A Novel Class of Linear MIMO Detectors With Boosted Communications Performance: Algorithm and VLSI Architecture

Dominik Auras, Rainer Leupers, Gerd Ascheid
Institute for Communication Technologies and Embedded Systems
RWTH Aachen University, 52056 Aachen, Germany
email: auras@ice.rwth-aachen.de

***Abstract*—This paper introduces a novel class of linear soft-input soft-output detectors with boosted communications performance. The detector showed an SNR gain of up to 2.4 dB compared to state-of-the-art linear detectors. We introduce a low-complexity algorithm tailored for VLSI implementation, and propose a suitable architecture. The developed ASIC demonstrates the feasibility and efficiency of the concept, achieving the IEEE 802.11n standard's peak data rate of 600 Mbit/s.**

***Keywords*-VLSI architecture; Digital signal processing; MIMO Systems; Wireless Communications; Data detection**

I. Introduction

Multi-antenna (MIMO) systems that employ bit-interleaved coded modulation (BICM) are already adopted for wireless standards such as IEEE 802.11n. The extension of BICM with Iterative Decoding (BICM-ID) is a promising option that provides superior communications performance at high spectral efficiency. However, the data detection within these systems poses particular challenges to VLSI implementation.

The exact MIMO detector suffers from an exponential complexity. Linear detectors, e.g. using MMSE filtering, are an economically reasonable option particularly suitable for VLSI designs. These offer a simple algorithmic structure, are purely feed-forward and thus deeply pipelineable. There is only simple signal processing, without any non-linear control flow. The linear algorithms exhibit inherent pipeline, symbol-level and stream-level parallelism.

Additionally, it was shown in [6] that systems using linear detectors can actually have a better spectral efficiency than with close-to-optimal detectors, for a given maximum area constraint. The reported analysis indicates that the VLSI designs' properties might be more important than the communications performance in certain cases.

Ideally, we would like to improve the communications performance of linear detectors while maintaining all beneficial implementation properties. Eventually, a recent publication [1] on detection algorithms based on the Expectation Propagation framework highlighted new opportunities to improve linear detection. In this paper, we propose the EPIC detector based on [1], termed the ***E**xpectation **P**ropagation based MIMO Detector AS**IC***.

Contribution: We show how to build a detector that combines the VLSI implementation efficiency of linear detectors with a greatly improved communications performance. To this end, we propose a low-complexity algorithm and provide an ASIC case study. Our post-layout results demonstrate our concept's competitive implementation complexity in terms of area and throughput compared to related work.

Outline: After the description of the assumed system model, we give an overview of related work. Subsequently, we present a low-complexity algorithm and a suitable architecture design. In the results section, we evaluate the algorithmic performance and hardware figures of our approach.

II. System Model

Fig. 1 depicts the considered spatial multiplexing $N_t \times N_r$ MIMO system with BICM-ID. A message $\boldsymbol{b} \in \{0,1\}^{N_b}$ is encoded with rate $r = N_b/N_c$ and interleaved, yielding the code word $\boldsymbol{c} \in \{0,1\}^{N_c}$. Let $\mathcal{X} \subset \mathbb{C}$ be a modulation alphabet with $K = \log_2 |\mathcal{X}|$ bits per symbol. The code word is partitioned into N_s subvectors $\boldsymbol{c}_n \in \{0,1\}^{KN_t}$. They are subsequently mapped to symbol vectors $\boldsymbol{x}_n \in \mathcal{X}^{N_t}$ that are transmitted independently. Assuming a frequency-flat fading channel characterized by $\boldsymbol{H}_n \in \mathbb{C}^{N_r \times N_t}$ and perfect synchronization in time and frequency, the received symbol vector at time n is $\boldsymbol{y}_n = \boldsymbol{H}_n \boldsymbol{x}_n + \boldsymbol{w}_n$ where $\boldsymbol{w}_n \in \mathbb{C}^{N_r}$ is a white Gaussian noise process with $\mathbb{E}[\boldsymbol{w}_n \boldsymbol{w}_n^H] = N_0 \boldsymbol{I}_{N_r}$. In the remainder, we drop the time index n for convenience.

Perfect channel knowledge at the receiver is assumed. Using iterative MIMO decoding, detector and channel decoder exchange extrinsic soft information $\boldsymbol{\lambda}^e = \boldsymbol{\lambda}^p - \boldsymbol{\lambda}^a$ in terms of log-likelihood ratios (LLRs), where $\boldsymbol{\lambda}^p$ are the detector's posterior LLRs and $\boldsymbol{\lambda}^a$ are the prior LLRs fed back from the decoder. A BICM-ID receiver thus requires soft-input soft-output (SISO) channel decoder and detector. In one receiver iteration marked with (A) in Fig. 1, detector and decoder are executed once, in that order.

