This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration

Esther P. Adeva , *Member, IEEE,* and Gerhard. P. Fettweis, *Fellow, IEEE*

***Abstract*—The application of the turbo principle allows to exploit the full potential of MIMO communications, at the cost of increasing the computational effort at the receiver. In the context of soft-input soft-output (SISO) tree search detection, the computation of metric values and of the optimal node order represent two of the most computationally demanding operations. Heuristic approaches may be applied to reduce the complexity, but their accuracy is compromised by the effect that the input *soft information* has on the node ordering. The design of adaptive, good-performing and cost-effective tree search detectors for iterative receivers represents hence a challenging task. To alleviate these complexity and performance loss drawbacks, an efficient MIMO sphere detector realization is proposed in this work. A novel smart-sorting enumeration approach offers a significant gain in terms of throughput (from 40% up to a factor 5) and energy efficiency (up to 80% energy saving in the low SNR regime) with regard to preceding implementations. Owing to the additional low delay and area cost reported, the proposed design represents a very promising candidate towards a fast, accurate, and efficient MIMO detector.**

***Index Terms*—Multiple-input multiple-output (MIMO), soft-input soft-output (SISO) sphere detection (SD), Schnorr-Euchner (SE) enumeration, ASIP architecture, VLSI design.**

I. Introduction

MULTI-ANTENNA detection belongs to the most computationally intensive constituents of the receiver's baseband signal processing, especially regarding spatial-multiplexing transmission. Designing adaptive, good-performing and cost-effective MIMO detectors represents a challenge, particularly concerning high-order systems (i.e., $\geq 4 \times 4$ MIMO configurations with ≥ 64-QAM modulations) in the context of iterative receivers. The turbo principle allows exploiting the full potential of MIMO communications by exchanging *soft-information* between the detector and the channel decoder. This enables enhancing the communication's reliability drastically or, alternatively, reducing the transmit energy substantially while guaranteeing a maximum target error rate. However, this benefit comes at the cost of increasing the receiver's overall complexity. Namely, the detector module has to be able to generate soft output information, as well as to process the soft input data received from the channel decoder. The presence of soft input information (known as *a priori* information) introduces a shuffling effect on the tree node enumeration sequence [1]. This sequence indicates the order in which nodes should be examined, and has a strong effect on the complexity and even the detection accuracy of tree-search algorithms [1]. Determining the ideal order in an accurate and efficient manner represents one of the major challenges in soft-input soft-output (SISO) detection. The Schnorr-Euchner (SE) enumeration is a widely known strategy to find the optimal sequence of symbols (i.e., sorted in ascending order of their metric values). Unfortunately, the complexity of exhaustive SE becomes unmanageable (especially for high-order modulations), since the metrics of all the constellation symbols have to be repeatedly computed and sorted during the tree search. In order to reduce the computational cost, the ideal SE ordering can be approximated by exploiting the geometrical properties of the considered QAM constellation. Some examples are the circular or column-wise zig-zag enumerations employed in [2] and [3], respectively, as well as the sector-based approach proposed in [4]. The latter, so-called search sequence determination (SSD), is a heuristic method which, combined with an estimation of the metric values, reduces the computational effort enormously [4], [1]. A common disadvantage of these geometry-based enumeration strategies is that they neglect the influence of the *a priori* information, consequently constraining the gain of iterative detection-and-decoding, as shown e.g., in [4]. A pragmatic low-complexity solution to this, firstly proposed in [5] and further analyzed in [6], consists in iteratively reusing a list of candidate tree paths. These are obtained by a soft-output detector, without re-running the tree search on every iteration. The gain provided by this approach is however very limited and a high number of candidates has to be collected in order to achieve an acceptable error-rate performance [6]. In [4], [7], [8] and [1], several corrective strategies have been proposed to enable the utilization of the SSD enumeration in iterative systems. While these solutions improve the error-rate performance to some extent, the benefit provided by the turbo principle is still not fully exploited. In [9], two enumeration approaches based either on the channel-state knowledge *or* on the *a priori* information are proposed. A hybrid enumeration strategy deriving from these is presented in [10]. In this case, a sequence of symbols is estimated by means of the SSD approach from [4] and the PAM-like enumeration algorithm from [3]. A second sequence is obtained by computing, sorting, and optionally storing the *a priori* information corresponding to all the constellation symbols at all tree layers. The effective enumeration sequence is then determined during runtime by computing and comparing the metrics of the symbols from both sorted successions and selecting the one with the lowest value. Even though this method approaches the ideal SE ordering, it entails a certain non-negligible complexity resulting from the need to determine two enumeration sequences, in addition to the required compare and sorting operations. To sum up, none of the state-of-the-art approaches provides an efficient and low-complexity solution

E.P. Adeva and G.P. Fettweis are with the Vodafone Chair for Mobile Communications Systems, Technical University Dresden, Dresden, Germany (email: esther.perez, fettweis@tu-dresden.de)

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

to enumerate the symbols correctly in the presence of *a priori* information. In contrast to this, the mechanism proposed in this work exploits the properties of quadrature modulations to provide the optimum enumeration sequence, while saving 95% of the computational effort required by an exhaustive SE search. In order to demonstrate the suitability of the proposed enumeration method, a low-complexity MIMO sphere detector [11] has been considered and analyzed throughout this work. A VLSI architecture concept is additionally presented and evaluated. After introducing the communications system model and the basic principles of MIMO detection in sections II and III, respectively, relevant node enumeration mechanisms are described in section IV. In section VI, an overview of key strategies to enable an efficient detector realization is provided, followed by a detailed description of the proposed architecture (section VII). Lastly, the resulting circuit's characteristics are analyzed and compared to state-of-the-art realizations in section VIII, before summarizing the contributions of this work in section IX.

II. System Model

In order to evaluate the performance of the proposed detection approach, a bit-interleaved coded modulation (BICM) transmission scheme [12] supporting iterative processing at the receiver [13], [14] is considered, as depicted in Fig. 1. The MIMO spatial-multiplexing system has $N_\mathrm{T} = N_\mathrm{R} = 4$ transmit/receive antennas. The coded and interleaved streams $\mathbf{c}$ of bits to be transmitted are mapped onto a vector $\mathbf{x}(\mathbf{c})$ of complex constellation symbols x from a QAM constellation set $\mathcal{X}$ with $Q = 64$ symbols (i.e., $L = 6$ bits per symbol). An uncorrelated, fast-, flat-fading Rayleigh channel model is considered and assumed to be perfectly known at the receiver. The channel is represented by $\mathbf{H} \in \mathbb{C}^{N_\mathrm{R} \times N_\mathrm{T}}$, with entries of a zero mean independent and identically distributed (i.i.d.) gaussian random process of variance 1. An AWGN vector $\mathbf{n} \in \mathbb{C}^{N_\mathrm{R} \times 1}$ comprised of zero-mean i.i.d. gaussian random variables of variance $N_0/2$ per real dimension is added at the receiver. The received signal $\mathbf{y}$ is therefore given by $\mathbf{y} = \mathbf{Hx} + \mathbf{n}$. The receiver is mainly comprised by the complex-valued tuple-search sphere detection algorithm [11] described in section III, in conjunction with a turbo channel decoder. The detector and the decoder are coupled through the corresponding (de-) interleaving blocks and may generate and exchange soft information ($L^\mathrm{Det/Dec}$, $L_\mathrm{a}^\mathrm{Det/Dec}$) in iterative fashion, in order to improve the communications error-rate performance cooperatively [13]. In the following, the notation $*^\mathrm{Det/Dec}$ will be dropped for the sake of simplicity. A simulation setup equivalent to the one applied in e.g., [5], [6] is considered[1] to ease comparing the results with previous works.

III. Tuple-Search Sphere Detection

The task of a MIMO detector is the determination of the most likely sent vector of bits $\mathbf{c}$, as well as of reliability information for each bit $c_{m,l}$. This can be accomplished by calculating the so-called log-likelihood ratios (LLRs)

$$L\left(c_{m,l}|\mathbf{y}\right) = \ln\left(\frac{P\left(c_{m,l} = +1|\mathbf{y}\right)}{P\left(c_{m,l} = -1|\mathbf{y}\right)}\right) \quad (1)$$

by means of a tree search detection strategy. The main idea behind tree search approaches is to represent the set $\mathcal{V}$ of all likely transmitted symbol vectors as a weighted tree structure, as exemplified in Figure 2. The number of levels of the tree is defined by the amount of MIMO layers or, equivalently, of transmit antennas N_T (assuming spatial multiplexing with one transmitted symbol stream per antenna). Every tree layer i comprises $2^{L(N_\mathrm{T}-i)}$ nodes, each representing a constellation symbol $x \in \mathcal{X}$. A set of Q *child* nodes descend from each *parent* node into the next layer $(i-1)$. The tree root is defined by the topmost layer ($i = N_\mathrm{T}$), while the *leaf* nodes compose the lowest layer ($i = 0$). Each of the tree *paths* (i.e., tree edges connecting parent and child nodes from the root to a leaf) is weighted by a metric λ. Instead of searching the complete set $\mathcal{V}$, tree-search detectors only consider a subset $\mathcal{L} \subset \mathcal{V}$ of candidates. By additionally applying the *max-log* approximation [15], the *maximum a posteriori* (MAP) solution of (1) (i.e., the exhaustive-search solution) is approximated [5] as (2). For the given received MIMO vector $\mathbf{y}$ and the estimated symbol vector $\hat{\mathbf{x}}(\mathbf{c})$ (represented by the vector of bits $\mathbf{c}$), the required metric values λ take then the form in (3).

$$L\left(c_{m,l}|\mathbf{y}\right) \approx -\frac{1}{N_0} \min_{\mathbf{x}\in\mathcal{L}|c_{m,l}=+1} \{\lambda\} + \frac{1}{N_0} \min_{\mathbf{x}\in\mathcal{L}|c_{m,l}=-1} \{\lambda\}. \quad (2)$$

$$\lambda\left(\mathbf{y}, \mathbf{c}, \mathbf{L}_\mathrm{a}\right) = \|\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}(\mathbf{c})\|^2 - \frac{N_0}{2} \sum_{i=0}^{N_\mathrm{T}-1} \sum_{j=0}^{L-1} c_{i,j} L_\mathrm{a}(c_{i,j}), \quad (3)$$

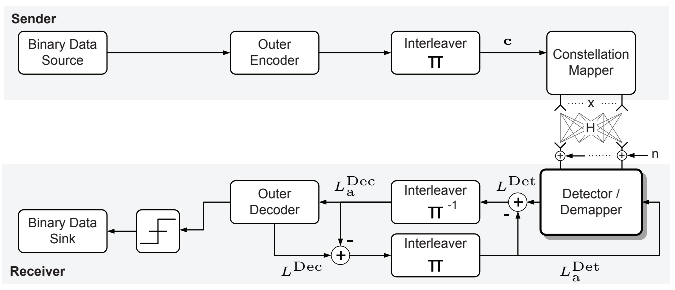


Fig. 1. Communications system model with BICM transmitter and iterative receiver.

