Layered Detection and Decoding in MIMO Wireless Systems

Nicholas Preyss and Andreas Burg
School of Engineering
EPF Lausanne, Lausanne, Switzerland
e-mail: {nicholas.preyss, andreas.burg}@epfl.ch

Christoph Studer
Dept. of Electrical and Computer Engineering
Rice University, Houston TX, USA
e-mail: studer@rice.edu

***Abstract*—Iterative detection and decoding (IDD) in multiple-input multiple-output (MIMO) wireless systems is known to achieve near channel capacity. The high computational complexity of IDD, however, poses significant challenges for practical implementations (in terms of circuit area, latency, throughput, and power consumption). While the implementation of the involved detector and decoder circuits have received attention in the literature, only little is known about the efficient combination of both blocks in an IDD architecture. In this paper, we propose a novel iterative receiver schedule, which simultaneously performs detection and decoding on the same code block. This novel IDD approach is referred to as layered detection and decoding (LDD) and achieves lower latency and better performance compared to conventional solutions. Moreover, LDD is able to automatically match the decoding effort to the wide range of different modulation schemes and code rates specified in modern MIMO wireless standards. To demonstrate the efficiency of LDD, we present an extensive case study based on the characteristics of existing reference designs for a soft-input soft-output MMSE detector and an LDPC decoder.**

I. Introduction

Multiple-input multiple-output (MIMO) wireless technology in combination with spatial multiplexing and channel decoding is the key to reliable, high-speed, and spectral efficient wireless communication [3]. Hence, many modern wireless communications standards, such as IEEE 802.11n [4] or 3GPP LTE [5], rely on MIMO technology. Data detection is amongst the main challenges in the design of practical implementations for MIMO wireless systems. In particular, the performance of the receiver critically depends on the employed detection and decoding algorithms, and the best-performing algorithms typically exhibit high computational complexity, which results in a large silicon area and high power consumption.

Iterative detection and decoding is known to achieve near channel capacity in MIMO wireless systems, e.g., [1], [6]. Unfortunately, the increase of computational complexity and latency (and, hence, the effective implementation costs) of IDD circuits by at least the number of performed iterations (compared to non-iterative detection schemes) renders corresponding implementations extremely challenging. Straightforward solutions are able to meet the throughput requirements at the cost of prohibitively large silicon area [7] and also entail a substantial increase in detection latency. This latency increase poses serious restrictions on the practical use, especially in systems that specify short time windows for receive acknowledgments, such as in IEEE 802.11n [4].

Furthermore, the fact that modern wireless standards specify a large number of modulation and coding schemes (MCSs) renders the design of efficient (with respect to throughput, latency, and power consumption) iterative architectures a challenging task.

While the implementation of the detector and decoder blocks (constituting the basis of iterative MIMO receivers) have recently gained attention in the literature, e.g., [1], [2], [7]–[10], only little is known about how to efficiently combine both blocks in a practical IDD architecture.

1) Contributions: In this paper, we propose a novel architecture for iterative detection and decoding (IDD) in MIMO wireless systems. We start by analyzing two conventional IDD architectures to highlight the challenges of building efficient IDD receivers for the large number of MCS in modern wireless systems. To overcome these challenges, we propose a novel schedule for iterative receivers, referred to as layered detection and decoding (LDD). We show that LDD is able to substantially simplify the task of matching the throughput of the detector and decoder unit, while being able to achieve lower latency and better performance than conventional IDD schemes. We finally demonstrate the superiority of LDD compared to existing solutions by presenting simulation results based on ASIC-implementation characteristics of existing detector and decoder reference designs.

2) Notation: Matrices are set in boldface capital letters, vectors in boldface lowercase letters. The superscript ^{H} denotes the conjugate transpose and $\mathbf{I}_M$ is the $M \times M$ identity matrix. $P[\cdot]$ stands for probability and $E[\cdot]$ for expectation.

II. Schedules and Architectures for Iterative MIMO Receivers

Iterative detection and decoding in MIMO wireless systems borrows the ideas of turbo decoding [11]. Specifically, relia-

The work of N. Preyss was supported by the Swiss National Science Foundation (SNSF) under Grant PP002-119057. The work of C. Studer was supported in part by the SNSF under Grant PA00P2-134155.

The authors would like to thank S. Fateh and D. Seethaler for the SISO MMSE-PIC detector implementation [1], and C. Roth and P. Meinerzhagen for the LDPC decoder implementation [2].

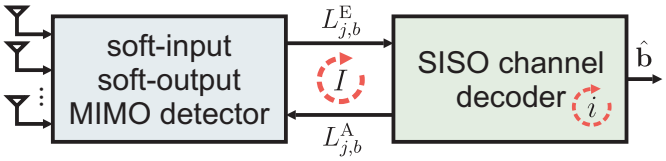


Fig. 1. Iterative MIMO receiver consisting of a soft-input soft-output MIMO detector and a (SISO) channel decoder block.

bility information for each coded bit is iteratively exchanged between a soft-input soft-output (SISO[1]) detector and a SISO channel decoder (see Figure 1). In each iteration, the SISO detector computes such reliability information in the form of log-likelihood ratios (LLRs) for each coded bit $x_{j,b}$ as [1], [6]

$$L^{\mathrm{E}}_{j,b} = \log\left(\frac{P[x_{j,b}=0|\mathbf{y}]}{P[x_{j,b}=1|\mathbf{y}]}\right) - L^{\mathrm{A}}_{j,b}, \tag{1}$$

where $L^{\mathrm{A}}_{j,b}$ designates a-priori information obtained from the SISO channel decoder (e.g., an LDPC decoder) and b stands for the index of the coded bit in the jth spatial stream. The computed LLR values $L^{\mathrm{E}}_{j,b}$ are then fed to the SISO channel decoder, which computes a new set of a-priori LLRs $L^{\mathrm{A}}_{j,b}$ that are used by the SISO detector in the next iteration. This iterative exchange of LLR-values between both blocks successively improves the reliability of the decoded bits. After a given number of *outer iterations*, denoted by I, the SISO channel decoder finally computes estimates $\hat{\mathbf{b}}$ for the transmitted information bits $\mathbf{b}$.

