A Performance Adjustable and Reconfigurable CDMA Receiver Concept for UMTS-FDD

Benjamin Heyne*, Marius Otte, Jürgen Götze
*University of Dortmund, Information Processing Lab
Otto-Hahn-Str. 4, 44221 Dortmund, Germany
E-Mail: benjamin.heyne@uni-dortmund.de
Phone: +49 231 755 7017, FAX: +49 231 755 3251

***Abstract*— Currently most of the terminals for UMTS-FDD implement a simple Rake receiver. This paper shows how the Rake, including despreading and descrambling, could be replaced by a least squares receiver that can be implemented on a Cordic based hardware architecture.**

The performance in conjunction with the computational requirements of the receiver is widely adjustable. Simulation results show that the performance of this concept can be significantly better than that of the Rake.

I. Introduction

A lot of modern signal processing applications require such a high computational power that only ASICs can fulfill the technical demands. Unfortunately, ASICs are inflexible, costly (development and debugging) and only economical for mass-products. As a consequence, system designers are striving to replace specialized hardware solutions with software based solutions as, e. g., developments in the field of software radio demonstrate.

Due to the fact that even the most commonly used programmable devices, i. e. DSPs, often lack the required processing power, one tries to develop a solution that lays somewhere in between the two extrema *programmable signal processing* and *dedicated hardware*. The efforts in this area are summarized under the term *reconfigurable computing*.

This paper presents such an alternative solution for the Rake receiver ([1][2] or [3]) as used for UMTS-FDD. The Rake is usually implemented as a hardwired SOC component which is not reconfigurable [4]. Furthermore adaptivity to the signal quality can only be realized by switching Rake fingers on or off.

We will present a receiver concept that is based on the least squares solution of a linear system. This concept allows us to adjust the receiver performance (and the power consumption associated with it) in a wide range depending on the channel impulse response. The performance of the receiver is adjusted by the number of used dominant taps of the channel impulse response, just like the fingers of the Rake that are also determined by these taps.

This project is partly funded from the German Ministry of Education and Research (BMBF) in the 'Mobile, Re-configurable Multi-standard Terminal for UMTS and WLAN (MoReTeX)'project, which is a part of the umbrella 'Software Defined Radio Based Architecture Studies for Re-configurable Mobile Communication Systems (RMS)'.

The receiver computes the symbols by an array of Cordic processors as described in [5][6] and is about to be implemented on the architecture proposed in [7]. Furthermore it is possible to use the processor array for other tasks due to its design as a coprocessor element.

The paper is organized as follows. At first the system model used for the algorithm is described in Section II. Then we will present the algoritm, the solution approach and the proposed implementation in section III. In Section IV, we will show some simulation results followed by the conclusions in SectionV.

II. System Model

Consider a QAM DS/CDMA downlink where all u users share the same channel and the first user is the desired one. Thus the received signal also contains the signal of the other users. The system model used is shown in Figure 1.

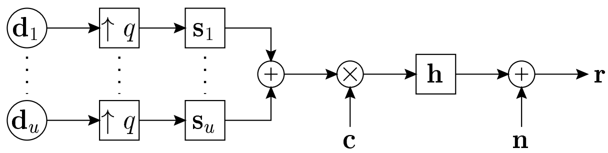

Fig. 1. Block diagram of the system model.

The incoming m complex data symbols of user i, collected in the vector $\mathbf{d}_i$, are first upsampled by the spreading factor q, so that one symbol now consists of q chips. Each upsampled symbol is now convolved with an OVSF code [8] contained in the vector $\mathbf{s}_i$ of length q. Finally, the summed data streams are scrambled with the complex data sequence in vector $\mathbf{c}$ which is repeated for every data frame (38400 chips).

The received chips are now obtained by propagating the signal through a channel, which is characterized by its complex valued channel impulse response vector $\mathbf{h}$ of length h_l, and adding an AWGN component $\mathbf{n}$.

