Architecture and FPGA-Implementation of a High Throughput K+-Best Detector

Nils Heidmann, Till Wiegand, Steffen Paul
Institute for Theoretical Electrical Engineering and Microelectronics (ITEM)
University of Bremen, Otto-Hahn-Allee, NW1, 28359, Bremen, Germany, +49(0)421/218-62535
Email: {heidmann, wiegand, steffen.paul}@item.uni-bremen.de

***Abstract*—Since Multiple Input Multiple Output (MIMO) transmission has become more and more popular for current and future mobile communication systems, MIMO detection is a big issue. Linear detection algorithms are less complex and well understood but their BER performance is limited. ML detectors achieve the optimum result but have exponential computational complexity. Hence, iterative tree-search algorithms like the sphere decoder or the K-Best detector, which reduce the computational complexity, has become a major topic in research. In this paper a modified K+-Best detector is introduced which is able to achieve the BER performance of a common K-Best detector with K=12, by using a sorting algorithm for K=8. This novel sorting approach based on Batchers Odd-Even Mergesort is less complex compared to other parallel sorting designs and saves valuable hardware resources. Due to an efficient implementation the throughput of the detector is about 455 Mbit/s which is twice as high as the LTE peak data rate of 217.6 Mbit/s for a 16-QAM modulated signal. In this paper the architecture and the implementation issues are demonstrated in detail and the BER performance of the K+-Best FPGA implementation is shown.**

***Index Terms*—K-Best Detector; MIMO; Odd-Even Mergesort; FPGA-Implementation.**

I. Introduction

The 3GPP Long Term Evolution (LTE) is the upcoming standard for cellular mobile communication. To meet the requirements of high data rate and low latency interactive services, Orthogonal Frequency-Division Multiplexing (OFDM) in combination with MIMO is used. The industry aims to achieve a peak throughput of 100 Mbit/s in the downlink and 50 Mbit/s in the uplink. The maximum theoretically achievable data rate using a 64-QAM modulated signal over a bandwidth of 20 MHz and a 4×4-MIMO configuration is stated in [1] as 326.4 Mbit/s. Hence, the peak data rate of a 16-QAM modulated signal is about 217.6 Mbit/s. To handle such data rates at the detector, linear detection methods like Zero Forcing (ZF) or Minimum Mean Square Error (MMSE) are widely used as their computational complexity is low. Unfortunately, these detection schemes have a relatively poor Bit-Error-Rate (BER) performance and therefore iterative tree-search strategies which approximate the Maximum Likelihood (ML) solution without exponential complexity become more interesting. Figure 1 shows exemplarily the Bit-Error-Rate (BER) performance of a K-Best detector with different K compared to the worst (ZF) and the best (ML) detection scheme for a 4×4 MIMO and 16-QAM modulated system.

978-3-9810801-7-9/DATE11/ © 2011 EDAA

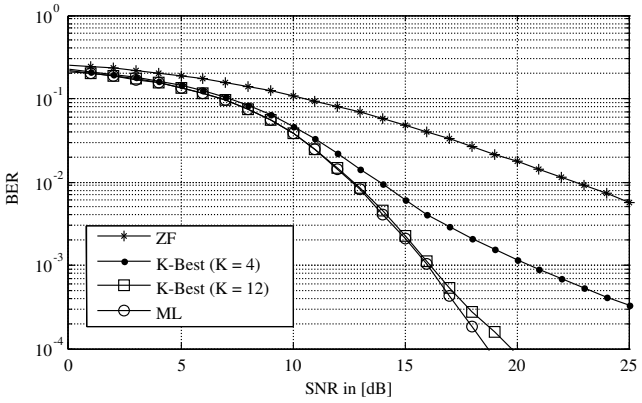

Fig. 1. BER performance of different 16-QAM 4×4 MIMO detector schemes

It can be seen that the BER performance increases with K and for $K = 12$ the ML solution is approximated quite well. Thus, the aim for the detector design is to handle a high K by keeping the complexity low.

The paper is structured as follows. Section II gives an overview of tree-search detectors including the K-Best detector. Section III describes the architecture of a detector, called K^+-Best, which is based on a modified search strategy. The FPGA implementation aspects are discussed in section IV and a complexity and performance analysis is shown in section V. Conclusions are made in section VI.

