Adv. Radio Sci., 4, 179–184, 2006
www.adv-radio-sci.net/4/179/2006/

Advances in Radio Science

Cordic based algorithms for software defined radio (SDR) baseband processing

B. Heyne and J. Götze

University of Dortmund, Information Processing Lab, Otto-Hahn-Str. 4, 44227 Dortmund, Germany

Abstract. This paper presents two Cordic based algorithms which may be used for digital baseband processing in OFDM and/or CDMA based communication systems. The first one is a linear least squares based multiuser detector for CDMA incorporating descrambling and despreading. The second algorithm is a pure Cordic based FFT implementation. Both algorithms can be implemented using solely Cordic based architectures (e.g. coprocessors or ASIPs). The algorithms exactly fit the needs of a multistandard terminal as they both are freely parameterizable. This regards to the accuracy of the results as well as to the parameters of the performed function (e.g. size of the FFT).

1 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 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 with the term reconfigurable computing.

We are using this approach to create a common software defined baseband implementation for UTRA/UMTS-FDD and WLAN as shown in Fig. 1. The most computational intensive tasks are performed on a dedicated hardware accelerator called RACE using Cordic processing elements. The implementation of a Cordic based the FFT and a Cordic based linear equalizer/multiuser detector for CDMA, including descrambling and despreading, is presented in this paper.

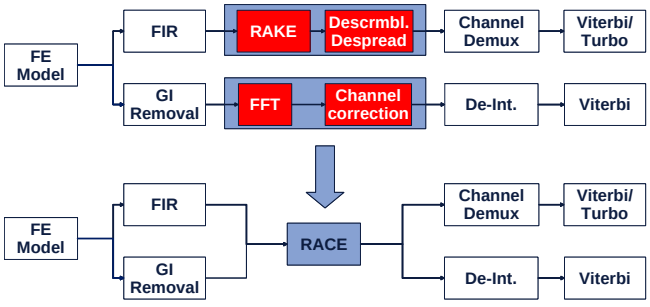

Fig. 1. Software defined baseband processing for WLAN and UMTS using the RACE accelerator.

As the equalizer and the FFT can now be build upon solely Cordics, we have derived a software defined architecture for a mobile multi-standard terminal. The main processing blocks of the WLAN and UMTS baseband can be replaced by this programmable architecture.

The paper is organized as follows. At first the multiuser detector including the system model, algorithm and simulations is described in Sect. 2. Secondly the Cordic based FFT algorithm and simulation results are presented in Sect. 3, followed by an overview of the RACE coprocessor in Sect. 4. Finally the conclusions are given in Sect. 5.

2 CDMA multiuser detection

2.1 System model

Consider a CDMA downlink where all u users share the same channel. Thus the received signal also contains the signal of

Correspondence to: B. Heyne
(benjamin.heyne@uni-dortmund.de)

Published by Copernicus GmbH on behalf of the URSI Landesausschuss in der Bundesrepublik Deutschland e.V.

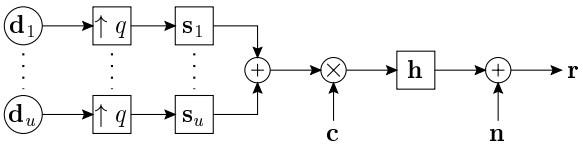

Fig. 2. Block diagram of the system model.

the other users. The system model used is shown in Fig. 2.

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 (3GPP, 2002) 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 in UTRA/FDD).

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

$$\mathbf{r} = \mathbf{n} + \mathbf{HC} \sum_{i=1}^{u} \mathbf{S}_i \mathbf{d}_i \tag{1}$$

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 $\mathbf{c}$ on its main diagonal and $\mathbf{S}_i$ is a block Toeplitz spreading matrix. In this matrix each block is one column wide and contains the OVSF code for the i-th data stream.