Therefore the received data vector $\mathbf{r}$ is given by

$$\mathbf{r} = \mathbf{n} + \mathbf{HC} \cdot \sum_{i=1}^{u} \mathbf{S}_i \mathbf{d}_i \qquad (1)$$

where $\mathbf{H}$ is a convolutional matrix describing the time variant complex channel, $\mathbf{C}$ is a complex valued diagonal matrix containing the scrambling code **c** on its main diagonal and $\mathbf{S}_i$ is a block Toeplitz spreading matrix. In this matrix each block is one column wide (see also Figure 2) and contains the OVSF code for the *i*-th data stream.

For the proposed algorithm it is assumed that the received signal **r** has already passed the chip matched filter and has been sampled at chip rate. We also assume that the channel impulse response **h** (or the strongest taps of it) is known, as the channel estimation is not a part of this paper.

III. Algorithm

Because we are just interested in the symbols of the first user, we will treat the other users as an additional noise component, so that **r** is simplified to

$$\mathbf{r} = \mathbf{n}' + \mathbf{HCS}_1\mathbf{d}_1 \tag{2}$$

with

$$\mathbf{n}' = \mathbf{n} + \mathbf{HC} \cdot \sum_{i=2}^{u} \mathbf{S}_i \mathbf{d}_i \,.$$

Of course it would be better to utilize the contribution of the other users in order to improve the performance of the receiver (i.e. multi user detection or interference cancellation). But for the ease of implementation, and to keep the computational complexity low, we will ignore the special properies of the noise component.

When we have a close look at the structure of the matrices involved in the computation, we will notice that there are several characteristic properties that can be exploited to simplify the calculation of the data symbols. Figure 2 clarifies the structure of the **H** and **S** matrices.

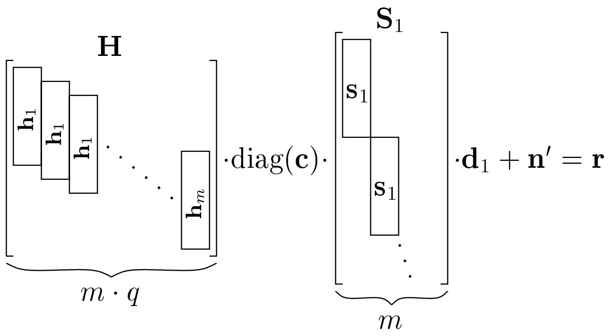

Fig. 2. Initial matrix structures.

The column vectors $\mathbf{h}_i$ of **H** are assumed to be constant in one symbol interval. The width of **H** equals $m \cdot q$, while the number of colums of $\mathbf{S}_1$ equals the number of symbols m. If we multiply these matrices we get the system matrix **K** of width m

$$\mathbf{K} = \mathbf{HCS}_1 \,. \tag{3}$$

This matrix will be used to calculate the desired data symbols. The structure of **K** is shown in Figure 3. It is obvious that **K** has got a very sparse structure which can be exploited to reduce the computational efford to solve the linear system.

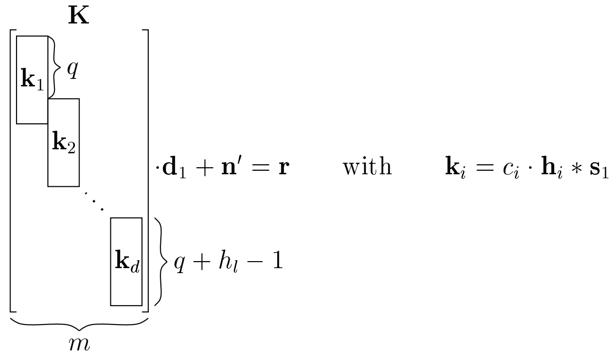

Fig. 3. Resulting structure of the system matrix **K**. The ∗-operator indicates a convolution.

A. Calculating the data symbols

By now the detection of the data symbols has been simplified to the form $\mathbf{K}\hat{\mathbf{d}}_1 = \mathbf{r}$. This overdetermined linear system could now easily be solved in the least squares sense by a QR-decomposition which can be implemented efficiently on a systolic processor array [5] using Cordic processor elements (PE) to perform Givens-transformations.