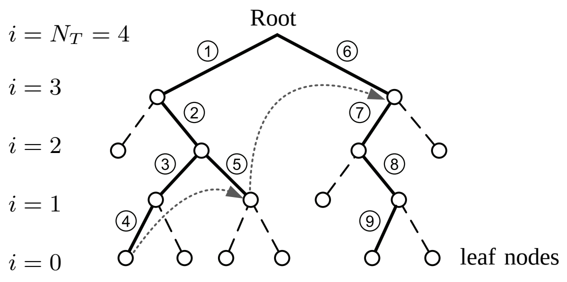


Fig. 2. Tree search example for a BPSK modulation ($L = 1$) and $N_\mathrm{T} = N_\mathrm{R} = 4$ antennas. Dashed lines represent pruned paths.

[1]The turbo channel decoder employs a BCJR (Bahl, Cocke, Jelinek and Raviv) algorithm with $(7_R, 5)$ convolutional codes and 8 internal iterations. A coding rate $R_\mathrm{c} = 1/2$ is applied. An information block size of 9216 bits (including tail bits), Gray mapping and random interleavers are considered.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, "Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration," in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

The vector $\mathbf{L}_\mathrm{a}$ contains the soft-information $L_\mathrm{a}(c_{i,j})$ generated by the channel decoder (i.e., the *a priori* information) for each bit $c_{i,j}$ of vector $\mathbf{c}$. To map all transmit symbols to a tree structure, the channel matrix must be decomposed in a way that a successive dependency among antennas can be established. This transformation can be performed by means of e.g., the QR decomposition (QRD) of the channel matrix $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ is unitary and $\mathbf{R}$ an upper triangular matrix with elements $r_{i,j}$ [16]. The triangular structure of $\mathbf{R}$ allows an ordered layer-wise exploration of the tree from the root to the leaves level. By modifying the received symbols as $\mathbf{y}' = \mathbf{Q}^\mathrm{H}\mathbf{y}$, the Euclidean distance $\|\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}(\mathbf{c})\|^2$ is reformulated as $\|\mathbf{y}' - \mathbf{R}\hat{\mathbf{x}}(\mathbf{c})\|^2$ and the metrics λ in (3) can be recursively calculated by accumulating each layer's contribution as

$$\lambda_i = \underbrace{\lambda_{i+1}}_{\substack{\text{metric from}\\\text{already estimated}\\\text{symbols}}} + \big|\underbrace{y_i''}_{\substack{\text{interference reduced}\\\text{symbol}}} - r_{ii}\hat{x}_i\big|^2 - \underbrace{\frac{N_0}{2}\sum_{j=0}^{L-1} c_{i,j} L_\mathrm{a}(c_{i,j})}_{\substack{\lambda_\mathrm{a}(\hat{x}_i)\\(\textit{a-priori}\text{ information})}}, \tag{4}$$

$$y_i'' = y_i' - \sum_{j=i+1}^{N_\mathrm{T}-1} r_{ij}\hat{x}_j. \tag{5}$$

In (4), the *partial* metric λ_i $(i > 0)$ represents the influence of the upper layers, whereas $\lambda_0 = \lambda(\mathbf{y}, \mathbf{c}, \mathbf{L}_\mathrm{a})$ denotes the total path metric, i.e., the metric corresponding to a complete estimated MIMO symbol vector $\hat{\mathbf{x}}(\mathbf{c})$. The interference among layers is successively suppressed by applying (5). Finding the MAP solution (i.e., the so-called detection *hypothesis*) is not sufficient to generate soft information, since this does not necessarily minimize the two terms in (2). Instead, an exhaustive search for all the $L \cdot N_\mathrm{R}$ required minima (the so-called *counter-hypotheses*) must be performed as well. Since an exhaustive search of all minima entails an impractically high complexity [17], the search space is reduced by applying a sphere detection approach. Sphere detectors introduce a certain constraint R (known as *radius*) to limit the maximum value of the nodes' metrics. All tree nodes whose partial metrics exceed the defined radius value ($\lambda_i > R$) are excluded from the search, as exemplified in Figure 2 by dashed lines. These excluded nodes, as well as the subtrees descending from them (which are also excluded from the search) are said to be *pruned*. From the large variety of existing tree search detection strategies, the tuple search sphere detector (TSD) proposed in [11] has demonstrated to outperform the error-rate-complexity trade-off of existing sphere detection strategies (such as single tree search (STS) [18], list sphere detection (LSD) [19] or K-best detection [20]), while representing a promising approach towards an efficient VLSI realization [21], [22]. The TSD strategy keeps a sorted list (or *tuple*) $\mathcal{T} := \{\lambda_0(\mathbf{c}_1), \lambda_0(\mathbf{c}_2), \ldots, \lambda_0(\mathbf{c}_{T-1})\}$ containing the best T candidate path metrics $\lambda_0(\mathbf{c}_t)$. The sphere radius is defined as the maximum metric in the tuple:

$$R = \max_{\mathbf{c}_t}\{\lambda_0(\mathbf{c}_t)\} = \lambda_0(\mathbf{c}_{T-1}). \tag{6}$$

For the LLR computation, an additional list is employed to store the best candidate metrics found for each of the $L \cdot N_\mathrm{R}$ bits $c_{m,l}$. Additionally applied mechanisms for further reduction of complexity are the sorted QR decomposition (SQRD) [16], the MMSE channel matrix extension [23] and the radius and LLRs clipping [24].

IV. Node Enumeration Strategies

The complexity of the tree search process is strongly influenced by the order in which the constellation symbols descending from a parent node are explored. To avoid spending computational effort on subtrees which will be eventually pruned by the radius constraint, tree paths which are advantageous for (2) (i.e., presenting small metrics) should be explored firstly. Consequently, tree nodes are preferably examined in ascending of their partial metrics. To accomplish this, numerous enumeration methods have been proposed (e.g., [9], [10], [3], [4]) which mainly differ in their complexity and accuracy. None of these state-of-the-art approaches provides, however, an efficient and low-complexity solution to enumerate the symbols correctly in the presence of *a priori* information. In contrast to this, the proposed smart-sorting enumeration with quadrature metric computation (SSE-QMC) strategy [25] guarantees perfect node ordering while keeping an acceptably low complexity, as shown in section VIII. In the following, the exact exhaustive Schnorr-Euchner computation approach, the geometry-based approximation of [4] and the proposed SSE-QMC mechanism are presented.

A. Exact Exhaustive Enumeration

The Schnorr-Euchner (SE) enumeration [26] is a widely known strategy to determine the ideal node ordering, consisting in a sequence of constellation symbols $[x_i^0, x_i^1, \ldots, x_i^{(Q-1)}]$ sorted in ascending order of their partial metrics ($\lambda_i(x_i^0) \leq \lambda_i(x_i^1) \leq \ldots \leq \lambda_i(x_i^{(Q-1)})$). The complexity of the (exhaustively applied) SE method grows exponentially, since the partial metrics of all Q constellation symbols have to be repeatedly computed and sorted for each parent node in the tree. This represents an enormous waste of computational resources, since symbols whose metrics violate the radius constraint $(\lambda_i(x_i^k) > R)$ will not be explored and, consequently, enumerating them is unnecessary.

B. Approximated Enumeration

Disregarding the contribution of the *a priori* information to the metrics, the node ordering is uniquely depending on the Euclidean distances between the (normalized) interference-reduced received signal $y_i''' = y_i''/r_{ii}$ and the constellation symbols:

$$(\Delta_i^k)^2 = \left|y_i''' - \hat{x}_i^k\right|^2, \quad \text{with } k = \{0, \ldots, Q-1\}. \tag{7}$$

The enumeration can be directly visualized on the constellation plane, where geometrical properties can be exploited in order to approximate the ideal SE ordering with a considerably low computational cost. An example of this is the so-called search sequence determination (SSD) method proposed in [4], which divides the constellation space into geometrical decision

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

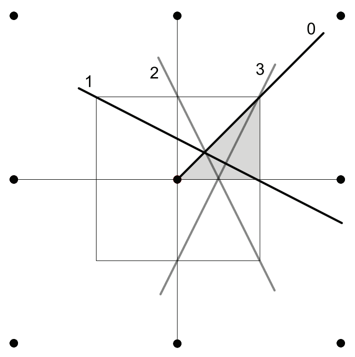


Fig. 3. Section of a 64-QAM constellation. The constellation's space denoted by a grey-shaded area is partitioned into 5 decision regions for the search sequence determination (SSD) approach.

regions (as illustrated in Figure 3). The node ordering is determined by a predefined node succession, which is associated to the decision region where y_i''' is found. The computational complexity can be further reduced by replacing the metric calculation in (4) by a metric estimation (ME) approach based on this sector-aided strategy. The distances Δ_i^k between the constellation symbols and the (normalized) interference-reduced received signal can be replaced by predefined geometrical distances $\Delta_i^{k'}$ [27] [21] between the constellation symbols and fixed reference points z_i^{ref} (such as the geometric centers of the defined decision regions):

$$\begin{aligned} r_{ii}^2(\Delta_i^k)^2 &= r_{ii}^2 \left\|y_i''' - \hat{x}_i\right\|^2 \\ &\approx r_{ii}^2 \left\|z_i^{\mathrm{ref}} - \hat{x}_i\right\|^2 = r_{ii}^2(\Delta_i^{k'})^2. \end{aligned} \tag{8}$$

It is additionally possible to precalculate r_{ii}^2 as well as $r_{ii}^2(\Delta_i^{k'})^2$ in order to simplify the complex-value products in (4) to a single real-value multiplication[2] or even to eliminate these operations completely.

The reduced computational complexity of the SSD and ME mechanisms make them seemingly attractive for hardware implementation, but these strategies also present some drawbacks. Firstly, in iterative scenarios the metric values do not depend only on the Euclidean distances, but also on the contribution of the *a priori* information $\lambda_{\mathrm{a}}(\hat{x}_i)$. Consequently, a sorted sequence of symbols can not be predicted by solely examining Δ_i^k. Euclidean-distance-based enumeration strategies are thus suboptimal and may lead to considerable error-rate performance degradation, as shown in [1]. To cope with this disadvantage, the *min-search* (MS) and *adaptive hypothesis* (AH) approaches proposed in [4] and [1], respectively, correct the hypothesis by taking the *a priori* contribution into account. These techniques compensate the performance loss to some extent, whereas the detection accuracy is still suboptimal [1]. An additional disadvantage is represented by the loss of accuracy caused by the estimation of the metrics, which leads to a degradation of the error-rate performance [21]. Additionally, the memory requirement increases since the precomputed distances $(\Delta_i^{k'})^2$ (or optionally the products $r_{ii}^2(\Delta_i^{k'})^2$) have to be stored.