II. Tree-Search Detectors

Considering a $N_T \times N_R$-MIMO communication system with $N_R \geq N_T$, the system equation is given by:

$$\mathbf{y} = \mathbf{Hs} + \mathbf{n}, \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^{N_R \times 1}$ and $\mathbf{s} \in \Omega^{N_T \times 1}$ denote the received and the transmitted signal, $\mathbf{H} \in \mathbb{C}^{N_R \times N_T}$ is the channel matrix and $\mathbf{n} \in \mathbb{C}^{N_R \times 1}$ represents the system noise; Ω denotes the symbol alphabet. The optimum solution is given by the ML detector:

$$\hat{\mathbf{s}} = \arg \min_{\mathbf{s} \in \Omega^{N_T}} \|\mathbf{y} - \mathbf{Hs}\|^2 . \tag{2}$$

The principal idea of the tree-search algorithms is to decrease the computational complexity compared to the

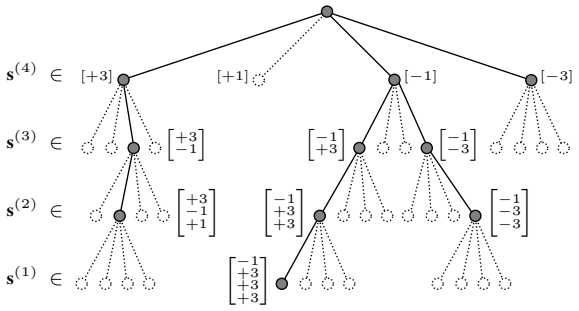

Fig. 2. Example of the search chain of a K-Best-detector with $\mathrm{K} = 3$

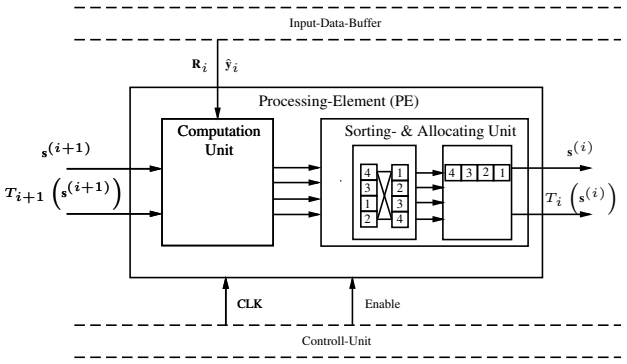

Fig. 3. Architecture of a Processing-Element

ML detector by setting partial symbol vectors $\mathbf{s}^{(i)} = \begin{bmatrix} s_i & s_{(i+1)} & \dots & s_{N_T} \end{bmatrix}^T$ in relation with nodes of an N_T-level search tree (Fig. 2). Whereby the search tree consists of $\mathrm{M}^{(N_T-i+1)}$ nodes at each level and M denotes the modulation level of the transmitted symbol. The criterion to find the optimum symbol vector $\mathbf{s}^{(1)}$ without searching the entire tree is the Euclidean Distance (ED) between the received vector $\mathbf{y}$ and the product $\mathbf{Hs}$ of the channel matrix $\mathbf{H}$ and the considered symbols $\mathbf{s}$. The ED can be split into Partial Euclidean Distances (PEDs). The aim is to find the symbol vector $\mathbf{s}^{(1)}$ with the smallest PED on the last level.

A. Partial Euclidean Distance

The PED for $N_R \geq N_T$ can be determined from the ML detector. Using a QR-decomposition $\mathbf{H} = \mathbf{QR}$, equation (2) can be transformed to:

$$\hat{\mathbf{s}} = \arg \min_{\mathbf{s} \in \Omega^{N_T}} \left\| \hat{\mathbf{y}} - \mathbf{Rs} \right\|^2, \tag{3}$$

where $\hat{\mathbf{y}} = \mathbf{Q}_1^H \mathbf{y}$, $\begin{bmatrix} \mathbf{R}^H & 0^{N_T \times N_R - N_T} \end{bmatrix}^H$ denotes a $N_R \times N_T$ upper triangular matrix and $\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix}$ denotes a $N_R \times N_R$ unitary matrix. This leads to:

$$\hat{\mathbf{s}} = \arg \min_{\mathbf{s} \in \Omega^{N_T}} \sum_{i=1}^{N_T} \left| \hat{y}_i - \sum_{j=i}^{N_T} r_{ij} s_j \right|^2. \tag{4}$$

A further advantage of equation (4) is, that it can iteratively be computed by the following procedure:

$$\begin{aligned} &T_i\left(\mathbf{s}^{(i)}\right) = T_{i+1}\left(\mathbf{s}^{(i+1)}\right) + \left| e_i\left(\mathbf{s}^{(i)}\right) \right|^2 \\ &\text{with} \\ &e_i\left(\mathbf{s}^{(i)}\right) = \hat{y}_i - \sum_{j=i}^{N_T} r_{ij} s_j, \end{aligned} \tag{5}$$

where $T_i\left(\mathbf{s}^{(i)}\right)$ denotes the Partial Euclidean Distance and $e_i\left(\mathbf{s}^{(i)}\right)$ can be interpreted as a distance increment [2], [3]. Hence, due to the iterative calculation each node can be assigned to a PED, which can be used as a criterion to prune branches early and therefore, to reduce the computational complexity.

