A PURE CORDIC BASED FFT FOR RECONFIGURABLE DIGITAL SIGNAL PROCESSING

Benjamin Heyne, 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

This paper presents a pure Cordic based architecture to calculate the FFT on a reconfigurable hardware accelerator.

The performance of this approach can compete with the ordinary MAC based implementation on this accelerator, although the main advantage is the possibility to implement the FFT on a reconfigurable Cordic-only processor array.

In a former publication it was already shown that the Rake receiver can be replaced by a Cordic based linear equalizer using the same architecture, which even results in a better performance. With the presented pure Cordic based FFT it is now possible to replace the main processing blocks of the WLAN and UMTS baseband by this programmable architecture.

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, 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*.

Within the framework of the MoReTeX project, this paper presents a solution to replace the MAC based FFT [1][2] computation of the DFT by a purely Cordic based FFT. Previous DFT implementations also used Cordics for parts of the calculations e.g. [3] , but not for the entire computation.

In [4] it was already shown 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 MAC based algorithm is described in Section 2. Then we will present the Cordic based algoritm, the solution approach and the proposed implementation in section 3. In Section 4, we will show some simulation results in a WLAN environment, followed by the conclusions in Section 5.

2. MAC BASED FFT

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 greatly reduced, and the well known Fast Fourier Transformation (FFT) is derived. An eight point FFT leads to the network shown in Figure 1. The twiddle factors ω_y^x are derived as

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

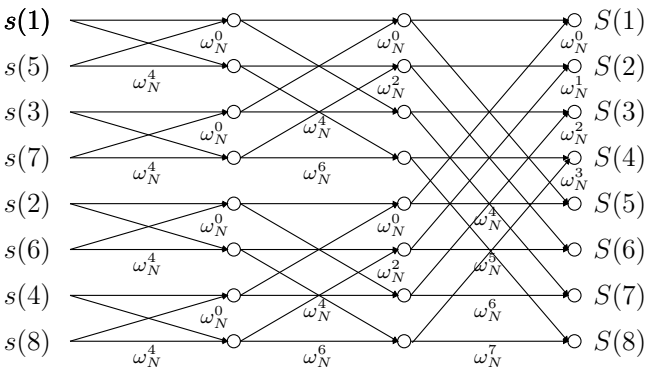

Figure 1: Regular FFT implementation

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

* The presented work was carried out together with the Nokia Research Center, Bochum within the German funded BMBF-project "Software Defined Radio Based Architecture Studies for Reconfigurable Mobile Communication Systems" (RMS), No. 01 BU171'.

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

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

$$= T \cdot \begin{bmatrix} a_r \\ b_r \end{bmatrix} + \mathrm{j}T \cdot \begin{bmatrix} a_i \\ b_i \end{bmatrix} \tag{6}$$

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 2. The scaling of the

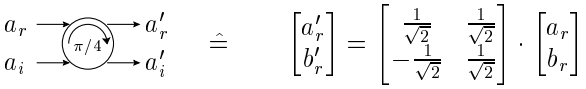

Figure 2: Symbol and function of one real valued Cordic

results can be performed at the end of the calculations as discussed in section 3.2. The symbol for the resulting complex Cordic, called type I, rotating two complex valued numbers by $\pi/4$ is shown in Figure 3.

As shown in equation 4, 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

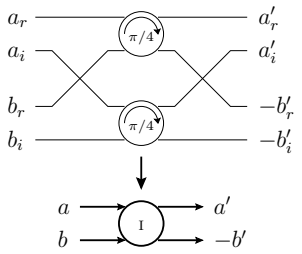

Figure 3: Inner structure of complex Cordic type I

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 $\frac{-3\pi}{4}$ 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 4).

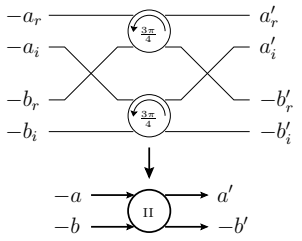

Figure 4: Inner structure of complex Cordic type II

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 5.

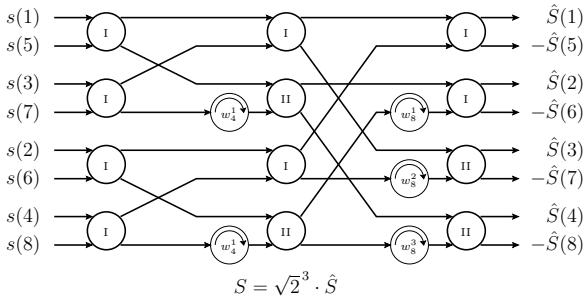

Figure 5: Cordic based FFT

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 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.

3.1 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 6. This processing element will be called type III.

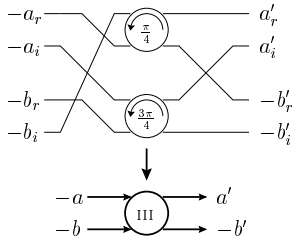

Figure 6: Inner structure of complex Cordic type III

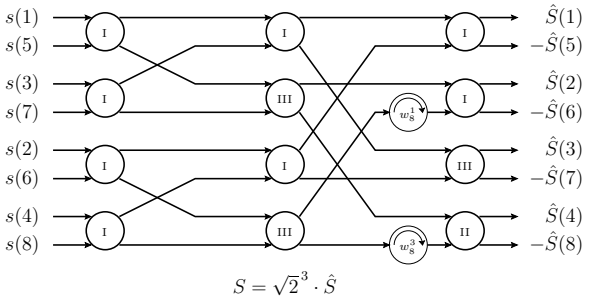

Figure 7: Optimized Cordic based FFT

The final optimized version of the FFT is shown in Figure 7. Note that there are no more twiddle factor multiplications after the first FFT stage, which saves one stage in a pipelined implementation.