A. Related Work

Approaches for MIMO detector design can be divided into two categories: linear and non-linear. Non-linear detectors

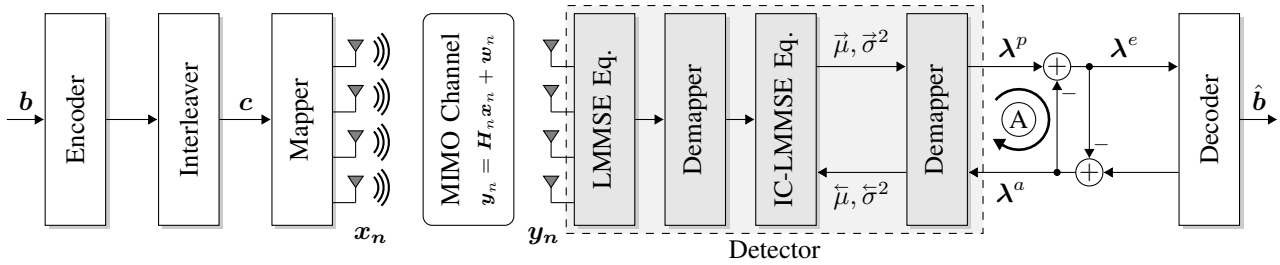

Figure 1. Assumed MIMO BICM-ID System Model. Detector and decoder iteratively exchange soft information (LLRs) to improve the final decoding result.

treat the detection problem as a search for the most likely symbol vector candidate $\boldsymbol{x}$ in the space $\mathcal{X}^{N_t}$. Their VLSI implementation is complex, however they regularly have better communications performance than linear detectors.

For example, the single tree search sphere decoder (STS-SD) [6] traverses a tree representing the search space in depth-first manner, to look for the most likely candidate and the best counter hypotheses. It is shown to achieve max-log optimality. However to guarantee a minimum throughput, the necessary search break-off results in significant losses.

To alleviate this problem, K-Best detectors perform a breadth-first tree search and keep only the K best candidates per transmit layer. This gives a constant deterministic run-time. The algorithmic performance is suboptimal, as it might miss many good candidates and fails to find good counter-hypotheses. No VLSI implementation of SISO K-Best detection is reported so far.

Fixed-complexity sphere decoders (FCSD) [4] also feature a deterministic run-time, but better communications performance. The computational effort is directed towards nodes that are more likely to change the final result (imbalanced expansion). Usually, a full search is performed for the two weakest transmit layers. On the remaining layers, only the best per-stream symbols are selected. The Parallel Candidate Adding scheme [4] helps to find counter-hypotheses.

Another approach [7] structures the search space as trellis diagram with N_t columns and $|\mathcal{X}|$ rows. For every trellis node, a specified number of shortest paths is found. This guarantees the availability of suitable information for every bit. The reported VLSI design is quite large while supporting only 16-QAM.

To get particularly small VLSI designs, Markov-Chain Monte-Carlo (MCMC) detection [5] is a good option. The search is performed randomly. Essentially, this removes almost all the control logic for candidate selection, and leaves only the evaluation logic. However, the detectors are rather slow, offering only a low throughput, and the architecture parameter choice is still an open question.

On the other side, linear detectors try to estimate the most likely symbol distribution. They assume that the transmit symbol distribution is Gaussian, i.e. $x \sim \mathcal{CN}$. The idea is to separate the spatially super-imposed transmit streams in order to process them individually. This greatly reduces the processing complexity, however also results in a suboptimal communications performance. A well-known VLSI design is the MMSE-PIC [2], that performs MMSE detection with parallel interference cancellation.

III. EPIC Detector

The following subsections introduce the proposed low-complexity detection algorithm suitable for VLSI implementation. We tailored the algorithm reported in [1] specifically for VLSI implementation. The concept was partially presented as two-pages abstract [12].

A. Algorithm of the EPIC Detector

The idea is to use regular SISO MMSE MIMO detection, but to *boost* it in the first receiver iteration, when the decoder's message is not available yet, using an initial SO-only MMSE MIMO detection.

The proposed detector, depicted in Fig. 1, in principle resembles a standard IC-LMMSE detector. It uses an interference cancellation followed by an LMMSE filter and per-stream demapping. We add an extra LMMSE detector at the front. Upon the reception of a symbol vector, this detector provides additional prior knowledge to the subsequent IC-LMMSE detector. While iterating between detector and decoder, only the IC-LMMSE detector is active.

The LMMSE detector corresponds to the first two blocks of the detector in Fig. 1, while the IC-LMMSE MIMO detector corresponds to the other two blocks. The second internal detector requires a slight modification in order to process the first detector's output. In the first receiver iteration, when the decoder's feedback is not available, i.e. $\boldsymbol{\lambda}^a = \mathbf{0}$, the first internal detector provides the prior knowledge in terms of LLRs denoted as $\boldsymbol{\lambda}^{\text{int}}$ and its LMMSE filter's output to the second internal detector.

In general, the four internal blocks exchange information in terms of Gaussian distributions. We parametrize the messages produced by the equalizers with $\vec{\mu}_t$ and $\vec{\sigma}_t^2$. The demappers' messages have the parameters $\overleftarrow{\mu}_t$ and $\overleftarrow{\sigma}_t^2$. Both messages combined according to

$$\frac{\mu_t}{\sigma_t^2} = \frac{\overleftarrow{\mu}_t}{\overleftarrow{\sigma}_t^2} + \frac{\vec{\mu}_t}{\vec{\sigma}_t^2} \quad \text{and} \quad \frac{1}{\sigma_t^2} = \frac{1}{\overleftarrow{\sigma}_t^2} + \frac{1}{\vec{\sigma}_t^2}. \tag{1}$$

express the EPIC detector's current belief on the transmit symbols with parameters μ_t and σ_t^2. The proposed EPIC detector's algorithm is listed in Fig. 2.