B. K-Best

The K-Best detector is a breath first iterative tree-search algorithm. Instead of searching the branches of the tree in a depth first manner, like the sphere decoder, the K-Best detector only calculates the child-nodes of a fixed number of K nodes in one level before it steps to the next level following the branches of the next nodes containing the K smallest PEDs. Figure 2 explains the search strategy of a K-Best detector with $\mathrm{K} = 3$. As shown in Figure 1 the K-Best detector does not reach the ML solution as a common sphere decoder does, as the BER performance depends on the chosen parameter K. But for a parameter $\mathrm{K} = 12$ the ML solution is approximated quite well. The main advantage of the K-Best detector compared to the depth first sphere decoder is its fixed throughput e.g. fixed number of required clock cycles to detect one symbol, which makes it suitable for a hardware implementation.

C. Real Value Decomposition

To avoid complex numbers and to reduce the amount of computations, the complex system equation (2) can be decomposed into an equivalent real-valued system by the following equation:

$$\begin{bmatrix} \Re\{\mathbf{y}\} \\ \Im\{\mathbf{y}\} \end{bmatrix} = \begin{bmatrix} \Re\{\mathbf{H}\} & -\Im\{\mathbf{H}\} \\ \Im\{\mathbf{H}\} & \Re\{\mathbf{H}\} \end{bmatrix} \begin{bmatrix} \Re\{\mathbf{s}\} \\ \Im\{\mathbf{s}\} \end{bmatrix} + \begin{bmatrix} \Re\{\mathbf{n}\} \\ \Im\{\mathbf{n}\} \end{bmatrix}. \tag{6}$$

The real-valued dimensions are $N_{2T} = 2N_T$, $N_{2R} = 2N_R$ and the number of constellation points is reduced to $\mathrm{M}_{\mathrm{RVD}} = \sqrt{\mathrm{M}}$.

III. Architecture

The K^+-Best detector is optimized for a complex 16-QAM modulated 4×4-MIMO system. As an RVD and QR-decomposed system is considered, the search tree levels are increased from four to eight and the child-nodes are decreased from $\mathrm{M} = 16$ to $\mathrm{M}_{\mathrm{RVD}} = 4$. Hence, the basic architecture of the K^+-Best detector consists of eight Processing-Elements (PEs) connected in serial. The parameter K is set to eight, but due to a modified sorting approach the performance of a common K-Best detector with $\mathrm{K} \approx 12$ is achieved. The schematic view of a single PE is shown in Figure 3. The PE consists of a Computation Unit, a Sorting Unit and an Allocating Unit.

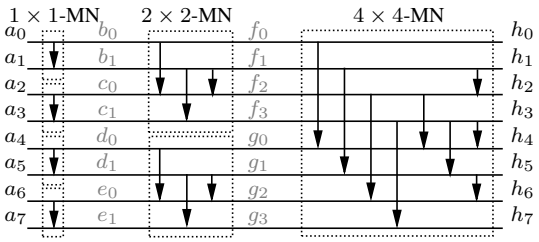

Fig. 4. Odd-Even-Mergesort Network for $n = 8$

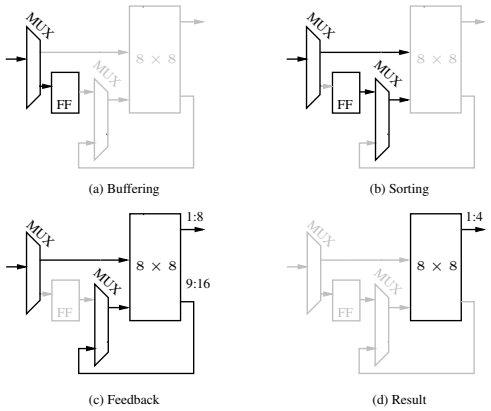

Fig. 5. Reducing the amount of hardware due to a feed back loop and minimizing the last Merge-Network

A. Computation-Unit

The main task of the Computation Unit (CU) is to calculate the PEDs of all child-nodes (eq. (5)), which can be further separated in a configurable and a static part for an effective hardware implementation (Fig. 8). The static part exists four times to calculate all child-nodes in parallel. Two CU are implemented in parallel to calculate eight PEDs in one step.