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

2.2 Multiuser detection

The derivation of the single user detector/equalizer from this model is described in Heyne et al. (2003). For the multiuser detection we are interested in the first j users and will treat the other users as an additional noise component, so that $\mathbf{r}$ is changed to

$$\mathbf{r} = \mathbf{n}' + \mathbf{HC} \sum_{i=1}^{j} \mathbf{S}_i \mathbf{d}_i \tag{2}$$

with

$$\mathbf{n}' = \mathbf{n} + \mathbf{HC} \sum_{i=j+1}^{u} \mathbf{S}_i \mathbf{d}_i \, .$$

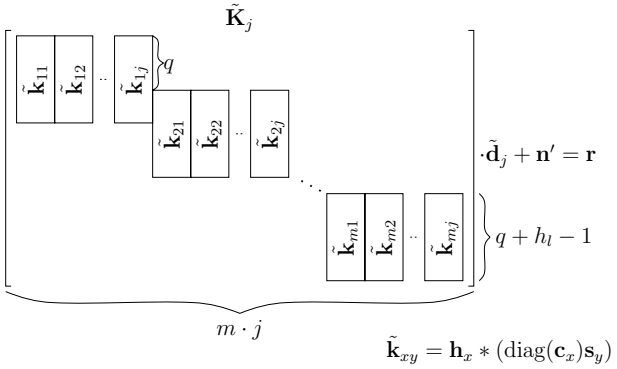

Fig. 3. Resulting structure of the system matrix $\tilde{\mathbf{K}}_j$. The $*$-operator indicates a convolution.

To describe $\mathbf{r}$ as a simple matrix-vector multiplication, we create a new symbol vector $\tilde{\mathbf{d}}_j$ which contains the time interleaved data symbols $\mathbf{d}_i$ of the first j users

$$\tilde{\mathbf{d}}_j = [d_{11} \;\; d_{12} \;\; \cdots \;\; d_{1j} \;\; d_{21} \;\; d_{22} \;\; \cdots \;\; d_{2j} \;\; \cdots \;\; d_{mj}]^T \, ,$$

and a new spreading matrix $\tilde{\mathbf{S}}_j$ containing the j corresponding spreading codes.

Now the vector $\mathbf{r}$ can be rewritten as:

$$\mathbf{r} = \mathbf{n}' + \mathbf{HC}\tilde{\mathbf{S}}_j \tilde{\mathbf{d}}_j \tag{3}$$

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. In our approach the H, C and $\tilde{S}_j$ matrices are multiplied to get the system matrix $\tilde{K}_j$ of width $m \cdot j$:

$$\tilde{\mathbf{K}}_j = \mathbf{HC}\tilde{\mathbf{S}}_j \, . \tag{4}$$

This matrix will be used to calculate the desired data symbols. The structure of $\tilde{\mathbf{K}}_j$ is shown in Fig. 3.

The nonzero column vectors are calculated by convolving the channel impulse response with the scrambled spreading code. $\mathbf{c}_x$ denotes the x-th code block of length q in $\mathbf{c}$. As the scrambled spreading code just has got $(\pm 1 \pm i)$ entries the system matrix can be build without using multiplications.

It is obvious that $\tilde{\mathbf{K}}_j$ has got a very sparse structure which can be exploited as described in Sect. 2.3, to reduce the computational effort to solve the linear system.

2.3 Implementation

The detection of the estimated data symbols $\tilde{\mathbf{d}}'_j$ can now be performed by solving the overdetermined linear system

$$\tilde{\mathbf{K}}_j \tilde{\mathbf{d}}'_j = \mathbf{r} \, . \tag{5}$$

The equation is solved in the least squares sense by a QR-decomposition which can be implemented efficiently on a

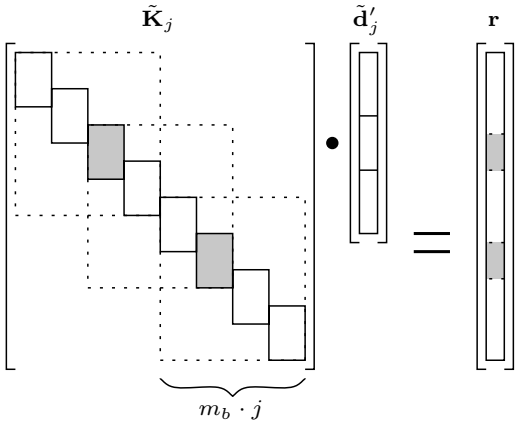

Fig. 4. Overlapping for easier calculation.