1: **if** First Receiver Iteration ($\boldsymbol{\lambda}^a = \mathbf{0}$) **then**
2: $\quad \vec{\mu}_t, \vec{\sigma}_t^2 = \text{LMMSE}(\boldsymbol{y}, \boldsymbol{H}, N_0)$
3: $\quad\quad \boldsymbol{\lambda}^{\text{int}} = \text{demap}(\vec{\mu}_t, \vec{\sigma}_t^2)$
4: $\quad \mu_t, \sigma_t^2 = \text{map}(\boldsymbol{\lambda}^{\text{int}})$
5: $\quad \overleftarrow{\mu}_t, \overleftarrow{\sigma}_t^2 = \text{gaussdiv}(\mu_t, \sigma_t^2, \vec{\mu}_t, \vec{\sigma}_t^2)$
6: **else**
7: $\quad \overleftarrow{\mu}_t, \overleftarrow{\sigma}_t^2 = \text{map}(\boldsymbol{\lambda}^a)$
8: **end if**
9: $\vec{\mu}_t, \vec{\sigma}_t^2 = \text{IC-LMMSE}(\boldsymbol{y}, \boldsymbol{H}, N_0, \overleftarrow{\mu}_t, \overleftarrow{\sigma}_t^2)$
10: $\quad \boldsymbol{\lambda}^e = \text{demap}(\vec{\mu}_t, \vec{\sigma}_t^2)$

Figure 2. The EPIC Detector's underlying algorithm.

The bridge between the two internal detectors lies in line 5. The function *gaussdiv* performs a "division" of Gaussian distributions. It computes

$$\overleftarrow{\sigma}_t^2 = \left(\frac{1}{\sigma_t^2} - \frac{1}{\vec{\sigma}_t^2}\right)^{-1} \quad \text{and} \quad \overleftarrow{\mu}_t = \overleftarrow{\sigma}_t^2\left(\frac{\mu_t}{\sigma_t^2} - \frac{\vec{\mu}_t}{\vec{\sigma}_t^2}\right) \tag{2}$$

using the inputs from the mapper and the filter.

Additionally, the algorithm assumes to obtain the decoder's posterior LLRs and neglects the prior knowledge in the final demapping. We omit both subtractions between detector and decoder in Fig. 1, thus directly compute $\boldsymbol{\lambda}^e$.

B. Algorithmic Building Blocks

In the following subsection, we introduce low-complexity versions of the algorithmic functions used in Alg. 2, that are particularly suitable for VLSI implementation.

1) IC-LMMSE and LMMSE Equalizer: The lines 2 and 9 of Alg. 2 use MMSE equalization. We describe first the IC-LMMSE equalizer and subsequently the differences to the LMMSE equalizer.

From the inputs $\boldsymbol{y}$, $\boldsymbol{H}$, N_0, $\overleftarrow{\mu}_t$ and $\overleftarrow{\sigma}_t^2$, we want to calculate the parameters $\vec{\mu}_t$ and $\vec{\rho}_t = 1/\vec{\sigma}_t^2$. First, compute the gram matrix $\boldsymbol{R}$ and the matched filter output $\boldsymbol{y}^{\text{mf}}$ as

$$\boldsymbol{R} = \boldsymbol{H}^H\boldsymbol{H} \in \mathbb{C}^{N_t \times N_t} \quad \text{and} \quad \boldsymbol{y}^{\text{mf}} = \boldsymbol{H}^H\boldsymbol{y} \tag{3}$$

which we refer to as preprocessing step in the remainder. Perform interference cancellation on $\boldsymbol{y}^{\text{mf}}$ to obtain

$$\boldsymbol{y}^{\text{mf,ic}} = \boldsymbol{y}^{\text{mf}} - \sum_{t=1}^{N_t} \boldsymbol{r}_t \overleftarrow{\mu}_t \tag{4}$$

where $\boldsymbol{r}_t$ denotes the t-th column of $\boldsymbol{R}$. Then decompose the matrix

$$\boldsymbol{C}_{\boldsymbol{y}} = \boldsymbol{R} + N_0\boldsymbol{C}_{\boldsymbol{x}}^{-1} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^H \tag{5}$$

where $\boldsymbol{C}_{\boldsymbol{x}} = \text{diag}(\overleftarrow{\sigma}_1^2, \ldots, \overleftarrow{\sigma}_{N_t}^2)$, into a lower triangular matrix $\boldsymbol{L} = [l_{ij}] \in \mathbb{C}^{N_t \times N_t}$ and a positive real-valued diagonal matrix $\boldsymbol{D} = \text{diag}(d_{ii}) \in \mathbb{C}^{N_t \times N_t}$. The matrix $\boldsymbol{L}$ has a unity diagonal $l_{ii} = 1$. We limit the noise density to a lower bound $N_{0,\text{min}}$, and the elements d_{ii} to d_{min} for numerical reasons. Subsequently, invert the matrices using forward and back substitution to obtain

$$\boldsymbol{G}^H = \boldsymbol{C}_{\boldsymbol{y}}^{-1} = (\boldsymbol{L}^H)^{-1}\boldsymbol{D}^{-1}\boldsymbol{L}^{-1} \tag{6}$$

