Noname manuscript No.
(will be inserted by the editor)

MIMO Detector for LTE/LTE-A Uplink Receiver

Tuomo Hänninen · Johanna Ketonen · Markku Juntti

Received: date / Accepted: date

Abstract We propose a carefully selected receiver structure, detector and detector implementation architecture for multiple-input multiple-output (MIMO) uplink base station receiver for fourth generation (4G) wireless cellular systems. First, we compare different receiver algorithms and structures for single-carrier frequency-division multiple access (SC-FDMA) uplink transmission to get a good understanding of the performance and complexity of these algorithms and their suitability for practical realization. One of those structures, namely the frequency domain MMSE equalization with sphere detection (SD), is proposed for implementation. The receiver consists of separate stages for inter-symbol interference (ISI) and inter-antenna interference (IAI) mitigation in frequency selective MIMO channels. Frame error rate (FER) performance is studied via simulations in realistic wireless channels and practical system parameters. K-best SD is selected as a detector algorithm for this receiver. There are several publications proposing a sort-free architecture for tree search type of detectors. Both a conventional K-best architecture and a sort-free architecture are implemented on a Xilinx Virtex-6 field-programmable gate array (FPGA) using High Level Synthesis (HLS) tool. Both architectures support 4×4 MIMO with 64-level modulation (64-QAM). Complexity results confirm that avoiding the sorter is not always recommended. The benefit of sort-free architecture depends on the system parameters.

Keywords K-best · MIMO detector · LTE · FPGA · High Level Synthesis (HLS)

Tuomo Hänninen, Johanna Ketonen, Markku Juntti
University of Oulu, Finland
E-mail: {tuomo.hanninen, markku.juntti}@oulu.fi

1 Introduction

Third generation partnership project (3GPP) Long Term Evolution (LTE) [1] uses single-carrier frequency-division multiple access (SC-FDMA) as the uplink transmission scheme [2]. The same is true for its further evolution LTE-Advanced (LTE-A) [3]. SC-FDMA has been selected in the uplink instead of orthogonal frequency division multiplexing (OFDM) mainly because of its reduced peak-to-average power ratio to reduce the mobile transmitter cost by allowing cheaper power amplifiers [4]. A consequence of the SC transmission is the fact that inter-symbol interference (ISI) is unavoidably introduced and an equalizer is needed in the receiver. Frequency domain (FD) linear minimum mean square error (MMSE) receivers have been studied extensively for a single carrier transmission [5,6] and are probably still the most predominant in practical realizations. Also more advanced turbo based receivers have been considered in basic research [7–10], but their complexity is still typically too high for most commercial products.

Multiple-input multiple-output (MIMO) communications [11,12] has been standardized also for LTE uplink to increase the peak data rates. The LTE-A standard specifies up to four transmit antennas in the user terminal. Similar to any spatial multiplexing based MIMO transmission, inter-antenna interference (IAI) is induced and a tailored spatial equalizer is required in the receiver. Both linear and nonlinear receiver structures have been extensively considered for MIMO receivers with an emphasis on ones operating in OFDM systems, wherein ISI is not a problem. In particular, different variants of so called *sphere detector* (SD) [13], which calculate the maximum likelihood (ML) solution with reduced complexity, have received a lot of attention in the literature. The *list sphere detector* (LSD)

[14] is useful in practical systems employing forward error control (FEC) coding, because it approximates the maximum *a posteriori* probability (MAP) detector producing soft outputs for the channel decoder. Implementations of different LSD versions and other tree-search algorithms have been considered earlier mostly in the the downlink MIMO-OFDM context in [15–20].

The introduction of the spatial multiplexing based MIMO concept to the LTE-A uplink means that the base station receiver is encountering further challenges. It must cope with both frequency-selectivity induced ISI and spatial domain IAI. This calls for joint consideration of both problems. The algorithm complexity will unavoidably increase and also the realization becomes a major challenge. The stringent real time and latency requirements of receiver processing make it necessary to perform joint algorithm and architecture optimization. The most conventional MIMO receiver structure consists of the frequency domain linear MMSE equalizer optimized for both ISI and IAI. It performs reasonably well, but suffers significant error rate increase for large numbers of antennas [21, 22]. A promising way to improve its error rate is to apply separate stages for ISI and IAI mitigation [23]; a similar idea has earlier been applied, e.g., in [24] in the wideband code-division multiple-access (WCDMA) context. The MMSE filter can be first applied to suppress the ISI as would be done in conventional single-input single-output SC-FDMA communications. Another, in general nonlinear, equalizer stage is then subsequently used for MIMO detection, i.e, equalizing the IAI between the spatial streams.

Our objective is to propose an efficient receiver structure, SD algorithm and SD implementation architecture for real LTE-A systems and their SC-FDMA based uplink base stations. We proposed a new receiver structure *frequency domain linear MMSE filter with sphere detection* for SC-FDMA uplink transmission and compared this to conventional *frequency domain linear MMSE equalization with soft demodulation* receiver structure in [23]. Two different tree search algorithms were considered for the latter one, namely the K-best [25] LSD algorithm and the selective spanning with fast enumeration (SSFE) algorithm [26]. In this paper, we have revised the frame error rate (FER) performance and complexity analysis of these receivers. As a completely new result, the K-best LSD algorithm used for the sphere detection part of the later receiver structure is chosen for more detailed analysis. Two different architectures for this K-best LSD algorithm are implemented on a Xilinx field-programmable gate array (FPGA) using High Level Synthesis (HLS) tool to get good understanding of their complexity and analyze whether so-called sort-free architecture has gain over straightforward implementation using common insertion sorter. The HLS tool enables the implementation and comparison of different architectures in relatively short time. The architecture optimization can be done in the C language level, which gives a clear benefit in terms of design time and effort. The area efficiency could probably be further optimized with traditional design approach but HLS is well suited for this type of architecture evaluation.

2 System Model

A single carrier based vertically encoded MIMO transmission system with T transmit and R receive antennas is considered. The system model is presented in Figure 1. The encoded data stream is interleaved and modulated into symbols. After the parallel-to-serial conversion, a cyclic prefix (CP) is added. At the receiver, a K-point DFT is performed and the symbols from the allocated carriers are selected. After the frequency domain equalization, the symbols are transformed into time domain and the detector is used to calculate the bit log-likelihood ratios (LLR) for the decoder.

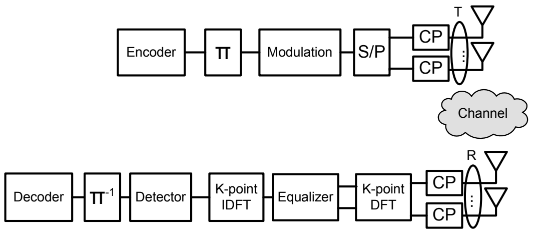

Fig. 1 The vertically encoded single carrier MIMO system model.