processor array (Otte et al., 2002) using Cordic processor elements (PE) performing Givens transformations. But, as the structure of $\tilde{\mathbf{K}}_j$ is known, a direct approach is used for the calculation of the QR decomposition. A new system matrix is build from $\tilde{\mathbf{K}}_j$ and $\mathbf{r}$. Then the required Givens transformations are applied only to the nonzero elements of $\tilde{\mathbf{k}}_{xy}$ in $\tilde{\mathbf{K}}_j$.

In each step one vector $\tilde{\mathbf{k}}_{xy}$ is annihilated. For each annihilation of one element in $\mathbf{k}_{xy}$ it is necessary to recompute two rows of the matrix composed of $\tilde{\mathbf{K}}_j$ and $\mathbf{r}$. The last step shows the matrix $\mathbf{R}$ and the vector $\mathbf{r}'$ which are used to perform the back-substitution.

2.4 Subdividing the calculation

It is obvious that $\tilde{\mathbf{K}}_j$ can grow to a very large matrix. A whole data frame of $m \cdot j = 2\,400 \cdot 16 = 38\,400$ symbols at spreading factor $q = 16$ and a channel of length $h_l = 10$ would require a $\tilde{\mathbf{K}}_j$ matrix of size $38\,409 \times 38\,400$.

To overcome this problem the system $\tilde{\mathbf{K}}_j \tilde{\mathbf{d}}'_j = \mathbf{r}$ is subdivided into overlapping subsystems of manageable size as shown in Fig. 4 and described in Vollmer et al. (2001).

The linear system is subdivided into blocks of size $m_b \cdot j$. 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. Overlapping of the blocks is necessary, as the independent calculation of the subproblems leads to higher symbol errors at the block edges. These errors are nearly eliminated by using this overlapping method.

This method involves a certain amount of computational overhead as the grey blocks have to be calculated twice. The overhead can be reduced by choosing larger block sizes m_b. For good results the overlapping factor should be chosen at least as high as the block overlapping factor p of $\tilde{\mathbf{K}}_j$.

Table 1. Computational complexity comparison.

	Rake	**Cordic based LS**
Operations / Symbol	4352	6023

2.5 Computational complexity

If the overlap method is used for a data frame with $q = 16$, $m_b = 8$, $p = 2$, $j = 16$ and $m = 2\,400$ about 400 blocks have to be calculated for one frame. It is also assumed that the channel length h_l is 10.

In this case the decomposition/back-substitution for each block needs $approx$ 179 000 (real valued) Cordic operations and $\approx 20\,500$ complex additions for the creation of the system matrix. Therefore the detection of a whole data-frame of $m \cdot j = 38400$ symbols uses

$$\approx 1865 \quad \frac{\text{Cordic Operations}}{\text{Symbol}}$$

and

$$\approx 214 \quad \frac{\text{Complex Additions}}{\text{Symbol}}.$$

Note that there is no further descrambling/despreading necessary. Furthermore the Cordic based QR decomposition can make nearly 100% use of two parallel Cordics.

The complexity comparison of our proposed algorithm to other implementations is based on the numbers given in Nahler et al. (2002) for Rake and PIC based receivers. An equivalent of three array multipliers for one Cordic operation is used to include some overhead. Therefore a Cordic operation would be equal to three operations, and a complex addition equals two operations. Then complexity of our approach for this example is on the same order than the conventional Rake receiver as shown in Table 1.

Of course this is only a rough estimate of the computational complexity. But it shows that it is about the same as for the conventional Rake receiver, while the performance is significantly increased as shown in Sect. 2.6. For a detailed description of the algorithm and the complexity analysis see Heyne and Goetze (2005).