by solving $\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^H\boldsymbol{G}^H = \boldsymbol{I}_{N_t}$ for $\boldsymbol{G}^H$. The algorithm reuses the reciprocals d_{ii}^{-1} computed for the decomposition in the back substitution step. Compute the bias term

$$\tilde{\mu}_t = \boldsymbol{g}_t^H\boldsymbol{r}_t \tag{7}$$

and its reciprocal $\tilde{\mu}_t^{-1}$, then compute new estimates of the per-stream symbol means and variances as

$$\begin{aligned} \vec{\mu}_t &= \overleftarrow{\mu}_t + \tilde{\mu}_t^{-1}\boldsymbol{g}_t^H\boldsymbol{y}^{\text{mf,ic}} \\ \vec{\sigma}_t^2 &= \overleftarrow{\sigma}_t^2(\tilde{\mu}_t^{-1} - 1) \end{aligned} \tag{8}$$

where $\boldsymbol{g}_t^H$ denotes the t-th row of $\boldsymbol{G}^H$. For numerical reasons, we limit $\tilde{\mu}_t$ to a lower value $\tilde{\mu}_{\text{min}}$, and the variances $\vec{\sigma}_t^2$ to $\vec{\sigma}_{\text{min}}^2$. The last step is to compute the per-stream SNRs $\vec{\rho}_t = 1/\vec{\sigma}_t^2$.

In line 2, perform LMMSE equalization since there is no prior knowledge available. To this end, we set $\overleftarrow{\mu}_t = 0$ and $\overleftarrow{\sigma}_t^2 = E_s$, where E_s denotes the average transmit symbol energy, in the above description and do not perform interference cancellation, i.e. assume $\boldsymbol{y}^{\text{mf,ic}} = \boldsymbol{y}^{\text{mf}}$.

2) Demapper: In the lines 3 and 10, we compute the per-stream LLRs $\boldsymbol{\lambda}^{\text{int}}$ and $\boldsymbol{\lambda}^e$ respectively. Using the max-log approximation and omitting the priors $\boldsymbol{\lambda}^a$, demap the equalizer's output $\vec{\mu}_t$ and $\vec{\sigma}_t^2$ according to

$$\lambda_{t,k} = \vec{\rho}_t\left(\min_{x \in \mathcal{X}_k^0} |x - \vec{\mu}_t|^2 - \min_{x \in \mathcal{X}_k^1} |x - \vec{\mu}_t|^2\right) \tag{9}$$

where $\mathcal{X}_k^0$ and $\mathcal{X}_k^1$ are subsets of $\mathcal{X}$ with the k-th bit set to zero or one respectively. The difference of the min-terms is computed with piece-wise linear functions [9].

3) Mapper: The *map* function computes symbol means μ_t and variances σ_t^2 from the LLRs. Select $\boldsymbol{\lambda}^{\text{int}}$ in line 4 or the decoder's LLRs $\boldsymbol{\lambda}^a$ in line 7. Convert the LLRs $\boldsymbol{\lambda}$ to bit probabilities according to

$$p_{t,k} = \text{logistic}(\lambda_{t,k}) = \frac{1}{1 + \exp(-\lambda_{t,k})}. \tag{10}$$

Then compute the terms $\mathbb{E}[x_t]$ and $\mathbb{E}[x_t^2]$ as in [10]. We limit the symbol variances to a lower bound σ_{min}^2 to improve the numerical stability of the next steps.

4) GaussDiv: The function *gaussdiv* implements (2). It calculates $\overleftarrow{\mu}_t$ and $\overleftarrow{\sigma}_t^2$ from the equalizer's outputs $\vec{\mu}_t$, $\vec{\sigma}_t^2$ and the mapper's outputs μ_t, σ_t^2. The variance $\overleftarrow{\sigma}_t^2$ might become arbitrarily large if $\sigma_t^2 \approx \vec{\sigma}_t^2$, and even negative e.g. due to rounding errors. This results in a largely increased dynamic range requirement for subsequent computations. We propose

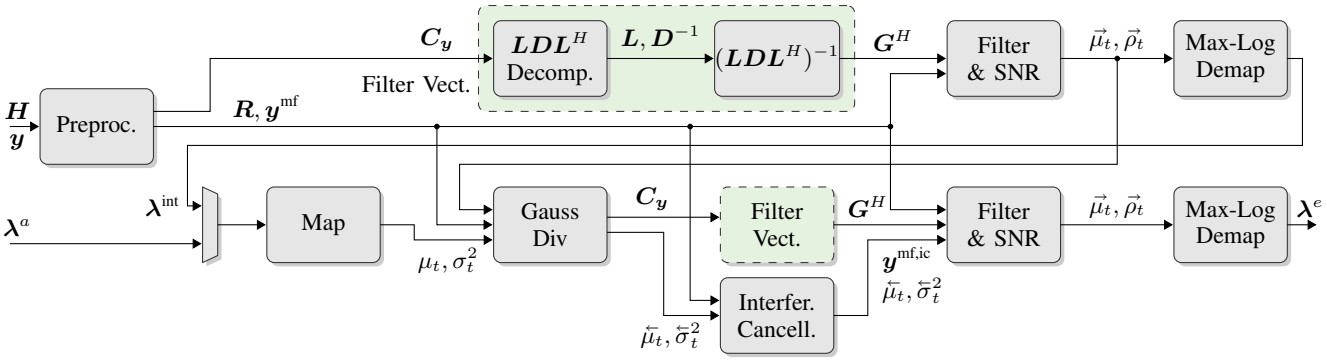

Figure 3. EPIC Architecture Design. The two blocks *Filter Vect.* are identical and each contain two PUs. Forwarding buffers are not drawn. In the first receiver iteration, data flows through the upper chain, then enters the lower one. In subsequent receiver iterations, only the Preproc. PU and the lower chain are active.

