Implementation Comparisons of the QR Decomposition for MIMO Detection

Gabriel L. Nazar[1,2]
glnazar@inf.ufrgs.br

Christina Gimmler[2], Norbert Wehn[2]
{gimmler, wehn}@eit.uni-kl.de

[1]Instituto de Informática - Universidade Federal do Rio Grande do Sul
Av. Bento Gonçalves, 9500, Bloco IV - Porto Alegre, RS, Brazil

[2]Fachbereich Elektrotechnik und Informationstechnik -Technische Universität Kaiserslautern
Erwin-Schrödinger-Straße - 67663 Kaiserslautern, Germany

ABSTRACT
In the context of decoding multiple-input multiple-output (MIMO) symbols, many approaches rise as promising, such as successive interference cancellation and sphere decoding. The QR decomposition (QRD) of the channel impulse response matrix is a critical point to ensure good performance of the subsequent decoding steps for both approaches. This paper presents a low-complexity hardware architecture for the basic QRD algorithm, which is extended to two improved versions, namely the sorted QR decomposition (SQRD) and the minimum mean-square error SQRD. The main contribution of this work is a comparison of hardware implementations of the three variants and an analysis of their impact on a MIMO-BICM system regarding system communications performance and computational complexity.

Categories and Subject Descriptors
B.4.1 [**Integrated Circuits**]: Input/Output and Data Communications – *data communication devices.*

General Terms
Algorithms, Performance, Design.

Keywords
MIMO detection, SQRD, MMSE-SQRD.

1. INTRODUCTION
Multiple-input multiple-output (MIMO) transmission is considered a key technology to reach the high data rates expected from future wireless technologies, especially due to its increased spectral efficiency [1]. However, in such systems the complexity of the receiver is significantly increased. The QR decomposition is of critical importance to many algorithms for MIMO detection that attempt to reduce this complexity. Such algorithms include sphere decoding, successive interference cancellation and extensions of V-BLAST [2]. It must be performed each time the channel impulse response matrix changes significantly. The time between changes is inversely proportional to the carrier frequency and the maximum supported receiver speed, which means that high carrier frequencies and fast moving receivers require a fast QR decomposition hardware. Also, extended versions of the algorithm, specifically the sorted QR decomposition (SQRD) and the minimum mean-square error SRQD (MMSE-SQRD), may result in even better decoding results for linear detectors. When dealing with tree-based search algorithms, such as sphere decoders, these techniques are used to reduce the computational effort [3].

There are already a number of QR decompositions architectures [4][5][6], which consider mainly the implementation results isolated. The main contribution of this work, on the other hand, is a comparison regarding the effects on the system level. This includes variations in the FER and in the computational effort associated with MIMO detection using a soft-input soft-output (SISO) detector. The hardware area and latency are compared as well, considering the basic QR decomposition algorithm and the mentioned extensions.

This paper is structured as follows: section 2 presents the communication chain model used to obtain the simulation results. Section 3 introduces the chosen QR decomposition algorithm and the SQRD and MMSE-SQRD extensions. Section 4 describes the implemented architectures. Section 5 presents the simulation results and section 6 the hardware implementation results, comparing the three different versions. Finally, section 7 presents the conclusions achieved from the presented results.

2. SYSTEM MODEL
Figure 1 shows the employed communication chain. The analyzed system uses bit-interleaved coded modulation (BICM). The source generates a random bit vector **b**, in which each bit has equal probability of being 1 or 0. The channel encoder uses a convolutional code with 64 states, non-systematic, non-recursive, and using the Maxlog MAP (BCJR) algorithm to encode **b** into **c'**. The encoded sequence is then interleaved and mapped by an M-QAM mapper into complex symbols **s**, which are transmitted over a noisy channel with Rayleigh fading. The $N_R \times 1$ received vector **y** is given by:

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \quad (1)$$

where **s** is the $N_T \times 1$ sent vector and **n** is the $N_R \times 1$ noise vector. All vectors and matrices are complex. **H** and **n** are composed of random values with mean zero. **H** has unit variance and **n** has variance N_0. In the remainder of this work, we consider $N_R = N_T = N$.