A. Architectures for the Conventional IDD Schedule

Non-iterative receivers ($I = 1$) resemble a coarse grained pipelined architecture consisting of two stages, where the first stage corresponds to a soft-output detector and the second stage to the channel decoder. In such an architecture, the overall throughput is limited by the maximum run-time of either the detector or the decoder unit, i.e., we have

$$T_{\mathrm{non}} = \frac{C}{\max\{t_{\mathrm{detector}}, t_{\mathrm{decoder}}\}}, \tag{2}$$

where C denotes the code-word size, t_{detector} stands for the time required by the detector to compute the LLR values (1), and t_{decoder} is the time required by the channel decoder to compute a set of new a-priori LLRs (and the estimates for the transmitted bits). In the following, we refer to both quantities t_{detector} and t_{decoder} as the *runtimes* of the two units.[2]

Due to the lack of a-priori reliability information, the initial detection phase can be performed by using a dedicated soft-output-only detector, which is, in general, less complex than a SISO detector. Hence in practical systems, the time required for the initial detection of the receive vectors will always be significantly shorter than the time required for decoding; the throughput is typically limited by t_{decoder}.

[1]In the remainder of the paper, SISO exclusively refers to "soft-input soft-output," rather than the conventional meaning "single-input single-output."

[2]The actual values of the two quantities heavily depend on the used algorithm and its implementation. Moreover, the runtimes can vary with the code rate, the modulation schemes, the number of antennas, and for certain algorithms, even with the channel and noise realization.

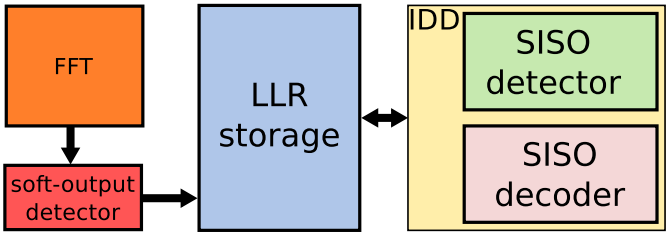


Fig. 3. IDD architecture in the context of a typical MIMO receiver.

The resulting processing latency is more difficult to characterize. For a thorough analysis, one has to consider the detector and decoder in the context of the entire receiver (see Figure 3). The MIMO symbol vectors for the initial detection phase are not available at once, but essentially arrive at a certain rate (e.g., delivered by an FFT in OFDM-based communication systems). Therefore, the initial detection phase is performed using a dedicated soft-output detection unit while the receive vectors keep arriving at a certain rate.

For the sake of simplicity of exposition, we ignore the processing latency required by the first detection phase (i.e., by the soft-output detector) and solely focus on the latency between the availability of the initially detected LLRs for the entire codeword and the generated hard-output estimates. Hence, in the non-iterative case the latency equals the time required by the decoding unit, which is given by $L_{\mathrm{non}} = t_{\mathrm{decoder}}$.

The design of efficient non-iterative MIMO receivers (in terms of throughput and latency) essentially amounts to minimizing t_{decoder}. For iterative receivers, however, the presence of a feedback path across the two-stage detector/decoder pipeline renders corresponding hardware-efficient designs challenging. We next analyze two architectures for the conventional IDD schedule and highlight the associated design challenges.

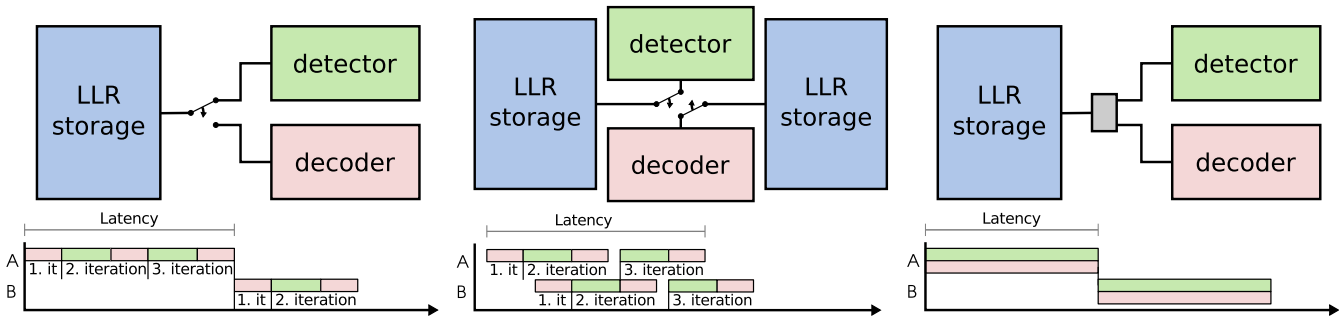
1) Serial Architecture: Figure 2(a) depicts a straightforward design for an IDD receiver employing the conventional schedule as proposed in [7]. A shared memory (used for storing the LLR-values) is connected to both, the detector and the decoder unit. One code-word block is processed in an alternating fashion in both units. Since fully-sequential iterative detection and decoding introduces a feedback loop across the two pipeline stages, the resulting throughput is not only reduced linearly with the number of iterations (compared to a non-iterative receiver), but also affected by the *total* latency of the feedback loop. Specifically, the throughput of this architecture corresponds to

$$T_{\mathrm{ser}} = \frac{C}{(I-1)\,t_{\mathrm{detector}} + I\,t_{\mathrm{decoder}}}. \tag{3}$$

The timing diagram shown in Figure 2(a) illustrates this behavior. The latency associated with the serial architecture behaves similarly and increases linearly in the number of iterations as

$$L_{\mathrm{ser}} = (I-1)\,t_{\mathrm{detector}} + I\,t_{\mathrm{decoder}}.$$

In addition to the rather poor throughput and latency behavior of the serial architecture, it is important to realize that one of the two units in this architecture is always idle. Hence, the serial architecture is highly sub-optimal from a resource-utilization point-of-view.



(a) Serial Architecture (b) Ping-pong Architecture (c) Layered Schedule/Architecture

Fig. 2. High-level architecture overview and corresponding time diagrams for the three considered iterative MIMO-receiver architectures.