After CP removal, the received signal vector $\mathbf{r} \in \mathbb{C}^{RK}$ can be expressed as

$$\mathbf{r} = \mathbf{H}\mathbf{x} + \mathbf{v}, \tag{1}$$

where $\mathbf{v} \in \mathbb{C}^{RK}$ is independent and identically distributed circularly symmetric complex Gaussian noise with variance σ^2 and zero mean, $\mathbf{x} \in \mathbb{C}^{TK}$ is the transmitted signal, $\mathbf{H} \in \mathbb{C}^{RK \times TK}$ is the circulant block channel matrix and K is the length of the DFT. The channel matrix $\mathbf{H}$ can be written as

$$\begin{bmatrix} \mathbf{H}^{1,1} & \cdots & \mathbf{H}^{1,T} \\ \vdots & \cdots & \vdots \\ \mathbf{H}^{R,1} & \cdots & \mathbf{H}^{R,T} \end{bmatrix},$$

where $\mathbf{H}^{r,t} \in \mathbb{C}^{K \times K}$ is a channel submatrix between tth transmit and rth receive antenna.

3 SC-FDMA MIMO Receivers

The most conventional MIMO receiver structure consists of the frequency domain linear MMSE equalizer with a soft demodulator. The ISI and IAI terms are both counteracted by the same linear MMSE filter. The structure is illustrated in Figure 2. The soft demodulator is used to calculate the log-likelihood ratios for the decoder. No further IAI suppression is performed in the soft demodulator as the LLRs are calculated separately for each stream. The structure is well known in the literature. It performs well but not optimally.

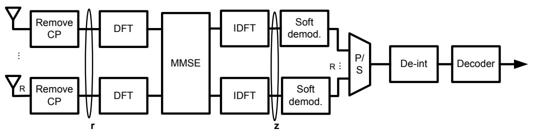

Fig. 2 Receiver with linear MMSE equalization and soft demodulation.

One option to improve the performance of the conventional MMSE based MIMO receiver would be a time domain sphere detector with combined mitigation of ISI and IAI. The time domain channel matrix for the QR decomposition (QRD) would have dimensions of $R \times T \times L$, where L is the length of the channel and R and T are the number of receive and transmit antennas, respectively. This means that the complexity explodes when the number of antennas and channel length increase. The time domain sphere detector would in principle give good communication performance, but it would be too complex for most practical implementations in the 4×4 MIMO case with the processing power available with the current technology.

Another option to improve the performance of a MIMO receiver, is the frequency domain MMSE filter with sphere detection. Therein, the ISI and IAI mitigation are performed in separate stages and complexity is much lower than that with time domain SD processing. This is the approach considered in the sequel. The MMSE filter is first applied to suppress the ISI like in conventional single-antenna SC-FDMA communications. Its operation can also be interpreted as a channel shortening filter, producing a shortened channel matrix for the sphere detector. The sphere detector is subsequently used for MIMO detection, i.e, removing the IAI between the spatial streams. Several different tree search algorithms could be used to perform the MIMO detection in this receiver structure. Here we consider the K-best LSD and SSFE algorithms as a candidate for our implementation. The receiver structure for vertically encoded $R \times T$ MIMO is illustrated in Figure 3. The MMSE filter and two different tree search algorithms are described in more detail below.

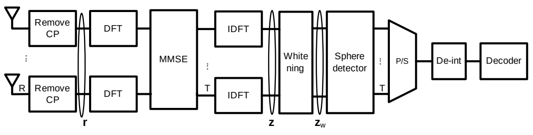

Fig. 3 Receiver with sphere detection.

3.1 MMSE Filter

The linear MMSE filter coefficients are derived to cancel ISI and the filter coefficients $\mathbf{\Omega} \in \mathbb{C}^{RK \times RK}$ can be determined according to the following criterion [10]

$$\mathbf{\Omega} = \arg\min_{\mathbf{\Omega}} \underbrace{\mathrm{tr}\{E_{\mathbf{x},\mathbf{v}}\{(\mathbf{F}_R^{-1}\mathbf{\Omega}^{\mathrm{H}}\mathbf{F}_R\mathbf{r} - \tilde{\mathbf{H}}\mathbf{x})(\mathbf{F}_R^{-1}\mathbf{\Omega}^{\mathrm{H}}\mathbf{F}_R\mathbf{r} - \tilde{\mathbf{H}}\mathbf{x})^{\mathrm{H}}\}\}}_{\mathbf{e}}, \tag{2}$$

where $\mathbf{e}$ is the mean square error (MSE), the expectation $E\{\cdot\}$ is respect to $\mathbf{x}$ and $\mathbf{v}$, $\tilde{\mathbf{H}} \in \mathbb{C}^{RK \times TK}$ is the target channel matrix and it consist of submatrices $\mathrm{diag}(\mathbf{H}^{r,t})$, i.e, the diagonal elements (1st channel taps) from $\mathbf{H}^{r,t}$, $\mathbf{F}_R \in \mathbb{C}^{RK \times RK}$ is a block diagonal DFT matrix $\mathbf{I}_R \otimes \mathbf{F}_K$, $\mathbf{F}_K$ is the DFT matrix, $\mathrm{tr}\{\cdot\}$ is the matrix trace operator and $\otimes$ is the Kronecker product. The MMSE filter can be written as

$$\mathbf{\Omega} = \mathbf{\Sigma}_r^{-1}\mathbf{\Gamma}\tilde{\mathbf{\Gamma}}^{\mathrm{H}} \tag{3}$$

where $\mathbf{\Sigma}_r = \mathbf{\Gamma}\mathbf{\Gamma}^{\mathrm{H}} + \sigma^2\mathbf{I} \in \mathbb{C}^{RK \times RK}$, $\mathbf{I} \in \mathbb{R}^{RK \times RK}$ is an identity matrix and the frequency domain channel matrix $\mathbf{\Gamma} = \mathbf{F}_R\mathbf{H}\mathbf{F}_T^{-1} \in \mathbb{C}^{RK \times TK}$. The (i, j) term of the equivalent channel $\mathbf{\Phi} \in \mathbb{C}^{R \times T}$ can be calculated as

$$\varphi_{i,j} = \frac{1}{K}\mathrm{tr}((\tilde{\mathbf{\Gamma}}\mathbf{\Gamma}^{\mathrm{H}}\mathbf{\Sigma}_r^{-1}\mathbf{\Gamma})_{i,j}) \tag{4}$$

where $i = 1, ..., R$ and $j = 1, ..., T$ and the (i, j) term of the covariance of residual interference $\mathbf{\Sigma}_w \in \mathbb{C}^{R \times T}$ as

$$\sigma_{i,j}^2 = \frac{1}{K}\mathrm{tr}((\tilde{\mathbf{\Gamma}}\mathbf{\Gamma}^{\mathrm{H}}\mathbf{\Omega})_{i,j}) - \frac{1}{K}\mathrm{tr}((\mathbf{\Omega}^{\mathrm{H}}\mathbf{\Gamma}\mathbf{\Gamma}^{\mathrm{H}}\mathbf{\Omega})_{i,j}). \tag{5}$$

The equalized signal $\mathbf{z} \in \mathbb{C}^{RK}$ after the IDFT, can be written as

$$\mathbf{z} = \mathbf{F}_R^{-1}\mathbf{\Omega}^{\mathrm{H}}\mathbf{F}_R\mathbf{r}. \tag{6}$$