SBCCI'10, September 6–9, 2010, São Paulo, Brazil.

The received sequence **y** then goes through the iterative decoder, in which the MIMO detector and the channel decoders exchange extrinsic soft information. When the decoding process is finished the estimated bit vector $\hat{\mathbf{b}}$ can be compared to **b**.

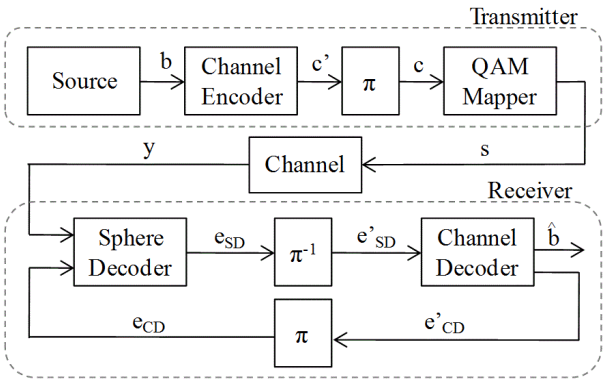


Figure 1. Communication Chain Model

As mentioned, this paper focuses on SISO detectors, which have been widely used as a method to decode MIMO symbols. Let $\mathbf{x}_{i,k}$ denote the *k*th bit of the *i*th symbol. SISO MIMO detectors attempt to find the bits $\mathbf{x}_{i,k}$ most likely sent and measure the reliability for each bit. This can be done using *logarithmic likelihood ratios* λ:

$$\boldsymbol{\lambda}_{i,k} = \log \frac{P(\mathbf{x}_{i,k} = +1 \mid \mathbf{y})}{P(\mathbf{x}_{i,k} = -1 \mid \mathbf{y})} \tag{2}$$

Equation 2 can be approximated by the difference of the minimum distance metric d(**s**) for each bit being 1 or 0. This metric, considering the a-priori information L^a, measures the likelihood that a sequence **s** was sent. It is given by:

$$d(s) = \left\| \mathbf{y} - \mathbf{H}\mathbf{s} \right\|^2 - \frac{N_0}{2} \sum_{i,k} \mathbf{x}_{i,k} L^a(\mathbf{x}_{i,k}) \tag{3}$$

The QR decomposition is required to map this problem into a tree search, based on the sphere decoding algorithm. This is performed in a tree with N+1 layers, in which each layer after the root represents one chosen symbol of the decoded **s** vector. Therefore, for an M-QAM constellation, each node has M children. A partial Euclidean distance (PED) can be recursively calculated for each node, and, when a leaf node is reached, its distance is associated to the entire symbol sequence that connects it to the root. The matrices **Q** and **R**, produced by the QRD, are used to calculate the PEDs. The PED increase $|e_i(\mathbf{s}^{(i)})|^2$ at the *i*th layer, starting with i=N-1, is given by:

$$\left| e_i(\mathbf{s}^{(i)}) \right|^2 = \left| \hat{\mathbf{y}}_i - \sum_{j=i}^{N-1} \mathbf{R}_{ij} \mathbf{s}_j \right|^2 + \sum_k \mathrm{LLR}(\mathbf{a}_{i,k}, \mathbf{x}_{i,k}) \tag{4}$$

$$\hat{\mathbf{y}} = \mathbf{Q}^H \mathbf{y}$$

In equation 4 the LLR function is responsible for considering the extrinsic information $\mathbf{a}_{i,k}$, for each bit $\mathbf{x}_{i,k}$, from the previous decoder iteration. During the search, whenever a node's PED is larger than the value defined as the sphere radius, its entire sub-tree can be excluded from further search, as those sequences are unlikely to contribute to the solution. Since the total amount of visited nodes is variable and is decisive to the total computation time, it can be seen as a good measure of the complexity of the search algorithm.