2) Ping-Pong Architecture: An architecture that uses pipeline interleaving [12], to process two different sets of codewords within the two pipeline stages as proposed in [7], is shown in Figure 2(b). This approach is able to partially mitigate the resource-utilization issue of the serial architecture. Specifically, while the LLR values associated with codeword A are processed in the detector, the codeword B is processed concurrently in the decoder unit. The interleaving of two code-word blocks allows to utilize both units simultaneously, which increase the throughput (compared to the serial schedule) to

$$T_{\text{pp}} = \frac{C}{I \max\{t_{\text{detector}}, t_{\text{decoder}}\}}.$$

However, to achieve full hardware utilization, the runtime of the detector and decoder unit t_{detector} and t_{decoder} must be matched. A mismatch between both runtimes forces one unit into an idle phase, which degrades the throughput.

We emphasize that architectural transforms applied to the detector and decoder units enable one matching the runtimes for a given MCS.[3] However, since the runtimes of detectors and decoders typically depend on the used MCS, it is very difficult to achieve good matching for the large number of modes specified in modern wireless communication standards. This fact is further aggravated if successive blocks use different modulation and coding schemes. In that case, the runtimes must further match across different MCS, which is, in general, difficult to achieve. Hence, the effective throughput realized by ping-pong-based architectures is limited by the worst-case accuracy of the matching achieved for (and across all) specified modulation and coding schemes.

While the ping-pong architecture is able to improve upon the throughput of the serial architecture, its latency

$$L_{\text{pp}} = (2I - 1) \max\{t_{\text{detector}}, t_{\text{decoder}}\}$$

is typically higher than that of the serial architecture, i.e., $L_{\text{pp}} \geq L_{\text{ser}}$. In particular, the mismatch between the runtime of the detector and the decoder is the reason for the increased latency compared to the serial schedule (see the idle phases between processing one code-word block in Figure 2(b)). Note that equality between both schedules can only be achieved by perfectly matching the runtimes of both units.

[3]Runtime matching can be achieved on different levels, such as on architecture level, by using, e.g., replicating or pipelining, or during synthesis of the circuit by using different timing constraints.

B. Throughput/Area Trade-off

The drastically decreased throughput of IDD compared to non-iterative MIMO decoding can be addressed (partially) by instantiating more detection and decoding units. A straightforward approach is to unroll the iterative loop into a pipeline of multiple pairs of detector and decoder units. In the extreme case, i.e., by unrolling all iterations into a single detection and decoding pipeline, one achieves a throughput corresponding to

$$T_{\text{unroll}} = \frac{C}{\max\{t_{\text{detector}}, t_{\text{decoder}}\}},$$

which is equivalent to that of a non-iterative receiver T_{non}. This brute-force approach, however, yields no improvement in terms of the latency and also comes at a substantial overhead in terms of silicon area (growing roughly linearly in the number of outer iterations I). To overcome the issues associated with the serial and ping-pong architecture, we next propose a novel schedule for iterative detection and decoding in MIMO wireless systems that lends itself to more efficient hardware implementations.

C. Layered Detection and Decoding (LDD)

The key idea of *layered detection and decoding* (LDD) is to get rid of the sequential dependency between detection and decoding altogether. LDD is not merely another architecture option for conventional IDD schedules, but a new schedule of its own. This concept is similar to the one proposed in [13] for joint decoding of MIMO space-time block-codes (STBC) and channel codes. With LDD, the SISO detector and channel decoder process the *same* block of LLR values *simultaneously* (see Figure 2(c)). Since both units can now operate independently and in parallel without requiring to be synchronized, the utilization of the detector and decoder units can be maximized without the need of matching the respective runtimes. Since LDD avoids the notion of iterations, one can get rid of the strict dichotomy between the SISO detector and the channel decoder that cases the rather long latency associated with IDD.

After an initial set of LLR values is computed by the soft-output detector (see Figure 3), the SISO detector and channel-decoder unit simultaneously access and process the LLR values stored in the (shared) LLR memory. The simultaneous detection and decoding of the *same* LLR values inevitably

leads to data-contention issues. In particular, information is potentially lost due to overlapping write-backs, caused by the parallel operation of the detector and the decoder units. In order to mitigate the detrimental effect of such data contentions, we propose to update the LLR values in the shared memory using an incremental LLR-value feedback strategy. This idea is inspired by the layered message-passing schedule proposed for LDPC decoding [14]. Specifically, let $L_{i,b}[k]$ be the a-posteriori LLR after the kth detection run generated by either the detector or the decoder unit. The instantaneous LLR value stored in the memory is updated using the following quantity

$$\Delta_{i,b}[k] = L_{i,b}[k] - L_{i,b}[k-1].$$

The LLR update amounts to adding $\Delta_{i,b}[k]$ to the LLR value stored at location i, b, which may have changed in the meantime (e.g., by processing in the other unit). The proposed incremental LLR update assures that all LLR updates are accounted for. Since all operations of the SISO detector and channel decoder are performed on the basis of (recently) updated LLR-values, the proposed layered decoding schedule generally leads to a faster exchange of reliability information between the detector and the decoder.

III. Case Study: Evaluation Methodology and System Description

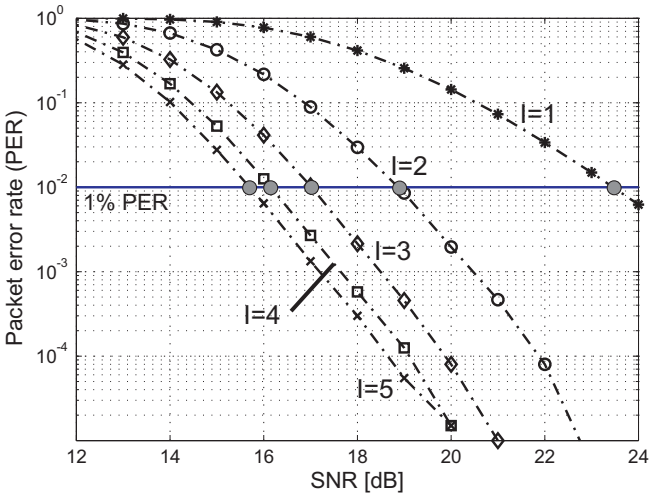
For evaluating the performance/complexity trade-offs of different iterative receiver architectures and to demonstrate the advantages of LDD (see Section IV), we rely on a case-study for a system that combines spatial multiplexing with LDPC codes. The assumptions on complexity and latency of the involved hardware units (detector and decoder) are based on state-of-the-art ASIC reference implementation results extracted from the literature [1], [2] .

