Area- and Throughput-Optimized VLSI Architecture of Sphere Decoding

Markus Wenk, Lukas Bruderer, Andreas Burg
Integrated Systems Laboratory
ETH Zurich, CH-8092 Zurich, Switzerland
E-mail: {mawenk,bruderer,apburg}@iis.ee.ethz.ch

Christoph Studer
Communication Technology Laboratory
ETH Zurich, CH-8092 Zurich, Switzerland
E-mail: studerc@nari.ee.ethz.ch

***Abstract*—Sphere decoding (SD) is a promising means for implementing high-performance data detection in multiple-input multiple-output (MIMO) wireless communication systems. In this paper, we focus on the register transfer level implementation of SD with minimum area-delay product for application in wideband MIMO communication systems, such as IEEE 802.11n, where multiple SD cores need to be instantiated. The basic architectural considerations and the proposed optimizations are explained based on hard-output SD, but are also applicable to soft-output SD. Corresponding VLSI implementation results (for both hard-output and soft-output SD) show an improvement in the area-delay product by almost 50 % compared to that of other SD implementations reported in the literature.**

I. Introduction

The ability to increase throughput and range without requiring more bandwidth or transmit power renders multiple-input multiple-output (MIMO) communication the key technology for wideband communication standards [1]. The MIMO gains come, however, at the cost of (often significant) complexity required for data detection. Maximum-likelihood (ML) detection provides excellent error-rate performance, but a straightforward implementation requires to exhaustively test all possible transmit symbols. For high spectral efficiencies, the exponential complexity increase of the number of candidate symbols (in the number of transmit antennas) is prohibitive, even for practical data-rates.

The sphere decoding (SD) algorithm [2] is one of the most promising methods for ML detection in MIMO systems, since its average complexity is far below that of an exhaustive search. The basic idea behind SD is to transform MIMO detection into a weighted tree-search problem, which is then solved efficiently by a branch-and-bound procedure. The main drawback of this approach lies in the fact that the decoding effort for SD is essentially determined by the *number of nodes* to be examined in that tree for each received symbol. For most VLSI implementations of SD, the number of visited nodes corresponds to the number of clock cycles required for each symbol [3]. This number depends on the channel and the noise realization. In the worst-case, all nodes in the tree must be examined, corresponding to the (often prohibitive) complexity of an exhaustive search. Since on-chip storage and higher-layer requirements limit the latency that may be inferred to support the processing of symbols for which the decoding effort lies far above the average, the worst-case complexity of SD renders its application in real-world systems difficult. This problem can be mitigated by limiting the maximum decoding effort through *early termination* of the decoding process, e.g., [4]. Such constraints, however, lead to a tradeoff between the maximum decoding effort and the receiver performance. A universally applicable VLSI architecture for a MIMO detector suitable for wideband MIMO systems must therefore be tailored to provide a straightforward solution to adjust this tradeoff and minimize overall silicon area for a given minimum performance requirement.

Outline and Contributions: In this paper, we describe the design and optimization of a SD core that is suitable for wideband MIMO systems. To this end, we first review the SD algorithm and we argue that the optimization target for each SD core in a wideband system differs from that usually employed in narrow-band MIMO systems, where a single SD-core can handle the throughput requirement (Sec. II). In Sec. III, we describe the register-transfer-level (RTL) architecture for hard-decision SD and propose a low-complexity approximation to the Schnorr-Euchner (SE) enumeration. We also introduce pipeline interleaving and analyze the level of pipelining required to yield the lowest area-delay product. In Sec. IV, we discuss our results and present a comparison to other SD implementations.

For better understanding and due to the limited page count, we focus on hard-output SD throughout the paper. The presented architecture, the proposed enumeration scheme, and pipeline interleaving can, however, also be applied to soft-output SD architectures (e.g.,single tree-search (STS) SD [5]). To support this claim, corresponding performance and implementation results are presented.

II. Sphere Decoding and Wideband MIMO Receiver Architecture

In the following, we introduce the MIMO system model, summarize the SD algorithm, and provide an overview of a wideband MIMO receiver architecture for the case where a single SD core is *insufficient* to meet the throughput-requirements associated with a high communication bandwidth.