2.6 Simulations

Figure 5 shows the frame error rate for a 16-QAM based system with $q = 16$, $j = 16$, $h_l = 10$, $m_b = 8$ and $p = 2$. The channel is assumed to be constant throughout the simulation and contains four, randomly distributed, strong taps. Hence the Rake receiver is using four fingers.

The Figure shows that the Rake has got no chance to detect the symbols in this case, and that the LS approach has got a large performance gain for rising SNR values.

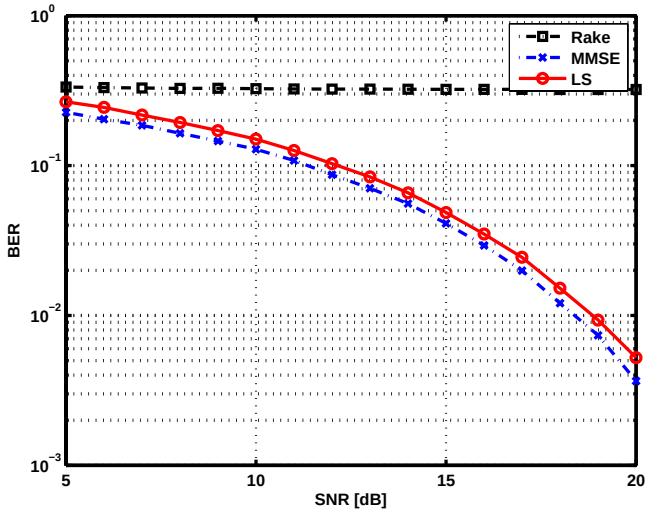

Fig. 5. Error rate for 16-QAM modulation.

3 FFT

3.1 MAC based FFT

A DFT with N input values **s** can be described as a matrix-vector multiplication. By exploiting the properties of the DFT matrix, operations can be greatly reduced and the well known Fast Fourier Transformation (FFT) (Oppenheim and Schafer, 1999) is derived.

The listing below shows a recursive implementation of a MAC based FFT in Matlab style.

```
1  function y=fft(x,n)
2    if n=1
3      y=x
4    else
5      m=n/2
6      w=exp(-2*pi*i/n)
7      om=diag(1,w,...,w^(m-1))
8      zt=fft(x(0:2:n-1),m)
9      zb=om*fft(x(1:2:n-1),m)
10     I_m=eye(m)
11     y=[I_m I_m;I_m -I_m]*[zt;zb]
12   end
13 end
```

3.2 Algorithm

To derive a Cordic based FFT we will stop the recursion at $n=2$. In this case line 11 will look like:

$$y = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} zt \\ zb \end{bmatrix} \tag{6}$$

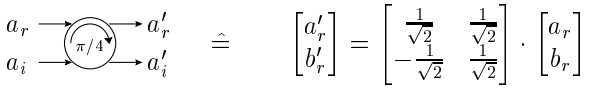

Fig. 6. Symbol and function of one real valued Cordic.

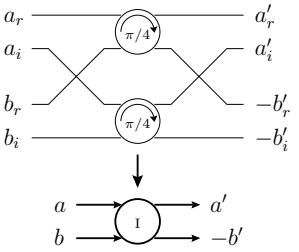

Fig. 7. Inner structure of complex Cordic type I.

The first matrix can then be decomposed to:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} \sqrt{2} & \\ & -\sqrt{2} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{7}$$

This equals a Cordic rotating the input values by $\pi/4$, followed by a scaling of $\sqrt{2}/-\sqrt{2}$.