After the frequency domain filtering, the noise is not white and has covariance matrix $\mathbf{\Sigma}_w$ from (5). The likelihood function term $1/\sigma^2||\mathbf{z} - \mathbf{\Phi}\mathbf{s}||_2^2$ then becomes $\mathbf{\Sigma}_w^{-1/2}||\mathbf{z} - \mathbf{\Phi}\mathbf{s}||_2^2$, where $\mathbf{s}$ is a transmitted symbol vector candidate. The covariance of residual interference can be taken into account either by whitening the noise or including it in the distance calculations. The whitening can be done by multiplying $\mathbf{z}$ and $\mathbf{\Phi}$ with the inverse square root of the covariance matrix $\mathbf{\Sigma}_w$, i.e, $\mathbf{z}_w = \mathbf{\Sigma}_w^{-1/2}\mathbf{z}$ and $\mathbf{\Phi}_w = \mathbf{\Sigma}_w^{-1/2}\mathbf{\Phi}$. The inverse square root can be obtained from $\mathbf{\Sigma}_w^{1/2} = \text{chol}(\mathbf{\Sigma}_w)$ or $\mathbf{\Sigma}_w^{1/2} = \mathbf{U}\Lambda^{1/2}$, when $\mathbf{\Sigma}_w = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}}$ and $\mathbf{\Lambda}$ contains the eigenvalues and $\mathbf{U}$ contains the eigenvectors of $\mathbf{\Sigma}_w$.

3.2 Sphere detector

The structure of the sphere detector is presented in Figure 4. In the QRD block, QR decomposition $\mathbf{\Phi_w} = \mathbf{QR}$ of the whitened channel matrix is performed where $\mathbf{Q} \in \mathbb{C}^{R\times R}$ and $\mathbf{R} \in \mathbb{C}^{T\times R}$. The QRD is performed only once in a block since the channel matrix $\mathbf{\Phi}$ is invariant, i.e, it is common for all symbol vectors in $\mathbf{z}$. Each symbol vector $\mathbf{z}_{w_{[n]}} \in \mathbb{C}^T$ from the whitened vector $\mathbf{z}_w$ is multiplied with matrix $\mathbf{Q}$ from QRD as $\mathbf{z}'_{w_{[n]}} = \mathbf{Q}\mathbf{z}_{w_{[n]}}$. The tree search is then performed separately for each whitened symbol vector.

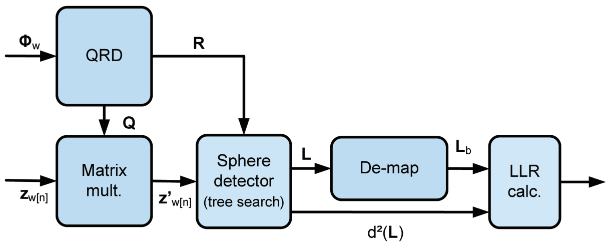

Fig. 4 The sphere detector structure.

The squared partial Euclidean distance (PED) of $\mathbf{s}_i^T$, i.e., the square of the distance between the partial candidate symbol vector and the partial received vector, can be calculated in the sphere detector block as

$$d(\mathbf{s}_i^T) = \sum_{j=i}^{T} \left| \mathbf{z}'_{w_{[n]_j}} - \sum_{l=j}^{T} R_{j,l} s_l \right|^2, \tag{7}$$

where $i = T\ldots,1$ and $\mathbf{s}_i^T$ denotes the last $T-i+1$ components of vector $\mathbf{s}$ [13].

The resulting list of candidate symbol vectors $\mathcal{L}$ is demapped into binary form and the LLR for the transmitted bit k is calculated as

$$L_{\mathrm{D}}(b_k) = \ln \frac{p(\mathbf{z}_{w_{[n]}}|b_k = +1)}{p(\mathbf{z}_{w_{[n]}}|b_k = -1)}, \tag{8}$$

where

$$p(\mathbf{z}_{w_{[n]}}|b_k = +1) = \sum_{\mathbf{s}\in\Theta, b_k=+1} e^{\frac{-d(\mathbf{s})}{2}} \tag{9}$$

and Θ is the set of possible transmitted symbol vectors. The LLRs can be updated from the decoder feedback L_A as:

$$\hat{L}_D(b_k|\mathbf{z}_{w_{[n]}}) = L_A(b_k) + \ln \frac{\sum_{\mathbf{b}\in\mathcal{L}_{k,+1}} \exp(\Lambda(\mathbf{b}, \mathbf{b}_{[k]}, \mathbf{l}_{A,[k]}|\mathbf{z}_{w_{[n]}}, \mathbf{\Phi_w}))}{\sum_{\mathbf{b}\in\mathcal{L}_{k,-1}} \exp(\Lambda(\mathbf{b}, \mathbf{b}_{[k]}, \mathbf{l}_{A,[k]}|\mathbf{z}_{w_{[n]}}, \mathbf{\Phi_w}))}, \tag{10}$$

where

$$\Lambda(\mathbf{b}, \mathbf{b}_{[k]}, \mathbf{l}_{A,[k]}|\mathbf{z}_{w_{[n]}}, \mathbf{\Phi_w}) = -\frac{1}{2}||\mathbf{z}_{w_{[n]}} - \mathbf{\Phi_w}\mathbf{s}||^2 + \frac{1}{2}\mathbf{b}_{[k]}^T \mathbf{l}_{A,[k]}, \tag{11}$$

$\mathbf{l}_{A,[k]}$ is a vector of L_A and $\mathbf{b}_{[k]}$ is a vector corresponding to k from the transmitted binary vector $\mathbf{b}$.

Two different tree search algorithms are considered in this paper. The K-best LSD algorithm [25] is a breadth-first search based algorithm, which keeps the K nodes which have the smallest accumulated Euclidean distances at each level. If the PED is larger than the squared sphere radius C_0, the corresponding node will not be expanded. We assume no sphere constraint or $C_0 = \infty$, but set the value for the list size K instead, as is common with the K-best algorithms. Figure 5 illustrates the K-best tree search structure for real valued 2×2 antenna system using 16-QAM and list size 4. In a complex valued system there would be only two levels but on each level the parent node would be expanded into 16 nodes.

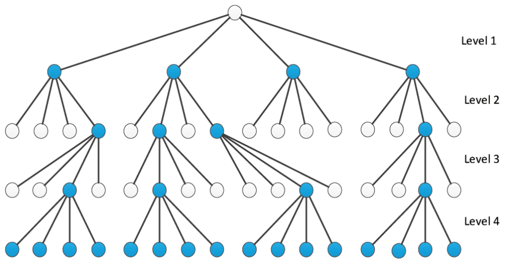

Fig. 5 4-best, 2×2 antenna system, 16-QAM, real system model.

The selective spanning with fast enumeration algorithm [26] is characterized by vector $\mathbf{m} = [m_1, ..., m_M]$, which defines the number of spans for each node on level i and also the length of the final candidate list. For example in a real 2×2 antenna and 16-QAM system, the node spanning vector $\mathbf{m} = [4, 4, 4, 4]$ would lead to a full search and the length of 256 candidates in the final list. The spanned nodes are never deleted and the number of nodes in the search tree can be determined using vector $\mathbf{m}$ i.e. $\prod_{j=i}^{T} m_j$. Figure 6 shows the tree search structure for the same system as in Figure

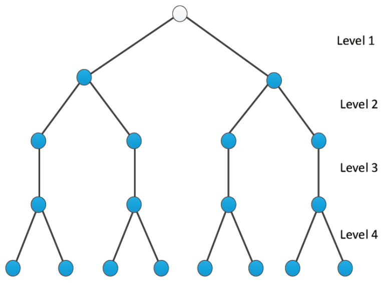

Fig. 6 SSFE[2,2,1,2], 2×2 antenna system, 16-QAM, real system model.

