Implementation of a Cordic Based FFT on a Reconfigurable Hardware Accelerator

Benjamin Heyne*, Jürgen Götze
University of Dortmund, Information Processing Lab

Martin Bücker
Nokia Research Center, Bochum

***Abstract*— This paper presents the implementation of a solely Cordic based FFT on a reconfigurable architecture.**

It was already shown in a former publication that the Rake receiver can be replaced by a Cordic based linear equalizer [1], which results in a better performance.

Hence, it is possible to replace the main processing blocks of the WLAN and UMTS baseband by this programmable architecture.

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 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*.

Within the framework of the MoReTeX project, this paper presents a solution to replace the MAC based FFT [2][3] computation of the DFT by a solely Cordic based FFT. Previous DFT implementations also used Cordics for parts of the calculations (e.g. [4]) but not for the entire computation.

It was already shown in [1] that the Rake receiver can be replaced by a Cordic based algorithm using a reconfigurable hardware accelerator [5][6], which even results in a better performance. This paper presents a way to implement the 64-FFT used in the WLAN baseband on the same architecture as used for the Rake alternative without a significant performance loss. Therefore, we have derived a reconfigurable architecture for a mobile multistandard terminal and 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 enhanced accelerator architecture is described in Section II. Then the derivation of the Cordic based algorithm, the solution approach and the proposed implementation will be shown in Sections III-IV. In Section V we will show some simulation results in a WLAN environment, followed by the conclusions in Section VI.

II. Accelerator

The FFT is implemented on the reconfigurable hardware accelerator (RACE) shown in Figure 1. 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. 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.

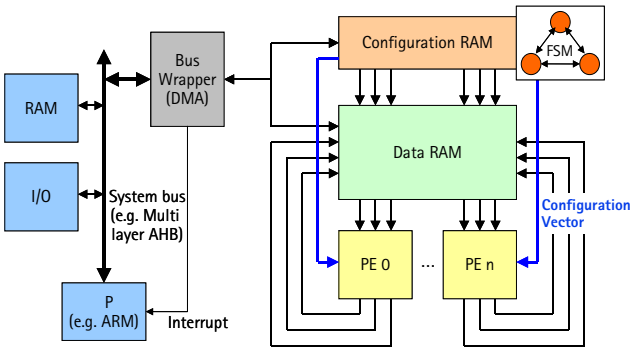

Fig. 1. Reconfigurable Accelerator

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

* This project is partly funded by the German Ministry of Education and Research (BMBF) in the 'Mobile, Reconfigurable 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), No. 01 BU171'.

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.

Compared to previous publications [5] the accelerator was enhanced in its configurability and flexibility. 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.

III. MAC BASED FFT

For the implementation of the Cordic based FFT on the accelerator described above, we will have a look at the MAC based implementation first. A DFT with N input values $\mathbf{s}$ can be described as the matrix-vector multiplication

$$\mathbf{S} = \mathbf{V} \cdot \mathbf{s} \qquad \text{with} \quad \mathbf{V}_{nk} = e^{-\mathrm{j}\frac{2\pi}{N}nk}. \tag{1}$$

By exploiting the properties of $\mathbf{V}$ the operations can be reduced significantly, and the well known Fast Fourier Transformation (FFT) is derived. An eight point FFT leads to the network shown in Figure 2. The twiddle factors ω_y^x are derived as