A. MIMO Detection as a Weighted Tree-Search Problem

System model: We consider a MIMO system employing spatial-multiplexing with M_{T} transmit and $M_{\mathrm{R}} \geq M_{\mathrm{T}}$ receive antennas. The data to be transmitted is mapped to M_{T}-dimensional transmit vectors $\mathbf{s} \in \mathcal{O}^{M_{\mathrm{T}}}$, where $\mathcal{O}$ is

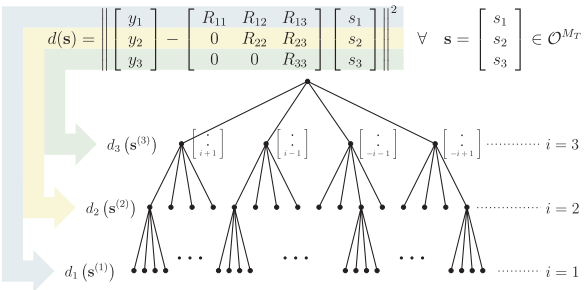

Fig. 1. MIMO detection as a weighted tree-search problem illustrated for $M_{\mathrm{T}} = 3$ and QPSK modulation.

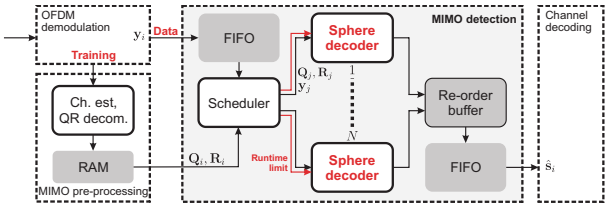

Fig. 2. System architecture of a wideband MIMO receiver.

the complex-valued scalar constellation. The baseband input-output relation, as seen by the MIMO detector, is given by

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

where $\mathbf{H}$ is the complex-valued $M_{\mathrm{R}} \times M_{\mathrm{T}}$ channel matrix and $\mathbf{n}$ is an i.i.d. circularly symmetric complex Gaussian noise vector of dimension M_{R}. The ML detection rule for the input-output relation in (1) is given by

$$\hat{\mathbf{s}} = \underset{\mathbf{s}\in\mathcal{O}^{M_{\mathrm{T}}}}{\arg\min} \left\|\mathbf{y} - \mathbf{H}\mathbf{s}\right\|^2. \tag{2}$$

Sphere decoding: SD [6] starts from the QR decomposition of the channel matrix $\mathbf{H} = \mathbf{Q}\mathbf{R}$, with $\mathbf{Q}$ being unitary of dimension $M_{\mathrm{R}} \times M_{\mathrm{T}}$ and $\mathbf{R}$ being $M_{\mathrm{T}} \times M_{\mathrm{T}}$ upper-triangular. This decomposition allows to rewrite (2) as

$$\hat{\mathbf{s}} = \underset{\mathbf{s}\in\mathcal{O}^{M_{\mathrm{T}}}}{\arg\min} \left\|\hat{\mathbf{y}} - \mathbf{R}\mathbf{s}\right\|^2 \tag{3}$$

with $\hat{\mathbf{y}} = \mathbf{Q}^H\mathbf{y}$. Thanks to the upper-triangularity of $\mathbf{R}$, the minimization problem (3) can be interpreted as a weighted tree-search problem where the nodes of the tree on level i are associated with a partial symbol vector $\mathbf{s}^{(i)} = [\, s_i \cdots s_{M_{\mathrm{T}}} \,]^T$ and with a corresponding partial Euclidean distance (PED) $d_i(\mathbf{s}^{(i)})$. Fig. 1 illustrates the corresponding weighted tree for a MIMO system with $M_{\mathrm{T}} = M_{\mathrm{R}} = 3$ using QPSK modulation. When starting from the root of the tree (at level $i = M_{\mathrm{T}} + 1$ with $d_{M_{\mathrm{T}}+1} = 0$), the PEDs can efficiently be computed in a recursive manner according to

$$d_i(\mathbf{s}^{(i)}) = d_{i+1}(\mathbf{s}^{(i+1)}) + |b_{i+1} - R_{i,i}s_i|^2 \tag{4}$$

using the definition

$$b_{i+1} = \hat{y}_i - \sum_{k=i+1}^{M_{\mathrm{T}}} R_{i,k}s_k \tag{5}$$

when proceeding from a parent node on level $i+1$ to one of its children on level i. The ML solution corresponds to the path through the tree leading to the leaf associated with the smallest PED. To find this leaf, SD traverses the tree in a depth-first manner. Complexity reduction (compared to an exhaustive search) is achieved by pruning those nodes from the tree for which $d_i(\mathbf{s}^{(i)})$ is larger than a *radius* $r > 0$. We use a technique known as radius-reduction [6], which initializes the radius to $r \leftarrow \infty$ (prior to detection) and performs the radius update $r \leftarrow d_1(\mathbf{s}^{(1)})$ whenever a leaf-node $\mathbf{s}^{(1)}$ is reached. In the following, we refer to the condition $d_i(\mathbf{s}^{(i)}) < r$ as the *sphere constraint* (SC).