In equation 4 the LLR function is responsible for considering the extrinsic information $\mathbf{a}_{i,k}$, for each bit $\mathbf{x}_{i,k}$, from the previous decoder iteration. During the search, whenever a node's PED is larger than the value defined as the sphere radius, its entire sub-tree can be excluded from further search, as those sequences are unlikely to contribute to the solution. Since the total amount of visited nodes is variable and is decisive to the total computation time, it can be seen as a good measure of the complexity of the search algorithm.

3. QRD ALGORITHMS

The QR decomposition of a matrix **H** is a factorization that produces two output matrices, namely the orthogonal matrix Q and the upper-triangular matrix R. It can be computed using several different algorithms, but more commonly Householder reflections, Givens rotations or the Gram-Schmidt process [7]. In this work, we propose a dedicated hardware architecture to compute the QR decomposition based on the modified Gram-Schmidt (MGS) algorithm, which has improved numerical stability when compared to its original version [7]. Algorithm 1 shows the basic MGS process.

Let $\mathbf{q}_k$ denote the *k*th column of matrix **Q**, $\mathbf{q}_k{}^H$ the Hermitian transpose of $\mathbf{q}_k$ and R(k, j) the element in the *k*th row and *j*th column of **R**.

```
(1)  R = 0; Q = H
(2)  for k=0:N-1
(3)     R(k, k) = ‖ qk ‖
(4)     qk = qk / R(k, k)
(5)     for j=k+1:N-1
(6)        R(k, j) = qk^H . qj
(7)        qj = qj – qk . R(k, j)
(8)     end
(9)  end
```

Algorithm 1. Modified Gram-Schmidt Process

The MGS algorithm can be extended to compute the sorted QR decomposition [8], which attempts to sort the elements in the main diagonal of **R** decreasingly in the order they are used by the decoding algorithm, i.e. from the bottom right corner to the top left corner. The purpose of this sorting is to assign the transmit antennas with strongest signal to the layers closer to the root in the sphere decoding search tree, since each transmit antenna is associated with one column of **H**, in an attempt to increase the signal-to-noise ratio (SNR) in those layers. This order is backwards to that in which these elements are calculated by the MGS algorithm. Therefore the SQRD algorithm attempts, at each iteration, to minimize the element in the main diagonal of **R** currently being calculated by swapping the *k*th column in **Q** with that with the smallest norm still not used. The permutation vector **p** is used only to keep track of the exchanges performed, information required later in the decoding process. In order to reduce the amount of squared norms to be calculated this algorithm uses a norm vector, which is filled before the rest of the execution and then only updated [2]. This is a greedy algorithm that does not ensure optimal ordering. Let *conj* denote the

complex conjugate of a value. The MGS algorithm, extended to compute the SQRD becomes:

(1) R = 0; Q = H; p = [0, 1, …, N-1]
(2) **for** k=0:N-1
(3) norm(k) = $\| \mathbf{q}_k \|^2$
(4) **end**
(5) **for** k=0:N-1
(6) i = $\arg \min_{j=k:N-1}$ norm(j)
(7) Exchange columns k and i in Q, R, p and norm
(8) R(k, k) = $\| \mathbf{q}_k \|$
(9) $\mathbf{q}_k = \mathbf{q}_k / R(k, k)$
(10) **for** j=k+1:N-1
(11) $R(k, j) = \mathbf{q}_k^H . \mathbf{q}_j$
(12) $\mathbf{q}_j = \mathbf{q}_j - \mathbf{q}_k.R(k, j)$
(13) norm(j) = norm(j) – *conj*(R(k, j)).R(k, j)
(14) **end**
(15) **end**

Algorithm 2. Sorted QR decomposition

The other extension of the QRD analyzed is the minimum mean-square error SQRD (MMSE-SQRD) [2], which couples the sorting previously described with the MMSE criterion. The MMSE can be seen as a compromise between complete interference cancellation and noise amplification [2], since it attempts to reduce the problem of amplified noise found in systems that attempt to completely suppress the interference of each transmit antenna.