$$\omega_y^x = \mathrm{e}^{-\mathrm{j}\frac{2\pi x}{y}}. \tag{2}$$

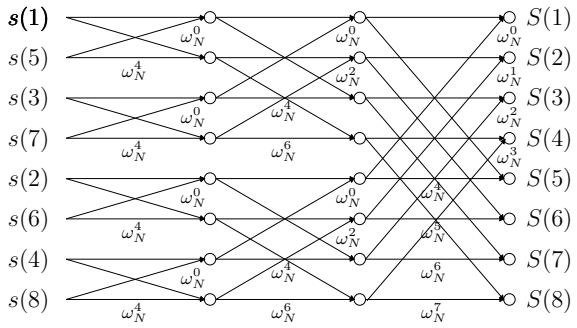

Fig. 2. Regular FFT implementation

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
```

IV. ALGORITHM

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

$$y = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} zt \\ zb \end{bmatrix} \tag{3}$$

The first matrix can then be decomposed to:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} \sqrt{2} & \\ & -\sqrt{2} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{4}$$

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 $t = \frac{1}{\sqrt{2}}$):

$$\begin{aligned} \underbrace{\begin{bmatrix} t & t \\ -t & t \end{bmatrix}}_{T} \cdot \begin{bmatrix} a_r + \mathrm{j}a_i \\ b_r + \mathrm{j}b_i \end{bmatrix} &= \begin{bmatrix} t(a_r + b_r) + \mathrm{j}t(a_i + b_i) \\ t(b_r - a_r) + \mathrm{j}t(b_i - a_i) \end{bmatrix} \\ &= T \cdot \begin{bmatrix} a_r \\ b_r \end{bmatrix} + \mathrm{j}T \cdot \begin{bmatrix} a_i \\ b_i \end{bmatrix} \end{aligned} \tag{5}$$

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 Figure 3. The scaling of the results can be performed at the

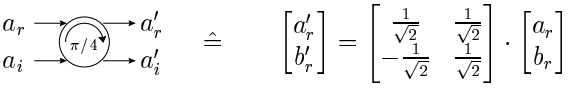

Fig. 3. Symbol and function of one real valued Cordic

end of the calculations as discussed in section IV-B. The symbol for the resulting complex Cordic, called type I, rotating two complex valued numbers by $\pi/4$ is shown in Figure 4.

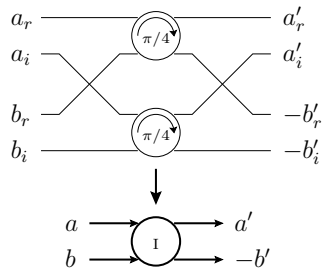

Fig. 4. Inner structure of complex Cordic type I

As shown in equation 4, the second scaling factor is negative. Therefore, all "lower" results of the complex Cordic operation have 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. Hence, a $\frac{-3\pi}{4}$ 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, this operation can be decomposed, too. The complex Cordic performing this operation is called type II (See Figure 5).

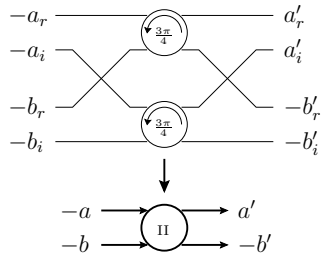

Fig. 5. Inner structure of complex Cordic type II

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

A complete 8-FFT based on Cordic operations is shown in Figure 6.

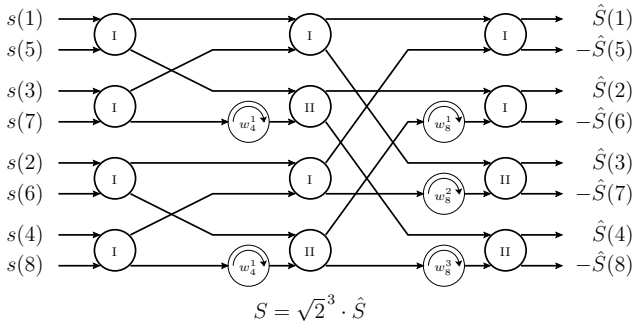

Fig. 6. Cordic based FFT

The twiddle factors shown are the same as used for the standard FFT (Equation 2). As they are located on the unity circle, the multiplication with ω_y^x can directly be replaced by a Cordic rotation.

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

A. *Further optimizations*

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 Cordic can be replaced by the structure shown in Figure 7. This processing element will be called type III.

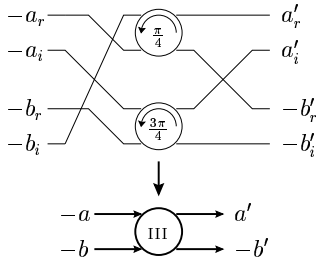

Fig. 7. Inner structure of complex Cordic type III

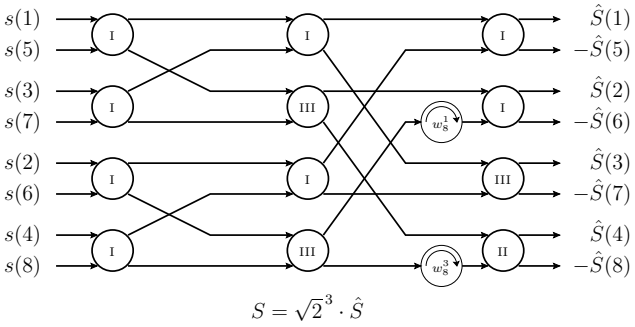

Fig. 8. Optimized Cordic based FFT

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

B. Numeric stability / scaling