B. Wideband MIMO Receiver Architecture

In wideband MIMO systems, such as IEEE 802.11n, a single SD core is usually insufficient to support both the bandwidth and the (error-rate) performance requirements, even for advanced process technologies. Hence, multiple SD-cores are necessary to meet the associated throughput and performance requirements.

Architecture overview: The high-level system architecture of a wideband MIMO receiver based on SD is illustrated in Fig. 2. The data flow starts with the OFDM *demodulation*. During a training phase, received training symbols are delivered to a *MIMO preprocessing* unit. This unit estimates the channel matrices $\mathbf{H}$ and performs necessary pre-computations on $\mathbf{H}$ (i.e., the QR decomposition). During the data phase, the *demodulation* unit and the *MIMO preprocessing* unit forward the received vectors and the results of the pre-computation of the corresponding channel matrices to the *MIMO detector* at a constant arrival rate, which is essentially given by the communication bandwidth of the system. In the *MIMO detector*, the information required to decode a symbol is first queued in a FIFO. A scheduler reads the entries of the FIFO and forwards them to the next idle SD core together with a runtime constraint (i.e., a constraint on the number of nodes that are allowed to be examined by SD). When the FIFO fills up, the runtime constraints are reduced to ensure that no data is lost. Note that this reduction degrades the quality of the detection.[1] The outputs from the N SD cores are collected and reordered since the variable runtime may cause decoded symbols to arrive out-of-order. The reordered symbol estimates are then forwarded to the channel-decoding block.

Implications on SD core optimization: With the above described architecture, the average decoding effort, i.e., the number of visited nodes that can be allocated for decoding of each symbol is determined by

$$\Phi \propto \frac{N}{T_{\mathrm{c}}B} \quad [\text{nodes}]$$

where B denotes the bandwidth of the system (i.e., the arrival-rate of the symbols to be decoded), T_{c} is the clock period of a SD core (assuming one node in the tree is checked in each cycle), and N is the number of SD instances. At the

[1]The particularities of the scheduling mechanism and the associated performance tradeoffs are outside the scope of this paper, which focuses on the implications on the RTL optimization of the SD cores.

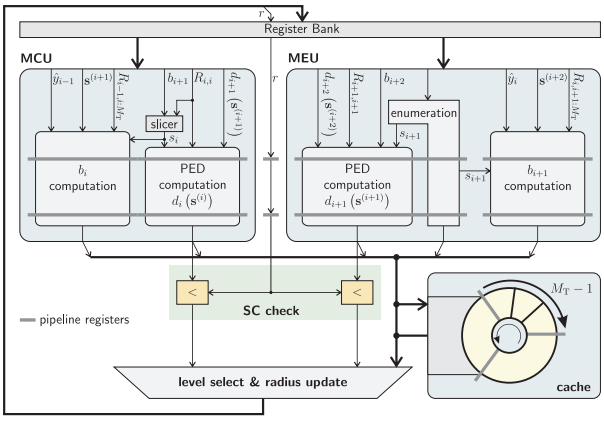

Fig. 3. High-level block diagram of the SD architecture. The shaded registers and the ring buffer (in the level cache) are only required when pipeline interleaving is applied.

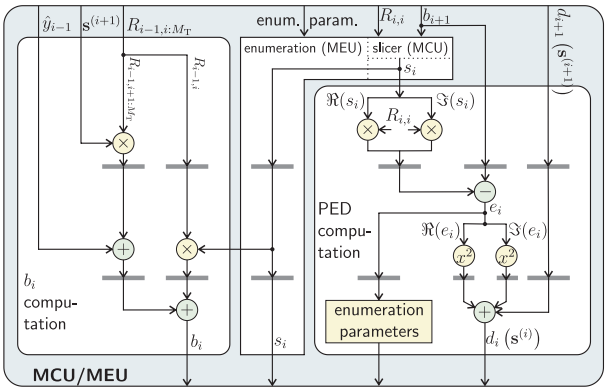

Fig. 4. RTL block diagram of the MCU and MEU. The shaded registers are only required when pipeline interleaving is applied.

system-level, the performance/complexity tradeoff can now be adjusted by the choice of N. The resulting area of such a system corresponds to $A_{\text{tot}} = NA_{\text{SD}}$, where A_{SD} denotes the silicon area of a single SD core. For large N, the overall silicon area for a guaranteed number of visited nodes $\bar{\Phi}$ that can be used for decoding received symbols, is given by

$$A_{\text{tot}} \propto \bar{\Phi} B \rho_{\text{SD}} \quad \text{with} \quad \rho_{\text{SD}} = T_{\text{c}} A_{\text{SD}}. \tag{6}$$

From (6), it follows that if multiple SD cores are necessary to meet the performance requirements of a wideband MIMO system, the focus for the optimization of the SD core shifts from minimizing the area or maximizing the throughput to minimizing the corresponding *area-delay (AT-)product* ρ_{SD}.