[2]For a conveniently chosen QRD, r_{ii} only contains positive real values.

C. Smart-Sorting Enumeration with Quadrature Metric Computation (SSE-QMC)

Existing tree search detection algorithms generally compute (4) repeatedly, disregarding the fact that the already determined quadrature components can be reused for other symbols with the same real or imaginary parts. The strategies proposed in this work, in contrast, exploit the latter property by initially determining and sorting the metrics' quadrature contributions, as described in the following. The proposed smart-sorting enumeration with quadrature metric computation (SSE-QMC) simplifies the metric computation significantly and saves an enormous amount of sorting operations, as demonstrated at the end of this section. The SSE-QMC mechanism relies on the following observations:

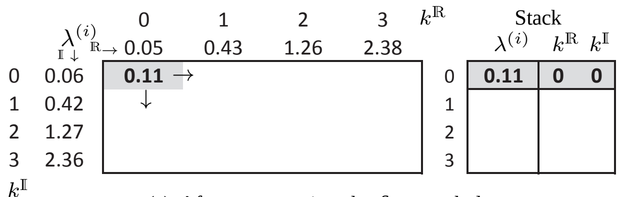
1) **Quadrature metric computation (QMC):**
   - For the considered QAM modulation, both the Euclidean distances and the *a priori* contributions λ_{a} in (4) can be decomposed in two additive quadrature components:
     $$\begin{aligned} \lambda_{(i)}^{\mathbb{R}}(x_i) &= \mathbb{R}\{r_{ii}^2(\Delta_i^{\mathrm{Eucl.}}(x_i))^2\} + N_0\lambda_{\mathrm{a}}^{\mathbb{R}}(x_i) \\ \lambda_{(i)}^{\mathbb{I}}(x_i) &= \mathbb{I}\{r_{ii}^2(\Delta_i^{\mathrm{Eucl.}}(x_i))^2\} + N_0\lambda_{\mathrm{a}}^{\mathbb{I}}(x_i) \end{aligned} \tag{9}$$
     with
     $$\lambda_{\mathrm{a}}^{\mathbb{R}}(x_i) = \sum_{\substack{j=0 \\ \text{with } c_{i,j}\neq\mathrm{sign}(L_{\mathrm{a}}(c_{i,j}))}}^{L/2-1} |L_{\mathrm{a}}(c_{i,j})|,\ \lambda_{\mathrm{a}}^{\mathbb{I}}(x_i) = \sum_{\substack{j=L/2 \\ \text{with } c_{i,j}\neq\mathrm{sign}(L_{\mathrm{a}}(c_{i,j}))}}^{L-1} |L_{\mathrm{a}}(c_{i,j})|.$$
   - The layer partial metric of any constellation symbol $\hat{x}_i^{(k^{\Re},k^{\Im})}$ can be hence determined by simply adding the corresponding quadrature components:
     $$\lambda_{(i)}\left(\hat{x}_i^{(k^{\Re},k^{\Im})}\right) = \lambda_{(i)}^{\Re}\left(\hat{x}_i^{k^{\Re}}\right) + \lambda_{(i)}^{\Im}\left(\hat{x}_i^{k^{\Im}}\right). \tag{10}$$
     By these means, the matrix of metrics illustrated by the example in Figure 4 can be composed (each matrix element corresponds to a symbol of a 16-QAM constellation). It should be noticed that, in order to obtain any of the Q partial metric values, only $\sqrt{Q}$ partial metric components have to be computed in each dimension.
2) **Smart-sorting enumeration (SSE):** By sorting the metric quadrature components in ascending order, the exact SE sequence can be determined *gradually*. Consequently, the tree nodes can be enumerated as they are required by the search process, instead of all at once (as required by the conventional SE enumeration approach).

The SSE-QMC procedure enumerates the tree nodes on the basis of these observations, by performing the following operations (exemplified in Figure 4):

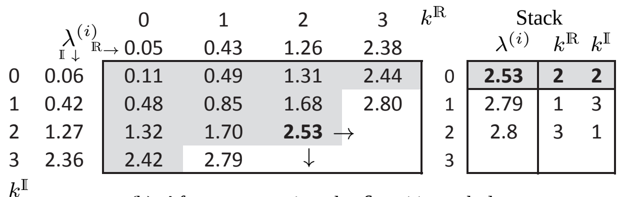
1) **Compute the $\sqrt{Q}$ quadrature components of the partial metrics** in each dimension ($\lambda_{(i)}^{\Re}$ and $\lambda_{(i)}^{\Im}$), according to (9).
2) **Sort the quadrature components in ascending order**, as depicted in Figure 4(a).

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904



(a) After enumerating the first symbol.



(b) After enumerating the first 11 symbols.

Fig. 4. Sample matrix of metrics and stack content for the SSE-QMC approach, after sorting the metrics’ quadrature components (each of the matrix elements corresponds to one symbol of a 16-QAM constellation). Grey-shaded cells correspond to symbols which have been already enumerated (i.e., examined by the tree search).

3) **Select the global minimum** (i.e., the element at the top-left corner of the metrics matrix) as the first symbol in the enumeration sequence.
4) **Expand two new candidates of the metrics matrix** (one along each quadrature component) departing from the previously selected symbol $\hat{x}_i^{(k^{\Re},k^{\Im})}$, i.e., compute the partial metrics $\lambda_i\left(\hat{x}_i^{(k^{\Re}+1,k^{\Im})}\right)$ and $\lambda_i\left(\hat{x}_i^{(k^{\Re},k^{\Im}+1)}\right)$ according to (10). The metric computation operation performed here consists in simply adding the already calculated quadrature components. In order to avoid generating new metric values unnecessarily (i.e., in case that better ones have been already computed) a *controlled-expansion* mechanism is applied.
5) **Add the newly determined enumeration candidates to the stack and sort them** in ascending order of the partial metrics, as illustrated in Figure 4(b).
6) **Select the next symbol to be explored within the tree search** (which is always contained in the first position of the stack), and remove it from the stack.
7) **Repeat steps 4 to 6** in order to enumerate further tree nodes, if required. The enumeration process finishes whenever all constellation symbols have been explored or the corresponding subtree is pruned (e.g., by the radius constraint).

The ***controlled-expansion*** mechanism mentioned in step 4 prevents unnecessary metric computations, hence avoiding stack overflows. Specifically, a new metric component is computed only if all the previously determined metrics within the same column/row have been already examined. In order to determine if a better metric has been already computed, no metric comparison is required. Due to the previous sorting of the metric quadrature components in ascending order, examining the element’s indexes within the matrix $(k^{\Re}, k^{\Im})$ is sufficient. By means of this controlled-expansion strategy, the stack size is kept reasonably small and overflow situations are prevented. It should be noticed that only the stack content needs to be stored, whereas the metrics matrix is employed here only for illustrative purposes.

V. Performance Analysis

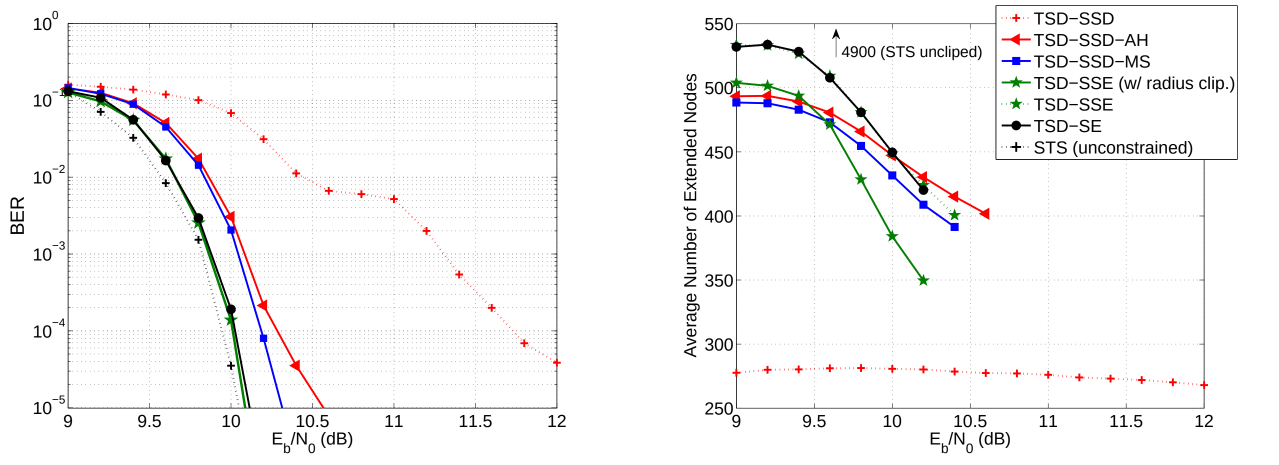
The overall complexity of a tree search algorithm can be assessed from two different perspectives, namely the computational complexity of each node extension (i.e., how computationally costly is to extend a single parent node of the tree), and the node count (i.e., the number of required parent node extensions). The latter is depicted in Figure 5, along with the error-rate performance. A scenario with $It = 4$ detection-decoding iterations is considered. The performance of the unconstrained single tree search (STS) strategy [18] with SE enumeration is included for reference, representing the optimal detection accuracy boundary (i.e., the performance of exhaustive *max-log-APP* detection [28]). In the considered iterative scenario, the SSD enumeration sequence is expected to diverge significantly from the exact SE ordering, resulting in a considerable error-rate performance loss. This effect is indeed exhibited in Figure 5(a), where the error rate of the TSD-SSD algorithm shows a clearly appreciable error floor. The corrections performed by the *Adaptive Hypothesis* (AH) [4] and the *Min-Search* (MS) [1] approach offer a noticeable enhancement of the error-rate performance, which is nonetheless still suboptimal. The SSE-QMC strategy, in contrast, achieves the same error-rate performance than the exact SE enumeration, but also a similar node count, as shown in Figure 5(b). The latter can be easily mitigated by simply applying the internal radius clipping mechanism of [6]. By these means, the average number of nodes accumulated over the 4 iterations is reduced by 20% (at $\mathrm{BER} = 10^{-5}$) with regard to the SE enumeration. As a result, the number of tree nodes explored by the proposed TSD-SSE-QMC approach (which provides the optimal node ordering) is comparable to the one of the TSD-SSD-ME strategy (which just approximates the ideal node sequence). The main advantage of the proposed SSE-QMC enumeration strategy resides, however, in the reduction of the computational complexity *per* node extension, rather than the in the reduction of the node count. Table I compares the computation count of the SSE-QMC approach against the one of exhaustive Schnorr-Euchner enumeration with state-of-the-art metric computation (4). To determine the number of compare and accumulate operations, the fast sorter architecture described in section VII has been assumed in both cases. The novel SSE-QMC mechanism achieves a drastic reduction of the computational effort, which causes a decrement of 80% in the complexity of the metric computation (on average), and a reduction of 97% in the complexity of the node enumeration process. To further reduce the complexity, the quadrature components of

	MC (4)	SE [26]	This work QMC	SSE
add	192	-	80	-
multiply	128	-	16	-
compare	-	4096	-	128
accumulate	-	512	-	16

TABLE I
Comparison of the computational complexity (in number of operations) of the Schnorr-Euchner enumeration (SE) with standard metric computation (MC) and smart-sorting enumeration (SSE) with quadrature metric computation (QMC).