5, but here using SSFE tree search algorithm. Here the node spanning vector $\mathbf{m} = [2, 2, 1, 2]$.

The slicer unit is an essential part of the SSFE algorithm. It selects a set of closest constellation points $\mathbf{s}^i$ such that the PED increment is minimized at each level e.g.,

$$\left\|e_i(\mathbf{s}^i)\right\|^2 = \left\| \underbrace{\mathbf{z}'_{w_{[n]_i}} - \sum_{j=i+1}^{T} R_{i,j} s_j}_{b_{i+1}(\mathbf{s}^{i+1})} - R_{i,i} s_i \right\|^2 . \quad (12)$$

Minimizing $\left\|e_i(\mathbf{s}^i)\right\|^2$ is equivalent to the minimization of $\|e_i(\mathbf{s}^i)/R_{ii}\|^2$

$$\left\| \frac{e_i(\mathbf{s}^i)}{R_{ii}} \right\|^2 = \left\| \underbrace{b_{i+1}(\mathbf{s}^{i+1})/R_{ii}}_{\varepsilon} - s_i \right\|^2 . \quad (13)$$

Equation (13) is essential for the slicer unit which selects the closest constellation points based on ε.

4 Communications Performance

The performances of the conventional MMSE receiver and the frequency domain MMSE filter with two different tree search algorithms were compared in Matlab simulations. K-best LSD algorithm was simulated with list size 8 and 16. SFFE algorithm was simulated with node spanning vector [8,8,1,1,1,1,1,1] and [4,3,2,2,1,1,1,1]. Iterations, antenna ordering or other optimization methods were not used for the algorithms. These methods would give performance gain for all the algorithms of interest if latency or extra complexity is not an issue.

The simulation parameters are presented in Table 1. Pedestrian A, Vehicular A and Pedestrian B channel models were used in the simulations [27]. The channel parameters are described in Table 2. As can be seen from the multipath profile values, Pedestrian A channel is the least and Pedestrian B channel the most frequency-selective causing powerful ISI term. The chosen azimuth spread values result in spatially correlated channels making the case both realistic and very challenging for the MIMO equalizer. The number of transmit and receive antennas is four. This illustrates the most challenging case of high data rate and significant IAI. True synchronization process is performed in the simulator. However, maximum uncertainty for time synchronization is set low enough to virtually eliminate the effect of synchronization error. The effect of sync error has not been studied in this paper.

Table 1 Simulation parameters

Coding	3GPP Turbo code
Code rate	1/2 and 2/3
Modulation scheme	64-QAM
Symbol duration	71.4 μs
Channel model	Pedestrian A and B, Vehicular A
Antenna configuration	4×4, 2×2 and 1×4

Table 2 Channel model parameters

	Ped A	Veh A	Ped B
Number of paths	4	6	6
Path delays [ns]	[0...410]	[0...2510]	[0...3700]
Path power [dB]	[0...−22.8]	[0...−20]	[0...−23.9]
BS/MS antenna spacing	4 λ/ 0.5 λ	4 λ/ 0.5 λ	
BS average angle of arrival	50 °	50 °	50 °
BS/MS azimuth spread	2 °/ 35 °	2 °/ 35 °	varies

The 4×4 performances of the different receivers in a correlated Pedestrian A channel are presented in Figure 7, in a correlated Vehicular A channel in Figure 8 and in a correlated Pedestrian B channel in Figure 9. The performances in a 2×2 MIMO and 1×4 (virtual MIMO) scenarios were also simulated, but the results are not reported herein, because they are basically similar as those for the 4×4 case. All the simulations were also performed with 1/2 code rate. The results followed these code rate 2/3 results, but the performance differences were slightly smaller.

In [23] we also simulated turbo receiver performances but those results had much more variance. Although being superior with limited set of parameters (code rate, ISI and IAI) there were many scenarios where the receiver did not converge even with high number of iterations. In summary the turbo receiver improved the performance especially in scenarios when there was no IAI. Therefore turbo receivers might be suitable for OFDM systems, wherein ISI is not a problem. Additionally, the full analysis whether it would be possible to accept the latencies of turbo structures in LTE Rel 8 - Rel 10 uplink receiver implementation would require further study.

The simulation results show that the 2-stage receiver with a time domain sphere detector using K-best algorithm (8-best, 16-best) or SSFE[4,3,2,2,1,1,1,1] algorithm outperforms the traditional MMSE receiver (frequency domain MMSE equalization with soft demodulator) in Pedestrian A and Vehicular A channel. With a large delay spread, as in the Pedestrian B channel, the performances of these algorithms are equal to that of the linear MMSE receiver. The reason is that ISI dominates over IAI and the proposed MIMO search algorithms cannot perform better than the linear receiver does.

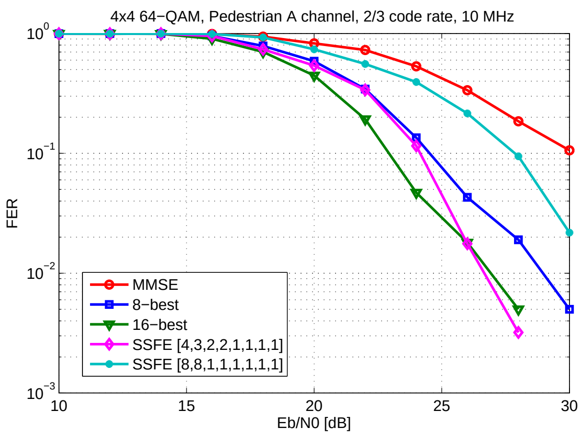

Fig. 7 4 × 4 performance in a correlated Pedestrian A channel.

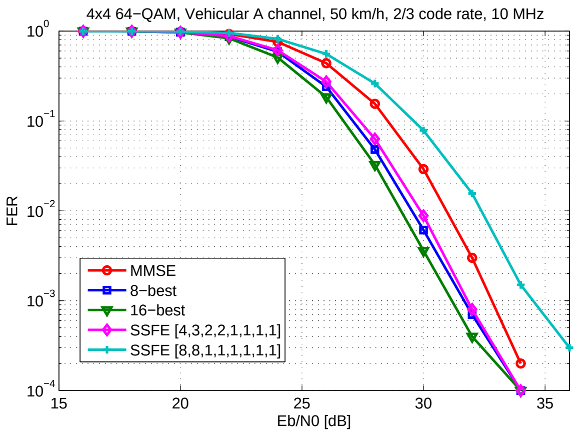

Fig. 8 4×4 performance in a correlated Vehicular A channel.

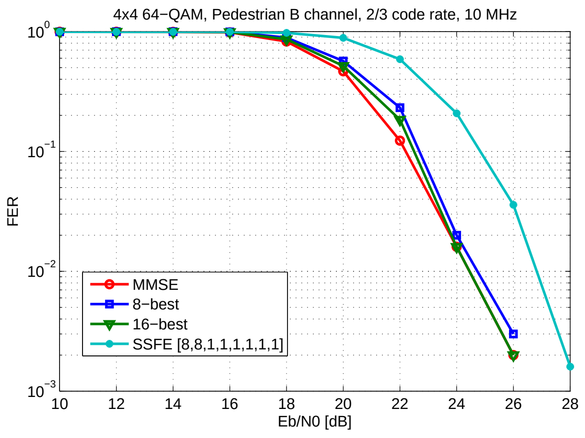

Fig. 9 4 × 4 performance in a correlated Pedestrian B channel.