III. VLSI Architecture of Hard-Output SD

On the first level of hierarchy, the proposed SD architecture is similar to the one proposed in [3]. In the following, we summarize this architecture and describe a number of optimizations that result in an improved AT-product compared to previously reported SD-implementations.

A. High-level Architecture

Fig. 3 shows the high-level block diagram of the proposed SD circuit. The design is comprised of a *metric computation unit* (MCU), a *metric enumeration unit* (MEU), an *SC check* unit, a *level-select multiplexer*, and a *cache*.

The MCU is responsible for the forward-iteration of the depth-first tree-traversal. In the implementation [3], this forward iteration includes the *sequential* evaluation of (5) and the computation of the PED in (4). In the present circuit (cf. Fig. 4), a slicer-unit performs a decision on the nearest constellation point and the MCU computes b_i (instead of b_{i+1}) in parallel to the PED of level $i+1$ as proposed in [7]. The resulting b_i is then used in the next iteration (provided that the SC is met); this optimization reduces the critical path without the need for additional hardware.

The MEU operates in parallel to the MCU. While the MCU is processing a node on layer i, the MEU selects the next-best constellation point on layer $i+1$ according to an enumeration scheme and computes its PED. Hence, once the SD algorithm needs to move upward in the tree, the MCU can directly start the next forward iteration as all required intermediate results have already been computed beforehand by the MEU. The RTL architecture of the MEU (cf., Fig. 4) is similar to the one of the MCU. However, the slicer-unit that determines the closest CP is replaced by an enumeration unit that determines which CP should be considered next on layer $i+1$.

The cache stores intermediate results for each level computed by the MEU and the MCU. The SC check is carried out immediately after the computation of the new PEDs. MEU, MCU, level cache, and the result of the SC check decide on which layer the SD algorithm proceeds next. If a leaf that fulfills the SC is found, the radius is updated. In this case an additional clock cycle is necessary, as the PEDs in the level cache need to be checked against the new radius.

B. Enumeration Strategy

The enumeration strategy (implemented by the *enumeration unit* in the MEU) defines the order in which the children of a node are visited. Radius reduction (cf. Section II-A) is most efficient in combination with the Schnorr-Euchner (SE) enumeration [8], which visits the children of a node in ascending order of their PEDs. An important advantage of this enumeration strategy is that leaves that are more likely to lead to the ML solution are found early, which expedites the pruning of the tree. Moreover, enumeration of the children of a node can terminate as soon as the first child violates the SC.

Implementation of Schnorr-Euchner enumeration: For each visited node, SE enumeration is comprised of two types of operations: The first operation is to initialize the enumeration of the children by identifying the child associated with the smallest PED. This task can easily be accomplished by comparing b_{i+1} in (5) to a number of decision boundaries, i.e., by performing a slicing operation in the MCU of Fig. 3. The second type of operation is to enumerate the remaining children in ascending order of their PEDs, which is a non-trivial task for complex-valued constellations. In order to minimize the AT-product ρ_{SD} of the SD core, an efficient implementation of this operation is of paramount importance.

Exhaustive enumeration: Exhaustive enumeration is a straightforward (but rather inefficient) solution to perform SE

enumeration [3]. The idea is to first compute the PEDs of *all* children of a node. During enumeration, a min-search (limited to the subset of children that have not yet been visited) identifies the next child. The main drawbacks of this solution are i) the area requirement to compute the PEDs of all children of a node, ii) the need to store them in the cache, and iii) the fact that a min-search is costly in terms of area and timing, especially for higher order constellations.

Subset enumeration: More elaborate solutions for SE enumeration were presented in [3], [6], and [9]. The main idea of these approaches is to divide the complex-valued (two-dimensional) constellation into one-dimensional subsets which only require to compute and store one PED per subset and consequently also reduce the complexity of the min-search. Unfortunately, the number of required subsets gets large for higher-order modulation schemes, which has considerable impact on circuit area and timing of implementations supporting 64-QAM.