The MMSE-SQRD is computed using an extended version of the input matrix **H**, which results in an extended output matrix **Q**, while **R** remains the same size. Let $\mathbf{I}_N$ denote the N×N identity matrix and σ the noise vector standard deviation. The MMSE-SQRD is computed with modified matrices as follows:

$$\overline{\mathbf{H}} = \begin{bmatrix} \mathbf{H} \\ \sigma \mathbf{I_N} \end{bmatrix} = \begin{bmatrix} \mathbf{Q_1} \\ \mathbf{Q_2} \end{bmatrix} \mathbf{R} \qquad \hat{\mathbf{y}} = \mathbf{Q_1}^H \mathbf{y} \tag{5}$$

$\mathbf{Q}_1$ and $\mathbf{Q}_2$ are N×N matrices that form the extended output. Note that $\mathbf{Q}_2$ can be discarded after the computation is done, since only $\mathbf{Q}_1$ is necessary to proceed with the decoding. The matrices are calculated using basically the same algorithm for the SQRD, with the exception that only the N+k-1 rows of **Q** are exchanged, in line 7 of Algorithm 2.

4. HARDWARE ARCHITECTURES

A fixed point hardware architecture was derived from Algorithm 1 to compute the basic QRD. This algorithm, however, has some clear hardware implementation issues: the vector norm in line 3 requires a square root hardware and the calculation of $\mathbf{q}_k$ in line 4 requires a vector division operation. These problems can be alleviated by replacing the operation in line 3 by an inverse square root, which also allows the replacement of the divisions in line 4 by multiplications [6]. The inverse square root hardware used in our architecture is of low complexity, as in [6], which computes an approximated value using first order polynomials. To achieve the required precision by this operation, the approximation hardware is followed by one Newton-Raphson iteration. The actual vector norm, required to form the main diagonal of **R**, is calculated by multiplying $\| \mathbf{q}_k \|^2$ and $1/\| \mathbf{q}_k \|$, which are the input and output of the inverse square root block, respectively.

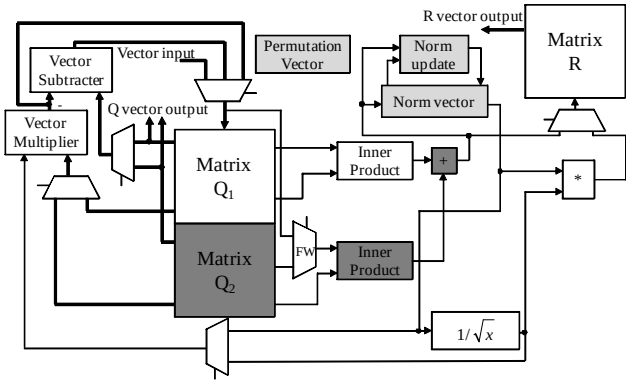


Figure 2. Hardware architecture for QRD. The blocks in light gray must be added to perform the SQRD. To perform the MMSE-SQRD, the blocks in dark gray must also be added

Figure 2 shows the implemented hardware, where the thick lines represent full columns and the thin ones represent scalar values. The storage blocks, which maintain **Q** and **R** at all times, are composed of registers grouped by lines, which allow quick access to entire columns. This way, vector operations can be parallelized, such as the multiplication of a vector by a scalar or the subtraction of two vectors. Matrix **Q** has two column and two scalar outputs. For the MMSE case, $\mathbf{Q_1}$ and $\mathbf{Q_2}$ have their own independent outputs. The scalar outputs are simply a selection of one of the positions from the column outputs. The multiplexer marked "FW" is for data forwarding, to bypass the first element in the input port of **Q**, saving cycles when the vector being written is required by the inner product block.

The inner (or dot) product is the multiplication of two vectors, where the first is represented as a row and the second as a column. It yields, therefore, a single scalar as output and can be seen in lines 3, where it is used to calculate the norm, and 6 of Algorithm 1. It is the only vector operation computed sequentially, also due to its scalar output.