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, "Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration," in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904



(a) BER performance.

(b) Average node count, accumulated during $It = 4$ detection processes.

Fig. 5. Performance comparison of different detection strategies in an iterative receiver with $It = 4$ turbo iterations (4×4 MIMO, 64-QAM, $T = 16$).

the *a priori* information ($\lambda_a^{\Re}$ and $\lambda_a^{\Im}$) can be pre-computed and stored (since they remain constant during the detection process). Throughout the next sections, the complexity of the proposed VLSI implementation will be evaluated.

VI. Towards an Efficient VLSI Implementation

The design's aspects contributing to restrict the incurred hardware complexity as well as to increase the processing capacity are described in the following.

A. Algorithm Partitioning

The sphere detection process can be decomposed, as originally proposed in [29], [27], into several functional blocks (ⓐ . . . ⓝ) comprising the regularized data flow diagram illustrated in Figure 6. The depicted operations *loop* is executed iteratively, examining one tree node in each iteration. Note that the data path has been explicitly divided into two branches. The path comprised by light-grey shaded blocks encloses the operations required by the tree node extension process, whereas the dark-grey shaded blocks contain the operations required to generate the soft-output information. The tree search procedure has been regularized [29], [27] in order to simplify the control flow and ease parallelization (e.g., single-instruction multiple-data -SIMD). The regularization concept consists in executing the same computation pattern in each iteration, regardless of the value of runtime data (such as e.g., the current tree layer). For this purpose, "dummy" operations are introduced and multiplexing logic is inferred in order to select only valid data for further processing.

As shown in Figure 6, the first and the second tree nodes (x_i^0, x_i^1) to be explored are determined by blocks ⓐ and ⓑ, respectively, according to the selected enumeration strategy (SSD or SSE). Note that in case the SSE approach is applied, the metric's quadrature components have to be previously computed, as denoted by block ⓐ'. In case the SSD approach is applied, the corresponding search sequence has to be determined instead (block ⓑ'). The partial metrics of the two first nodes ($\lambda_i(x_i^0)$, $\lambda_i(x_i^1)$) are subsequently computed, by means of the QMC or ME approach, within the respective blocks ⓒ and ⓔ. Analogously, the corresponding reduced-interference received signals (y_i''') are calculated according to (5) in blocks ⓓ and ⓕ. The previously determined partial metric $\lambda_i(x_i^0)$ is subsequently considered in order to update the radius tuple (6) within block ⓖ. This operation is performed at each tree layer for the sake of regularization, whereas it only takes effect at layer $i = 0$. The next step consists in selecting the tree layer to be processed within the next loop (block ⓗ). In case the search proceeds on the same layer or traces back to upper tree layers, a new sibling node needs to be enumerated, its corresponding partial metric has to be determined, and the inter-antenna interference has to be computed. It should be noticed that these operations (performed by blocks ⓘ, ⓙ and ⓚ, respectively), are rather unconditionally executed in order to keep the regularized control flow. Likewise, the operations involved in the soft-output generation, which are actually required only at layer $i = 0$, are nevertheless continuously executed regardless of the tree layer. In particular, the bit-specific metrics required for the LLR values are determined by block ⓛ, and the stored subset of candidates is thereafter updated by block ⓜ. Finally, the LLR values are computed in block ⓝ. As shown in Figure 6, the end-condition (which terminates the execution as soon as the root node $i = N_T - 1$ is reached) is the only conditional operation required.

B. Memory Access

The designs proposed in section VII are accelerated by means of application-specific instruction-set processors (ASIPs). The data path, depicted in Figure 8, comprises the MIMO detection module, an address generation unit and several data memories, particularly:

- **Input vector memory (VMEMI):** this memory contains the vectors of N_T received signals $\mathbf{y}$.
- L_a **vector memory (VLAMEM):** it stores the vectors of $N_T \cdot L$ *a-priori* values $L_a(c_{m,l})$, generated by the channel decoder in order to enable the application of the turbo principle.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

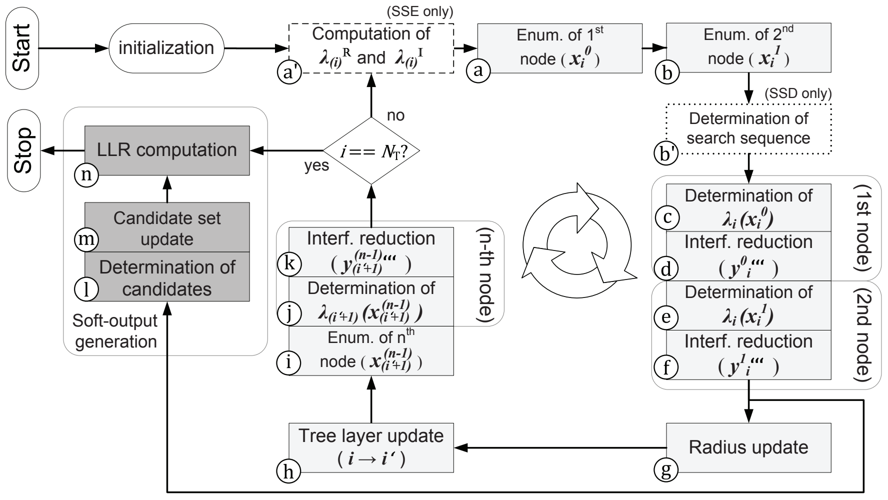


Fig. 6. TSD regularized flow diagram.

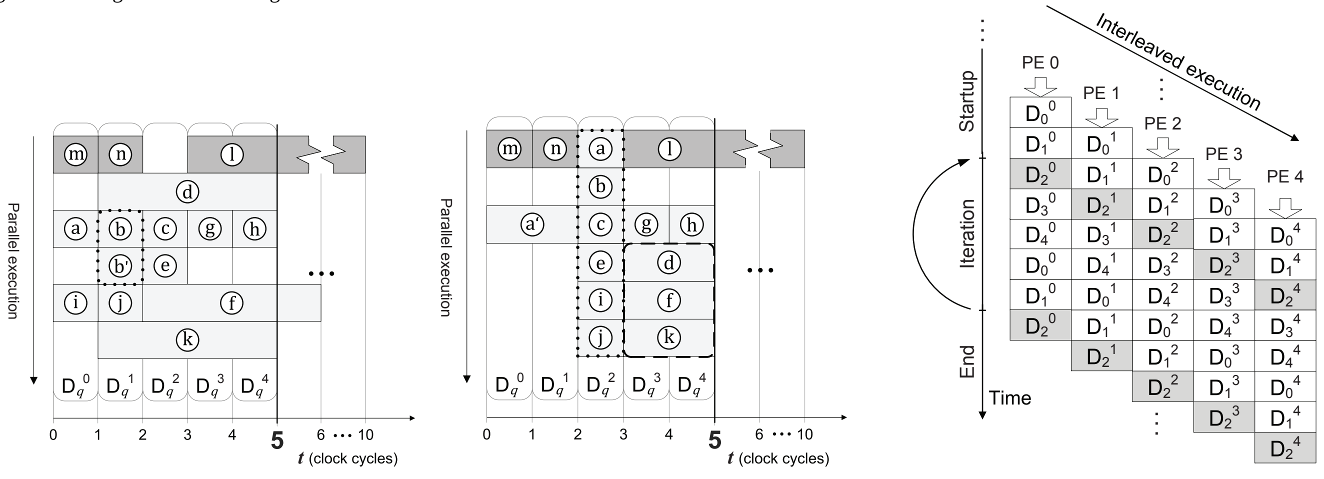


Fig. 7. TSD’s task scheduling and pipelining schemes.

- **Output vector memory (VMEMO):** the vectors of LLR values $L\left(c_{m,l}|\mathbf{y}\right)$ generated by the MIMO detector are contained in this memory.
- **Scalar memory (SMEM):** this memory contains configuration data (such as N_T, Q and T) and channel information, including the channel matrix $\mathbf{R}$ and the precomputed products $r_{ii}^2(\Delta_i^{k'})^2$ (8) (required by the ME approach). A cache which holds a partial copy of the SMEM content was additionally introduced in [30]. This storage element allows faster data access and saves costly (in terms of latency) memory access operations. It should be emphasized that no precomputed data are required by the QMC strategy, in which case the configuration information can be relocated into VMEMI, while SMEM and the cache (256 bytes each[3]) can be completely eliminated from the design.

C. Scheduling and Parallelization

In [30] the computational complexity and the latency of the operations comprising the TSD-SSD-ME detection algorithm have been investigated. A time budget (in clock cycles) has been defined for each functional block, based on the number of sequential add-equivalent[4] operations (assuming a sufficient degree of internal bit-level parallelism[5]). The overall latency is reduced by partially overlapping the execution of the defined functional blocks (i.e., through instruction-level parallelism) [30], [22]. The proposed scheduling scheme is illustrated in Figure 7(a), presenting an overall delay of 10 clock cycles. The data generated by blocks ⓜ, ⓝ and ⓕ are not required by the immediately subsequent detection loop(s) [30] and, consequently, a new iteration can be triggered every 5 clock cycles instead. In the case of the TSD-SSE-QMC detection approach, the node enumeration as well as the metric computation modules have been modified in order to implement the SSE and QMC strategies, respectively. The different computational effort and data dependencies of these

[3]The stored information is represented with 8-bit precision [30], [1].

[4]The specific definition of add-equivalent operation as well as a detailed breakdown of the blocks’ latencies can be found in [30]. The latter are also depicted in Figure 7. Each add-equivalent computation is assumed to consume one clock cycle.