The architecture used to implement the FFT is based on fixed point/fixed wordlength arithmetics ranging from -1 to 1. Therefore, overflows and rounding errors have to be taken care of. The FFT should not have a scaling error in order to replace a standard FFT.

Due to the numeric properties the input/output values e.g. for a 64-FFT have to be scaled as shown in Figure 9.

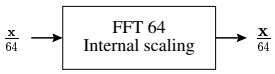

Fig. 9. Scaled FFT

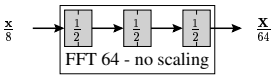

Fig. 10. Unscaled FFT

To avoid overflows, the input values have to be divided by 64. After the computation the final result can be obtained by multiplying the output values by 64.

For the Cordic based FFT this behaviour can be achieved by multiplying the results of every second stage by two. On the other hand, it might be better to make use of the inherent scaling of the Cordics as shown in Figure 10. In this case, the input values have to be divided by 8 and the Cordic stages will automatically scale the final result down to $1/64$. Using this procedure a kind of graceful degradation of the accuracy can be achieved without any additional computational overhead as the simulations presented in section V show.

C. Complexity

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

$$\mathrm{OP}_{\mathrm{Cordic}} = \frac{3N}{2}(\log_2(N) - 1) + 2 \qquad (6)$$

The same architecture used to implement the Cordic array also provides a MAC PE [7]. This PE needs

$$\mathrm{OP}_{\mathrm{MAC}} = (N + 2)\log_2(N) \qquad (7)$$

activations for an FFT of size N. These two functions are compared in Figure 11. It can be seen that for small FFTs the operation count $\mathrm{OP}_{\mathrm{Cordic}}$ is even lower than $\mathrm{OP}_{\mathrm{MAC}}$. For the 64-FFT used in a WLAN receiver $\mathrm{OP}_{\mathrm{Cordic}}$ is 482 and $\mathrm{OP}_{\mathrm{MAC}}$ is 396. In case that the hardware accelerator is implemented with two PEs in parallel these numbers can be halved to get the number of accelerator activations (198 for the MAC, 241 for the Cordic). Accordingly, the number of activations would be $\frac{1}{4}$ for an implementation with four PEs.

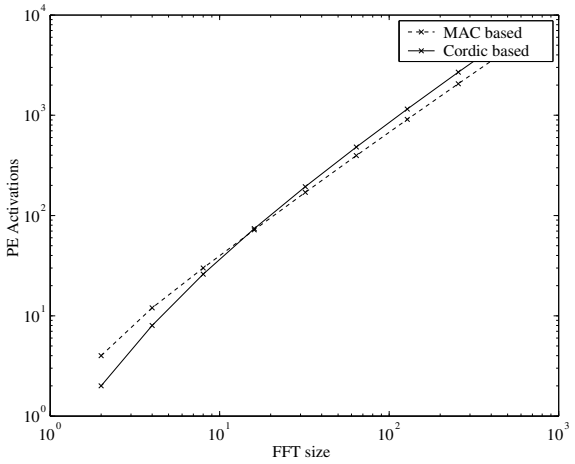

Fig. 11. Number of PE activations for MAC and Cordic based FFTs

So, the Cordic based FFT is slightly 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.

D. Memory requirements

The Data RAM has to provide a sufficient amount of memory to store input data, constants (which are the angles of rotations in this case), intermediate results and the final output data.

Since the input and output data are complex valued and the scheduling can be arranged in a way that the output data overwrites input data that has already been used, $2N$ words have to be reserved for this data.

An optimized implementation on the RACE requires $2N + 2$ words for the intermediate values and $\frac{N}{2} + 1$ constant values for the rotation angles.

V. Simulation results