B. Sorting- and Allocation-Unit

The Sorting Unit (SU) aims to find the K smallest PEDs and the Allocation Unit (AU) forwards it to the next PE. In this implementation a modified Odd-Even Mergesort algorithm is used for the SU, which is explained in detail in the following section. Due to the modification the AU, which is negligible in common K-Best detector designs, is of major interest for the K$^+$-Best detector.

1) Modified Odd-Even Mergesort: The basis for the SU is the K. E. Batchers Odd-Even Mergesort algorithm stated in [4], [5], [6] as it needs the minimal number of comparators for a parallel sorting process of $n = 2^p$ $(p = 1, 2, \ldots, N)$ elements compared to other sorting algorithms.

The algorithm merges gradually two lists, with ascending order, into a sorted one. Figure 4 shows an example of the sorting process of eight elements. The unsorted list **a** is converted into a sorted list **h**. Each arrow denotes the smallest element of a Merge-Network (MN). A 1×1-MN sorts two elements and can be realized by a simple comparator. Due to the fact, that each $2^p \times 2^p$-MN is split into two $2^{p-1} \times 2^{p-1}$-MNs, where the first network contains the elements with an odd- and the second with an even-index, the algorithm is called Odd-Even Mergesort. Comparing the outputs leads to the final list in ascending order. To generate a $2^p \times 2^p$-MN, $p \cdot 2^p + 1$ comparator elements are needed which are arranged in $p + 1$ levels.

For a K $= 8$ i.e. 32 PEDs SU $16 \cdot 1 \times 1$-MNs $+ 8 \cdot 2 \times$ 2-MNs $+ 4 \cdot 4 \times 4$-MNs $+ 2 \cdot 8 \times 8$-MNs $+ 1 \cdot 16 \times 16$-MNs would be necessary which consist of 191 comparators.

To reduce the amount of comparators a pipelined structure can be used which is described in [6] and expanded in [7]. In the first instance 2^p elements are sorted and saved in a register. Within the next clock cycle another set of 2^p elements is sorted and forwarded together with the buffered elements to the input of a $2^p \times 2^p$-MN. Due to this architecture an entire sorting network of $n = 2^p$ elements is saved by the cost of on clock cycle.

To further reduce the complexity of the sorting algorithm for a K-Best detector with K $= 8$ $(8 \cdot 4 = 32$ PEDs), the last 16×16-MN can be replaced by a smaller 8×8-MN, as only the eight smallest PEDs of the entire list are of interest. Thus, only the eight smallest PEDs of the penultimate 8×8-MN are buffered and forwarded together with the eight smallest PEDs within the next clock cycle to the last 8×8-MN. Due to this modification the same result is achieved by reducing the complexity (Table I, 1. Modification).

The principle idea to achieve a BER performance of a K $= 12$ K-Best detector using the sorting algorithm for K $= 8$ is explained in Figure 5. In the first instance (Figure 5a) 16 elements are buffered in a register and subsequently sorted with the next 16 elements (Figure 5b). Hence, the eight smallest PEDs of the first eight nodes are available. Moreover the PEDs $9 - 16$ are also available which can be fed back and sorted with the eight smallest PEDs of the nodes $9 - 12$ (Figure 5c) within the next clock cycle. Thus, the four residual smallest PEDs of the nodes $1 - 12$ are available additionally to the eight smallest PEDs of the nodes $1 - 8$ (Figure 5d).

As only the eight smallest PEDs of the first eight nodes are correctly estimated, a few simplifications are made to increase the BER performance of the detector from eight to twelve nodes without additional hardware:

- Four nodes contain no more than eight of the twelve smallest PEDs
- The last four nodes contain no more than four of the twelve smallest PEDs. This is assisted by the fact, that these four nodes contain the relatively biggest PEDs of the twelve nodes from the previous level.