C. Approximate SE Enumeration

The goal of considering approximations to SE enumeration is to perform the enumeration without the need for computing, caching, and comparing PEDs for multiple candidate CPs on the same level. Such, approximations based on geometrical considerations were first proposed in [10] and [11]. The basic idea is to store predefined enumeration sequences in one or multiple look-up tables (LUTs). A fixed sequence is chosen based on several geometric rules that analyze the position of the received point b_{i+1} relative to the closest CP. The accuracy of these techniques can be adjusted by the number and complexity of the associated selection criteria together with the number of predefined LUTs. The major drawback of this approach is the rather poor scaling behavior of the size of the LUTs required for higher-order modulation schemes.

Ordered $l^{\widetilde{\infty}}$-Norm Enumeration: In the following, we describe an approximation to SE enumeration that can be implemented efficiently in hardware without the need for LUTs and therefore, scales well to higher-order constellations (i.e., constellations including and beyond 64-QAM).

Inspired by the $l^{\widetilde{\infty}}$-norm SD algorithm [3], [12], we define the $l^{\widetilde{\infty}}$-norm of a vector $\mathbf{x}$ according to $\|\mathbf{x}\|_{\widetilde{\infty}} = \max\{|\Re(\mathbf{x})|, |\Im(\mathbf{x})|\}$, where $\Re(\mathbf{x})$ and $\Im(\mathbf{x})$ denote the real and imaginary part of the entries of $\mathbf{x}$, respectively. The starting point for the enumeration is trivially determined by the closest CP (in Euclidean distance). However, the CPs are enumerated[2] according to their $l^{\widetilde{\infty}}$-norm distance

$$\begin{aligned} d_{\widetilde{\infty}} &= |b_{i+1} - R_{i,i}s_i|_{\widetilde{\infty}} \\ &= \max\{|\Re(b_{i+1} - R_{i,i}s_i)|, |\Im(b_{i+1} - R_{i,i}s_i)|\} \end{aligned}$$

from b_{i+1}. To this end, the area around the closest CP is first subdivided into eight sectors as illustrated in the lower right corner of Fig. 5. The sector containing b_{i+1} is identified with simple geometric rules to define the second CP in the enumeration and the direction for the ordered $l^{\widetilde{\infty}}$-norm enumeration. CPs with identical $l^{\widetilde{\infty}}$-norm form one-dimensional subsets.

[2]We use the $l^{\widetilde{\infty}}$-norm only for enumeration, whereas the algorithm in [3], [12] also uses it for distance computations.

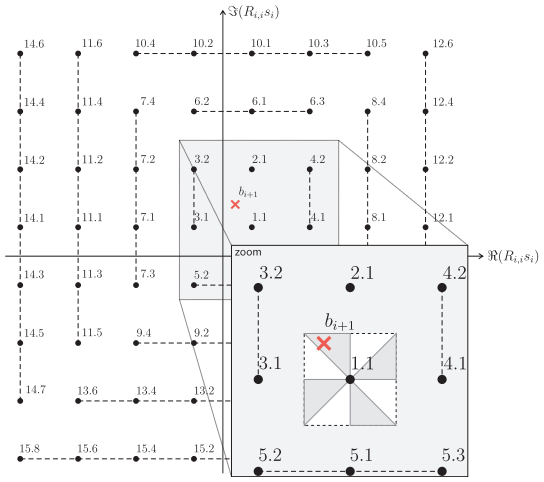

Fig. 5. Principle of ordered $l^{\widetilde{\infty}}$-norm enumeration for 64-QAM modulation.

All nodes within the same subset are processed before the algorithm selects the next subset. In the example provided in Fig. 5, the processing order of the one-dimensional subsets is illustrated by the leading number attached to each CP. Within each subset, zig-zag enumeration is applied around the CP closest to b_{i+1}; this is illustrated by the trailing number in Fig. 5. The members of each subset are returned in SE order and subsets are enumerated in order of increasing $l^{\widetilde{\infty}}$-norm.

RTL architecture: For the RTL implementation, the above-described enumeration algorithm can be split into two basic tasks: i) tracking of the position, size, and orientation of the linear subsets, and ii) zig-zag enumeration within the subsets and checking for the boundaries of the finite-size modulation alphabet. Both tasks can be implemented using simple combinational logic, comparators, and three counters. Hence, the required circuit complexity is low.