a simple ad-hoc heuristic that effectively avoids the increase. To this end, we define that if

$$\overleftarrow{\rho}_t = \frac{1}{\sigma_t^2} - \frac{1}{\vec{\sigma}_t^2} \overset{?}{<} \overleftarrow{\rho}_{\text{thr}} \tag{11}$$

where $\overleftarrow{\rho}_{\text{thr}}$ is a constant threshold, then we set $\overleftarrow{\mu}_t = \mu_t$ and $\overleftarrow{\sigma}_t^2 = \sigma_t^2$, i.e. we forward the mapper's unprocessed output. Otherwise, compute

$$\overleftarrow{\sigma}_t^2 = \frac{1}{\overleftarrow{\rho}_t} \quad \text{and} \quad \overleftarrow{\mu}_t = \overleftarrow{\sigma}_t^2 \left(\frac{\mu_t}{\sigma_t^2} - \frac{\vec{\mu}_t}{\vec{\sigma}_t^2} \right). \tag{12}$$

The threshold $\overleftarrow{\rho}_{\text{thr}}$ is chosen as power of two for further simplification.

C. Explanations

The crux in the EPIC detector is the *GaussDiv* process. Without it, we simply have two chained linear detectors. Our experiments confirm that the full performance enhancement is only available using the *GaussDiv* method.

Thus, we want to give a possible interpretation of this phenomenon. Basically, the first LMMSE filter produces a continuous-valued distribution with mean $\mu_t \in \mathbb{C}$, that describes the most likely transmit symbol distribution. The subsequent demapper discretizes the continuous distribution, i.e. it evaluates it at every valid point $x_t \in \mathcal{X}$. The mapper computes a new continuous-valued estimate, which is assumed to be the mean of a new continuous distribution. The innovation contained in this new message potentially results from imposing the validity constraint $x_t \in \mathcal{X}$. The *gaussdiv* function extracts this innovation, by "dividing" the two distributions. Note that instead of computing symbol probabilities, we compute bit probabilities, which is essentially an approximation of the process, as it neglects symbol correlations.

IV. Architecture

As a case study, we implemented the algorithm to support the IEEE 802.11n standard's peak data rate of $\Theta_b = 600\,\text{Mbit/s}$, which is a code bit throughput of $\Theta_c = 720\,\text{MBit/s}$ (for $N_t = 4$, 64-QAM, $r = 5/6$). There are two possible workloads, for the first and the subsequent iterations. Our implementation has a constant deterministic throughput, independent of the receiver iteration and SNR.

A. Overview

We use a coarse-grained pipeline of multi-cycle processing units (PUs) that take 18 clock cycles per pipeline cycle, depicted in Fig. 3. For a reasonable clock frequency of $f_{clk} = 540\,\text{MHz}$, we achieve a data rate of $\Theta_b = \frac{rN_tK}{18} f_{\text{clk}} = 600\,\text{Mbit/s}$.

The architecture runs in two possible configurations. For the first receiver iteration, data flows through the *Preprocessing* PU into the upper internal detector composed of four PUs. Then it enters the lower internal detector comprising seven PUs. For any subsequent receiver iterations, we disable the upper chain. The mapper PU takes the input $\boldsymbol{\lambda}^a$. In this mode, the *GaussDiv* unit always forwards $\overleftarrow{\mu}_t, \overleftarrow{\sigma}_t^2$ and computes $\boldsymbol{C_y}$. The processing latency depends on the configuration. In the first case, the latency is $L = 11 \cdot 18$ clock cycles, while in the second it is $L = 7 \cdot 18$ clock cycles. The architecture's throughput is identical for both configurations.

B. Processing Units

The PUs exchange data in one dedicated clock cycle. They use a simple handshake based protocol. All data is kept in registers. Forwarding buffers are used where required, but are not drawn in Fig. 3. All resource allocations of arithmetic units and schedules have been devised manually for each PU.

Beneath regular multiplier and adder units, we use lookup tables e.g. for the logistic function in (10). Newton-Raphson based reciprocal units perform the required divisions. This unit type first normalizes the input x, then performs an initial table lookup $\tilde{x}_0 \approx 1/x$, followed by one Newton-Raphson iteration $\tilde{x}_1 = 2\tilde{x}_0 - x\tilde{x}_0^2$ to obtain an approximate reciprocal $\tilde{x}_1 \approx 1/x$.

We use saturation for the input, output and intermediate LLRs $\boldsymbol{\lambda}^a$, $\boldsymbol{\lambda}^e$ and $\boldsymbol{\lambda}^{\text{int}}$ respectively.