[5]Bit-level parallelism of the proposed architecture is depicted in section VII.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, "Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration," in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

blocks with respect to the analogous modules in the TSD-SSD-ME design (described in detail in section VII) require a modified task scheduling scheme (as depicted in Figure 7(b)). Particularly, the computation of the quadrature partial metrics is firstly performed in block ⓐ', with a time budget of 2 clock cycles. The first and second enumerated nodes, as well as their corresponding partial metrics, are subsequently determined in blocks ⓐ, ⓑ, ⓒ and ⓔ, with a time budget of 1 clock cycle. In parallel to this, successive sibling nodes are analogously enumerated (ⓘ, ⓙ). The operations to determine the radius and the tree layer (ⓖ, ⓗ) are scheduled exactly as in the case of the TSD-SSD-ME approach, whereas the execution of the interference-reduction units has been delayed by 1 (ⓕ) and 2 clock cycles (ⓓ, ⓚ). In order to maintain the proposed 5-cycle architecture, the time budget for these units has been reduced to 2 clock cycles, as illustrated in Figure 7(b). As later discussed in section VIII, these modules do not lie within the critical path and, consequently, no decay of the maximum achievable clock frequency is expected. Lastly, the units computing the bit-specific metrics and LLRs (ⓛ, ⓜ, ⓝ) keep the same timing structure as in the previous case. Note that the newly proposed TSD-SSE-QMC approach presents a fairly more compact and structured scheduling scheme than the preceding TSD-SSD-ME design, which allows grouping the execution (and thereby the hardware resources) of similar functional blocks easily.

Pipeline-interleaving [31] has been applied to the proposed TSD detector in order to enhance the design's throughput [30]. Similarly as done in e.g., [32] and [33], several independent data paths are pipelined and executed in interleaved fashion. The previously described task blocks have been clustered according to their execution times t, as illustrated in Figures 7(a) and 7(b). Resulting from this, 5 sets of parallel operations are composed, each defining a pipeline stage. On this basis, the throughput can be easily enhanced by simply interleaving the pipelined execution of the 5 task-sets for different detection paths. This procedure is illustrated in Figure 7(c), where $D_q^p \in \{D_q^0 \dots D_q^4\}$ denotes the p-th task-set of the q-th detection path. The resource utilization is maximized when the number of interleaved data paths equals the number of available pipeline stages $P = 5$, achieving an average throughput $\overline{\tau}$:

$$\overline{\tau} = \frac{L N_\mathrm{T} P}{\mathbb{E}[n] P + P - 1} f'_\mathrm{CLK} \quad \rightarrow \quad \frac{L N_\mathrm{T}}{\mathbb{E}[n]} f'_\mathrm{CLK} \quad \text{[bps].} \quad (11)$$

Ideally, the clock frequency reached by the pipelined circuit $f'_\mathrm{CLK} > f_\mathrm{CLK}$ increases by a factor of nearly P with respect to the non-pipelined architecture[6] ($f'_\mathrm{CLK} \approx P \times f_\mathrm{CLK}$). Assuming that a new detection starts as soon as another has finished, no processing element (PE) is idle after filling in the pipe, leading to the sustained throughput expression on the right-hand side of equation (11). The memory address computation and access operations are embedded in the pipeline structure, in parallel to the detection algorithm's computations. By these means, the execution of a new detection path can be triggered as soon as another one is concluded, without stalling the pipeline execution.

[6]The actual speedup factor is constrained by the physical limitations of the underlying technology.

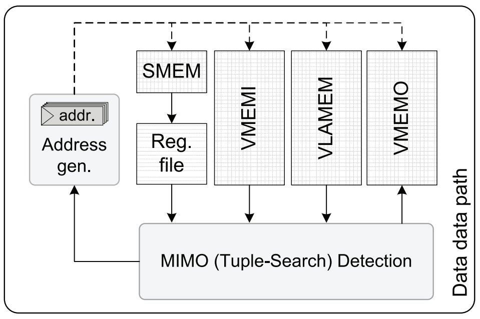


Fig. 8. Block diagram of the MIMO detector ASIP architecture.

VII. Architectures for MIMO Detection

The newly proposed SSE-QMC detector has been shown to be a promising solution to cope with the disadvantages of the SSD-ME approach. However, its efficiency has not yet been assessed from the implementation point of view. For this reason, both TSD-SSD-ME and TSD-SSE-QMC architectures will be analyzed and compared[7] in the following. Both MIMO detector realizations have been optimized for a 4×4 MIMO system employing a 64-QAM modulation. A dedicated functional unit (FU) has been defined for each of the task blocks ⓐ – ⓝ shown in Figure 6, with the exception of ⓑ and ⓑ' - which are merged together. In the following, the architecture of those units involved in the node enumeration and the metric determination tasks will be described and illustrated, highlighting in each case the critical path by means of a red arrow. Remaining units of the MIMO detector are common to both designs and are thus omitted here (a complete description can be found in [30], [22] and [1]).

A. Node Enumeration Unit (NEU)

The NEU block enumerates the nodes to be examined during the tree search in the appropriate order. In the case of the SSD approach, the pre-computed enumeration sequences are stored in a small ($\approx$ 32 bytes) look-up-table (LUT). The first node x_i^ref is directly obtained by rounding the normalized reduced-interference received signal y_i''' to the closest constellation symbol, whereas subsequent symbols are directly selected from one of the enumeration series contained in the LUT, as illustrated in Figure 9(a). The appropriate node sequence is identified by determining which of the quantized constellation regions in Figure 3 contains the received signal y'''. For this purpose, simple comparisons of the real and the imaginary components of $\Delta^\mathrm{r} = y''' - x_i^\mathrm{ref}$ are required, as described in detail in [4]. In the case of the SSE approach, this unit implements the node enumeration procedure described in

[7]An architecture concept for the soft-output TSD-SSD-ME approach was firstly presented in [30], [22] and brought to silicon in [34]. An extended design was proposed in [1], adapted to support processing of *a-priori* information. In this regard, the adaptive hypothesis strategy proposed in [1] was incorporated in order to enable achieving an acceptable error-rate performance in iterative detection↔decoding receivers. On-chip measurements of the corresponding silicon realization can be found in [35].

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

section IV-C (steps 3 to 6). The first node in the enumeration sequence is directly obtained (step 3) after sorting the metrics' quadrature components (previously determined by the metric computation unit). In order to enumerate further nodes, steps 4 to 6 are executed. As illustrated in Figure 9(b), only two parallel adders are required to compute the partial layer metrics of successive nodes. These values are added to the stack, which is subsequently sorted by means of the fast sorter circuit described below. The next sibling symbol (which occupies the first position in the stack after the sorting process) is then selected and removed from the stack. Two additional parallel adders are required to add the metrics accumulated throughout the upper tree layers λ_{i+1}. The complexity of this unit is thus comparable to the one of the SSD, which requires a similar amount of adders and comparators. The SSD requires, in addition, a LUT to store the pre-defined enumeration sequences.

B. Metric Computation Unit (MCU)

As described in section IV, the ME approach pre-computes the products $r_{ii}^2(\Delta_i^k)^2$ and eventually stores them in a cache (65 bytes). Consequently, (4) is simplified to a single addition operation, as illustrated in Figure 10(a). The TSD-SSE-QMC design, in contrast, performs an exact computation of the metrics according to (9) and (10), instead of the estimation in (8). The metric computation introduces two multipliers for each of the quadrature components (namely the squared operation $|y_i''' - \hat{x}_i|^2$ and the subsequent multiplication with r_{ii}). As illustrated in Figure 10(b), the SSE-QMC approach proposed in section IV-C requires a total of $2\sqrt{Q}$ multipliers per dimension, thus increasing the computational complexity considerably with respect to the SSD-ME approach. Furthermore, the parallelism of the block which computes the *a priori* contribution increases to $\sqrt{Q}$ parallel instances per dimension. The quadrature metric components are sorted by employing the fast sorter circuit described below, which also entails a complexity increase with respect to the SSD-ME approach. The memory requirement is reduced, on the contrary, since no pre-computed Euclidean distances need to be stored. It should be noticed that this unit (corresponding to task block ⓐ' in Figure 6) computes only the metrics' quadrature components, whereas the final partial metric values (including the upper layer's metric) are subsequently obtained in the NEU, as previously described.

C. Soft-Output Computation Unit (SOCU)

This unit is involved in the generation of soft-information (i.e., LLR values) for the channel decoder. In particular, it determines the L most favorable counter-hypotheses among all candidate leaf symbols, enumerating them according to predefined leaf-specific sequences [11]. For this purpose, instances of the MCU and parts of the NEU (Figures 10(a) and 9, respectively) are utilized. The SSD-ME approach estimates the metric values of the L candidates, as described in section VII-B. For this purpose, 32 bytes are required to store the pre-computed products $r_{ii}^2(\Delta_i^k)^2$ for the leaf-specific sequences. It should be emphasized that these pre-computed values do not include the contribution of the *a priori* information, which has to be additionally calculated. The SSE-QMC approach, on the contrary, simplifies the computation of (4) to two addition operations (to add the quadrature components and the metric of the upper tree layer, respectively), as illustrated in Figure 9(b). In contrast to the TSD-SSD approach, the *a priori* contribution is not computed (since it is already contained in the metric quadrature components provided by the MCU), and less memory is required (since no pre-computed distances $r_{ii}^2(\Delta_i^k)^2$ are stored).

D. Fast Sorter Circuit

The fast sorter circuit depicted in Figure 11 sorts a vector of N positive values $\mathbf{z} = [z_0, z_1, \ldots, z_{N-1}]$ in a highly-parallel manner. For this purpose, all the N values in the considered set are compared against each other by a bank of N^2 parallel comparators. The output of this operation is the $N \times N$ matrix of binary flags depicted in Figure 11. Each binary value $b_{n,m}$ indicates if the condition $z_n > z_m$ is met ($b_{n,m} = 1$) or not ($b_{n,m} = 0$). The number of flags $b_{n,m} = 1$ within column m represents hence the position (or *index*) that element z_m should occupy within the sorted vector $\mathbf{z}'$. In order to generate this sorting index, the binary values within each column are added together by a bank of N accumulators $\sum_{n=1}^{N} b_{n,m}, \forall m = \{1, 2, \ldots, N\}$. Lastly, N parallel multiplexers are employed to relocate the elements of $\mathbf{z}$ within the output vector $\mathbf{z}'$ according to the computed indexes. This highly-parallel circuit enables a very fast sorting operation at the cost of having a complexity which scales quadratically with N. This is obviously disadvantageous if a large number of elements have to be sorted, as in the case of the exhaustive Schnorr-Euchner enumeration ($N^2 = Q^2$) for high-order modulations (e.g., 64-QAM). In contrast to this, the number of values to be sorted by the presented SSE-QMC strategy is reduced to only $N^2 = (\sqrt{Q})^2 = Q$ elements, as discussed in section IV-C. This fact makes the proposed sorting circuit affordable even for high-order modulations.