Impact on error-rate performance and number of visited nodes: Besides a reduction of the hardware complexity, the approximation to the SE enumeration has an impact on the number of visited nodes and on the (error-rate) performance of the SD algorithm. The reason for this impact lies in the fact that the approximation does not guarantee that the children of a node are always enumerated strictly in ascending order of their PEDs (only the first three CPs always correspond to the first three CPs obtained by SE enumeration). Hence, numerical simulations are performed to verify that the error-rate implementation loss due to the approximation of the enumeration is low and that the number of visited nodes does not increase substantially. Corresponding results[3] for hard- and soft-output SD are shown in Fig. 6. It can be seen that the loss in terms of the coded frame-error rate (FER) performance is negligible and that the number of visited nodes with $l^{\widetilde{\infty}}$-norm

[3]We consider coded (rate $2/3$ convolutional code, constraint length 7, generator polynomials [133_o 171_o], and random interleaving across space and frequencies) MIMO-OFDM transmission with $M_{\mathrm{R}} = M_{\mathrm{T}} = 4$, 64-QAM (Gray mapping), 64 OFDM tones. One frame corresponds to 1536 coded bits. A TGn type C [13] channel model is used. We assume perfect channel state information at the receiver and employ minimum mean-square error sorted QR decomposition (MMSE-SQRD) [14] for SD-preprocessing. The SNR is per receive antenna.

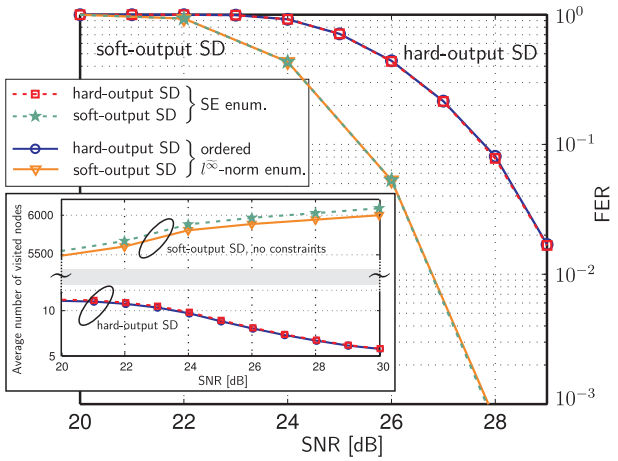

Fig. 6. FER performance and average number of visited nodes for ordered $l^{\widetilde{\infty}}$-norm and SE enumeration ($M_{\mathrm{T}} = M_{\mathrm{R}} = 4$ using 64-QAM).

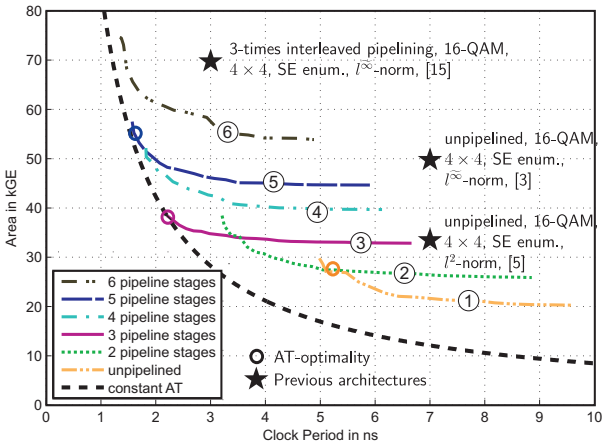

Fig. 7. AT-diagram of hard-output SD with different number of pipeline stages. The optimal synthesis results (in terms of the area/delay tradeoff) are highlighted by circles and implementation results of previous architectures are indicated by stars. All designs are scaled to 0.13 µm CMOS technology.

enumeration is slightly less (i.e., approximately 5 %) compared to exact SE enumeration.

D. Pipeline Interleaving

Pipelining cannot directly be applied to SD due to the first-order feedback path present in the architecture. Nevertheless, symbol-wise pipeline interleaving can be used to shorten the critical path. The main idea of this approach is to process multiple (independent) symbol-vectors in parallel within the same circuit. This basic idea has already been suggested for SD [15], [16], but neither details on suitable locations of the pipeline registers, nor a discussion of the number of pipeline stages yielding the optimal AT-product has been provided.

Fig. 3 and Fig. 4 show the location of the pipeline registers (in light grey) in the RTL architecture for three pipeline stages. The location was manually chosen to approximately balance the path delays between the pipeline stages and register-retiming during synthesis was allowed for further optimization. Besides adding the pipeline registers in the datapath, the level cache in Fig. 3 was extended to a ring-buffer in which each entry is associated with one of the symbols in the pipeline and corresponds to one instance of the original level cache.