Due to these simplifications, the detector is called a K$^+$-Best detector. Figure 6 compares the BER performance between the K$^+$-Best detector and a common K-Best detector. It can be seen that the introduced K$^+$-Best detector approximately achieves the same BER performance as the K-Best detector

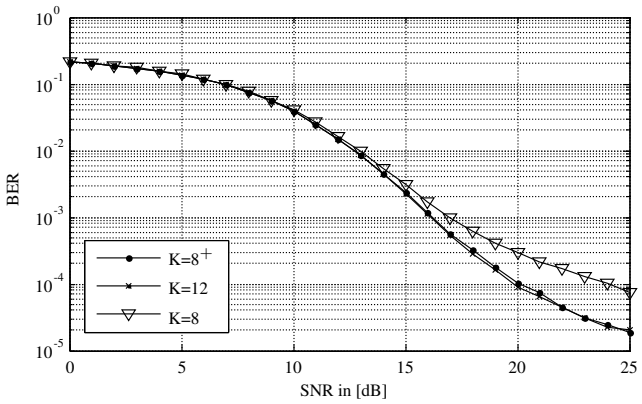

Fig. 6. BER performance of a 16-QAM 4×4 MIMO K-Best detector with a modified Odd-Even-Mergesort ($\mathrm{K} = 8^{+}$) compared to an ideal sorting with $\mathrm{K} = 8$ and $\mathrm{K} = 12$

TABLE I
OVERVIEW ON THE AMOUNT OF HARDWARE AND THE SIZE OF K FOR DIFFERENT SORTERS FOR RVD AND 16-QAM.

Odd-Even-Mergesort	Node	Comparator	Level	Clock (ideal)
W/o Modification	$\mathrm{K} = 8$	191	15	1
Configurable cf. [7]	$\mathrm{K} = 1 - 16$	109	15	$1 - 16$
1. Modification	$\mathrm{K} = 8$	69	14	4
2. Modification (K^{+}-Best) cf. Fig. 5	$\mathrm{K} \approx 12$	69	14	6

with $\mathrm{K} = 12$. Table I compares the different sorting strategies. It is clearly traceable how the amount as well as the number of comparators are reduced by the explained modification. Furthermore, it must be pointed out that the sorter can be further optimized depending on the modulation and search level, which will be explained in section V.

2) Allocation-Unit: Caused by the modifications and the assumptions of the SU the AU is of importance as the order, in which the nodes are forwarded to the CU, affects the result of the K^{+}-Best detector. The description of the SU shows that only the four smallest PEDs of the nodes $9-12$ are considered. Furthermore, these PEDs do not affect the eight smallest PEDs of the first eight nodes. In case that the smallest PED within the level i is generated by a parent-node with the smallest PEDs $9 - 12$ of the level $i + 1$, it would not be possible, that this PED could be counted to the eight best nodes. As it is important that the smallest PED is counted to the first eight nodes a reallocation is necessary. Figure 7 shows the schematic of the AU. In the first step the four smallest PEDs of the first eight nodes are forwarded to the CU, followed by the two smallest PEDs of the nodes $9-12$ (including the PEDs $9 - 16$ of the nodes $1 - 8$). Afterwards the residual nodes are forwarded to the CU in natural order. Therefore, the PEDs (7:8, 9:10, 11:12) need to be buffered.

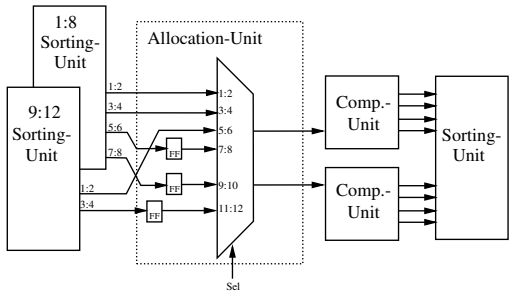

Fig. 7. Architecture of the Allocation-Unit

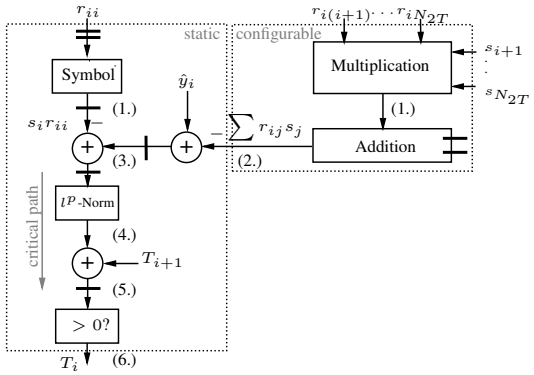

Fig. 8. Schematic of the pipelining levels and the critical path of the Compuation-Unit