In order to measure the hardware complexity by the detector and decoder unit in a way that is agnostic about the variety of architectural transformations that maintain the area-delay product, we consider a complexity measure defined as

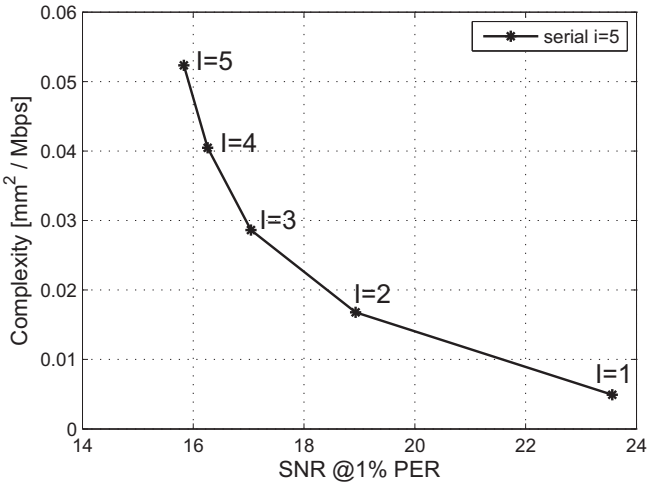
$$\text{complexity} = \frac{\text{circuit area}}{\text{throughput}} \quad [\text{mm}^2/\text{Mbps}]. \tag{4}$$

In the remainder of the paper, the throughput always relates to an LDPC decoder running at a speed of 280 MHz and a codeword size of 1296 bits. The reference MMSE detector is assumed to be clocked internally at twice the rate of the base clock. All statements regarding clock cycles are made with respect to the slower clock frequency of the decoder unit. The circuit area is derived from the respective ASIC implementations in a 90 nm CMOS process [1], [2].

Note that the complexity measure Figure 4 is directly related to the computational effort to perform I outer and, in the case of an LDPC decoder, i inner iterations. Both parameters determine also the achievable packet error-rate (PER) and therefore adjust the the performance/complexity trade-off. To simplify the comparison of different IDD architectures and schedules, we consider the minimum SNR that is necessary to achieve a 1% PER (see Figure 4(a)). Defining a PER target



(a) PER performance for different numbers of iterations I.



(b) Performance/complexity trade-off.

Fig. 4. Illustration of the performance/complexity trade-off of iterative detection and decoding in MIMO wireless systems using the serial schedule. (Simulated for a 1296 bit codeword with 16-QAM.)

allows us to study the necessary hardware complexity for a given PER-performance requirement (see Figure 4(b) for an illustration of this performance measure).

A. Iterative MIMO System

We consider a spatially-multiplexed MIMO system with M_{T} transmit and $M_{\mathrm{R}} \geq M_{\mathrm{T}}$ receive antennas. At the transmit-side, the information bits $\mathbf{b}$ are encoded (e.g., using an LDPC code) and the resulting encoded bit-stream $\mathbf{x}$ is mapped to a sequence of transmit vectors $\mathbf{s} \in \mathcal{C}^{M_{\mathrm{T}}}$, where Q designates the used constellation of size 2^Q. Each transmit vector $\mathbf{s}$ is associated with $M_{\mathrm{T}}Q$ coded bits $x_{j,b} \in \{0,1\}$, $j = 1, \ldots, M_{\mathrm{T}}$, $b = 1, \ldots, C$. The input-output relation of the MIMO channel is given by $\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}$, where $\mathbf{H} \in \mathbb{C}^{M_{\mathrm{R}} \times M_{\mathrm{T}}}$ models the MIMO channel, $\mathbf{y} \in \mathbb{C}^{M_{\mathrm{R}}}$ corresponds to the receive vector, and $\mathbf{n} \in \mathbb{C}^{M_{\mathrm{R}}}$ models zero-mean Gaussian noise with variance N_0 per dimension. The iterative receiver is illustrated in Figure 1. The PER simulations shown in the remainder of

the paper are carried out in a symmetric MIMO system with $M_{\mathrm{T}} = M_{\mathrm{R}} = 4$ and a TGn type-C channel model [15] is used.

B. SISO MMSE-PIC Detector

The SISO MMSE-PIC detection algorithm proposed in [1] is a high-performance low-complexity detection algorithm whose complexity scales only with $\mathcal{O}(M_T^3)$ for symmetric systems with $M_{\mathrm{R}} = M_{\mathrm{T}}$ and does not depend on the channel and noise realization, which is in stark contrast to the LSD and STS-SD algorithms in [6], [16].

1) Algorithm Summary: The algorithm computes the LLR-values in several steps [1]. First, the Gram matrix $\mathbf{G} = \mathbf{H}^H\mathbf{H}$ and the matched-filter output $\mathbf{y}^{\mathrm{MF}} = \mathbf{H}^H\mathbf{y}$ are precomputed to avoid redundant computations. Then, soft-symbols for each spatial stream $j = 1, \ldots, M_{\mathrm{T}}$, are computed as $\hat{s}_j = E[s_j]$ with the aid of the a-priori LLRs[4] $L_{j,b}^A$; analogously, the algorithm computes the variances $V_j = \mathrm{Var}[s_j]$ associated with the soft-symbols. In order to remove interference in each spatial stream j, the algorithm performs PIC as

$$\hat{\mathbf{y}}_j^{\mathrm{MF}} = \mathbf{y}^{\mathrm{MF}} - \sum_{i,j\neq i} \mathbf{g}_i \hat{s}_i, \tag{5}$$