IV. Implementation Results and Comparison

A. Results for Hard-Output SD

The AT-diagram in Fig. 7 shows the implementation results of hard-output SD with ordered $l^{\widetilde{\infty}}$-norm enumeration and pipeline interleaving with different number of pipeline stages[4]. The proposed architectures have been implemented with support for multiple modulation schemes (BPSK, QPSK, 16-QAM, and 64-QAM) and for up to four spatial streams.

The architecture with three pipeline stages achieves the best AT-product. But also the architectures with more than three pipeline stages come close to AT-optimality, whereas the architectures with fewer pipeline stages are clearly outperformed in terms of hardware-efficiency. For comparison, implementation results of previous hard-output SD implementations are also included in Fig. 7 (the results are also summarized in Tbl. I). It can be seen that the proposed unpipelined hard-output SD architecture outperforms previous unpipelined designs by a least 23% in terms of area and by at least 28% in terms of clock frequency[5]. Furthermore, the AT-product [kGE/MHz] of the proposed architecture with pipeline interleaving is more than a factor two better than that of a previously reported implementation [15] with pipeline interleaving.

B. Application to Soft-Output STS-SD

The proposed enumeration scheme and pipeline interleaving can also be applied to soft-output SD. The corresponding architecture is based on the soft-output single tree-search (STS) algorithm proposed in [5]. Fig. 6 demonstrates that also for STS-SD, the FER performance loss due to the proposed $l^{\widetilde{\infty}}$-norm enumeration scheme is negligible and the average number of visited nodes is slightly reduced. Implementation results for soft-output STS-SD with the proposed $l^{\widetilde{\infty}}$-norm enumeration scheme are shown in Tbl. II and compared to previous soft-output detection implementations. The presented implementation is clearly superior in terms of area and clock frequency compared to the soft-output detector shown in [11]. The original implementation of soft-output STS-SD in [5] only supports 16-QAM modulation, which is the main reason for the smaller area in the unpipelined case. For hard-output SD, pipeline interleaving with three pipeline stages showed to be pareto-optimal. As the additional units required for STS-SD do not influence the critical path, STS-SD was also implemented with three pipeline stages. Tbl. II shows that the AT-product is improved by more than 30 % due to pipeline interleaving.

C. The Case for Multiple SD-Cores

In Section II, we argued that a single SD core is insufficient to meet the bandwidth and error-rate performance requirements of modern wireless communication standards

[4]The results were obtained by synthesizing the RTL description in VHDL with different timing constraints.

[5]The clock frequencies of the designs are scaled to a 0.13 µm CMOS technology.

TABLE I
IMPLEMENTATION RESULTS AND COMPARISON OF HARD-OUTPUT SD.

	[3]	[5]	[15]	This work		
CMOS Tech.	0.25 µm	0.25 µm	0.13 µm	0.13 µm		
Antennas	4×4	4×4	4×4	1×1 to 4×4		
Modulation	16-QAM	16-QAM	16-QAM	BPSK to 64-QAM		
Norm	$l^{\widetilde{\infty}}$	l^2	$l^{\widetilde{\infty}}$	l^2		
Enumeration	SE	SE	SE	ordered $l^{\widetilde{\infty}}$-norm		
Pipeline stages	no	no	3×	no	3×	5×
Area[a] [kGE]	50	34.4	70	27.1	38.4	55.3
Freq. [MHz]	137[b]	140[b]	333	196	455	625
[kGE/MHz]	0.37	0.25	0.21	0.14	0.08	0.09
Throughput for $D_{\text{avg}} = 7$[c] [Mbps]	470	480	1141	672	1560	2143

TABLE II
IMPLEMENTATION RESULTS AND COMPARISON OF SOFT-OUTPUT SD FOR A 4×4 MIMO-OFDM SYSTEM.

	[11]	[5]	This work	
CMOS Technology	0.13 µm	0.25 µm	0.13 µm	
Modulation	64-QAM	16-QAM	BPSK to 64-QAM	
Enumeration	tabular	SE	ordered $l^{\widetilde{\infty}}$-norm	
Pipeline stages	no	no	no	3×
Area[a] [kGE]	350	56.8	70.4	97.1
Max. frequency [MHz]	198	137[b]	183	383
AT-product [kGE/MHz]	1.77	0.41	0.38	0.25

[a] One GE corresponds to the area of a two-input drive-one NAND gate.

[b] Scaled from 0.25 µm to 0.13 µm by multiplying by 0.25/0.13.

[c] D_{avg} denotes the average number of nodes used for block processing [4].

such as IEEE 802.11n, where a throughput of 600 Mbps is required. From Tbl. I, we observe that hard-output SD meets the throughput requirement when early-termination and block-processing according to [4] are applied.