VIII. Implementation Results

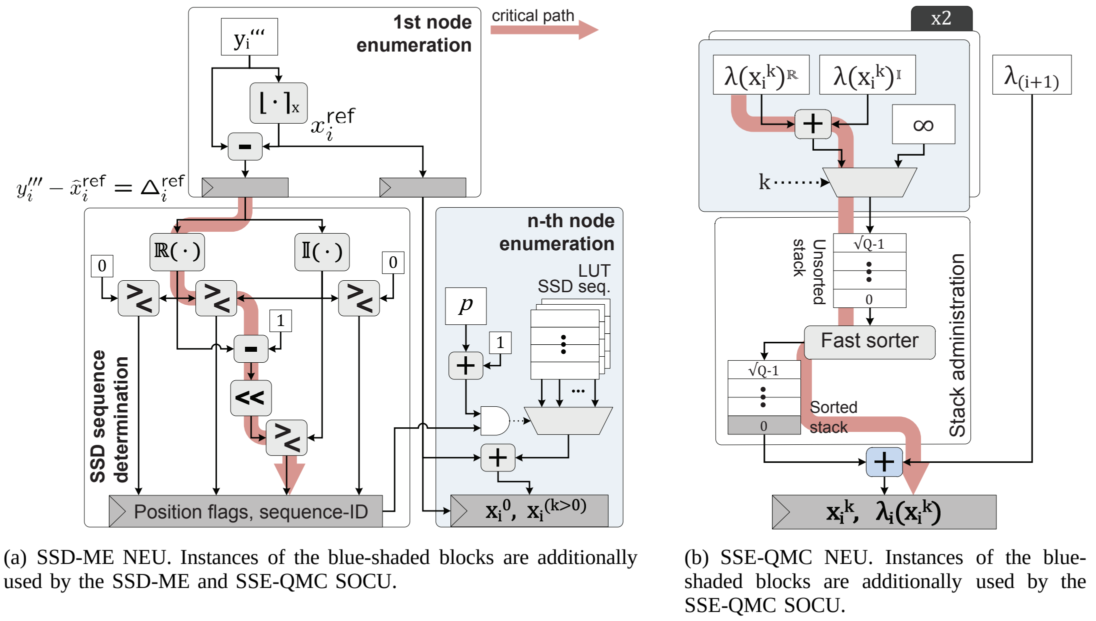
In order to assess the hardware cost of the proposed SISO TSD-SSE-QMC architecture, the most relevant design's characteristics (area, delay, throughput and energy) will be evaluated in the following. Additionally, a comparison against the SISO TSD-SSD-ME design of [1] will be provided. For this purpose, the previously described SSD-ME and SSE-QMC modules have been implemented and synthesized using TSMC low-power 65nm libraries under typical case operating conditions.

A. Area

The area (in GEs) required by the complete SSD-ME and SSE-QMC modules is depicted in Figure 12. Note that the area corresponding to the SOCU block accounts only for the logic required by the metrics' computation, whereas remaining operations do not differ between the compared approaches and are consequently omitted. As shown by these results, the

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

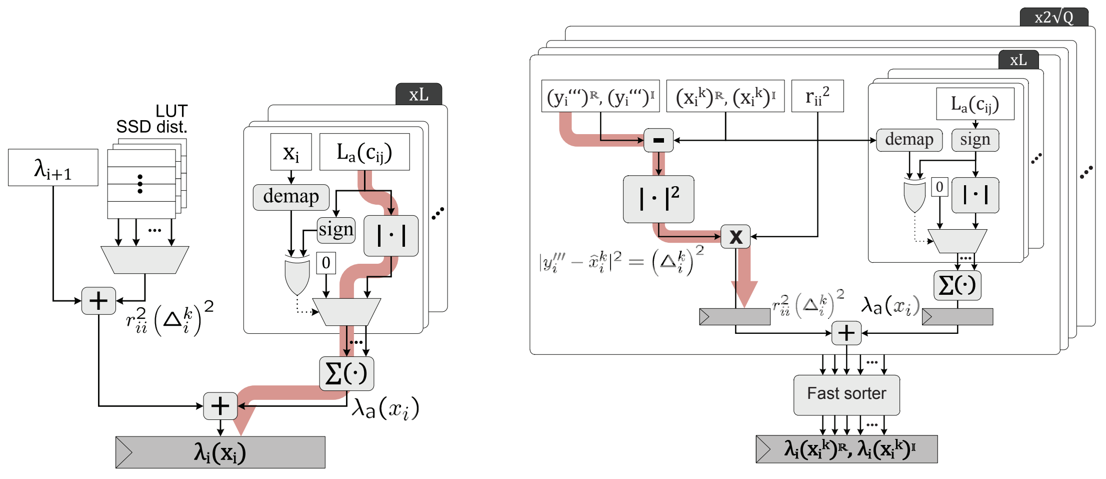
This document is a preprint of: E. Perez Adeva and G. Fettweis, "Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration," in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904



(a) SSD-ME NEU. Instances of the blue-shaded blocks are additionally used by the SSD-ME and SSE-QMC SOCU.

(b) SSE-QMC NEU. Instances of the blue-shaded blocks are additionally used by the SSE-QMC SOCU.

Fig. 9. Architecture of the TSD's node enumeration unit (NEU).



(a) SSD-ME MCU. Instances of this unit are additionally used by the SSD-ME SOCU.

(b) SSE-QMC MCU.

Fig. 10. Architecture of the TSD's metric computation unit (MCU).

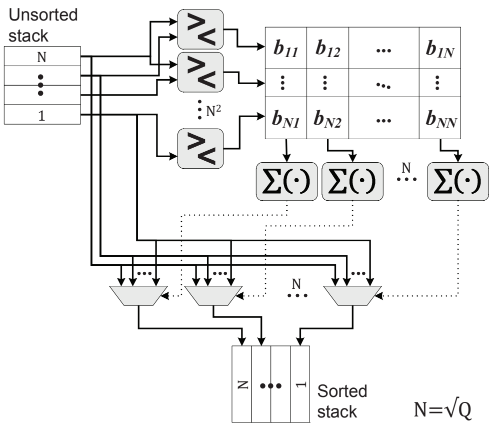


Fig. 11. Architecture of the TSD-SSE-QMC's fast sorting circuit.

size of the SSE-QMC's MCU has grown significantly with respect to that of the SSD-ME approach, as a consequence of the complex-valued multiplications and the considerably increased level of parallelism. The area increase exhibited by the NEU due to the introduction of the stack maintenance and sorting operations is less pronounced but still considerable. As previously presumed, the area of the SOCU has been in contrast reduced, since the LUT storing the Euclidean distances has been eliminated and the *a-priori* information processing is entirely accomplished within the MCU. As previously discussed, the SMEM, its associated cache, and the corresponding memory access control logic is not required by the SSE-QMC approach, making the area of these components (enclosed in the category *Other* in Figure 12) vanish. To sum up, the proposed SSE-QMC strategy requires nearly 4 and 8 times larger NEU and MCU modules, respectively, but a slightly smaller SOCU block. Taking the additional memory saving

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

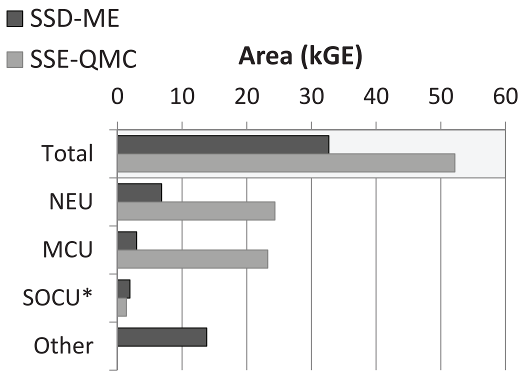


Fig. 12. Area comparison of the SSD-ME and SSE-QMC modules (synthesized under typical case operating conditions at $f_{\mathrm{clk}} = 450\mathrm{MHz}$). *The area of the SOCU block corresponds to the logic required exclusively for the computation (estimation) of the Euclidean distances.

into account, the area balance corresponding to the complete SSE-QMC block exhibits an overall 60% increase with regard to the SSD-ME module. It should be however noticed that the node enumeration and metric computation units represent less than $1/4$ of the complete TSD design. A rough estimation ($A^{\mathrm{total}}_{\mathrm{TSD-SSE-QMC}} = A^{\mathrm{total}}_{\mathrm{TSD-SSD-ME}} - A_{\mathrm{SSD-ME}} + A_{\mathrm{SSE-QMC}} = 184\mathrm{kGE} = 0.26mm^2$, with $A^{\mathrm{total}}_{\mathrm{TSD-SSD-ME}} = 165\mathrm{kGE}$ [1]) indicates that replacing the SSD-ME block by the SSE-QMC one within the TSD core would lead to an overall area increase of only 12%.

B. Delay

In comparison to the estimation of the metrics, the metric computation causes a considerable increase of the critical path delay. In order to avoid this, the SSE-QMC block has been partitioned and pipelined according to the structure presented in section VI-C. Each module has been individually optimized during synthesis for its maximum clock frequency. The resulting critical path delays are illustrated in Figure 13. The delay of the interference-reduction unit (IRU) is additionally depicted, since it is affected by the redefinition of the pipelining structure (section VI-C). In particular, the delay obtained by applying different pipelining depths P (with $P = \{1, \ldots, 4\}$) is illustrated (denoted by dashed lines). As evinced here, even in the worst case $P = 1$ (non-pipelined circuit), the IRU delay is still slightly inferior than the design's critical path delay, originally defined by the SOCU block. It can be hence concluded that the design's f^{max}_{clk} will not be affected by the proposed IRU's retiming. As additionally depicted, the delay of the SOCU block stays constant, whereas the critical paths of the NEU and the MCU architectures have been extended. In particular, the NEU's delay has nearly doubled and the MCU's delay is approximately 40% higher. As a result, the MCU defines the critical path of the SSE-QMC's design, which is 14% higher than that of the SSD-ME circuit (determined by the SOCU block). This increment can nevertheless be considered as irrelevant, since the longest path delay ($1.65\mathrm{ns}$) is still far from the target timing constraint ($2.2\mathrm{ns}$). The SSE-QMC block has been in fact successfully synthesized for $f^{max}_{clk} = 465\mathrm{MHz}$ and, consequently, the maximum clock frequency of the complete TSD-SSE-QMC

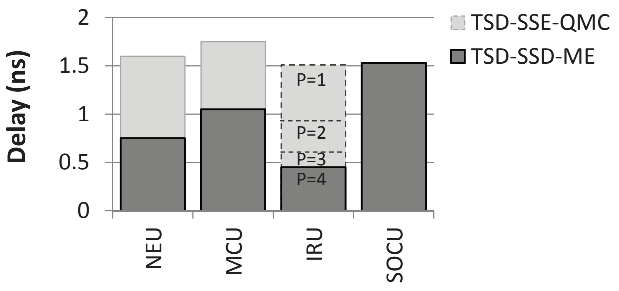


Fig. 13. Delay of selected TSD-SSD-ME and TSD-SSE-QMC modules, synthesized for the maximum achievable frequency in each case under typical-case operating conditions.

module is not expected to decay below the original frequency of the TSD-SSD-ME design ($f^{max}_{clk} = 454\mathrm{MHz}$) [1].