where $\mathbf{g}_j$ denotes the jth column of $\mathbf{G}$. For each vector $\hat{\mathbf{y}}_j^{\mathrm{MF}}$, noise and residual interference is suppressed by an MMSE filter. To this end, one computes the inverse

$$\mathbf{A}^{-1} = (\mathbf{G}\mathbf{\Lambda} + N_0\mathbf{I}_{M_{\mathrm{T}}})^{-1},$$

with $\mathbf{\Lambda}$ being an $M_{\mathrm{T}} \times M_{\mathrm{T}}$ diagonal matrix having $\Lambda_{j,j} = V_j$. Using the rows $\mathbf{a}_j^H$ of $\mathbf{A}^{-1}$, which correspond to the M_{T} MMSE filter vectors, the SISO MMSE-PIC finally computes approximates to the LLR-values in (1) as

$$\tilde{L}_{j,b}^E = \rho_j \min_{a\in\mathcal{Z}_b^{(0)}} |z_j - a|^2 - \rho_j \min_{a\in\mathcal{Z}_b^{(1)}} |z_j - a|^2, \tag{6}$$

where $z_j = \mu_j^{-1}\mathbf{a}_j^H\hat{\mathbf{y}}_j^{\mathrm{MF}}$, $\mu_j = \mathbf{a}_j^H\mathbf{g}_j$, $\rho_j = \frac{\mu_j}{1-V_j\mu_j}$, and the sets $\mathcal{Z}_b^{(0)}$ and $\mathcal{Z}_b^{(1)}$ designate to the subsets of $\mathcal{C}$, where the bth bit is equal to 0 and 1, respectively.

2) Implementation Summary: The SISO MMSE-PIC algorithm as summarized above can efficiently be implemented in VLSI using the parallel and (coarse-grained) pipelined architecture proposed in [1]. This architecture computes a set of new LLRs for each receive vector $\mathbf{y}$ every 18 clock cycles, independent from the antenna setup and the modulation scheme. The resulting maximal throughput corresponds to 757 Mb/s (per iteration) at only 1.5 mm^2 when implemented in 90 nm CMOS technology. Table I summarizes the key implementation characteristics of this SISO MMSE-PIC detector implementation.

C. LDPC Decoder

For the SISO channel decoder we focus on low-density parity check (LDPC) codes. LDPC codes are linear block codes originally proposed in [17] and have been shown to be able to achieve near channel-capacity in practical systems [18]. Another advantage of LDPC codes is the fact that they do not require an interleaver that scrambles the LLRs between detector and decoder; this is in contrast to systems relying on convolutional or turbo codes requiring an interleaver in order achieve good error-correction performance in the presence of burst errors. In what follows, we focus on the binary quasi-cyclic (QC) LDPC codes as specified in the IEEE 802.11n standard [4], which can be decoded efficiently in VLSI while achieving high throughput [2].

[4]Note that the SISO MMSE-PIC algorithm exhibits better performance when using the intrinsic a-priori LLRs from the SISO channel decoder rather than the extrinsic LLRs (see [1] for the details).

TABLE I
KEY CHARACTERISTICS OF THE USED DETECTOR AND DECODER CIRCUITS

Algorithm	SISO MMSE-PIC detector [1]	SISO LDPC decoder [2]
CMOS technology	90 nm	90 nm
Clock frequency	560 MHz	280 MHz
Core area	1.5 mm^2	1.77 mm^2
Throughput	746 Mbps	989 Mbps

1) Algorithm Summary: LDPC codes are based on the null space of a large, sparsely populated parity-check matrix $\mathbf{H}$ of dimension $r_H \times c_H$ for which valid code words $\mathbf{x}$ in GF(2) satisfy the parity-check equation $\mathbf{H}\mathbf{x} = \mathbf{0}$. For the codes specified in IEEE 802.11n [4], various code rates are achieved by using 12 different parity-check matrices. More specifically, the standard specifies 3 possible code-block sizes $\{648, 1296, 1944\}$ at 4 different code rates $\{1/2, 2/3, 3/4, 5/6\}$.

Decoding of LDPC codes is commonly carried out by an iterative algorithm known as belief propagation or message passing [19]. Here, reliability information is exchanged between two types of nodes, namely parity and check nodes, in the sparse graph specified by $\mathbf{H}$. The iterative nature of LDPC decoding introduces a second level of iterations in the iterative MIMO receiver (cf. Figure 1). We refer to the iterations inside the LDPC decoder *inner iterations* and denote the maximum number of inner iterations by i (cf. Figure 1).

2) Implementation Summary: LDPC decoding for QC-LDPC codes, as specified in IEEE 802.11n [4], can efficiently be implemented in VLSI using the layered architecture proposed in [2] (see, e.g., [14] for details about layered processing). This architecture consists of two main units that implement the operations required by the check and parity nodes, whereas the connections in the sparse graph are realized using programmable permutation units, rather than using a hard-wired network. The advantage of this architecture is its flexibility to support all specified code rates, while only requiring roughly 110 clock cycles per inner iteration.[5] The maximal throughput achieved by the decoder ASIC in [2] corresponds to 989 Mb/s at $i = 5$, while only requiring 1.77 mm^2 in 90 nm CMOS technology. Table I summarizes the key characteristics of this QC-LDPC decoder implementation.

IV. CASE STUDY: RESULTS

We now compare the performance, complexity, and latency of iterative MIMO-receiver architectures based on the conven-

[5]The exact number of clock cycles depends on the used code rate.

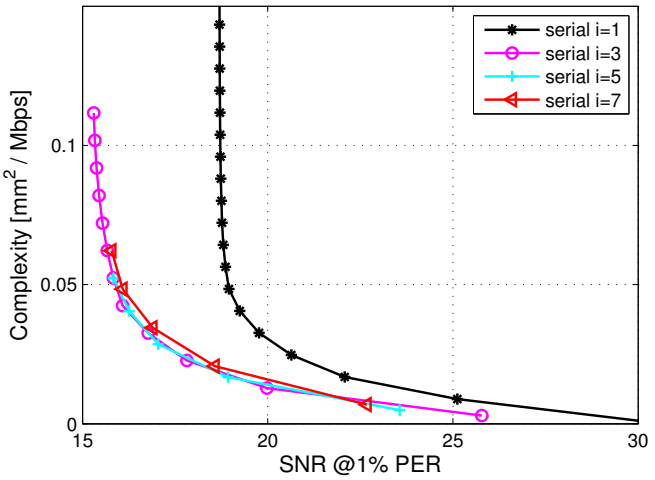


Fig. 5. Performance/complexity trade-off of the serial IDD schedule. Each simulation point corresponds to the number of outer iterations I, whereas each curve is associated to a different number of inner iterations i. (Simulated for a 1296 bit codeword with 16-QAM.)

tional sequential schedules and the proposed layered detection and decoding (LDD) approach.