IV. FPGA-IMPLEMENTATION

A. Fix-Point Implementation

To achieve a high performance, low complexity detector, a Q-fomat fix-point arithmetic is used for the FPGA-implementation [8]. The notation $Q_{m.n}$ denotes a fix-point number, where m denotes the number of integer-bit and n denotes the number of fractional-bit. Figure 8 shows a schematic view of the CU. The input data is assumed to be in a $Q_{0.15}$ format, as the RVD 16-QAM symbols s_i are given by a $Q_{2.0}$ number, the multiplication *(1.)* of an element r_{ii} with a symbol s_i leads to an 18 bit wide $Q_{2.15}$ result, which is afterwards shifted to the right by two ($Q_{2.13}$) to reduce the number of bits. The following two operations, the summation *(2.)* of the separate products and the addition *(3.)* of $\hat{y}_i$ with the product $s_i r_{ii}$ imply the risk of an overflow, but in terms of the accuracy the preventive right shift is left out. In case of an l^2-norm calculation *(4.)*, squaring a $Q_{2.13}$ number leads to a $Q_{4.26}$ result. As the result is always positive the sign bit is known in advance and therefore, the upper 16 bit can be used storing the result. The risk of an overflow in step *(5.)* is reduced, as T_{i+1} is stored as an 11 bit value which saves hardware resources in the SU. Thus, the 16 bit result needs to be reduced to an 11 bit representation *(6.)*. Therefore, PEDs which are greater than the range of an 11 bit value are clipped.

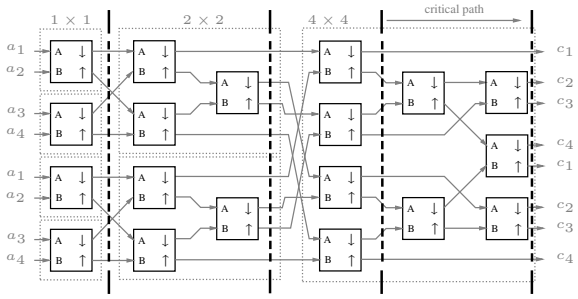

Fig. 9. Schematic of the pipelining levels and the critical path of the Merge-Network

B. Pipelining

To increase the throughput of a digital circuit, pipelining is a well known strategy. Due to the insertion of pipelining registers, the combinatorial circuit is split into several sub-circuits which can be clocked by a higher frequency and thus, the throughput is increased by the cost of the size and the latency.

The variation of the throughput Θ due to the pipelining levels, which is defined as the number of bits d per second, can be described by:

$$\Theta = d \cdot \frac{1}{t_{cp}} \leq d \cdot \frac{1}{t_{lp}} = d \cdot \frac{p}{t_f + t_{reg}}, \tag{7}$$

where t_{cp} denotes the computation time, t_{lp} the longest path of the circuit, t_f the time of a combinatorial function and t_{reg} is the delay time of a single register. As stated before, equation (7) shows that the throughput of the circuit is increased by p under the assumption that $t_{reg} \ll t_f/p$ [9].

1) Throughput Considerations: The maximum theoretically achievable throughput of an LTE system using a 16-QAM modulated signal over a bandwidth of 20 MHz and a 4×4-MIMO configuration is about 217.6 Mbit/s. Thus, assuming the latency $L = 1$ to detect one symbol, the minimal clock frequency

$$f_{min} = \frac{217.6 \text{ Mbit}}{16 \text{ Bit} \cdot s} = 13.6 \cdot 10^6 \frac{1}{s} = 13.6 \text{ MHz} \tag{8}$$

is required to reach the LTE peak data rate. As the introduced K$^+$-Best detector needs six cycles ($L = 6$) to detect one symbol the minimal clock frequency should be:

$$f_{min} = 13.6 \text{ MHz} \cdot 6 = 81.6 \text{ MHz}. \tag{9}$$

To meet these requirements the K$^+$-Best detector is highly pipelined. The pipelining levels of the CU and a 4×4-MN are shown in Figure 8 and Figure 9. For example, as the maximum frequency f_{max} of a 2×11 Bit comparator is expected as approx. 233 MHz, the critical path of the 4×4-MN does not exceed more than two comparators connected in serial.

V. Complexity and Performance

Figure 10 shows the VHDL-construction of the PEs. As only the levels $6 - 2$ of the search tree are fully constructed (32 PEDs per tree level), only the PE $6-2$ are fully populated.

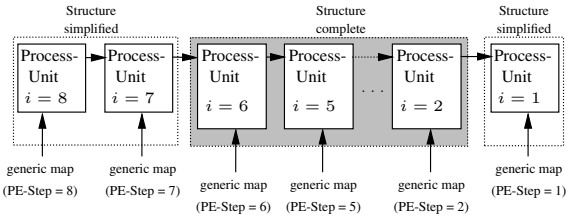

Fig. 10. VHDL-construction of the PEs using generics to configure the word width and the Computation Unit

TABLE II
Utilization of the Xilinx XC2 V6000 by the K$^+$-Best detektor.