As the Cordic elements are real valued but the input values are complex valued, the complex Cordic operation has to be separated into real valued operations. Due to the special structure of the rotation matrix, this is quite easy to perform. If we assume two complex numbers a, $b \in \mathbb{C}$, the result of the complex rotation will be (with $t = 1/\sqrt{2}$):

$$\underbrace{\begin{bmatrix} t & t \\ -t & t \end{bmatrix}}_{T} \cdot \begin{bmatrix} a_r + \mathrm{j}a_i \\ b_r + \mathrm{j}b_i \end{bmatrix} = \begin{bmatrix} t(a_r + b_r) + \mathrm{j}t(a_i + b_i) \\ t(b_r - a_r) + \mathrm{j}t(b_i - a_i) \end{bmatrix} \tag{8}$$

$$= T \cdot \begin{bmatrix} a_r \\ b_r \end{bmatrix} + \mathrm{j}T \cdot \begin{bmatrix} a_i \\ b_i \end{bmatrix} \tag{9}$$

Thus the operation can be applied to the real and the imaginary part of the input values independently, and the complex "butterfly" can be performed by using two of the real valued Cordics shown in Fig. 6. The scaling of the results can be performed at the end of the flow graph. The symbol for the resulting complex Cordic, called type I, rotating two complex valued numbers by $\pi/4$ is shown in Fig. 7.

As shown in Eq. 7, the second scaling factor has got a negative sign. Therefore all "lower" results of the complex Cordic operation have got a reversed sign. Fortunately, because of the structure of the FFT stages, this compensation does not result in additional computational overhead. In each stage of the FFT the sign reversed results are just combined with other sign reversed results. Therefore a $-3\pi/4$ rotation is applied to the results in these cases to project the result from the third quadrant back into the first one. This equals a multiplication of the complex input value by $-T$. Therefore

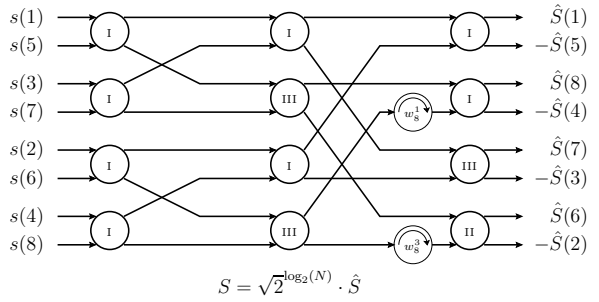

Fig. 8. Optimized Cordic based FFT.

this operation can be decomposed, too. The complex Cordic performing this operation is called type II.

The structure is the same compared to the type I Cordic, except that the real valued Cordics now perform a rotation by $-3\pi/4$.

Further optimizations are possible for $w_y^x = -\mathrm{j}$. The result of this operation can also be obtained by swapping the real and imaginary part of the input value, and then inverting the sign of the new imaginary part. A closer look at the algorithm reveals, that this operation is always performed on sign reversed results of the previous stage. This also implies that the result of the multiplication with $w = -\mathrm{j}$ is always provided to complex Cordics of type II.

Therefore the input value of the real valued Cordic is $-a$. Thus a $w = -\mathrm{j}$ multiplication followed by a type II complex Cordic can be replaced by another complex Cordic, called type III.

The final optimized version of the FFT is shown in Fig. 8. Note that there are no more twiddle factor multiplications after the first FFT stage, which saves one stage in a pipelined implementation.

The twiddle factors shown are the same as used for the standard FFT. As they are located on the unit circle, the multiplication with ω_y^x can be replaced by a Cordic rotation directly.

It is obvious that the FFT like butterfly structure is kept. The $\sqrt{2}^{\log_2(N)}$ scaling of the result can be performed after the computation of the three stages.

3.3 Complexity

If N is the size of the FFT, the number of Cordic operations is:

$$\mathrm{OP}_{\mathrm{Cordic}} = \frac{3N}{2}(\log_2(N) - 1) + 2 \tag{10}$$

The same architecture used to implement the Cordic array also provides a MAC processing element (PE) (Lange et al., 2002). This PE needs

$$\mathrm{OP}_{\mathrm{MAC}} = (N + 2)\log_2(N) \tag{11}$$

activations for an FFT of size N.