A. Performance Characteristics of the sequential Schedule

1) Serial Architecture: As detailed in Section II-A1, the throughput of the serial architecture given in (3) is directly determined by the number of outer iterations I and the runtimes of both, the detector and decoder. The runtime of the LDPC decoder is given by $t_{\text{LDPC}} = i \cdot t_{\text{iteration}}$, where i is the number of inner iterations and $t_{\text{iteration}}$ is the time required by a single iteration. The number of inner and outer iterations can be adjusted individually. Different combinations of I and i can result in similar hardware complexity, but with different SNR performance. Hence, it is of practical interest to identify those configurations of inner and outer iterations, which correspond to the Pareto-optimal solution for each complexity/performance target.

Figure 5 shows this performance/complexity trade-off for the serial architecture. It is interesting to see that except for a single inner iteration, i.e., $i = 1$, all other choices of inner and outer iterations exhibit a similar trade-off characteristic. In other words, many different combinations of inner and outer iterations lie on the Pareto-optimal curve. Hence, there is no *single* best combination of inner and outer iterations. We furthermore see from Figure 5 that for a large number of outer iterations, the complexity starts to increase without offering a significant gain in terms of SNR performance. This observation implies that using a large number of iterations is not beneficial in practice.

2) Ping-Pong Architecture: The results for the serial schedule shown in Figure 5 suggest that a given performance target can be achieved with many different combinations of inner and outer iterations. From the discussion in Section II-A2, we know that matching the runtime of the decoder and the detector minimizes the complexity (and the latency) for the ping-pong architecture. Hence, one can take advantage of the plurality

TABLE II
RUNTIMES OF THE SISO MMSE-PIC DETECTOR FOR DIFFERENT MODULATION AND ANTENNA CONFIGURATIONS (1296 BIT CODEWORD)

Antennas:	2×2		4×4	
	Vectors	Cycles	Vectors	Cycles
QPSK	324	2961	162	1503
16-QAM	162	1503	81	774
64-QAM	108	1017	54	531

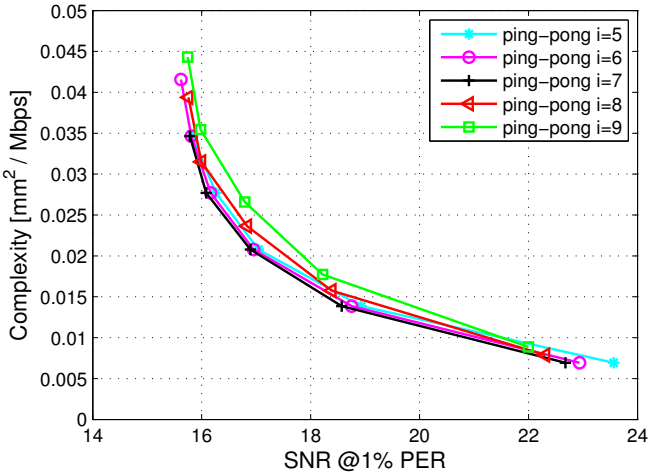
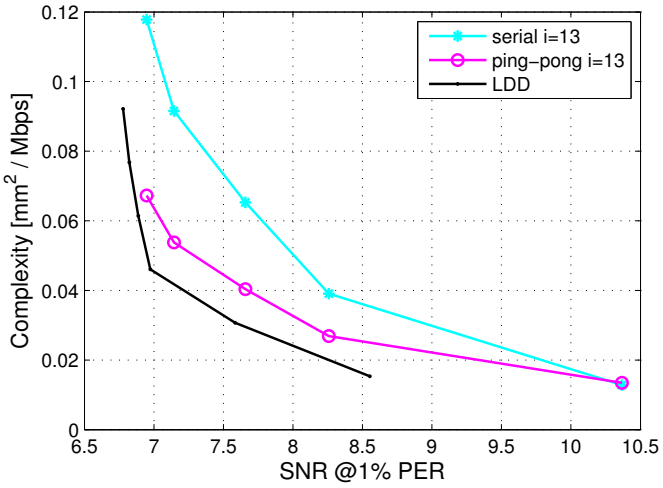


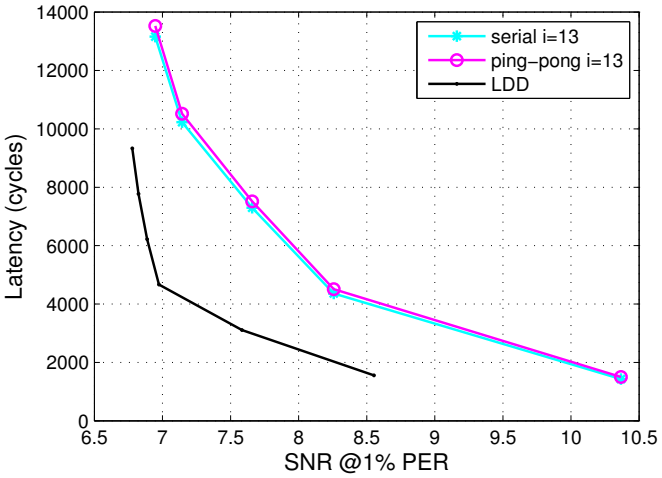
Fig. 6. Performance/complexity trade-off of the ping-pong IDD schedule. Each simulation point corresponds to the number of outer iterations I, whereas each curve is associated to a different number of inner iterations i. (Simulated for a 1296 bit codeword with 16-QAM)

of different Pareto-optimal configurations by choosing the parameters that best match the runtimes of both units; this approach not only minimizes the complexity but also reduces the latency of the ping-pong architecture.

