map the transceiver to the platform, first we derived a feasible mapping based on the Flavor characteristics. This computed mapping had only a minimal overhead margin included and was verified with a detailed hardware simulation based on the TLM platform simulator.

Besides the identification of Nuclei and the implementation of Flavors, the capabilities of the many-core computing fabric have been investigated when implementing an SDR application. Similar to previous investigations we determined the vector unit to be of vital importance when real time behavior is required. Real time behavior has been achieved for the inner modem part of a 4x4 MIMO OFDM transceiver by using 8 processor cores, whereas 3 of these 8 can be switched off after processing the first 27 OFDM data slots.

In our future work, we will investigate and develop flexible hardware architectures optimized for the specific needs of the determined Nuclei. In addition, we will investigate the porting of the complete transceiver to another platform by making use of the Nucleus methodology and tools.

Finally, we use the characterization of Nuclei and Flavors in our automatic mapping tools [20] and will elaborate the generated mapping results with the handcrafted solution.

References

[1] Ramakrishnan, V., et al. Efficient and Portable SDR Waveform Development: The Nucleus Concept. In *IEEE Military Communications Conference (MILCOM 2009)*, Boston, USA, Oct 2009.

[2] IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999) - Wireless LAN MAC and PHY Specifications. 12 2007.

[3] F. Adachi. Evolution Towards Broadband Wireless Systems. In *Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on*, volume 1, pages 19 – 26 vol.1, 2002.

[4] Kees van Berkel, Frank Heinle, Patrick P. E. Meuwissen, Kees Moerman, and Matthias Weiss. Vector Processing as an Enabler for Software-Defined Radio in Handheld Devices. *EURASIP J. Appl. Signal Process.*, 2005:2613–2625, January 2005.

[5] Ulrich Ramacher. Software-Defined Radio Prospects for Multistandard Mobile Phones. *Computer*, 40:62–69, 2007.

[6] C. H. (Kees) van Berkel. Multi-Core for Mobile Phones. DATE '09.

[7] O. et al. Gustafsson. Architectures for Cognitive Radio Testbeds and Demonstrators - An Overview. In *CROWNCOM 2010*, june 2010.

[8] Y. Lin, H. Lee, M. Woh, Y. Harel, S. Mahlke, T. Mudge, C. Chakrabarti, and K. Flautner. Soda: A high-performance dsp architecture for software-defined radio. *Micro, IEEE*, 27(1):114 –123, jan.-feb. 2007.

[9] Texas Instruments Inc. C6000 High Performance Multicore DSP, 2011.

[10] Freescale Semi. Inc. MSC8156: Six Core High Performance DSP, 2011.

[11] Azimi, M.; Cherukuri, N.; Jayasimha, D.N.; Kumar, A.; Kundu, P.; Park, S.; Schoinas, I.; Vaidya, A. Integration Challenges and Tradeoffs for Tera-scale Architectures. In *Intel Technology Journal.*, http://www.intel.com/technology/itj/2007/, Aug 2007.

[12] P. S. Paolucci. The Diopsis Multiprocessor Tile of SHAPES. In *MPSoC'06*, Aug. 2006.

[13] STMicroelectronics and CEA. Platform 2012: A Many-core programmable accelerator for Ultra-Efficient Embedded Computing in Nanometer Technology. White paper, November 2010.

[14] A.J. Paulraj, D.A. Gore, R.U. Nabar, and H. Bolcskei. An Overview of MIMO Communications - a Key to Gigabit Wireless. *Proceedings of the IEEE*, 92(2):198 – 218, feb 2004.

[15] Tensilica Inc. ConnX Baseband DSP and Xtensa Customizable Processor, May 2011.

[16] STEricsson. Low-power Embedded Vector DSP, EVP VD32041 32-bit Embedded-Vector Processor for SoCs, Feb. 2009.

[17] STMicroelectronic. *STxP70-4 Core and Instruction Set Architecture. 8154580 Rev. B*. http://www.st.com/, June 2010.

[18] Gordon Blair, Thierry Coupaye, and Jean-Bernard Stefani. Component-based Architecture: the Fractal Initiative. *Annals of Telecommunications*, 64:1–4, 2009.

[19] Bradford Nichols, Dick Buttlar, and Jacqueline Proulx Farrell. *Pthreads Programming*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1996.

[20] Castrillon, J., et al. Component-based Waveform Development: the Nucleus Tool Flow for Efficient and Portable SDR. In *2010 Wireless Innovation Conference and Product Exposition (SDR'10).*, Washington D.C., USA, Dec 2010.

[21] PrismTech Ltd. Spectra CX - SCA Development Tool (Visited 2011).

[22] Zeligsoft. Zeligsoft CE 2.4 (Visited 2011).

[23] CRC Canada. Scari Software Suite (Visited Dec. 2010).

[24] Lyrtech Inc. *SFF SDR DP API Guide*, 1.0 edition, January 2007.

[25] HyperX Development System, 2010.

[26] Peter Jan Luethi. *VLSI circuits for MIMO preprocessing*. ETH, 2009.

[27] Hun Seok et al. Kim. A practical, hardware friendly MMSE detector for MIMO-OFDM-based systems. *EURASIP J. Adv. Signal Process*, 2008:94:1–94:14, January 2008.

[28] E. Perahia and R. Stacey. *Next Generation Wireless LANs: Throughput, Robustness, and Reliability in 802.11n*. Cambridge University Press, Cambridge, UK, 2010.

[29] I.B. Collings, M.R.G. Butler, and M. McKay. Low Complexity Receiver Design for MIMO Bit-Interleaved Coded Modulation. In *International Symposium on Spread Spectrum Techniques and Applications, 2004*.

[30] David Tse and Pramod Viswanath. *Fundamentals of Wireless Communications*. Cambridge University Press, 2004.

[31] P. et al. Almers. Survey of Channel and Radio Propagation Models for Wireless MIMO Systems. *EURASIP J. Wirel. Commun. Netw.*, 2007:56–56, January 2007.

[32] Simon Haene. *VLSI circuits for MIMO-OFDM physical layer*. ETH, 2007.