The resulting mean deviation between the scaled and the unscaled version of the 64-FFT is shown in Figure 12. In order to minimize the influence of the rounding error a 16 bit fixed point implementation was used. The simulation shows that the unscaled version of the FFT generates a smaller and more uniformly distributed deviation compared to the floating point version.

Finally , the unscaled FFT has been implemented in a WLAN transmitter/receiver simulation to replace the regular FFT. The simulation shown in Figure 13 has been performed using an AWGN channel and the coding parameters defined in [8].

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.

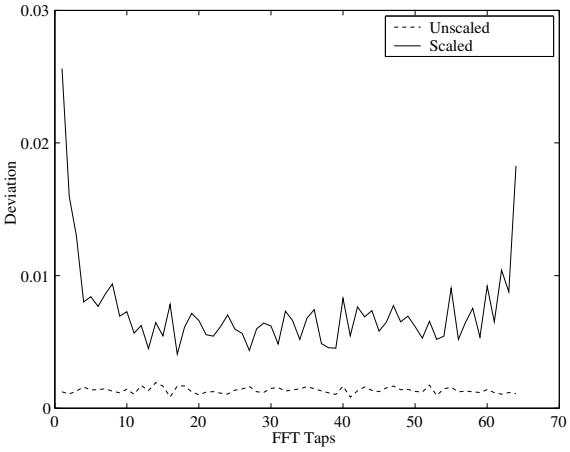

Fig. 12. Scaled vs. unscaled version - Simulation of deviation

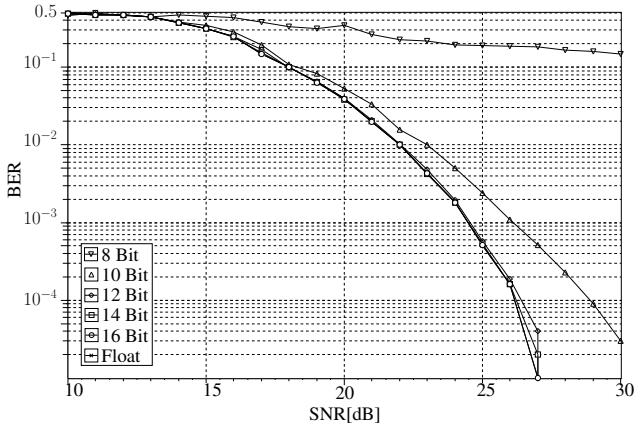

Fig. 13. BER for 54MBit data rate

Furthermore, the number of iterations of the Cordic PE has been reduced in order to find the minimum number of necessary iterations. The results are shown in Figure 14. One can see that about six iterations are enough to achieve a satisfying BER performance. Other simulations have shown that the number of necessary Cordic iterations is independent from the wordlength. Therefore, six iterations would also be the minimum number of iterations for a 16 bit implementation.

VI. Conclusions

We have presented a solely Cordic based FFT algorithm implemented on a reconfigurable hardware accelerator to replace a MAC based FFT in a multi-standard terminal.

It was shown how the algorithm is derived from the recursive FFT definition. Optimizations can be applied to save computational resources.

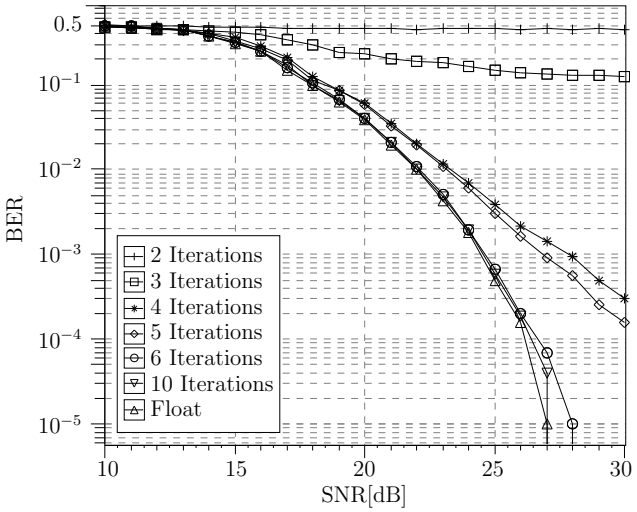

Fig. 14. BER for 54MBit at 12 bit wordlength for different numbers of iterations

Simulations show that the Cordic inherent scaling improves the accuracy of the FFT result in a fixed wordlength/sign environment.

In a WLAN environment simulations show that the wordlength of the optimized Cordic based implementation just needs to be 12 bit and the number of Cordic iterations at this wordlength can be reduced to six.

References

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

[2] A. V. Oppenheim and R. W. Schafer, *Discrete Time Signal Processing*, 3rd ed. Upper Saddle River, New Jersey: Prentice-Hall, 1999.

[3] C. V. Loan, *Computational Frameworks for the Fast Fourier Transform*, 1st ed. Philadelphia, Pennsylvania: SIAM, 1992.

[4] A. M. Despain, "Fourier Transform Computers Using CORDIC Iterations," *IEEE Transactions on Computers*, vol. 10, pp. 993–1001, 1974.

[5] B. Oelkrug, M. Buecker, D. Uffmann, A. Droege, J. Brakensiek, and M. Darianian, "Programmable Hardware Accelerator for Universal Telecommunication Applications," in *2nd Workshop on Software Radios*, Karlsruhe, Germany, 2002.

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

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

[8] "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, Tech. Rep., 1999.