To illustrate the idea of matching both runtimes, consider the following example. For 16-QAM modulation in a 4×4 antenna setup, each 1296 bit LDPC codeword is associated with a total number of 81 symbol vectors. For this configuration, the SISO MMSE-PIC implementation requires 774 clock cycles to detect all 81 symbol vectors. The number of clock cycles required by the SISO MMSE-PIC for other MCS are summarized in Table II. The best-possible runtime match can now be achieved by using $i = 7$ inner iterations, for which the LDPC decoder requires roughly 770 clock cycles. The simulation of the performance/complexity trade-off for a ping-pong architecture with different number of inner iterations can bee seen in Figure 6. As expected from the runtime matching calculations shown above, the configuration with $i = 7$ inner iterations achieves the best hardware utilization and requires the lowest complexity. Moreover, we see that the ping-pong schedule nearly halves the complexity compared to the serial architecture shown in Figure 5. Nevertheless, the ping-pong schedule does not resolve the issue of having a very long processing latency, which is even larger than that of the serial schedule (see Figure 7(b)).

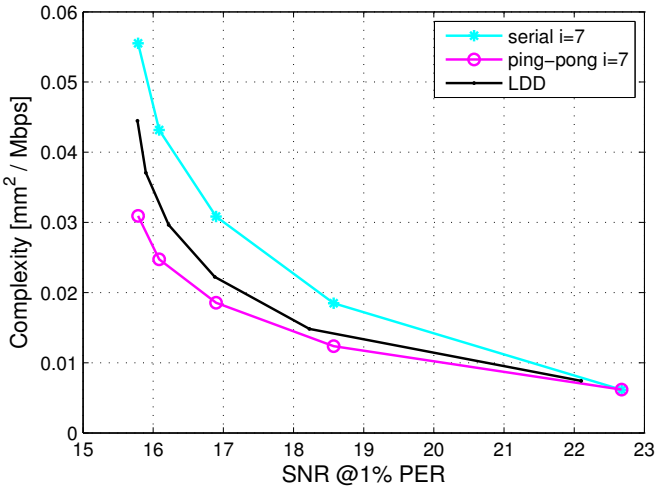


(a) Performance/complexity trade-off.

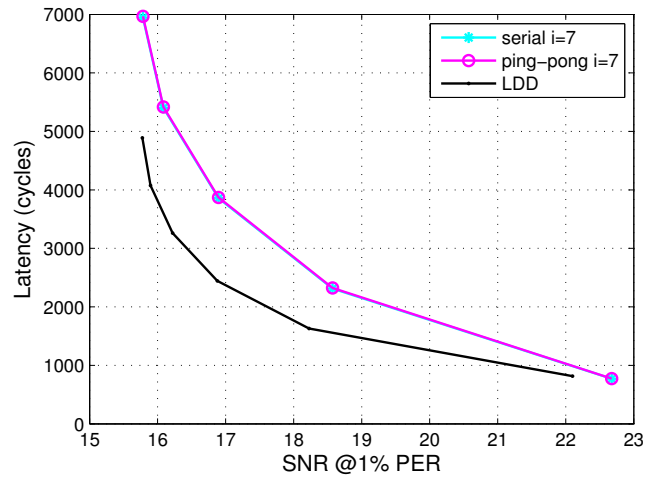


(b) Performance/latency trade-off.

Fig. 7. Trade-off comparison of conventional serial and ping-pong architectures to LDD using QPSK modulation.



(a) Performance/complexity trade-offs.



(b) Performance/latency trade-offs.

Fig. 8. Trade-off comparison of conventional serial and ping-pong architectures to LDD using 16-QAM.

B. Performance of LDD

The proposed LDD schedule is particularly suited for iterative receivers based on LDPC decoders. In that case, the fact that no interleaving is required by the employed LDPC decoder simplifies the concurrent memory access of the detector and the decoder and enables its efficient implementation.

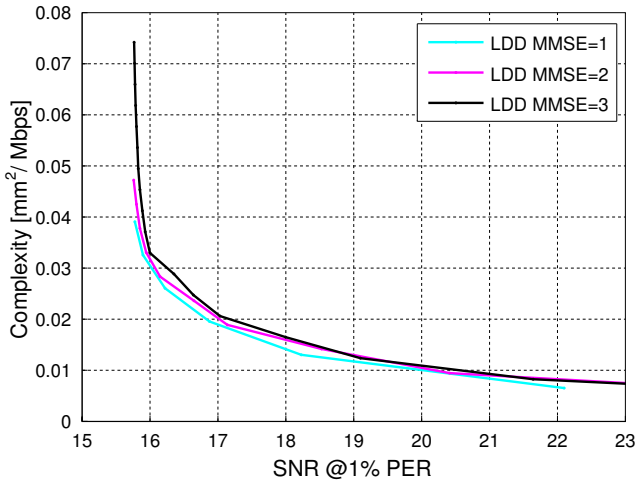
1) QPSK Modulation: Figure 7 demonstrates the complexity and latency advantages of LDD compared to that of the serial and ping-pong architectures for QPSK modulation. One can observe that LDD substantially outperforms both sequential IDD architectures with respect to the complexity (see Figure 7(a)). One drawback of LDD is the fact that due to data contentions and differential LLR updates, a more sophisticated memory-access scheme is required, which leads to an overhead in terms of circuit area and power consumption. Note that the increase in performance is observed despite of this drawback. LDD also slightly improves upon conventional schedules in terms of the achievable SNR performance. More importantly, LDD significantly reduces the processing latency compared to that of both conventional IDD schemes (see Figure 7(b)).

2) 16-QAM: We next consider the simulation results for 16-QAM shown in Figure 8(a). The observed complexity achieved by LDD is comparable to that of the conventional IDD schedule with a ping-pong architecture. However, LDD significantly reduces the latency compared to the architectures of the conventional schedule; we observe a reduction to less than 70 % of the latency of the conventional schedule. Hence, LDD can be considered as an effective strategy for latency-constrained MIMO receivers.