Due to the special structure of **K**, this array does not even need to compute the whole **R** matrix. As shown in Figure 4, the computation of **R** leads to a banded matrix of band width w which is equal to the degree of overlapping of the column vectors $\mathbf{k}_i$ in **K**.

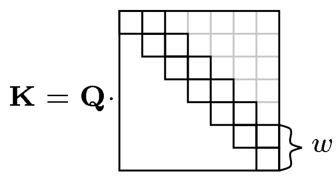

Fig. 4. Resulting structure of the **R** matrix.

Hence the processor array that is required to compute the result only needs to have active processor elements in this band. An example of such an array that can compute 4 symbols is shown in Figure 5. A detailed description of the array and its function can be found in [5]. The implementation was done on the hardware accelerator described in [7].

Unfortunately this systolic array is not capable of entirely exploiting the special structure of **K**. An example of $q = 16$, $h_l = 8$ and $m = 4$, which results in a length $k_l = (q + h_l - 1) + (m-1)q = 71$ of **K** would take

$$a_{\text{Systolic}} = k_l\left(2m + 3mw + \frac{3}{2}w(1-w)\right) = 2059 \tag{4}$$

Cordic PE activations.

Obviously a lot of computation time can be saved if the sparsity of **K** can be exploited. As the structure of **K** is known exactly, it is possible to apply the required Givens-Transformations only to the nonzero elements of $\mathbf{k}_i$ in **K**. This approach is shown in Figure 6. In each step one vector $\mathbf{k}_i$ is

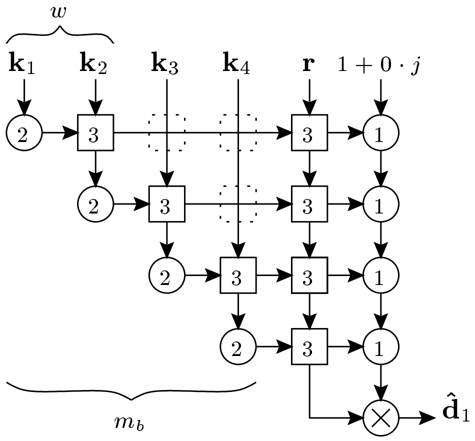

Fig. 5. Complex valued systolic array to calculate the symbols. The numbers indicate the number of real valued Cordic PEs for one complex operation. The phase correction Cordics on the right side are needed for the backsubstitution only.

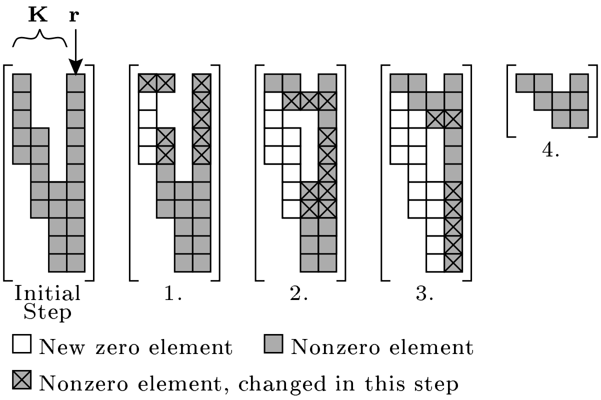

Fig. 6. Computations for the direct solution of the linear system.

annihilated. For each annihilation of one element in $\mathbf{k}_i$ it is necessary to recompute two rows of the matrix composed of $\mathbf{K}$ and $\mathbf{r}$. The last step shows the matrix that is used to perform the backsubstitution. Thus for the first symbol, corresponding to the first column of $\mathbf{K}$,

$$\begin{aligned} a_{\text{First}} &= (q+h_l-2)\cdot 2+(q-1)\cdot 3\cdot 2 \\ &\quad +(h_l-1)\cdot 3\cdot 3) \end{aligned} \tag{5}$$