C. Throughput and Energy

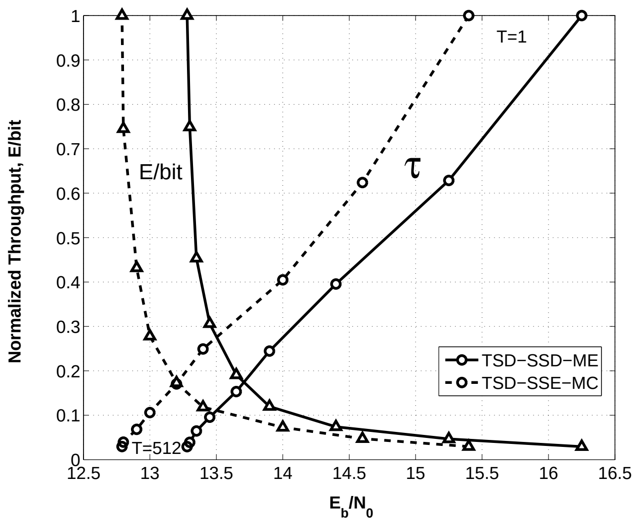
The normalized throughput and energy-per-bit corresponding to the TSD-SSD-ME and TSD-SSE-QMC strategies are depicted in Figure 14(a) for different values of T, in a non-iterative signal-processing scenario. The non-normalized values are included for reference in Table 14(b). As evinced here, the higher accuracy of the SSE-QMC approach allows reducing the value of T with respect to the SSD-ME strategy. This benefit becomes especially noteworthy in the low SNR regime. At e.g., $E_{\mathrm{b}}/N_0 = 13.4\mathrm{dB}$, the TSD-SSD-ME approach requires a search tuple of size $T = 64$ in order to meet the target error-rate ($\mathrm{BER} = 10^{-5}$), whereas a tuple size $T = 8$ is sufficient in the case of the proposed TSD-SSE-QMC strategy. This allows achieving a 3.4 times higher throughput while requiring only $1/4$ of the energy-per-bit. In the high SNR regime ($\mathrm{SNR} > 13.5\mathrm{dB}$), the benefit of the TSD-SSE-QMC approach is mainly restricted to the throughput gain ($\geq 40\%$ increase), since the energy dissipation is in both cases comparably low. For a particular throughput value (i.e., for the same T), the proposed TSD-SSE-QMC detector presents a performance gain of approximately $0.4\mathrm{dB}$ with regard to the TSD-SSD-ME detection strategy.

D. Comparison to the State-of-the-Art

The most relevant and representative MIMO detector realizations reported to date in literature are listed for comparison in Table II. The cost of input/output memories (i.e., on-chip SRAM macrocells) has been removed from the gate count. All hardware characteristics are presented as reported in the respective works, whereas area-throughput and energy efficiency have been scaled to 65nm and $V_{\mathrm{DD}} = 1.2\mathrm{V}$ according to the scaling factors $S = \frac{\text{reported tech.}}{65}$ and $U = \frac{\text{reported } V_{\mathrm{DD}}}{1.2}$ (scaled values are enclosed within brackets). It should be emphasized that results from literature correspond to silicon implementations, while the results reported in this work correspond to pre-layout synthesis of the proposed VLSI design. The TSD-SSE-QMC detector has been configured with $T = 8$. The area-throughput and energy efficiency can be nevertheless improved with respect to this configuration by reducing the value of T, as discussed in section VIII-C.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, "Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration," in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904



(a) Normalized average throughput and energy-per-bit.

	T	1	2	4	8	16	32	64	256	512
SSD-ME	SNR (dB)	16.3	15.3	14.4	13.9	13.7	13.5	13.4	13.3	13.3
	τ (Mbps)	1360	855	538	333	209	130	88	53	40
	E/bit (nJ/b)	0.12	0.19	0.30	0.48	0.77	1.24	1.83	3.03	4.05
SSE-QMC	SNR (dB)	15.4	14.6	14.0	13.4	13.2	13.0	12.9	12.8	12.8
	τ (Mbps)	1206	753	488	300	206	128	82	48	36
	E/bit (nJ/b)	0.12	0.19	0.29	0.46	0.68	1.09	1.70	2.93	3.93

(b) Average throughput, energy-per-bit and SNR values.

Fig. 14. TSD's average throughput and energy-per-bit.

The MIMO STS sphere detector (SD) proposed in [36] represents the best-suited competitor for comparison against the TSD-SSE-QMC realization, due to the high number of algorithmic and implementation similarities. The much higher throughput and considerably reduced area of the TSD-SSE-QMC design results in a clearly superior throughput-area efficiency $\eta_{\tau,\mathrm{A}}$, which excels that of the contending circuit by factor ≈ 8, while doubling the energy efficiency η_{E}. As shown in Table II, the efficiency of the proposed TSD-SSE-QMC design is additionally comparable or even slightly superior than that of other similar detectors from literature ([18], [37]). In contrast to this, the pipelined hard-output K-Best detector of [38] outperforms the efficiency of the proposed detector, but it is not capable of incorporating and generating soft-information. The best-first sphere detector (BeF-SD) proposed in [39], on the other hand, doubles the gate count of the TSD-SSE-QMC design. This system is dimensioned to support a higher MIMO order (8×8), but can not process *a priori* information. Lastly, an MMSE parallel interference cancellation (PIC) detection approach is regarded for comparison. In contrast to the TSD-SSD-ME circuit, the MMSE-PIC design integrates all the circuitry required for channel pre-processing, including a LU-decomposition-based matrix inversion approach similar to the QR-decomposition required by the sphere detector. For the sake of a fair comparison, the corresponding pre-processing area penalty (68.1kGE [28]) is added to the gate count of the TSD-SSE-QMC design, yet totalling nearly 40% of the MMSE-PICs gate count. An additional disadvantage of the PIC approach is that it presents nearly 3dB loss (in terms of SNR) with regard to sphere detection[8] in the non-iterative case.

IX. Conclusion

The efficiency of the soft-input soft-output MIMO detector proposed in this work has been shown to be superior than that of several contending depth-first detector realizations, while approaching the complexity of significantly less accurate detection methods (such as K-Best and PIC detectors). In contrast to the preceding TSD-SSD-ME approach (which presents suboptimal performance and is practically restricted to non-iterative receivers), the proposed TSD-SSE-QMC strategy provides nearly optimal detection performance in iterative scenarios ($< 0.1\mathrm{dB}$ loss with regard *max-log-APP*). Additionally, significant gains are provided, both in terms of throughput (from 40% up to a factor 5) and energy efficiency (up to 80% energy saving in the low SNR regime). The proposed approach represents hence a viable solution for low-complexity MIMO detection, which avoids sacrificing accuracy or performance.

References

[1] E. P. Adeva, T. Seifert, and G. Fettweis, "VLSI Architecture for MIMO Soft-lnput Soft-Output Sphere Detection," *in Journal of Signal Processing Systems (JSPS)*, vol. 70, no. 2, pp. 125–143, February 2013, dOI 10.1007/s11265-012-0709-z.

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

[3] C. Hess, M. Wenk, A. Burg, P. Luethi, C. Studer, N. Felber, and W. Fichtner, "Reduced-Complexity MIMO Detector with Close-to ML Error Rate Performance," in *ACM Great Lakes Symposium on VLSI (GLSVLSI)*, 2007, pp. 200–203.

[4] B. Mennenga and G. Fettweis, "Search Sequence Determination for Tree Search based Detection Algorithms," in *Proceedings of the IEEE Sarnoff Symposium 2009*, Princeton, USA, 2009.

[5] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Transactions on Communications*, vol. 51, pp. 389–399, 2003.

[6] B. Mennenga, R. Fritzsche, and G. Fettweis, "Iterative Soft-In Soft-Out Sphere Detection for MIMO Systems," in *Proceedings of the IEEE 69th Vehicular Technology Conference, (VTC'09-Spring)*, Barcelona, Spain, 2009.

[7] M. Shah, B. Mennenga, J. Werner, and G. Fettweis, "Complexity Reduction in Iterative Soft-In Soft-Out Sphere Detection," in *Proceedings of the IEEE 73rd Vehicular Technology Conference, (VTC Spring)*, 2011.

[8] T. Seifert, E. P. Adeva, and G. Fettweis, "Towards Complexity-Reduced Soft-Input Soft-Output Sphere Detection," in *Proceedings of the 9th International ITG Conference on Systems, Communications and Coding 2013, (SCC'13)*, Munich, Germany, 2013.

[9] K. Nikitopoulos, D. Zhang, I.-W. Lai, and G. Ascheid, "Complexity-Efficient Enumeration Techniques for Soft-Input, Soft-Output Sphere Decoding," *IEEE Communications Letters*, vol. 14, no. 4, pp. 312–314, 2010.

[10] C.-H. Liao, I.-W. Lai, K. Nikitopoulos, F. Borlenghi, D. Kammler, M. Witte, D. Zhang, T.-D. Chiueh, G. Ascheid, and H. Meyr, "Combining orthogonalized partial metrics: Efficient enumeration for soft-input sphere decoder," in *International Conference on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2009, pp. 1287–1291.

[11] B. Mennenga, A. von Borany, and G. Fettweis, "Complexity reduced Soft-In Soft-Out Sphere Detection based on Search Tuples," in *Proceedings of the IEEE International Conference on Communications (ICC'09)*, Dresden, Germany, 2009.

[8]Measured at $\mathrm{PER} = 10\%$ in a typical 40-MHz IEEE 802.11n scenario with four spatial streams, a 16-QAM modulation, and a rate-1/2 convolutional code over a TGn type C channel [28].

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

[12] G. Caire, G. Taricco, and E. Biglieri, “Bit-Interleaved Coded Modulation,” *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 927–946, 1998.

[13] S. ten Brink, J. Speidel, and R.-H. Yan, “Iterative Demapping and Decoding for Multilevel Modulation,” in *IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 1, 1998, pp. 579–584.

[14] X.Li and J. Ritcey, “Bit-Interleaved Coded Modulation with Iterative Decoding Using Soft Feedback,” *Electronics Letters*, vol. 34, no. 10, pp. 942–943, 1998.

[15] P. Robertson, E. Villebrun, and P. Hoeher, “A Comparison of Optimal and Sub-optimal MAP Decoding Algorithms Operating in the Log Domain,” in *Proceedings of the IEEE International Conference on Communications, 1995 (ICC ’95)*, 1995.

[16] D. Wübben, R. Böhnke, J. Rinas, V. Kühn, and K. Kammeyer, “Efficient Algorithm for Decoding Layered Space-Time Codes,” *Electronics Letters*, vol. 37, pp. 1348–1350, 2001.