Table I
EPIC ALGORITHM TO PU MAPPING

Processing Unit	Implemented Functionality
Preprocessing	Eq. (3), $\boldsymbol{C_y}$ in Eq. (5)
$\boldsymbol{LDL}^H$-Decomposition	$\boldsymbol{C_y} = \boldsymbol{LDL}^H$ in Eq. (5)
Solver $(\boldsymbol{LDL}^H)^{-1}$	Eq. (6)
Filter & SNR	Eq. (7), (8), $\vec{\rho}_t$
Max-Log Demap	Eq. (9)
Map	Eq. (10), $\mathbb{E}[x_t]$, $\mathbb{E}[x_t^2]$, μ_t, σ_t^2
GaussDiv	Eq. (11), (12), $\boldsymbol{C_y}$ in Eq. (5)
Interference Cancellation	Eq. (4)

Due to lack of space, we cannot show the PUs' implementation details here. However, the architectural style resembles those presented in [2] and in [3], [11].

C. Partitioning

The proposed algorithm partition and PU mapping for this case study is given in Tbl. I. It follows approximately the functional bounds. Note that several PUs are instantiated twice, as they are in use by both internal detectors. Where appropriate, we customized the unique instances individually, e.g. assuming $\overleftarrow{\mu}_t = 0$ for the upper *Filter & SNR*.

V. IMPLEMENTATION RESULTS

As proof of concept, we designed an ASIC that meets the IEEE 802.11 standard's peak data rate, in order to show the feasibility and efficiency of the proposed algorithm.

A. Conditions and Assumptions

A 40 MHz 802.11n-like scenario is considered assuming a 4×4 MIMO system with gray-mapped 4-/16-/64-QAM modulation, max-log demapping, a spatially uncorrelated Rayleigh channel and perfect channel knowledge. We use a tail-biting convolutional code with polynomials $[133, 171]_8$ and puncturing for the code rates 1/2, 2/3, 3/4 and 5/6, and a max-log BCJR decoder. The frame length equals the interleaver's length, given by the IEEE 802.11n standard, which depends on the modulation and coding scheme. The average signal-to-noise ratio (SNR) per receive antenna is defined as SNR $= \mathbb{E}[\|\boldsymbol{Hx}\|^2]/(N_r N_0)$. We determined the required word lengths to obtain an SNR loss of ≤ 0.1dB compared to the floating-point model at a frame error rate of 10%. All simulations have been performed with 10^5 frames.

B. Algorithmic Performance

Fig. 4 shows the frame error rate (FER) over SNR for the $\Theta_b = 600$ Mbit/s mode of the IEEE 802.11n standard, assuming $N_t = 4$, 64-QAM ($K = 6$) and the code rate $r = 5/6$. The simulation includes the bit-accurate model of our EPIC, and models of the MMSE-PIC [2] detector and the fixed-complexity sphere decoder (FCSD) [4] that both neglect any finite word length effects. The EPIC shows a reasonable communications performance in-between the FCSD and the MMSE-PIC as expected [1].

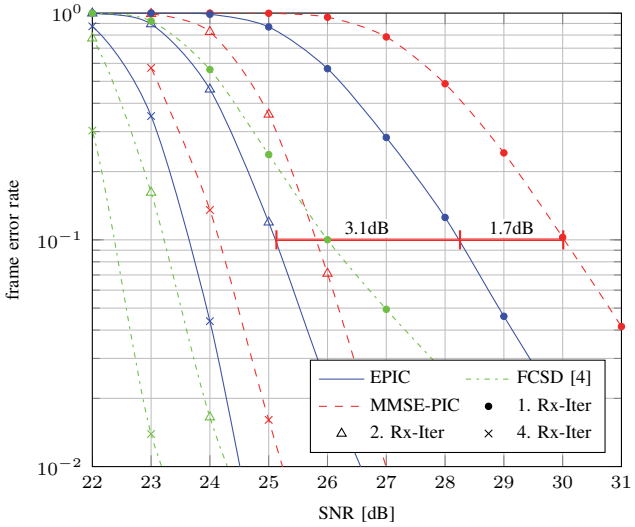

Figure 4. Frame Error Rate over SNR. The *MMSE-PIC* model excludes any finite word length effects. *EPIC* is our VLSI implementation including all finite word length effects.

Tbl. II summarizes the communications performance gain of the EPIC over the MMSE-PIC, measured at 10% FER. We observe a possible gain of more than 2 dB. The largest gain is achieved in the first receiver iteration, as could be expected. For subsequent receiver iterations, the EPIC is a default IC-LMMSE detector, thus the observed gains result from the improved initial performance. The gain diminishes over receiver iterations. The EPIC exhibits greater gain at lower code rates. The gain from iterating between detector and decoder, e.g. visible in Fig. 4, appears to be weaker for the EPIC than for the MMSE-PIC, for example approximately 4.2 dB (MMSE-PIC) to 3.1 dB (EPIC) from first to second receiver iteration. This is in accordance with the original algorithm [1] and not due to our implementation.

We also conducted simulations using the low-density parity check (LDPC) codes defined in the IEEE 802.11n standard. The observations are similar. The gain values are lower, but always greater than zero, thus the EPIC performs at least as good as the MMSE-PIC.