Cordic PE activations are needed. For the middle and last symbol(s), the number of activations are derived similarly (See also Figure 3):

$$\begin{aligned} a_{\text{Middle}} &= (q+h_l-1)\cdot 2+q\cdot 3\cdot 2 \\ &\quad +(h_l-1)\cdot 3\cdot 3 \end{aligned} \tag{6}$$

$$a_{\text{Last}} = (q+h_l-1)\cdot 2+(q+h_l-1)\cdot 3\cdot 2 \tag{7}$$

Hence if the same algorithm is used, but this time exploiting the structure of $\mathbf{K}$, just

$$a_{\text{Direct}} = a_{\text{First}} + (m-2)\cdot a_{\text{Middle}} + a_{\text{Last}} = 791 \tag{8}$$

activations are neccessary for our example to calculate $\mathbf{R}$. The backsubstitution, when performed on the Cordic processor array, will need

$$a_{\text{Back}} = m(3m+3mw+\frac{3}{2}w(1-w)) = 132 \tag{9}$$

activations in both cases. A complexity estimation to compare the Rake and our proposed algorithm is quite hard to perform as it is strongly depended on the respective implementation. Therefore the following is only a rough estimate of the computational complexity. For our example we consider a filter Rake with 4 fingers. According to [9] this receiver will use

$$op_{\text{Rake}} = 8\cdot 16(4+1) = 640\,\frac{\text{FLOPs}}{\text{Symbol}}. \tag{10}$$

If one Cordic PE operation is equated to x FLOPs, the LS solver will consume

$$op_{\text{LS}} = \frac{a_{\text{Direct}}+a_{\text{Back}}}{m}\cdot x\,\frac{\text{FLOPs}}{\text{Symbol}}. \tag{11}$$

As the Cordic PE's complexity should be about the same as two array multipliers, a rough estimate for x could be ≈ 3, including some computational overhead. This means the LS receiver would use

$$op_{\text{Example}} = \frac{791+132}{4}\cdot 3 \approx 693\,\frac{\text{FLOPs}}{\text{Symbol}} \tag{12}$$

for our example, using the complete channel impulse response. This is a remarkable result considering the reconfigurability and the performance of this approach as shown in Section IV. Currently this optimized algorithm is implemented using the accelerator of [7].

As mentioned before the accuracy of the detection can be varied by the number of channel taps taken into account for the calculation of $\mathbf{K}$. However, the impact on the symbol calculation time is only noticable for large values of h_l and small spreading factors q, as every tap of $\mathbf{h}$ considered for $\mathbf{K}$ generates a block of q non-zero data values in the vector $\mathbf{k}_i$.

B. Partitioning the calculation

It is obvious that $\mathbf{K}$ can grow to a very large matrix. A whole data frame of 150 symbols at spreading factor $q = 256$ and a channel of length $h_l = 40$ would require a $\mathbf{K}$ matrix of size 38439×150.

To overcome this problem the system $\mathbf{K}\cdot\mathbf{d}_1 = \mathbf{r}$ is subdivided into overlapping subsystems of manageable size as shown in Figure 7 and described in [10].

The linear system is subdivided into blocks of size m_b. Using this method it is possible to solve the complete system without the need to store the whole matrix which would also involve large latency and memory needs. The overlapping of the blocks is neccessary, as the independent calculation of the subproblems leads to higher symbol errors at the block edges as shown in Figure 8.