The extension of the architecture to compute the SQRD requires the blocks in light gray. It must allow quick swapping operations between columns in **Q** and **R**, as well as support the norm vector and the norm update operation (line 13 in Algorithm 2). The swaps are performed by internal connections in the storage blocks, using the same multiplexers of the outputs. This allows swaps to be performed in a single cycle. The norm update is also performed by dedicated hardware, allowing it to be calculated in parallel with the computation of $\mathbf{q}_j$, in line 12. The permutation vector must also be created and perform all the same exchanges done in the matrices, even though it is not connected to any other block in the data path.

To extend the SQRD architecture to the MMSE-SQRD version the blocks in dark gray must also be added. Also, in order to not increase the area excessively, since we are now using vectors with 2N elements, the operations that were performed fully in parallel

have to be modified. More specifically, the vector multiplications and vector subtractions are now performed in two steps, each operating over half vector. Another inner product block was added, allowing each block to operate over half vector in parallel, thus eliminating increases in the amount of cycles required by this operation.

All operations use saturation arithmetic. Also, all multiplications are followed by rounding operations to keep the bit widths constant.

5. SIMULATION RESULTS

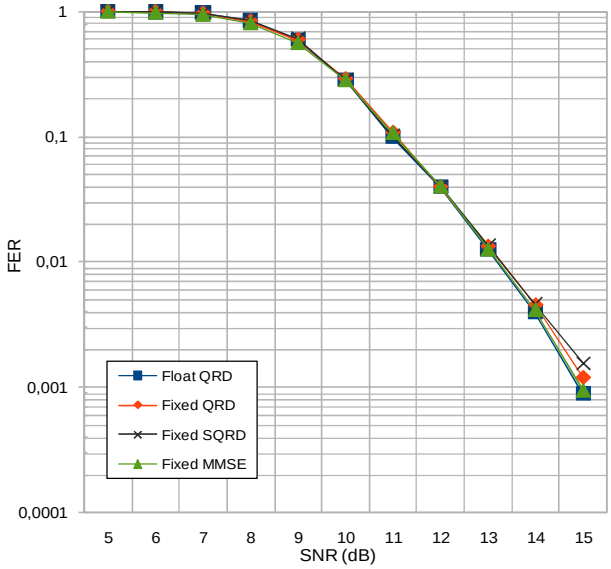
In order to determine the required amount of bits in the fixed point quantization, as well as evaluate the variations due to the usage of SQRD or MMSE-SQRD, a floating point software simulation model was created. The model comprises the entire MIMO-BICM communication chain. The replacement of the original floating point QR decomposition by the fixed point version, also considering the approximated inverse square root, allows analyzing the isolated effects of these changes. The remainder of the chain still uses floating point. Also, the full floating point system is used as reference.

The simulations consider frames with 994 bits before the channel encoder, which become 2000 after encoding. These results consider a 16-QAM constellation and the number of iterations in the receiver is limited to 5. The system uses 4×4 antennas, i.e. 4 for reception and 4 for transmission. The sphere radius is constant throughout the execution. It was dimensioned for each algorithm to the smallest value that ensured FER degradation smaller than 0.3dB, when compared to an unconstrained search. The chosen radii are 0.8 for QRD and SQRD and 0.4 for MMSE-SQRD. The usage of MMSE significantly alters the distance metrics and therefore allows a significant reduction in the sphere radius.