	Used	Available	Utilization
Slices	15682	33792	46%
Flip-Flops	20682	67584	30%
LUTs	23699	67584	35%
HW-Multiplier	116	144	80%

For example, the PE 8 needs no SU as only four PEDs are calculated. Therefore, to save valuable hardware resources the first two and the last PEs are reduced using generic map commands in VHDL.

The K$^+$-Best detector is synthesized and mapped on a Xilinx XC2 V6000 and a Xilinx XC4 VFX60 FPGA of a multi processor board to verify the functionality directly on the hardware. Figure 11 shows the BER of the detector generated by the FPGA and simulations of a common K-Best detector with $\mathrm{K} = 8, 10, 12$. The BER generated by an FPGA is still comparable with the BER performance of a K-Best detector with $\mathrm{K} = 12$. The minor tolerances compared to the simulated BER of the K$^+$-Best detector (Figure 6), are caused by the fix-point implementation due to the bounded number of used bit.

Exclusively for a complexity estimation and comparison, as in other K-Best detector designs, the l^1-norm[1] is used to calculate the PEDs. Thus, the dedicated multiplier to calculate the l^2-norm can be neglected and the 56 3 Bit $\times$ 16 Bit multiplications of the CU, implemented as dedicated multiplier within the FPGA, can be approximated by $56 \cdot 39$ LUTs $=$ 2184 LUTs.

Table II shows the number of resources used for a Xilinx XC2 V6000 FPGA synthesis. The average utilization is about 30% of the flip-flops (FF), 46% of the slices and 35% of the Lookup tables (LUTs). Using the following equation, the Gate Equivalent (GE) can be estimated [12].

$$(\text{LUTs with FF}) \cdot 12 \text{ GE} + (\text{LUTs w/o FF}) \cdot 6 \text{ GE} = \#\text{GE}, \tag{10}$$

with #LUTs w/o FF $=$ LUTs $-$ FFs $+$ LUTs of MULT. Applied to the K$^+$-Best detector the number on GEs can be determined as:

$$20682 \cdot 12 \text{ GE} + 5201 \cdot 6 \text{ GE} = 279390 \text{ GE}. \tag{11}$$

Table IV shows the performance characteristics of the implementations. As it can be seen the maximum clock frequency

[1]The l^2-norm is approx. by the l^1-norm ($\sqrt{\mathbf{n}^2 + \mathbf{m}^2} \approx |\mathbf{n}| + |\mathbf{m}|$)

TABLE III
COMPARISON OF DIFFERENT K-BEST-DETECTORS FOR A 4×4 MIMO SYSTEM.

Reference		**[3]**[a]	**[3]**[b]	**[10]**	**[11]**	**[6]**	**[7]**	**Virtex II This work**	**Virtex IV This work**
Modulation		16-QAM	16-QAM	16-QAM	16-QAM	16-QAM	QPSK, 16-QAM, 64-QAM	16-QAM	16-QAM
K		5	10	4	4	$1-8$	$1-16$	12	12
Frequency	[MHz]	132	52	100	120	15	47	128.4	170.9
Throughput	[Mbit/s]	424	83	400	480	$240-30$	$564-12$	342.4	455.7
Latency	[μs]	0.4	1.71	-	0.02	$0.53-3.8$	$0.37-4.78$	1.14	0.86
Gate Equivalent	GE	93k	135k	188k	150k	460k	300k	279k	279k
Platform		ASIC	ASIC	FPGA	FPGA	FPGA	ASIC	FPGA	FPGA

TABLE IV
PERFORMANCE CHARACTERISTICS OF THE XC2 V6000 AND THE XC4 VFX60 IMPLEMENTATION OF THE 16-QAM K^+-BEST DETECTOR.

		Xilinx XC2 V6000	**Xilinx XC4 VFX60**
Frequency	[MHz]	128.4	170.9
Throughput	[Mbit/s]	342.4	455.7
Critical path	[ns]	7.7	5.8
Latency	[μs]	1.14	0.86

f_{max} of the XC4 VFX60 is about 170.9 MHz and respectively the throughput is about 455.7 Mbit/s.

A comparison of different K-Best detectors is given in Table III. All of them make a compromise between the throughput and the number of searched nodes K. This can be reached for a small K by a serial sorting process. The advantage of the introduced detector is its high throughput for a 16-QAM modulated signal, searching a high number of nodes, by requiring a moderate complexity compared to other parallel sorting designs.