The performance of the SSFE algorithm is not optimal with the node spanning vector [8,8,1,1,1,1,1,1]. With a different vector [4,3,2,2,1,1,1,1], the SSFE algorithm performs better than the MMSE algorithm at the expense of additional computational complexity. The performance of the K-best with list size 8 or 16 is better than that of the MMSE algorithm. Also SSFE[8,8,1,1,1,1,1,1] performance could be increased to meet the performance of these algorithms with the cost of extra complexity. [28] is a good example of SSFE[8,8,1,1,1,1,1,1] implementation in which good performance has been achieved with pre-processing. However, the main compute complexity of this implementation is in the antenna ordering, the pre-processor being 3.3x more complex than the actual SD. Based on the simulations 8-best, 16-best and SSFE[4,3,2,2,1,1,1,1] would all be suitable for the 2-stage receiver implementation without any additional pre-processing. However, our complexity estimation results for these algorithm [23] show that 16-best algorithm would be twice as complex and the SSFE[4,3,2,2,1,1,1,1] algorithm would be 50% more complex than the 8-best algorithm. As a result, K-best list sphere detector with list size of 8 was decided to be implemented on an FPGA. It offers the best performance-complexity ratio for the practical implementation in the channel conditions studied herein.

5 Development environment

The HLS tool was decided to be used for generating the RTL instead of hand written RTL. The HLS tools are gaining popularity and they are challenging the traditional design approach. There are several studies showing that these tools increase the design productivity and reduce the development time, while producing competitive quality of results compared to hand written RTL [29,30].

The implementation tool flow can be seen in Figure 10. The algorithms were first written using Matlab to enable comprehensive simulations and comparisons in our SC-FDMA Matlab link level simulator. The selected HLS tool uses C code as a source. Thus, C versions of selected algorithms had to be written. MEX interface in Matlab enabled us to verify the C version was identical to original Matlab version.

Xilinx Vivado HLS tool was used for converting the C code into RTL (in this case VHDL). The tool gives a new abstraction level and it hides some of the complexity of design implementation. The HLS tool generates a high-performance pipelined architecture based on the constraints, directives and implementation C/C++ code. The constraints include, for example, the target FPGA family and the target clock frequency. The directives guide the HLS tool, for example, to unroll loops or partition arrays. The input is not the original reference C/C++ code. Instead, the reference code has been restructured so that it represents an architecture targeted by the designer. Figure 11 shows iterative code restructuring phase of the design flow. The HLS tool generates the RTL output based on these inputs and reports the throughput performance and estimate of the complexity of the architecture. The designer can then iteratively change the directives and the C/C++ source code as long as the throughput requirements have been satisfied. With HLS tool it is possible to generate a valid high complex solution in relatively short time, but a highly optimized low complex solution requires many iterations. The iterative design approach enables the trade of between the quality of results and development time.

In the next phase, the output RTL is used as an input for the FPGA implementation tool (Xilinx ISE/EDK). The final achievable clock frequency and resource usage are reported after logic synthesis and Place & Route. If the result do not satisfy the designer, directives or implementation C/C++ code could be modified further.

6 Implementation

The K-best LSD algorithm was implemented on a Virtex-6 XC6VLX240T FPGA with speed grade -2. Implementation started with requirement specification and input/output (I/O) specification. After that an initial architecture was planned. Matlab model of the 8-best LSD algorithm was written again using C code. The C code was verified again after every modification.

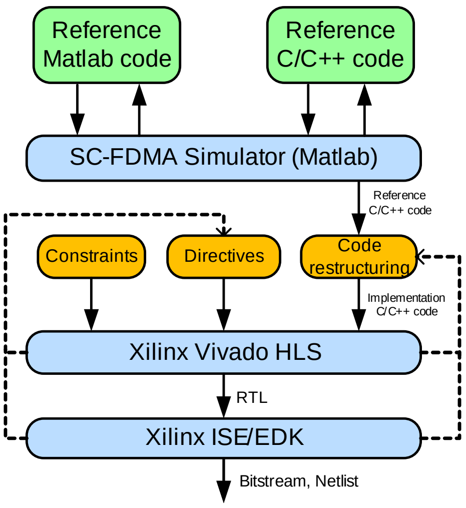

Fig. 10 Tool flow.

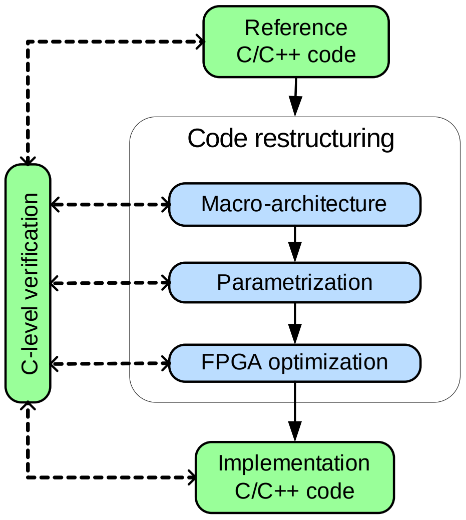

Fig. 11 Code restructuring.

HLS tool gave the possibility to generate several different solutions and choose the best one.

6.1 Implementation requirements

A SC-FDMA 64-QAM 4×4 single-user MIMO system was assumed with a 20 MHz bandwidth with 1200 subcarriers. A slot (0.5 ms) consists of 6 or 7 symbols (depending on cyclic prefix length) giving maximum of 83 μs for receiving the symbol. One symbol consists of 1200 subcarriers. There are 4 transmit antennas and 6 bits/symbol. As a result, the minimum throughput needed is

$$1/83\ \mu\text{s} \times 1200 \times 4 \times 6 = 347\ \text{Mbps} \qquad (14)$$

The K-best LSD can then use

$$83\ \mu\text{s}/1200 = 69.4\ \text{ns} \qquad (15)$$

to process one received symbol vector **y**. Real valued 4×4 tree search is feasible to schedule in $N \times 8$ ($N = 1, 2, 3...$) cycles so minimum clock frequency for different numbers of available clock cycles can be calculated

$$f(\min) = \frac{num\_cycles}{69.4 \text{ ns}}, num\_cycles = 8, 16, 32, ... \quad (16)$$

The throughput of 347 Mbps can be achieved for example with the following parameter combinations: 115 MHz/8 cycles, 230 MHz/16 cycles or 460 MHz/32 cycles. 460 MHz is somewhat too high a frequency target for FPGA and 115 MHz very loose one. Scheduling the design in 8 clock cycles wastes resources, because a higher frequency could be achieved. We decided to implement two different architectures which both achieve 347 Mbps: Architecture I including a challenging sorting operation of the tree search and Architecture II without sorting operation. Both architectures were implemented with the similar amount of optimization.