Table II
EPIC COMMUNICATIONS PERFORMANCE GAIN

Modulation	Code Rate	SNR Gain [dB] to MMSE-PIC			
		1. It	2. It	3. It	4. It
QPSK	1/2	0.9	0.2	0.0	0.0
	3/4	2.4	0.7	0.3	0.2
16-QAM	1/2	0.9	0.3	0.1	0.0
	3/4	2.0	0.8	0.6	0.4
64-QAM	2/3	1.0	0.4	0.3	0.1
	3/4	1.4	0.5	0.4	0.2
	5/6	1.7	0.6	0.5	0.5

Table III
COMPARISON TO OTHER REPORTED MIMO DETECTORS

	This work	[2]	[3]	[4]	[5]	[6]	[7]	[8]
Number of antennas	4×4	4×4	4×4	4×4	$\leq 4 \times 4$	$\leq 4 \times 4$	4×4	4×4
Modulation order	≤ 64	≤ 64	≤ 64	64	≤ 64	≤ 64	16	64
Algorithm	EPIC	MMSE-PIC	IC-LMMSE	FCSD	MCMC	STS-SD	Trellis	K-Best
Iterative MIMO Decoding	yes	yes	yes	yes	yes	yes	yes	no
Technology	90 nm	90 nm	90 nm	90 nm	90 nm	90 nm	65 nm	130 nm
Core area [mm^2]	1.08	1.5	–	2.61	–	–	1.58 (3.03[f])	–
Throughput Θ_c [$\mathrm{Mbit/s}$]	733	757	833	2200	avg. 34.2[e]	avg. 51.1[d]	1700 (1228[f])	946[f]
Preprocessing area [kGE]	345.8	384.2[a]	178.3	–[b]	–[c]	–[b]	–[b]	–[b]
Detection area [kGE]				555	265	175	1097	174
Efficiency [$\mathrm{Mbit/s/kGE}$]	2.11	1.97	4.67	3.96	0.13	0.29	1.12	5.44

[a] Area for chip IO interface excluded
[b] Area for required QRD not included
[c] Area for optional initial detection not included
[d] We assume 100 visited nodes per symbol vector [2].
[e] We assume $N_s = 64, N_{gs} = 8, N_p = 8$ [5].
[f] Scaled to 90nm CMOS technology assuming $t_{pd} \sim 1/s$ and $A \sim 1/s^2$.

C. ASIC Implementation

We synthesized the proposed architecture with Synopsys Design Compiler G-2012.06 in topographical mode and obtained a layout with Cadence SoC Encounter 10.1 using a 1.0V standard-performance standard cell library for the UMC 90nm SP-RVT LowK CMOS process. The ASIC achieves a maximum clock frequency of 550 MHz, yielding a maximum throughput of $\Theta_c = 733\,\mathrm{Mbit/s}$. This variant occupies $345.8\,\mathrm{kGE}$ and $1.08\,\mathrm{mm}^2$. One gate-equivalent (GE) corresponds to the area of one two-input drive-one NAND gate. The per-unit area breakdown is given in Tbl. IV. The matrix inversion, performed on the decomposition and solver units, requires up to $68.9\,\mathrm{kGE}$, which is about the half of the MMSE-PIC's inversion circuitry. This results most likely from the exploitation of the symmetries $\boldsymbol{C}_{\boldsymbol{y}}^H = \boldsymbol{C}_{\boldsymbol{y}}$ and $\boldsymbol{G}^H = \boldsymbol{G}$. With about 5.1%, the additional GaussDiv unit constitutes only a small overhead, but is crucial for the improved communications performance.

Table IV
EPIC IMPLEMENTATION RESULTS

Processing Unit	1st	2nd
Preprocessing [kGE]	32.5	
$\boldsymbol{LDL}^H$ Decomposition [kGE]	38.3	27.5
Solver $(\boldsymbol{LDL}^H)^{-1}$ [kGE]	30.6	29.4
Filter & SNR [kGE]	39.5	52.3
Max-Log Demap [kGE]	7.8	7.1
Map [kGE]	12.2	
GaussDiv [kGE]	17.8	
Interference Cancellation [kGE]	17.6	
Forwarding & Multiplexer [kGE]	33.2	
Total area [kGE]	345.8	
Clock frequency [MHz]	550	
Throughput Θ_c [Mbit/s]	733	
Efficiency [Mbit/s/kGE]	2.11	

Tbl. III shows data of other reported MIMO detectors and our work. We use the area-throughput efficiency for comparisons. The EPIC is smaller than the MMSE-PIC [2], and slightly slower. Both detectors show a similar efficiency around $2\,\mathrm{Mbit/s/kGE}$. The reduced area might be due to the data symmetries of $\boldsymbol{C}_{\boldsymbol{y}}$ and $\boldsymbol{G}^H$ present in the EPIC algorithm, which are not available in the MMSE-PIC's algorithm formulation. Post-synthesis results for an IC-LMMSE detector are reported in [3]. Our detector internally comprises two MMSE detectors, and thus seems to have roughly double the area of that design.

Compared to the FCSD, the EPIC is smaller, but has a lower area-throughput efficiency. Combined with the better communications performance of the FCSD, the EPIC seems to be the less efficient architecture. However, neither the required QR decomposition, the required matrix multiplication $\boldsymbol{Q}^H\boldsymbol{y}$ nor the overhead for storing $\boldsymbol{Q}$ are included in the FCSD's figures, while the EPIC does not need any further preprocessing. Thus, the FCSD has a better detector area-throughput efficiency, however its system efficiency depends on additional circuitry, which is not needed for the EPIC (and for the MMSE-PIC).