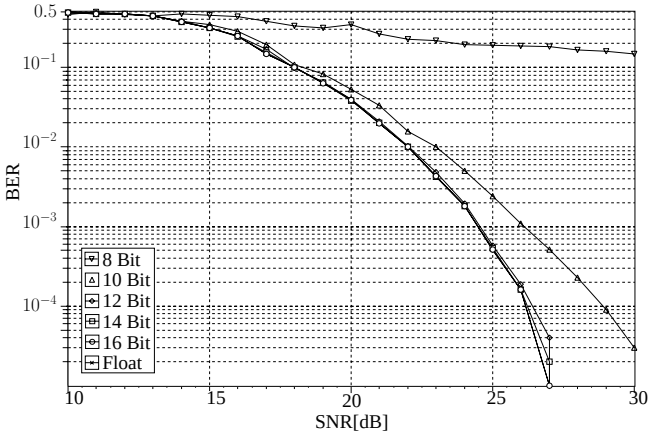

Fig. 9. BER for 54 MBit.

For small FFTs the operation count $\mathrm{OP}_{\mathrm{Cordic}}$ is even lower than for $\mathrm{OP}_{\mathrm{MAC}}$. For the 64-FFT used in a WLAN receiver $\mathrm{OP}_{\mathrm{Cordic}}$ is 482 and $\mathrm{OP}_{\mathrm{MAC}}$ is 396. As the hardware accelerator currently provides two parallel PEs these numbers can be halved to get the number of accelerator activations (198 for the MAC, 241 for the Cordic).

So the Cordic based FFT is slower than the MAC based implementation, but on the other hand one Cordic based reconfigurable hardware architecture can now be used to implement the FFT for WLAN and the Rake substitute for UMTS. Also the OFDM channel correction can be implemented very efficiently on a Cordic, as it supports division. Hence, in the case of a FFT followed by an OFDM channel correction, the computational overhead is considerably small. A more detailed description of the algorithm and it's properties can be found in Heyne and Goetze (2004).

3.4 Simulation results

Finally, the proposed FFT has been implemented in a WLAN transmitter/receiver simulation to replace the regular FFT. The simulation shown in Fig. 9 has been performed using an AWGN channel and the coding parameters defined in IEEE Std 802.11a-1999 (1999).

The results for the 54 MBit case show that a wordlength ≥ 12 Bit is enough to achieve the same BER performance than the floating point FFT implementation. So the Cordic based FFT has to use just 12 Bit arithmetics to replace the standard FFT in a WLAN environment.

4 Hardware coprocessor

Both algorithms have been implemented on the reconfigurable hardware accelerator (RACE) shown in Fig. 10. The RACE can be described as an algorithm specific instruction set processor (ASIP) with a limited instruction set that is optimized for different classes of algorithms.

Table 2. Basic operations of the Cordic PE.

Mode	Operation
Ortogonal Rotation	$x_{out} = x_{in}\cos(\phi_z) + y_{in}\sin(\phi_z))$ $y_{out} = -x_{in}\sin(\phi_z) + y_{in}\cos(\phi_z))$ $z_{out} = z_{in}$
Ortogonal Vector	$x_{out} = \sqrt{x_{in}^2 + y_{in}^2}$ $y_{out} = 0$ $z_{out} = \arctan_2(x_{in}, y_{in})$
Linear Rotation	$x_{out} = x_{in}$ $y_{out} = -x_{in}z_{in} + y_{in}$ $z_{out} = z_{in}$
Linear Vector	$x_{out} = x_{in}$ $y_{out} = 0$ $z_{out} = y_{in}/x_{in}$

In our case the class of algorithms is composed of matrix based algorithms that can be implemented by enhanced Cordic operations. For this purpose the accelerator contains Cordic processing elements which are based on simple shift-add operations, but there are other PE types such as e.g. MACs that can be used as well. The operations that can be performed, and which are used to implement the QR decomposition of the system matrix, are given in Table 2. For example in the "Orthogonal Rotation" mode the Cordic rotates a two dimensional input vector $a = (x, y)$ by an angle ϕ_z.