6.2 Macroarchitecture Specification

6.2.1 Architecture I

Architecture I describes the structure of the K-best algorithm without trying to avoid sorting operation. Eight PEDs are calculated on the first level. On levels 2–8, 8 more distances are calculated resulting in 64 PEDs. These 64 PEDs need to be sorted resulting in 8 surviving PEDs. Levels 2-8 need to have a sorter. Sorting N samples requires N operations if there is no pre-information about the samples. With synchronous logic this means that a level including sorter can not be scheduled in less than 64 cycles (pipeline initiation interval $\geq$ 64). The selected sorter here is the insertion sorter. For large data sets asymptotically efficient sorters like quicksort, heap sort and merge sort could be used. However, 64 samples can be considered as a relatively small data set and insertion sort is one of the most common and efficient sorter algorithms available for small data sets.

The macroarchitecture shown in Figure 12 was designed for the Architecture I. PED 1 calculates 8 distances and does not require sorting. PEDs 2–8 calculate 64 distances and include an insertion sorter.

6.2.2 Architecture II

Alternative Architecture II was also implemented in which the goal was to avoid the sorting operation. In the Architecture I, all the 64 PEDs are sorted and then

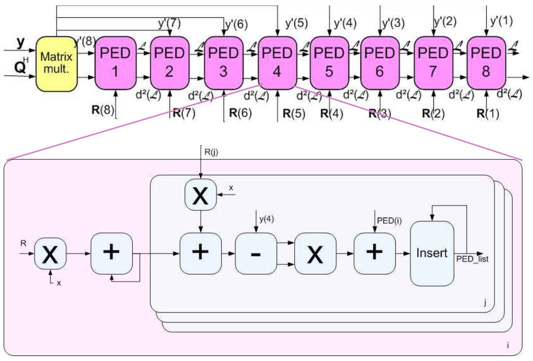

Fig. 12 Macroarchitecture I.

the 8 smallest ones are selected. Whereas, in the Architecture II 8 smallest ones are selected directly. For the sort-free architecture we are using the method used in [31], [32] and [33]. It is possible to find K smallest PEDs in less than K cycles, if we use regularities of constellation points and pre-sorted PEDs. Slicing operation, used in Schnorr-Euchner enumeration, is used to find the smallest child from parent node. Next min-search can be used to find the smallest out of different parents pre-ordered childs. Figure 13 [33] shows the idea of the sort-free method for the K-best algorithm. The key idea of the distributed K-best scheme is to find the first child of each node in $K_l + 1$. Among these first children the one with the lowest PED is definitely one of the K best candidates in K_l. That child is selected and is replaced by its next best sibling. This process is repeated K times to find the K best candidates in level l (K_l). This structure finds the K best candidates in just K clock cycles. Figure 14 shows the planned macroarchitecture for the alternative structure, which we here call Architecture II.

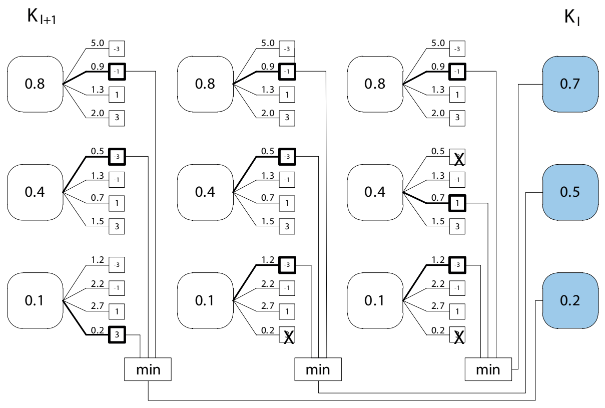

Fig. 13 Distributed K-best scheme.

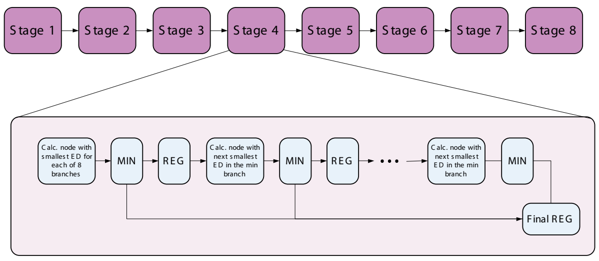
Stage 1
Stage 2
Stage 3
Stage 4
Stage 5
Stage 6
Stage 7
Stage 8
Calc. node with smallest ED for each of 8 branches
MIN
REG
Calc. node with next smallest ED in the min branch
MIN
REG
...
Calc. node with next smallest ED in the min branch
MIN
Final REG

Fig. 14 Macroarchitecture II.

6.3 Parameterization

Parameterization was used in re-writing of the C code. Example below shows the C++ template function for levels PED 2–8 (PED 1 has its own function). *K-best_LSD_8* gets the level of the tree search as an template parameter. Parametrization gives the ability for HLS tools to use more resource sharing and that way reduce the FPGA resources.

```
/* Stages 2-8 of the tree search (PED 2-8)*/
template<int level> void Kbest_LSD_8(
ap_fixed<10,2> x[8],
ap_ufixed<3,3>,
cand_final_812[level*8],
ap_ufixed<16,6> ED_list88[8],
ap_fixed<16,6> r2,
ap_fixed<16,5> R8[level],
ap_ufixed<3,3> cand_final21[(level-1)*8],
ap_ufixed<16,6> cand_temp_PED3[8])
{
<function body>
}
```

6.4 FPGA optimization

Two examples of FPGA optimization, used in these implementations, are bit-width optimization and efficient use of embedded DSP blocks. The reference C/C++ uses normal C/C++ data types (e.g. short, int). To optimize the bit-widths, fixed point data types are used in the implementation C/C++ code. Arbitrary bit-widths are required so that, for example, 32-bit multipliers are not wasted when less bits are required. ap_int.h header enables arbitrary precision integers and ap_fixed.h enables arbitrary precision fixed-point data types to be used in the C code.

A specific example of efficient use of embedded DSP blocks is DSP48s usage. The use of DSP48s improves timing and FPGA resource utilization significantly. The structure of DSP48 block is shown in Figure 15. Here is an example of a multiplication followed by an addition.

```
/* Multiplying candidate symbol with index
    ind8 from level k with R*/
temp_ed8_2 = temp_ed8_2 + R8[k+1]*x[ind8];
```

These two operations can be mapped in to a single DSP48 block. The modified function call looks like this

```
/* Multiplying candidate symbol with index
    ind8 from level k with R*/
temp_ed8_2 = macc25x18<5,2,true>(R8[k+1],x[
    ind8],temp_ed8_2);
```

Template function *macc25x18* converts the fixed point values into integer values. It gets the following inputs: values of three variables (R8, x, temp_ed8_2), integer part widths of those variables and parameter true (=addition) or false (=subtraction) which specifies the desired operation.

```
//**********macc25x18**********
template<int iwidth_a, int iwidth_b, bool
    ADDSUB>
ap_fixed<48,iwidth_a+iwidth_b+5> macc25x18(
    ap_fixed<25,iwidth_a> A, ap_fixed<18,
    iwidth_b> B, ap_fixed<48,iwidth_a+
    iwidth_b+5> C){

ap_int<25> ia = (ap_int<25>)A.range(24,0);
ap_int<18> ib = (ap_int<18>)B.range(17,0);
ap_int<48> ic = (ap_int<48>)C.range(47,0);

ap_int<48> id = multadd25x18<ADDSUB>(ia,ib,
    ic);

ap_fixed<48,iwidth_a+iwidth_b+5> r;
r.range(47,0)=id.range(47,0);

return r;
}
```