For good results the overlapping factor should be chosen at least as high as the overlapping factor w of the columns of $\mathbf{K}$. As the columns of $\mathbf{K}$ can be calculated independently, it

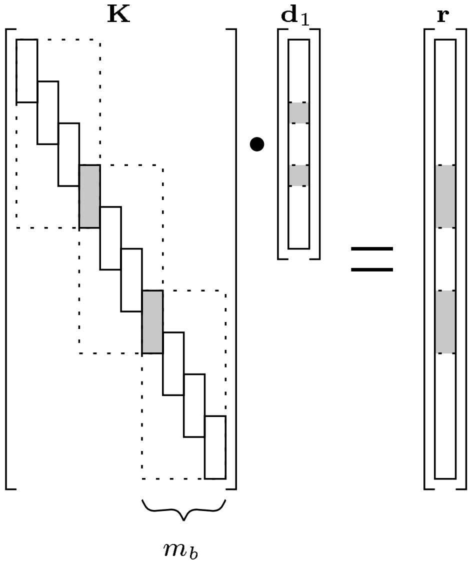

Fig. 7. Overlapping for easier calculation.

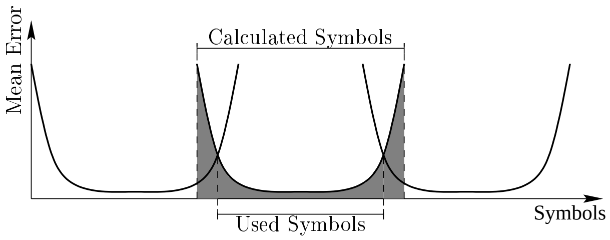

Fig. 8. Expected error function.

is also possible to update the channel $\mathbf{h}$ during the calculation of the blocks.

IV. Simulations

The BERs of the proposed receiver are presented in Figure 9. They were evaluated using an 8 tap channnel with 4 strong taps. The system consists of 8 users, each transmitting with the same power.

A spreading factor of 16 was chosen. The scrambling and coding sequences have been generated according to the UMTS standards [8], using the scrambling sequence number 0 and channelization codes 0-7. The taps of the channel are assumed to be known perfectly. A size of $m_b = 8$ was chosen for the blocks. This leads to an overlapping factor w of 2 which was used during the calculations.

The results show that the performance is better than the 4 finger Rake receiver in any case. “LS - strongest taps” uses the same 4 taps of the channel impulse response in the least squares solution as the 4 finger Rake, and “LS - all taps” uses all of the taps for the solution. If the complete channel impulse response is known, a significant better performance can be achieved while the computational complexity for the calculation of the LS solution does not increase very much.

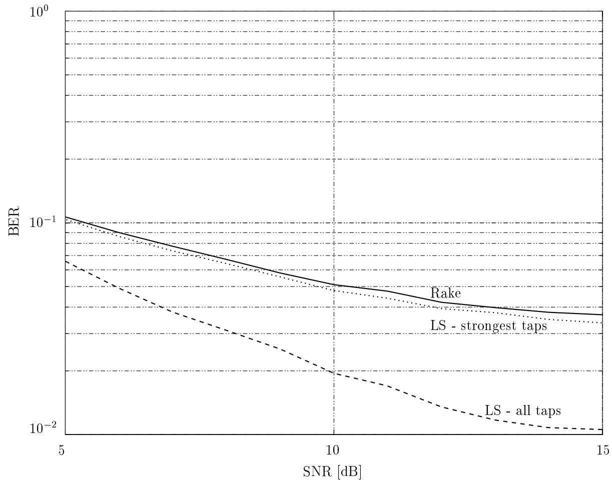

Fig. 9. BER as a function of the SNR.

Figure 10 shows the absolute symbol deviation of the first 50 decoded symbols if no overlapping is used (no AWGN, no other users, $m_b = 8$).

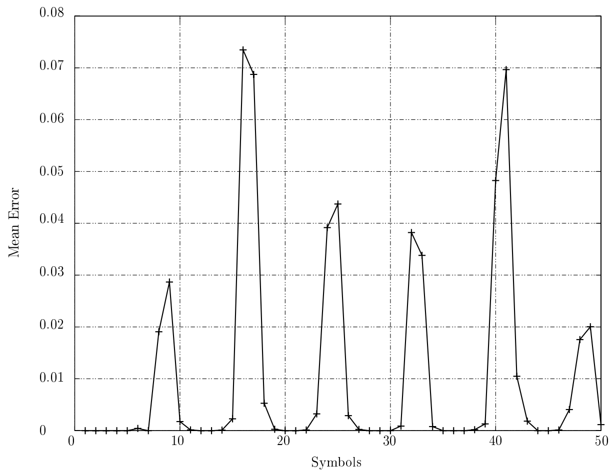

Fig. 10. Symbol errors using no overlap, block size 8.