The accelerator contains several processing elements (PE) in parallel that perform the computations, a Data RAM to store values and a Configuration RAM in conjunction with a finite state-machine (FSM) to control the data flow.

Here, the RACE is embedded in a processor environment where it is connected to the system bus via a bus wrapper, which has direct memory access (DMA) capability and thereby controls the dataflow into and out of the RACE. The processor itself is freed from all data moving tasks and is just informed by an interrupt when the results of an operation are available.

The number and the interfaces of the PEs as well as the amount and structure of the memory inside the Data-RAM can be parameterized. Hence it exactly fits the needs of the multistandard terminal where it is used.

5 Conclusions

We have presented two Cordic based algorithms usable for communication systems based on OFDM and CDMA, namely a FFT and a linear least squares based CDMA multiuser detector/equalizer. The good performance and the low computational complexity of both algorithms make an implementation feasible.

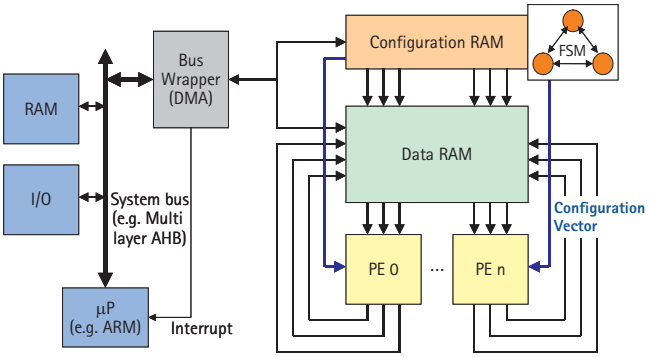

Fig. 10. Reconfigurable accelerator.

Hence, they can be used to implement a reconfigurable (software defined) architecture for multi standard terminal digital basebands, replacing dedicated hardware by using Cordic coprocessors like the RACE accelerator.

References

3GPP: TS25.213 V5.0.0 - Spreading and Modulation (FDD), Technical specification group radio access network, 3rd Generation Partnership Project (3GPP), 2002.

Heyne, B. and Goetze, J.: A Cordic Based Equalizer For Multiuser Detection in WCDMA Systems, in: IEEE Workshop on Signal Processing Systems (SiPS2005), Athens, Greece, 2005.

Heyne, B. and Goetze, J.: A Pure Cordic Based FFT For Reconfigurable Digital Signal Processing, in: 12th European Signal Processing Conference (EUSIPCO2004), Vienna, Austria, 2004.

Heyne, B., Otte, M., and Goetze, J.: A Performance Adjustable and Reconfigurable CDMA Receiver Concept for UMTS-FDD, in: 14th IEEE Intern. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC2003), Beijing, China, 2003.

IEEE Std 802.11a-1999: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, High-speed Physical Layer in the 5 GHz Band, IEEE, 1999.

Lange, H., Franzen, O., Schröder, H., Bücker, M., and Oelkrug, B.: Reconfigurable Multiply-Accumulate-based Processing Element, in: IEEE Workshop on Heterogeneous Reconfigurable Systems on Chip, Hamburg, Germany, 2002.

Nahler, A., Irmer, R., and Fettweis, G.: Reduced And Differential Parallel Interference Cancellation For CDMA Systems, IEEE J. on Select. Areas in Commun., 20, 237–247, 2002.

Oppenheim, A. V. and Schafer, R. W.: Discrete Time Signal Processing, Prentice-Hall, Upper Saddle River, New Jersey, third edn., 1999.

Otte, M., Goetze, J., and Buecker, M.: Matrix Based Signal Processing on a Reconfigurable Hardware Accelerator, in: 10th Digital Signal Processing Workshop, Pine Moutain, Georgia, USA, 2002.

Vollmer, M., Haardt, M., and Götze, J.: Comparative Study of Joint-Detection Techniques for TD-CDMA Based Mobile Radio Systems, IEEE J. on Selected Areas in Communications, 19, 1461–1475, 2001.