The MCMC detector and the STS-SD offer a significantly lower throughput and lower area-throughput efficiency. Furthermore, the STS-SD has a strong run-time variability of the throughput, which makes system design challenging, and requires additional circuitry to manage the variations (e.g. dynamic data distribution and collection).

The trellis-based detector [7] shows good communications performance, similar to the FCSD, but the reported design supports only 16-QAM. For this modulation order, it already

occupies a large area of $> 1\,\mathrm{MGE}$. Transitioning to 64-QAM, the internal trellis will have 64 instead of 16 states, which suggests that the area occupation might grow largely. The area-throughput efficiency is lower than that of the EPIC.

If only SO-only detection is required, then the K-Best detector in [8] is a lot more efficient than the EPIC. We expect that a SO-only variant of the EPIC would save only few area (basically a few multiplexers). This is opposing to SO-only MMSE detectors, that could save more (like the mapper), but the EPIC requires most of the functionality also in the first receiver iteration.

Clearly, the fact that the design is customized for one maximum throughput target is a limitation of our architectural approach. To support higher or lower code rates, beyond simple frequency scaling, we need to redesign the PUs considering less clock cycles per pipeline cycles. E.g. if the target ist $\Theta_b = 300\,\mathrm{Mbit/s}$, then the PUs might take 36 clock cycles instead of 18. The architectural concept is scalable in principle, but requires quite some development effort.

VI. Conclusions

A novel class of linear MIMO detectors for BICM-ID systems is introduced. It keeps the desirable VLSI implementation properties of linear detectors, while additionally improving the communications performance by up to 2.4 dB for the considered IEEE 802.11n-like scenarios. The presented ASIC has a similar area-throughput efficiency as the best linear MIMO detector reported in literature. It proves that similar area and throughput figures combined with significant SNR gains over regular linear MMSE detection is possible.

Future MIMO receivers can take advantage of the detectors's observed SNR gains in order to largely improve the network's transmission range, or increase the data rate in presence of strong inter-cell interference. In fact, the largest gains have been observed for the non-iterative receiver case. Thus an adoption to current MIMO systems is possible without requiring the complexity of iterative MIMO decoding.

Acknowledgment

This work has been supported by the Ultra High-Speed Mobile Information and Communication Research Centre, RWTH Aachen University.

References

[1] M. Senst and G. Ascheid, "How the framework of expectation propagation yields an iterative IC-LMMSE MIMO receiver," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, 2011, pp. 1–6. DOI:10.1109/GLOCOM.2011.6133608

[2] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1754 –1765, 2011. DOI:10.1109/JSSC.2011.2144470

[3] D. Auras, R. Leupers, and G. Ascheid, "A novel reduced-complexity soft-input soft-output MMSE MIMO detector: Algorithm and efficient VLSI architecture," in *Communications, 2014. ICC '14. IEEE International Conference on*, 2014, in press.

[4] X. Chen, G. He, and J. Ma, "VLSI implementation of a high-throughput iterative fixed-complexity sphere decoder," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 60, no. 5, pp. 272–276, 2013. DOI:10.1109/TCSII.2013.2251954

[5] U. Deidersen, D. Auras, and G. Ascheid, "A parallel VLSI architecture for markov chain monte carlo based MIMO detection," in *Proceedings of the 23rd ACM International Conference on Great Lakes Symposium on VLSI*, ser. GLSVLSI '13. New York, NY, USA: ACM, 2013, pp. 167–172. DOI:10.1145/2483028.2483084. [Online]. Available: http://doi.acm.org/10.1145/2483028.2483084

[6] E. M. Witte, "Efficiency and flexibility trade-offs for soft-input soft-output sphere-decoding architectures," Ph.D. dissertation, RWTH Aachen University, 2013.

[7] Y. Sun and J. Cavallaro, "Trellis-search based soft-input soft-output MIMO detector: Algorithm and VLSI architecture," *Signal Processing, IEEE Transactions on*, vol. 60, no. 5, pp. 2617–2627, 2012. DOI:10.1109/TSP.2012.2187646

[8] D. Patel, V. Smolyakov, M. Shabany, and P. Gulak, "VLSI implementation of a WiMAX/LTE compliant low-complexity high-throughput soft-output k-best MIMO detector," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, May 2010, pp. 593–596. DOI:10.1109/ISCAS.2010.5537524

[9] I. Collings, M. R. G. Butler, and M. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *Spread Spectrum Techniques and Applications, 2004 IEEE Eighth International Symposium on*, Aug 2004, pp. 12–16. DOI:10.1109/ISSSTA.2004.1371654

[10] A. Tomasoni, M. Ferrari, D. Gatti, F. Osnato, and S. Bellini, "A low complexity turbo MMSE receiver for W-LAN MIMO systems," in *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 9, June 2006, pp. 4119–4124. DOI:10.1109/ICC.2006.255726

[11] D. Auras, R. Leupers, and G. Ascheid, "Efficient VLSI architectures for matrix inversion in soft-input soft-output MMSE MIMO detectors," in *Circuits and Systems (ISCAS), Proceedings of 2014 IEEE International Symposium on*, 2014, in press.

[12] D. Auras, D. Rieth, R. Leupers, and G. Ascheid, "Extended abstract: VLSI implementation of linear MIMO detection with boosted communications performance," in *Proceedings of the 24th ACM International Conference on Great Lakes Symposium on VLSI*, ser. GLSVLSI '14, 2014, in press.