VI. CONCLUSION

The architecture and the FPGA implementation aspects of a moderate complexity, high performance and high throughput K-Best detector based on a modified Odd-Even Mergesort algorithm are shown. Due to a few simplifications in the search process a detector called K^+-Best is introduced which achieves the BER performance of a common $K = 12$ K-Best detector by using a sorting algorithm for $K = 8$. The throughput of the K^+-Best detector is twice as high as the LTE peak data rate of 217.6 Mbit/s for a 16-QAM modulated signal. Therefore, it will be suitable for future mobile communication standards like LTE-Advanced. The detector is implemented on a Xilinx XC2 V6000 and a Xilinx XC4 VFX60 FPGA of a multi processor board and the FPGA generated BER is shown. As the introduced architecture by now implies the possibility to generate a symbol candidate list, future research will address to expand the detector by a soft-output unit to make it suitable for coded systems. Furthermore, the opportunity of a configurable modulation scheme will be considered.

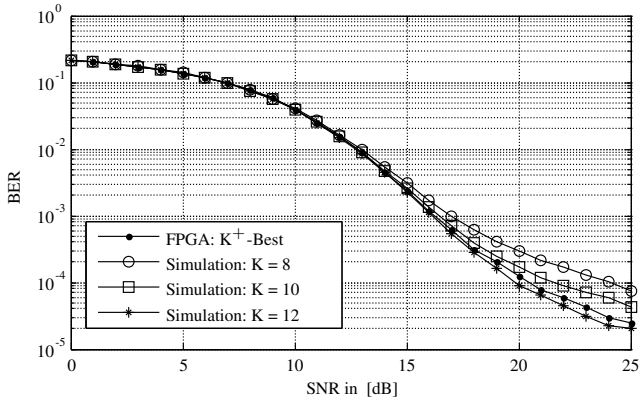

Fig. 11. BER performance of the 4×4 16-QAM K^+-Best detector generated by the FPGA using the l^2-norm

REFERENCES

[1] 3rd Generation Partnership Project. Technical specification group radio access network; feasibility study for evolved universal terrestrial radio access (utra) and universal terrestrial radio access network (utran). Technical Report TR 25.912, 3GPP, 12 2008.

[2] A. Burg. *VLSI Circuits for MIMO Communication Systems*. PhD thesis, Swiss Federal Institute of Technology, 2006.

[3] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner. K-best mimo detection vlsi architectures achieving up to 424 mbps. In *Proc. IEEE International Symposium on Circuits and Systems ISCAS 2006*, pages 4 pp.–1154, 2006.

[4] K. E. Batcher. Sorting networks and their applications. In *AFIPS '68 (Spring): Proceedings of the April 30–May 2, 1968, spring joint computer conference*, pages 307–314, New York, NY, USA, 1968. ACM.

[5] Qingwei Li and Zhongfeng Wang. Reduced complexity k-best sphere decoder design for mimo systems. *Circuits, Systems, and Signal Processing*, 27:491–505, 2008.

[6] R. Shariat-Yazdi and T. Kwasniewski. Reconfigurable k-best mimo detector architecture and fpga implementation. In *Proc. International Symposium on Intelligent Signal Processing and Communication Systems ISPACS 2007*, pages 349–352, 2007.

[7] R. Shariat-Yazdi and T. Kwasniewski. Configurable k-best mimo detector architecture. In *Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on*, pages 1565 – 1569, 12-14 2008.

[8] N. Kehtarnavaz. *Real-time digital signal processing based on the TMS320C6000.* Elsevier, Amsterdam, 2005. xii, 306 p.

[9] H. Kaeslin. *Digital integrated circuit design: from VLSI architectures to CMOS fabrication*. Cambridge Univ. Press, Cambridge [u.a.], 2008. XXII, 845 S. ; 25 cm : zahlr. Ill., graph. Darst.

[10] L.G. Barbero and J.S. Thompson. Rapid prototyping of a fixed-throughput sphere decoder for mimo systems. In *Communications, 2006. ICC '06. IEEE International Conference on*, volume 7, pages 3082 – 3087, june 2006.

[11] R. Shariat-Yazdi and T. Kwasniewski. Dual mode k-best mimo detector architecture and vlsi implementation. In *14th IEEE ICECS 2007. 14th*, pages 735 –738, 11-14 2007.

[12] Xilinx. *Gate Count Capacity Metrics for FPGAs*, v1.1 edition, 02 1997.