3.2 Numeric stability / scaling

The architecture used to implement the FFT is based on fixed point/fixed wordlength arithmetics ranging from -1..1. Therefore care has to be taken for overflows and rounding errors. The FFT should not have a scaling error to be able 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 8.

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 9. In this case

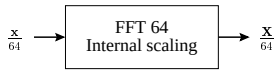

Figure 8: Scaled FFT version

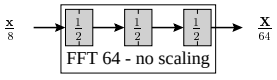

Figure 9: Unscaled FFT version

the input values have to be divided just by 8 and the Cordic stages will automatically scale the final result down to $1/64$. This way a kind of graceful degradation of the accuracy can be achieved without any additional computational overhead as the simulations presented in section 4 show.

3.3 Complexity

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

$$\mathrm{OP}_{\mathrm{Cordic}} = \frac{3N}{2}(\log_2(N) - 1) + 2 \tag{7}$$

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

$$\mathrm{OP}_{\mathrm{MAC}} = (N+2)\log_2(N) \tag{8}$$

activations for an FFT of size N. These two functions are compared in Figure 10. It can be seen that for small FFTs the

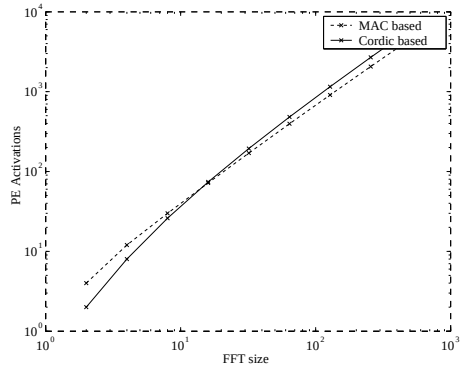

Figure 10: Number of PE activations for MAC and Cordic based FFTs

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 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.

4. SIMULATION RESULTS

The resulting mean deviation between the scaled and the unscaled version of the 64-FFT is shown in Figure 11. To keep the influence of the rounding error small, a 16 Bit fixed point

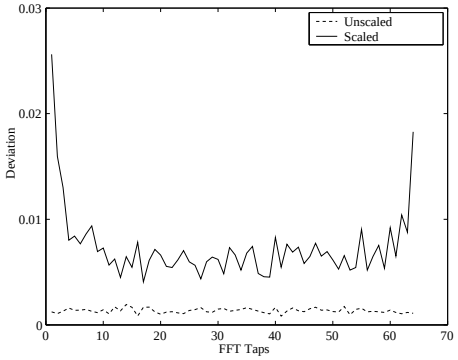

Figure 11: Scaled vs. unscaled version - Simulation of the deviation

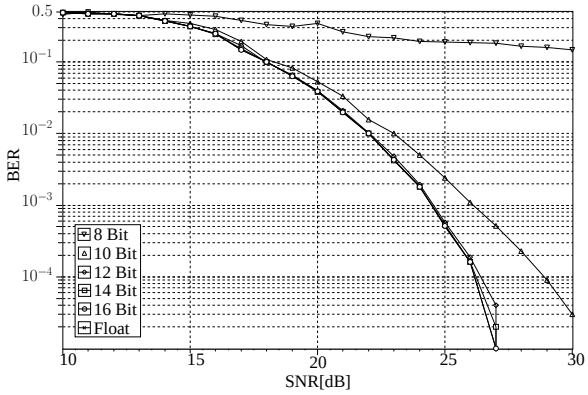

Figure 13: BER for 54MBit

implementation was used. The Figure shows that the unscaled version of the FFT generates a smaller and more uniformly distributed deviation compared to the floating point version.

The simulation in Figure 12 has been done using a software simulation of the hardware accelerator. The mean deviation of the MAC-PE based FFT is compared to the unscaled Cordic-PE based FFT (in relation to the floating point implementation). The deviation of the Cordic based FFT is about five times smaller.

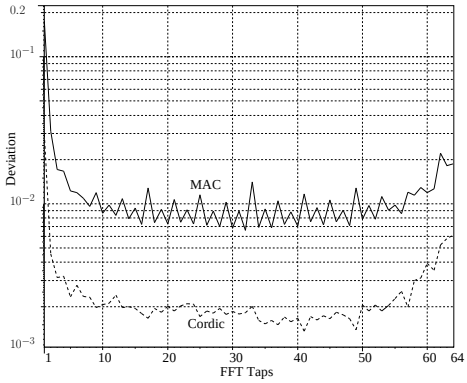

Figure 12: Deviation of the MAC based and the Cordic based FFT

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 $\geq$ 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.

5. CONCLUSIONS

We have presented a solely Cordic based FFT algorithm. This algorithm is best suited for the implementation in reconfigurable Cordic processor fields.

It was shown how the algorithm is derived from the recursive FFT definition. Optimizations can be applied to save computational resources. Simulations show that the Cordic inherent scaling improves the accuracy of the FFT result in a fixed wordlength/sign environment. Additionally the accuracy of the result is much better than the MAC FFT based result.

In a WLAN environment simulations show that the wordlength of the optimized Cordic based implementation just needs to be 12 Bit.

REFERENCES

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

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

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

[4] B. Heyne, M. Otte, and J. Götze, "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.

[5] 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.

[6] 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.

[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.