Finally, template function *macc25x18* calls *multadd25x18* to perform the actual calculation. Two directives in *multadd25x18* instruct the high-level synthesis tool to use a maximum of two cycles to schedule these operations and use a register for the output return value.

```
//**********multadd25x18**********
template<bool ADDSUB>
ap_int<48> multadd25x18(ap_int<25> A, ap_int
    <18> B, ap_int<48> C){
#pragma AP INTERFACE ap_none port=return
    register
#pragma AP LATENCY max=2
  if(ADDSUB)
    return C + A * B;
  else
    return C - A * B;
}
```

6.5 Implementation results

The Architecture I, including insertion sorter, schedules in 64 cycles, which means that every 64th cycle a new input vector y is taken in, where y includes one symbol

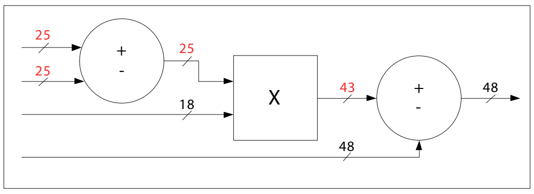

Fig. 15 Xilinx Virtex-6 DSP48 block.

from four different antennas and each symbol consists of 6 bits (64-QAM). Our implementation achieves a 247 MHz clock frequency which means that the achieved throughput is

$$\frac{247 \text{ MHz} \times 4 \text{ antennas} \times 6 \text{ bits/symbol}}{64} = 93 \text{ Mbps}, \tag{17}$$

However, several parallel blocks can be used. 20MHz SC-FDMA transmission consists of 1200 subcarriers. The same channel matrix is used while 1200 data symbols from each of the four antennas are processed. One data symbol from each of these four antennas is used at the time in SD processing. All the 1200 data symbols are received during the same symbol period 83 μs. One data symbol occupy full BW for a 1/1200 symbol period and the channel matrix is the same during this 83 μs period. In order to achieve the required 347 Mbps, four detectors can be used. Each of these process 300 data symbols from four antennas. This leads to a total 372 Mbps detection rate. In case of short cyclic prefix, five detectors can be used. The sort free Architecture II schedules in 16 cycles and achieves 231 MHz clock frequency. Therefore, single Architecture II detector is enough to achieve the throughput of 347 Mbps.

Architecture I and II are compared in Table 3. Four Architecture I blocks in parallel achieve the target throughput with less resources than the sort-free Architecture II. Furthermore, it also adds the value of scalability for the design. [31] claims that the architecture without traditional sorter for the K-best algorithm is adequate when K is smaller than the number of constellation points. Here the K is 8 and the number of constellation points in real valued 64-QAM system is also 8. Thus, our system parameters create somewhat a borderline case for the comparison.

Regarding the complexity distribution of the discussed receiver structure the following notes were made for the blocks shown in Figure 4. QRD has very low throughput requirements compared to SD. QRD is performed only once while SD tree search algorithm will run 1200 times in the meantime. This enables high latency/low complexity directives for the design and in our HLS implementation QRD complexity is only 15-20% of the 8-best SD. Matrix multiplication has low complexity compared to the tree search. The de-mapper complexity depends on the number of remaining ED paths after the last level of the tree search. The number of remaining paths in the 8-best and 16-best tree search algorithms is 8 and 16, respectively. Likewise, in SSFE[8,8,1,1,1,1,1,1] and SSFE[4,3,2,2,1,1,1,1] tree search algorithms, the number of remaining paths is 64 and 48, respectively. Therefore, the de-mapper for SSFE[8,8,1,1,1,1,1,1] is almost eight times more complex than the de-mapper for the 8-best tree search algorithm. The same is true also for the LLR algorithm. In our 8-best scenario, the de-mapper and LLR complexity is less than 10% of the SD algorithm.

Table 3 Architecture comparison

	4× Architecture I	Architecture II
LUT	34476	69383
FF	50044	97676
DSP48	216	228
BRAM	28	287
Max freq (MHz)	247	231
Throughput(Mbps)	372	347

7 Discussion

We compared different receiver algorithms and structures for SC-FDMA uplink transmission. The novel frequency domain MMSE equalization with sphere detection receiver is a remarkable improvement over the conventional linear MMSE receiver. The K-best LSD algorithm and the SSFE algorithm were considered as possible tree search algorithms for this receiver. Two different list sizes were used for the K-best algorithm and two different node spanning vectors for the SSFE algorithm. As a result, K-best algorithm with a list size of 8 was considered to give the best performance-complexity ratio and was chosen for the implementation.

The 8-best LSD algorithm was implemented on a Xilinx Virtex-6 FPGA with Xilinx Vivado High Level Synthesis tool using several optimization methods. The used HLS implementation was used to compare two architectures with each other. The design process and the amount of effort used for both of these implementations were identical. Both architectures might have potential for further optimization with traditional design methods. Yet, we assume that with hand-written RTL implementation the same conclusions about the benefits of both architectures could be made.

The target throughput was 347 Mbps. HLS tool enabled us to implement two different equally optimized architectures with moderate amount of work. The Architecture I, including conventional sorter, schedules in 64 cycles and achieves 93 Mbps detection rate. However, the SC-FDMA receiver allows parallel processing of the subcarriers and four 93 Mbps designs can be used to achieve the throughput target of 347 Mbps. The Architecture II, without conventional sorter, schedules in 16 cycles and achieves 347 Mbps with one design. Hence, the equalization algorithms and their realizations on an FPGA fulfills both the latency and performance requirements of LTE/LTE-A base stations with 64-QAM and 4×4 MIMO set-up. The recommendations for the most suitable algorithm and architecture were made based on performance and implementation complexity. Due to uplink scenario and RF front-end domination in base station energy consumption, the energy efficiency was not used as factor.

The sort-free architecture did not give any gain and was actually more complex than the architecture including a sorter. The sort-free architecture would be efficient only if $K <$ number of constellation points. This is often the case when operating on the complex-valued constellation points. In our system $K{=}8$ and the number of constellation points was also 8. Based on the simulation results the efficient value for K in common $4{\times}4$ 64-QAM system is somewhat minimum of 8. There are basically two options to get gain from the sort-free architecture in a common 4×4 64-QAM system. Either K value should be dropped down to e.g. 4 or complex valued tree search should be applied. In 64-QAM system this would change the number of constellation points from 8 to 64. Although, complex valued processing would create other implementation challenges. The trade-off in the performance by reducing the K value or applying the complex valued tree search as well as determination of how much smaller the K should be to gain significant difference in the complexity, would require further study.

8 Conclusion

Our objective was to give a recommendation for an efficient and realistic receiver structure, detector algorithm and detector implementation architecture for 4×4 64-QAM MIMO LTE-A systems and their SC-FDMA based uplink base stations. The *Frequency domain linear MMSE filter with sphere detection* was chosen as the receiver structure based on our earlier work in [23]. The 8-best LSD algorithm was chosen as the detector algorithm based on the analysis done in this paper. With these system parameters a sort-free implementation architecture for 8-best LSD algorithm is not recommended. Thus, for practical realizations our recommendation is to focus on optimizing the conventional 8-best 4×4 64-QAM architecture without trying to avoid the sorting operation.