[17] J. Jaldén and B. Ottersten, “Parallel Implementation of a Soft Output Sphere Decoder,” in *Proceedings of Asilomar Conference on Signals, Systems, and Computers*, 2005.

[18] C. Studer, A. Burg, and H. Bölcskei, “Soft-output sphere decoding: Algorithms and VLSI implementation,” *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 290–300, Feb. 2008.

[19] M. S. Yee, “Max-log-MAP sphere decoder,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’05)*, vol. 3, 18.-23. Mar. 2005.

[20] Z. Guo and P. Nilsson, “A VLSI Architecture of the Schnorr-Euchner Decoder for MIMO Systems,” in *IEEE 6th CAS Symposium on Emerging Technologies: Mobile and Wireless Communications*, 2004.

[21] E. P. Adeva, B. Mennenga, and G. Fettweis, “Survey on an Efficient, Low-complex Tuple Search Based Sphere Detector,” in *Proceedings of the IEEE 34th Sarnoff Symposium*, Princeton, USA, 2011.

[22] E. P. Adeva, M. A. Shah, B. Mennenga, and G. Fettweis, “VLSI Architecture for Soft-Output Tuple Search Sphere Decoding,” in *Proceedings of the IEEE Workshop on Signal Processing Systems (SIPS’11)*, Beirut, Lebanon, 2011.

[23] E. Zimmermann and G. Fettweis, “Unbiased MMSE Tree Search Detection for Multiple Antenna Systems,” in *Proceedings of the International Symposium on Wireless Personal Multimedia Communications (WPMC’06)*, San Diego, USA, Sep. 2006.

[24] Y. de Jong and T. Willink, “Iterative tree search detection for MIMO wireless systems,” *IEEE Transactions on Communications*, vol. 53, no. 6, pp. 930–935, 2005.

[25] E. P. Adeva and G. Fettweis, “Verfahren zur Bestimmung der optimalen Suchreihenfolge von Knoten bei einem Baumsuch-Algorithmus mit geringen Komplexität,” Technische Universität Dresden, Patent pending DE 10 2015 109 015.5, 8. June 2015.

[26] C. Schnorr and M. Euchner, “Lattice basis reduction: Improving practical lattice basis reduction and solving subset sum problems,” *Mathematical Programming*, vol. 66, pp. 181–199, Aug. 1994.

[27] B. Mennenga, *Aufwandsgünstige Detektion in Mehrantennensystemen mittels komplexitätsreduzierter Baumsuchverfahren*. Dissertation, Technische Universität Dresden, 2010.

[28] C. Studer, S. Fateh, and D. Seethaler, “ASIC Implementation of Soft-Input Soft-Output MIMO Detection Using MMSE Parallel Interference Cancellation,” *Solid-State Circuits, IEEE Journal of*, 2011.

[29] B. Mennenga, E. Matus, and G. Fettweis, “Vectorization of the Sphere Detection Algorithm,” in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS’09)*, Taipei, Taiwan, 2009.

[30] E. P. Adeva, B. Mennenga, and G. Fettweis, “Scalable ASIP Implementation and Parallelization of a MIMO Sphere Detector,” in *Proceedings of the 11th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XI)*, Samos, Greece, 2011.

[31] E. Lee and D. Messerschmitt, “Pipeline interleaved programmable DSP’s: Architecture,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, pp. 1320–1333, 1987.

[32] A. Burg, M. Wenk, and W. Fichtner, “VLSI Implementation of Pipelined Sphere Decoding with Early Termination,” in *Proceedings of the 14th European Signal Processing Conference, 2006 (EUSIPCO 2006).*, Florence, Italy, 2006.

[33] J. Lee, “Area Efficient Pipelined VLSI Implementation of List Sphere Decoder,” in *Proceedings of the Asia-Pacific Conference on Communications, 2006 (APCC ’06).*, Busan, Korea, 2006.

[34] M. Winter, S. Kunze, E. P. Adeva, B. Mennenga, E. Matus, G. Fettweis, H. Eisenreich, G. Ellguth, S. Höppner, S. Scholze, R. Schüffny, and T. Kobori, “A 335Mb/s 3.9mm^2 65nm CMOS Flexible MIMO Detection-Decoding Engine Achieving 4G Wireless Data Rates,” in *Proceedings of the 59th IEEE International Solid-State Circuits Conference (ISSCC’12)*, San Francisco, USA, 2012, pp. 216–218.

[35] B. Noethen, O. Arnold, E. P. Adeva, T. T. Seifert, E. Fischer, S. Kunze, E. Matus, G. Fettweis, H. Eisenreich, G. Ellguth, S. Hartmann, S. Hoppner, S. Schiefer, J.-U. Schlusler, S. Scholze, D. Walter, and R. Schüffny, “A 105GOPS 36mm^2 Heterogeneous SDR MPSoC with Energy-Aware Dynamic Scheduling and Iterative Detection-Decoding for 4G in 65nm CMOS,” in *Proceedings of the 61th IEEE International Solid-State Circuits Conference (ISSCC’14)*, San Francisco, USA, 2014, pp. 188–189.

[36] F. Borlenghi, E. Witte, G. Ascheid, H. Meyr, and A. Burg, “A 2.78 mm^2 65 nm CMOS gigabit MIMO iterative detection and decoding receiver,” in *Proceedings of the 38th European Solid-State Circuits Conference (ESSCIRC’12)*, 2012, pp. 65–68.

[37] C. Studer, M. Wenk, and A. Burg, “VLSI implementation of hard- and soft-output sphere decoding for wide-band MIMO systems,” in *VLSI-SoC: Forward-Looking Trends in IC and Systems Design - IFIP Advances in Information and Communication Technology*, vol. 373, 2012, pp. 128–154.

[38] M. Shabany and P. Gulak, “A 675 Mbps, 4×4 64-QAM K-Best MIMO Detector in 0.13 μm CMOS,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 1, pp. 133–147, 2012.

[39] C.-H. Liao, T.-P. Wang, and T.-D. Chiueh, “A 74.8 mW Soft-Output Detector IC for 8×8 Spatial-Multiplexing MIMO Communications,” *IEEE Journal of Solid-State Circuits*, vol. 45, no. 2, pp. 411–421, 2010.

Esther P. Adeva received her M.Sc. in Electrical Engineering from the Miguel Hernndez University (Spain) in June 2009. She developed her diploma thesis at the TU Dresden, focusing on the analysis and implementation of tree-search-based MIMO detection algorithms. She is currently pursuing a PhD at the Vodafone Chair and she continues her research on the exploration of efficient architectures for MIMO detection.

Gerhard P. Fettweis earned his Ph.D. from RWTH Aachen in 1990. Thereafter he was at IBM Research and TCSI Inc., California. Since 1994 he is Vodafone Chair Professor at TU Dresden, Germany, with main research interest on wireless transmission and chip design. He is IEEE Fellow and an honorary doctorate of TU Tampere.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This document is a preprint of: E. Perez Adeva and G. Fettweis, “Efficient Architecture for Soft-Input Soft-Output Sphere Detection with Perfect Node Enumeration,” in IEEE Transactions on Very Large Scale Integration Systems (VLSI), Mar 2016. DOI:10.1109/TVLSI.2016.2526904

	This work 2015[a]	Borlenghi et al. [36] 2012[b]	Studer et al. [18] 2008[c]	Studer et al. [37] 2012[d]	Shabany et al. [38] 2012[e]	Liao et al. [39] 2010[f]	Studer et al. [28] 2011[g]
Antennas	4×4	$\leq 4 \times 4$	4×4	$\leq 4 \times 4$	4×4	$\leq 8 \times 8$	4×4
Modulation	64-QAM	$\leq$ 64-QAM	16-QAM	$\leq$ 64-QAM	64-QAM	$\leq$ 64-QAM	$\leq$ 64-QAM
Detector	DF-SD (TSD)	DF-SD (STS)	DF-SD (STS)	DF-SD (STS)	BF-SD (K-Best)	BeF-SD	MMSE PIC
Input/Output	SISO	SISO	SO	SO	HO	SO	SISO
Performance	close to *mlAPP*	close to *mlAPP*	close to *mlAPP*	close to *mlAPP*	close to ML	close to *mlAPP*	subopt.
Tech. (nm)	65	65	250 (65)	130 (65)	130 (65)	130 (65)	90 (65)
V_{DD} (V)	1.2	1.2	-	-	1.2	1.3 (1.2)	1.2
Area (kGE)	184	802	47	97.1	114	350	410
$f_{\mathrm{clk}}^{\max}$ (MHz)	450	135	71	383	258	198	568
τ @ $f_{\mathrm{clk}}^{\max}$		SO	SO	sustained	sustained	peak[i]	peak
(Mbps)	300	194	10	92	620	431.8	757
$\eta_{\tau,\mathrm{A}}$ (Mbps/kGE)	1.6	0.2	0.2 (0.8)	0.95 (1.9)	5.4 (10.9)	1.2[h] (2.3)	1.9 (2.6)
η_{E} (b/nJ)	2.17	1.1	-	-	5.9 (11.8)	7.4[h] (16.2)	4.0 (5.5)

TABLE II
COMPARISON OF MIMO DETECTORS ON CMOS AND THE VLSI DESIGN PROPOSED IN THIS WORK (VALUES IN BRACKETS RESULT FROM TECHNOLOGY SCALING TO 65NM).

HO - hard output / SO - soft output.
mlAPP - abbreviated form of *max-log-APP*.

a At $\mathrm{BER} = 10^{-5}$ (information block size of 9216b) over a Rayleigh fading channel, employing a 1/2-rate turbo decoder with 8 internal iterations and random interleaving.
b At $\mathrm{BLER} = 1\%$ (code block size of 1944b) over a Rayleigh fading channel, employing a 1/2-rate LDPC decoder with 10 internal iterations.
c At $\mathrm{FER} = 1\%$ (frame size of 1024b), employing a 1/2-rate convolutional code with a soft-input Viterbi decoder and random interleaving.
d At $\mathrm{BER} < 10^{-5}$ over a TGn type C (frequency-selective) channel, employing a 2/3-rate convolutional code with a (soft-input) Viterbi decoder and random interleaving.
e At $\mathrm{BER} = 10^{-3}$ (data packet size of 96b) over a Rayleigh fading channel.
f At $\mathrm{BER} = 10^{-5}$ (information block size of 9216b) over a Rayleigh fading channel, employing a 1/2-rate convolutional code and block interleaving.
g At $\mathrm{PER} = 10\%$ (packet block size of 864b) over a Rayleigh fading channel, employing a 5/6-rate convolutional code and block interleaving.
h Reported results correspond to a 4×4 MIMO configuration.

© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.