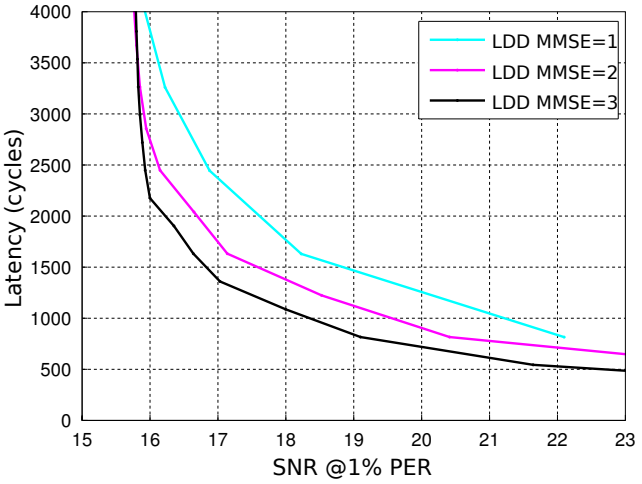
C. Latency/Complexity Trade-off

A short receiver latency is especially critical in wireless systems, in which the MAC layer mandates receive acknowledgments within a short time window. A corresponding example is IEEE 802.11n [4], which requires to transmit an acknowledgment of each received frame within only 10 µs.

In such systems, it is likely to happen that an IDD receiver with a conventional schedule cannot achieve the full perfor-



(a) Performance/complexity trade-offs.



(b) Performance/latency trade-offs.

Fig. 9. LDD complexity and latency with different ratios between MMSE-PIC and LDPC operations for 16-QAM.

mance gain offered by iterative detection and decoding, as it is unable to carry out a sufficiently large number of iterations. The LDD receiver can provide significantly better performance figures in such a scenario. The proposed LDD architecture offers the opportunity to even further decrease the latency in return for a slightly higher complexity. In all previous simulations, we exclusively assumed a *single* SISO MMSE-PIC detector instance. However the detection of the symbol vectors is a very self-contained problem and can therefore be parallelized easily. Since LDD avoids the tedious task of matching the runtimes of the detector and decoder, one can easily instantiate multiple detector units.

While this technique also increases the throughput, its behavior in combination with LDD is fundamentally different to just having multiple instances of the same circuit. From Figure 9(b) we can observe that the use of multiple MMSE-PIC decoder units leads to considerable reductions in terms of the processing latency.

V. Conclusions

We have proposed a novel schedule and architecture for iterative detection and decoding (IDD) in MIMO wireless systems, referred to as layered detection and decoding (LDD). LDD simultaneously performs detection and the decoding on the same code block, which significantly reduces the processing latency compared to existing IDD architectures. Moreover, LDD is able to achieve comparable or even better error-rate performance than standard IDD architectures, while avoiding the tedious task of runtime matching between the SISO detector and channel decoder. The proposed LDD scheme is particularly well suited for wireless standards which mandate stringent latency constraints, such as IEEE 802.11n.

References

[1] 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*, no. 99, 2011.

[2] C. Roth, P. Meinerzhagen, C. Studer, and A. Burg, "A 15.8 pj/bit/iter quasi-cyclic LDPC decoder for IEEE 802.11n in 90 nm CMOS," in *Proc. IEEE Asian Solid State Circuits Conf. (A-SSCC)*, 2010, pp. 1–4.

[3] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge Univ. Press, 2003.

[4] *IEEE Draft Standard; Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications; Amendment 4: Enhancements for Higher Throughput*, IEEE P802.11n/D3.0, Sep. 2007.

[5] *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding (Release 9)*, 3GPP Organizational Partners TS 36.212, Rev. 8.3.0, May 2008.

[6] B. M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Comm.*, vol. 51, no. 3, pp. 389–399, Mar. 2003.

[7] C. Studer, "Iterative MIMO decoding: Algorithms and VLSI implementation aspects," Ph.D. dissertation, ETH Zürich, Switzerland, Series in Microelectronics, vol. 202, Hartung-Gorre Verlag Konstanz, 2009.

[8] E. M. Witte, F. Borlenghi, G. Ascheid, R. Leupers, and H. Meyr, "A scalable VLSI architecture for soft-input soft-output single tree-search sphere decoding," *IEEE Trans. Circ. Systems II*, vol. 57, no. 9, pp. 706–710, Sept. 2010.

[9] C. Studer, C. Benkeser, S. Belfanti, and Q. Huang, "Design and implementation of a parallel turbo-decoder ASIC for 3GPP-LTE," *IEEE J. Solid State Circuits*, vol. 46, no. 1, pp. 8–17, Jan 2011.

[10] C. Studer, S. Fateh, C. Benkeser, and Q. Huang, "Implementation trade-offs of soft-input soft-output MAP decoders for convolutional codes," *IEEE Trans. Circ. Systems I*, vol. 59, no. 12, Dec. 2012, Early Access.

[11] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proc. IEE Int. Conf. on Comm. (ICC)*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[12] H. Kaeslin, *Digital Integrated Circuit Design: From VLSI Architectures to CMOS Fabrication*. Cambridge Univ. Press, 2008.

[13] J. Yang, C. A. Nour, and C. Langlais, "Joint factor graph detection for LDPC and STBC coded MIMO systems: A new framework," in *Proc. 6th Int Turbo Codes and Iterative Information Processing (ISTC) Symp*, 2010, pp. 122–126.

[14] E. Sharon, S. Litsyn, and J. Goldberger, "Efficient serial message-passing schedules for LDPC decoding," *IEEE Trans. Inf. Theory*, vol. 53, no. 11, pp. 4076–4091, 2007.

[15] *TGn channel models*, IEEE 802.11 TGn Std. 904, Rev. 43, May 2004.

[16] C. Studer and H. Bölcskei, "Soft–input soft–output single tree-search sphere decoding," *IEEE Trans. Inf. Th.*, vol. 56, no. 10, pp. 4827–4842, Oct. 2010.

[17] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[18] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Th.*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[19] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Sig. Proc. Mag.*, vol. 21, no. 1, pp. 28–41, Jan. 2004.