Acknowledgements This research was partially done in the CoMIT project which was supported by Elektrobit, Nokia, Xilinx and the Finnish Funding Agency for Technology and Innovation (Tekes). The authors would like to thank Xilinx Inc., especially Juanjo Noguera, Baris Ozgul and Jari Keskinen, for the possibility to evaluate Xilinx Vivado High Level Synthesis tool and for their co-operation.

References

1. 3rd Generation Partnership Project (3GPP), “Physical layer procedures,” 3GPP TS 36.213 V11.0.0, Tech. Rep., 2012.
2. Z. Zvonar, “Multiuser detection in asynchronous CDMA frequency-selective fading channels,” *Wireless Personal Communications*, vol. 2, no. 4, pp. 373–392, 1995.
3. 3rd Generation Partnership Project (3GPP), “Further advancements for E-UTRA physical layer aspects,” 3GPP TR 36.814 V9.0.0, Tech. Rep., 2010.
4. D. Falconer, S. Ariyavisitakul, A. Benyamin-Seeyar, and B. Eidson, “Frequency domain equalization for single-carrier broadband wireless systems,” *IEEE Communications Magazine*, vol. 40, no. 4, pp. 58–66, Apr. 2002.
5. H. Sari, G. Karam, and I. Jeanclaude, “Frequency domain equalization of mobile radio and terrestrial broadcast channels,” in *Proceedings of the IEEE Global Telecommunication Conference*, Nov. 1994.
6. M. Tüchler, A. Singer, and R. Koetter, “Minimum mean squared error equalization using a priori information,” *IEEE Signal Processing Letters*, vol. 50, no. 3, pp. 673–683, Mar. 2002.
7. R. Koetter, A. Singer, and M. Tüchler, “Turbo equalisation,” *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 67–80, Jan. 2003.
8. T. Abe and T. Matsumoto, “Space-time turbo equalization in frequency-selective MIMO channels,” *IEEE Transactions on Vehicular Technology*, vol. 52, no. 3, pp. 469–475, May 2003.
9. K. Kansanen and T. Matsumoto, “An analytical method for MMSE MIMO turbo equalizer EXIT chart computation,” *IEEE Transactions on Wireless Communications*, vol. 6, no. 1, pp. 59–63, Jan. 2007.
10. J. Karjalainen, N. Veselinovic, K. Kansanen, and T. Matsumoto, “Iterative frequency domain joint-over-antenna detection in multiuser MIMO,” *IEEE Transactions on Wireless Communications*, vol. 6, no. 10, pp. 3620–3631, Oct. 2007.
11. E. Telatar, “Capacity of multi-antenna Gaussian channels,” *European Transactions on Telecommunications*, vol. 10, no. 6, pp. 585–595, Nov. 1999.
12. D. Gesbert, M. Shafi, D. Shiu, P. J. Smith, and A. Naguib, “From theory to practice: An overview of MIMO space-time coded wireless systems,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, Apr. 2003.
13. M. O. Damen, H. E. Gamal, and G. Caire, “On maximum–likelihood detection and the search for the closest lattice point,” *IEEE Transactions on Information Theory*, vol. 49, no. 10, pp. 2389–2402, Oct. 2003.

14. B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Communications Letters*, vol. 51, no. 3, pp. 389–399, Mar. 2003.
15. Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 491–503, Mar. 2006.
16. S. Chen, T. Zhang, and Y. Xin, "Relaxed K -best MIMO signal detector design and VLSI implementation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 3, pp. 328–337, Mar. 2007.
17. C. Studer, A. Burg, and H. Bolcskei, "Soft-output sphere decoding: algorithms and VLSI implementation," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 2, pp. 290–300, Feb. 2008.
18. M. Myllylä, M. Juntti, and J. Cavallaro, "Implementation aspects of list sphere decoder algorithms for MIMO-OFDM systems," *Elsevier Journal on Signal Processing*, vol. 90, no. 10, pp. 2863–2876, Oct. 2010.
19. M. Myllylä, J. Cavallaro, and M. Juntti, "Architecture design and implementation of the metric first list sphere detector algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 5, pp. 895–899, May 2011.
20. J. Ketonen, M. Juntti, and J. Cavallaro, "Performance-complexity comparison of receivers for a LTE MIMO-OFDM system," *IEEE Signal Processing Letters*, vol. 58, no. 6, pp. 3360–3372, Jun. 2010.
21. F. Pancaldi, G. Vitetta, R. Kalbasi, N. Al-Dhahir, M. Uysal, and H. Mheidat, "Single-carrier frequency domain equalization," *IEEE Signal Processing Magazine*, vol. 25, no. 5, pp. 37–56, Sep. 2008.
22. W. Gerstacker, P. Nickel, F. Obernosterer, U. L. Dang, P. Gunreben, and W. Koch, "Trellis-based receivers for SC-FDMA transmission over MIMO ISI channels," in *Proceedings of the IEEE International Conference on Communications*, May 2008.
23. J. Ketonen, J. Karjalainen, M. Juntti, and T. Hänninen, "Mimo detection in single carrier systems," in *Proceedings of the 19th European Signal Processing Conference*, Aug. 2011.
24. M. Juntti, K. Hooli, K.Kiiskilä, and J. Ylioinas, "Space-time equalizers for MIMO high speed WCDMA downlinks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2582–2592, Jul. 2007.
25. K. Wong, C. Tsui, R. K. Cheng, and W. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, May 2002.
26. M. Li, B. Bougart, E. Lopez, and A. Bourdoux, "Selective spanning with fast enumeration: A near maximum-likelihood MIMO detector designed for parallel programmable baseband architectures," in *Proceedings of the IEEE International Conference on Communications*, May 2008.
27. 3rd Generation Partnership Project (3GPP), "Spatial channel model for multiple input multiple output (MIMO) simulations," 3GPP TR 25.996 V7.0.0, Tech. Rep., 2007.
28. C. Dick, M. Trajkovic, S. Denic, D. Vuletic, R. Rao, F. Harris, and K. Amiri, "Fpga implementation of a near-ML shere detector for 802.16e broadband wireless systems," in *Proceedings of the SDR 09 Technical Conference and Product Exposition*, 2009.
29. Berkeley design technology Inc, "An independent evaluation of: High-level synthesis tools for Xilinx FPGAs," Tech. Rep., 2010.
30. J. Noguera, S. Neuendorffer, S. Haastregt, J. Barba, K. Vissers, and C. Dick, "Implementation of sphere decoder for MIMO-OFDM on FPGAs using high-level synthesis tools," *Analog Integrated Circuits and Signal Processing*, vol. 69, no. 2, pp. 119–129, Sep. 2011.
31. M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "K-best MIMO detection VLSI architectures achieving up to 424 mbps," in *IEEE International Symposium on Circuits and Systems*, May 2006.
32. M. Shabany, K. Su, and P. Gulak, "A pipelined scalable high-throughput implementation of a near-ML K-best complex lattice decoder," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Mar. 2008.
33. M. Shabany and P. Gulak, "A 675 mbps, 4x4 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. 135–147, Jan. 2012.