For soft-output STS-SD, the number of visited nodes is significantly increased: from seven for hard-output SD to a least 100 for soft-output STS-SD[6]. To illustrate the necessity for multiple soft-output STS-SD cores, we hypothetically assume $D_{\text{avg}} = 100$ for 64-QAM modulation. The throughput of one STS-SD core is then 92 Mbps. To fulfill the throughput requirement of 802.11n, up to seven STS-SD cores are required.

V. CONCLUSION

To meet the throughput and latency requirements of wide-band systems (e.g., IEEE 802.11n) with sphere decoding (SD), multiple detection cores need to be instantiated. Therefore, the efficiency or the area-delay product of a single SD core needs to be optimized. To this end, two techniques, namely ordered $\ell^{\widetilde{\infty}}$-norm enumeration and pipeline interleaving, have been proposed. The enumeration scheme significantly reduces circuit area and the critical path-delay. Simulations also showed, that the performance loss due to the new enumeration scheme is negligible. With pipeline interleaving multiple independent symbol vectors are processed in parallel and the available hardware resources are better exploited. A design-space exploration with different number of pipeline stages revealed that the architecture with three pipeline stages is the most efficient. With these two approaches, the area-delay product is improved by almost 50 % compared to that of other SD implementations. Finally, we showed that both approaches can also be applied to soft-output SD.

[6] A in-depth evaluation for the number of visited nodes for 64-QAM goes beyond the scope of this paper and involves optimizations of different parameters (e.g., clipping level, run-time constraint, SNR requirement). We expect, based on simulation results, a hundred to a few hundreds of nodes to be visited. For 16-QAM, the numbers have been presented in [5].

ACKNOWLEDGMENT

The authors thank H. Friederich, P. Luethi, N. Felber, W. Fichtner, and H. Bölcskei for their support during the design of the SD architecture. This work was partially supported by the STREP project MASCOT (IST-026905) within the Sixth Framework of the European Commission and by the Swiss National Science Foundation project No. PP002-119052.

REFERENCES

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

[2] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Mathematics of Computation*, vol. 44, no. 170, pp. 463–471, Apr. 1985.

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

[4] A. Burg, M. Borgmann, M. Wenk, C. Studer, and H. Bölcskei, "Advanced receiver algorithms for MIMO wireless communications," in *DATE '06: Proc. of the conf. on design, automation and test in Europe*, Mar. 2006, pp. 593–598.

[5] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithms and VLSI implementation," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 2, pp. 290–300, Feb. 2008.

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

[7] E. M. Witte, F. Borlenghi, G. Ascheid, R. Leupers, and H. Meyr, "A scalable VLSI architecture for soft-input soft-output depth-first sphere decoding," 2009, available online at http://arxiv.org/abs/0910.3427.

[8] E. Agrell, T. Eriksson, A. Vardy, and K. Z. r, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[9] 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 *Proc. 17th ACM Great Lakes Symposium on VLSI*, 2007, pp. 200–203.

[10] B. Mennenga and G. Fettweis, "Search sequence determination for tree search based detection algorithms," in *IEEE Sarnoff Symposium*, Apr. 2009, pp. 1–6.

[11] L. Chun-Hao, W. To-Ping, and C. Tzi-Dar, "A 74.8 mw soft-output detector IC for 8×8 spatial-multiplexing MIMO communications," *IEEE J. Solid-State Circuits*, vol. 45, no. 2, pp. 411–421, Feb. 2010.

[12] D. Seethaler and H. Bölcskei, "Performance and complexity analysis of infinity-norm sphere-decoding," *IEEE Trans. Inf. Theory*, vol. 56, no. 3, pp. 1085–1105, Mar. 2010.

[13] V. Erceg and et al., *TGn channel models*. IEEE 802.11-03/940r4, May 2004.

[14] D. Wübben, R. Böhnke, V. Kühn, and K.-D. Kammeyer, "Mmse extension of v-blast based on sorted QR decomposition," in *IEEE 58th Vehicular Technology Conference*, Oct. 2003, pp. 508–512.

[15] A. Burg, M. Wenk, and W. Fichtner, "VLSI implementation of pipelined sphere decoding with early termination," *Proceedings of the European Signal Processing Conference*, Sep. 2006, invited paper.

[16] J. Lee, S.-C. Park, and S. Park, "A pipelined VLSI architecture for a list sphere decoder," in *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS'06)*, Sep. 2006, pp. 397–400.