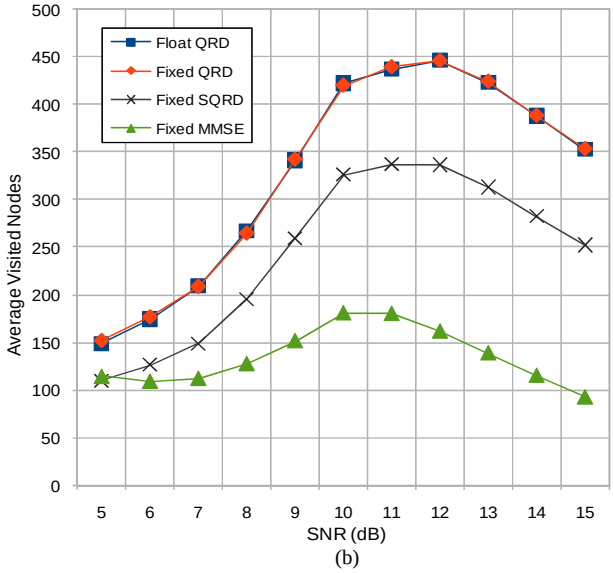
Figure 3a shows the FER associated with each of the different QR decomposition variations, after iterative MIMO detection and channel decoding. It shows that the degradation is smaller than 0.4dB for the chosen SNR interval, when compared to the full floating point version. The QRD and MMSE-SQRD fixed point versions use 4 integer and 8 fractional bits, including the sign bit. The SQRD requires one extra fractional bit to achieve satisfactory performance and therefore uses 4 integer and 9 fractional bits. Figure 3b shows the average computational effort measured by average visited nodes per tree built for each SNR point. As expected, the floating and fixed point versions of the basic QRD show no significant variation. The SQRD and MMSE-SQRD versions, on the other hand, show significant reduction. Table 1 shows the average and peak node counts *NC* for each algorithm and the speed up *S*, compared to the QRD version.

Table 1. Node counts statistics for each QRD variation

	NC_{avg}	NC_{peak}	S_{avg}	S_{peak}
QRD	329	446	1	1
SQRD	245	337	1.34	1.32
MMSE-SQRD	136	182	2.42	2.45



(a)



(b)

Figure 3. FER and average visited nodes for different QR decomposition versions

6. IMPLEMENTATION RESULTS

The proposed architectures were implemented in hardware and synthesized using the Synopsys Design Compiler and a state-of-the-art 65nm library. To allow greater flexibility in the system, the implemented hardware works with *up to* N antennas, which means that it can deal with smaller matrices when there are not enough antennas available. In our implementation, the hardware performs the QRD for 4×4 and 2×2 matrices.

Table 2 shows the cycle count for each variation of the algorithm. The increase observed when comparing QRD and SQRD is due to the vector exchange operations required. On the other hand, the variation between SQRD and MMSE-SQRD is due to the division of the vector multiplications and subtractions performed in two steps, to reduce the increase in hardware.

Table 2. Cycles required by each QRD variation for 2×2 and 4×4 matrices

	Cycles 2×2	*Cycles 4×4*
QRD	43	138
SQRD	45	147
MMSE-SQRD	48	157

These architectures were able to reach up to 500MHz for the target technology, which leads to a total computation latency of 276ns for QRD, 294ns for SQRD and 314ns for MMSE-SQRD. They may also be synthesized for lower frequencies, leading to reduced area and power consumption. Figure 4 shows the area required by each algorithm, in mm^2, considering the fixed point quantizations mentioned in section 5, for different target frequencies.

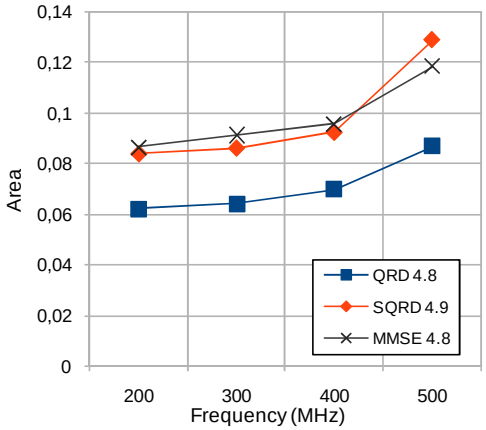


Figure 4. Total area for different versions of the QRD algorithm

The QR decomposition total latency can be compared to a real time requirement, such as presented in [6]. The time in which the channel impulse response in essentially invariant is given by the coherence time t_{coh}:

$$t_{coh} = \frac{c}{v_r f_c} \quad (6)$$

Where c is the speed of light (3×10^8m/s), v_r is the maximum receiver speed and f_c is the carrier frequency. Considering, for example, f_c =2.4GHz and v_r =500km/h, t_{coh} =0.9ms. 3G LTE MIMO systems will make use of OFDM modulation, where the maximum number of data carrying sub-carriers is 1021 [9]. This leaves us with 881ns to perform the QR decomposition of the **H** matrix of each sub-carrier. Since the variation that takes the longest, the 4×4 MMSE-SQRD, takes 157 cycles, its total computation time, at 200MHz, is 785ns, respecting the real time requirement. Evidently, if higher carrier frequencies or more OFDM sub-carriers are considered, the versions with higher clock frequency may be required.

The area and timing results showed in this section, coupled with the complexity analysis in Figure 3, show that both improved algorithms provide significant reduction in computational effort at the expense of an increase in the area, compared to the basic QRD. The SQRD algorithm, however, provides smaller gain in detection complexity, while reaching similar area results, when compared to the MMSE version. Due to this we consider the MMSE-SQRD to be, so far, the best choice for pre-processing, among the considered options.

7. CONCLUSIONS

MIMO systems are a promising approach to achieve very high data rates in future wireless systems. The QR decomposition is the vital component for the preprocessing of a MIMO detector. We considered a QR decomposition hardware for a MIMO-BICM system with iterative detection and decoding. We compared QRD, SQRD and MMSE-SQRD regarding communications performance, detection complexity, area and latency of the hardware implementations. The area for the MMSE-SQRD implementation increases by 40% compared to the basic QRD implementation. However, the algorithmic complexity of MIMO detection is decreased by 58.7% at the same time. Thus, an overall complexity reduction is achieved, showing that the MMSE-SQRD algorithm and corresponding hardware architecture presented are good choices for MIMO detecting systems.

REFERENCES

[1] E. Telatar, “Capacity of multi-antenna Gaussian channels,” European Trans. Telecommun., vol. 6, pp. 585–595, Nov.-Dec. 1999.

[2] D. Wübben, R. Böhnke, V. Kühn, and K. D. Kammeyer, “MMSE Extension of V-BLAST based on Sorted QR Decomposition,” in IEEE Proc. Vehicular Technology Conference (VTC), Orlando, Florida, USA, October 2003.

[3] B. Mennenga, R. Fritzsche, G. P. Fettweis, "Iterative Soft-In Soft-Out Sphere Detection for MIMO Systems," in Proceedings of 69th IEEE Vehicular Technology Conference (VTC-Spring'09), Barcelona, Spain, 26.-29. April 2009.

[4] P. Luethi, C. Studer, S. Duetsch, E. Zgraggen, H. Kaeslin, N. Felber, and W. Fichtner, "Gram-Schmidt-Based QR Decomposition for MIMO Detection: VLSI Implementation and Comparison," In Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Macao, China, November 2008.

[5] P. Luethi, A. Burg, S. Haene, D. Perels, N. Felber, and W. Fichtner, “VLSI Implementation of a High-Speed Iterative Sorted MMSE QR Decomposition,” in Proc. of IEEE ISCAS, May 2007, pp. 1421–1424.

[6] P. Salmela, A. Burian, H. Sorokin, and J. Takala, "Complex-valued QR decomposition implementation for MIMO receivers," in proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Las Vegas, NV, USA, Mar. 30-Apr. 4, 2008, pp. 1433-1436.

[7] G. H. Gollub and C. F. Van Loan, “Matrix Computations,” Baltimore, MD, USA: John Hopkins University Press, 1996.

[8] D. Wübben, R. Böhnke, J. Rinas, V. Kühn, and K. D. Kammeyer, “Efficient Algorithm for Decoding Layered Space-Time Codes,” IEE Electronics Letters, vol. 37, no. 22, pp. 1348–1350, Oct. 2001.

[9] R. Bachl, P. Gunreben, S. Das, and S. Tatesh, “The long term evolution towards a new 3GPP* air interface standard,” Bell Labs Technical Journal, vol. 11, no. 4, pp. 25–51, Mar. 2007.