As expected the symbol deviation is comparatively high at the block edges, as the symbols , corresponding to the columns of $\mathbf{K}$, are overlapping by 2. Thus if the symbols are calculated overlapping the blocks by one symbol, the result can be greatly improved as shown in Figure 11.

V. Conclusions

We have presented a way to replace the ordinary Rake receiver, including despreading and descrambling, by a performance adaptive and fully reconfigurable algorithm. The alternative receiver concept does not need more information than the Rake receiver for equal performance. If more information of the channel is used to compute the symbols, the performance can be increased significantly, while the computational complexity stays almost the same.

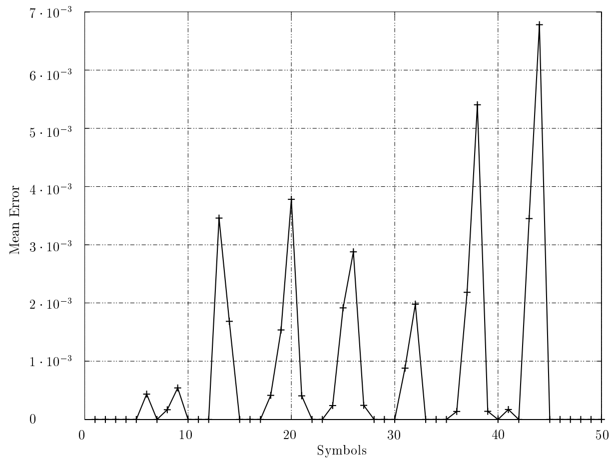

Fig. 11. Symbol errors using an overlap of 1, block size 8.

It was shown how the algorithm is derived from the system model, and how it could be implemented on a Cordic based architecture. The structure of the involved matrices can be exploited to save computation time. Further work will include the optimized implementation of the algorithm on a hardware accelerator. Due to the sparse structure of the system matrix, iterative methods for solving linear systems are also considered for the implementation.

REFERENCES

[1] J. G. Proakis, *Digital Communications*, 2nd ed. New york: McGraw-Hill, 1989.

[2] K. K. Parhi and T. Nishitani, *Digital Signal Processing for Multimedia Systems*. New York: Marcel-Dekker, 1999.

[3] K. Kammeyer, *Nachrichtenübertragung*, 2nd ed. B.G. Teubner, 1996.

[4] H. Holma and A. Toskala, *WCDMA for UMTS*. New York: Wiley & Sons, 2001.

[5] M. Otte, J. Götze, and M. Bücker, "Matrix Based Signal Processing on a Reconfigurable Hardware Accelerator," in *10th Digital Signal Processing Workshop*, Pine Moutain, Georgia, USA, October 2002.

[6] S. Haykin, *Adaptive Filter Theory*. Prentice Hall (3rd edition), 1995.

[7] B. Oelkrug, M. Bücker, D. Uffmann, A. Dröge, J. Brakensiek, and M. Darianian, "Programmable hardware accelerator for universal telecommunication applications," in *2nd Workshop on Software Radios*, Karlsruhe, Germany, 2002.

[8] "TS25.213 V5.0.0 - Spreading and Modulation (FDD)," 3rd Generation Partnership Project (3GPP)," Technical Specification Group Radio Access Network, 2002.

[9] R. Machauer, "Multicode-Detektion im UMTS," Ph.D. dissertation, University of Karlsruhe, INT, 2002.

[10] M. Vollmer, M. Haardt, and J. Götze, "Comparative Study of Joint-Detection Techniques for TD-CDMA Based Mobile Radio Systems," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1461–1475, August 2001.