AREA-EFFICIENT HW-IMPLEMENTATION OF THE FFT FOR OFDM APPLICATIONS

Rainer Storn (ROHDE & SCHWARZ GmbH & Co. KG
Radiocommunications Systems Division
81671 Muenchen, Germany, Email: rainer.storn@rohde-schwarz.com)

ABSTRACT

The Fast Fourier Transform (FFT) plays a central role in OFDM systems, and for today's high-speed waveforms a parallel HW-implementation of it , either in ASICs or FPGAs, is often imperative. OFDM systems frequently employ so-called "Real-Valued FFTs" which exploit that the time-sequence is real-valued and the frequency-sequence is conjugate complex. In addition a parallel/serial conversion has to be performed to accommodate the discrepancy between serial sample streams and the block processing nature of the FFT. Pipeline architectures for the FFT perform the required parallel/serial conversion as a by-product and lend themselves to efficient HW-implementations.
The Cooley Tukey FFT of length $N=2^{\nu}$, ν integer, is one of the most widespread FFT algorithms for OFDM systems and allows for an efficient pipeline implementation. In this paper it will be shown that a different algorithm, the Bruun-FFT of length $N=2^{\nu}$, ν integer, also lends itself to a very efficient pipeline architecture that exhibits chip area savings of up to 50% over the Cooley Tukey FFT.

1. INTRODUCTION

OFDM systems use the IDFT/DFT transform pair to generate a multicarrier modulation signal with very high spectral efficiency [1].
The N-point discrete Fourier Transform (DFT) is defined by the transform of the time sequence f_n into the frequency domain $F_{N,m}$ via

$$F_{N,m} = \sum_{n=0}^{N-1} f_n \cdot W_N^{mn}; \quad m = 0,1,...,N-1; \quad W_N = e^{-j\frac{2\pi}{N}}$$

The inverse transform (IDFT) is defined by

$$f_n = \frac{1}{N}\sum_{m=0}^{N-1} F_{N,m} \cdot W_N^{-mn}; \quad n = 0,1,...,N-1$$

Besides its spectral efficiency another asset of DFT-based OFDM is the simple equalization process which can be performed by a single complex multiplication per frequency bin. This type of equalization is possible if either the channel itself or the signal to be convolved with the channel is cyclic. While a cyclic channel does not exist in the RF-domain a cyclic signal can be mimicked by prepending a cyclic prefix having the length of the linear channel impulse response minus one. Employing a cyclic prefix offers the capability to exploit multipath signal reflections rather than suffering from them. These benefits along with the possibility to compute the IDFT and DFT with fast and efficient algorithms, called FFT-algorithms, have made OFDM the modulation method of choice for many modern wireless communication systems, especially in urban environments where multipath reflections abound. Several of these communication systems are listed in Table 1 along with the number of tones, i.e. carriers, that are used.

Table 1: No. of carriers for various wireless OFDM-based standards [2], [3].

Standard	No. of carriers
DAB	192, 384, 768, 1536
DRM	181, 203
DVB-T	2048, 4096, 8192
WLAN, HiperLAN	52
WiMAX	256, 2048

By approach the FFT/IFFT is a block oriented computation which, in the case of OFDM transmission, requires a serial data input stream to be parallelized before it can be transformed by the IFFT. Prior to transmission the parallel time samples rendered by the IFFT must be serialized again. The reverse is true for the FFT and signal reception. Fig. 1 depicts a simplified OFDM system illustrating the parallelization, transformation and serialization of data for transmission and reception.

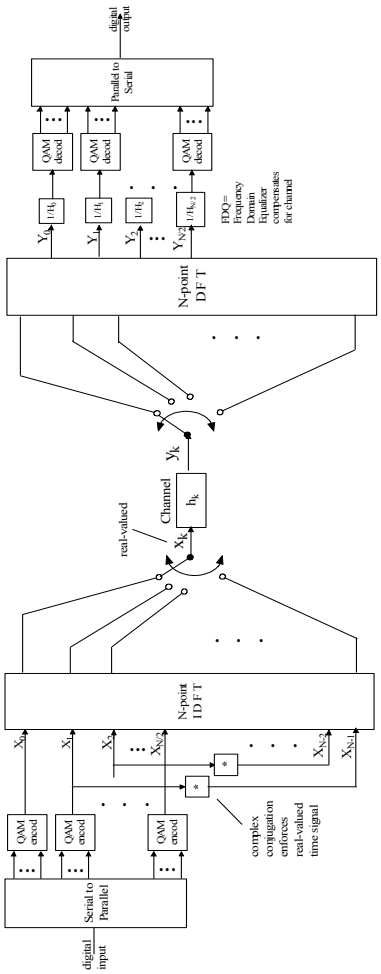

Figure 1: Simplified block diagram of the OFDM signal processing portion.

2. THE COOLEY-TUKEY-FFT AND ITS PIPELINE IMPLEMENTATION

A class of very efficient FFT-algorithms have transform lengths of $N=2^{\nu}$, ν integer, with the Cooley-Tukey-FFT, or CT-FFT, representing one of its most often used implementation variants [4]. Besides its favourable complexity and numerical properties a pipeline architecture [4] for the CT-FFT exists which not only allows for a HW-efficient implementation but also yields the required serial to parallel and parallel to serial data conversion as a by-product. Fig. 2 shows the pipeline architecture for the CT-FFT for $N=2^{\nu}$(left hand side) and its CT-IFFT counterpart (right hand side). The architecture consists of delay elements $z^{-(N/n)}$, twiddle-factor multipliers W_n^k, adders, and switches. For the FFT the switch S1 switches periodically between the upper and lower position. The transition from one switch position to the next occurs after N/2 time samples. The subsequent switches SW2, SW3, etc. change between straight through and crisscross positions with the switching frequency doubling from stage to stage.

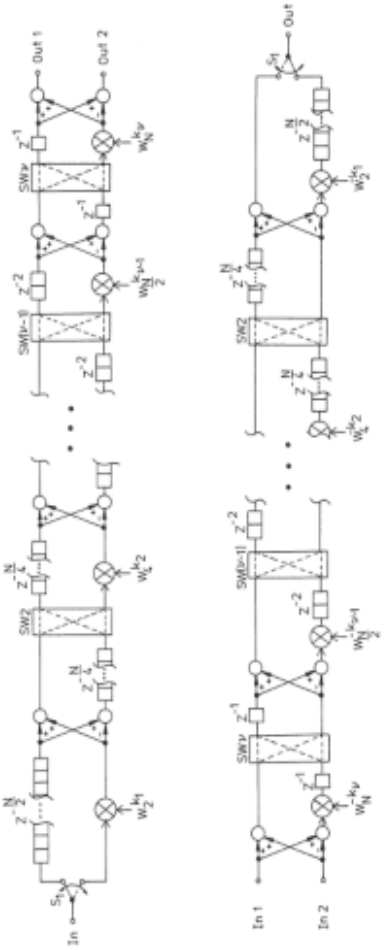

Figure 2: Cooley-Tukey-Pipeline for transform length $N=2^{\nu}$.

An important property of an OFDM-signal to be transmitted is that its samples are real-valued. To this end the frequency domain signal must be conjugate complex according to

$$F_{N,N-m} = F_{N,m}^{*}$$

where the superscripted asterisk denotes complex conjugation. This symmetry property requires the IDFT transform length N to be twice the number of complex input data.

Fortunately a group of FFT algorithms - called “Real-Valued FFTs” [5] - exists which exploit the symmetry properties ensuing from the real-valued time signal and complex conjugate frequency domain signal. One of these algorithms – the Bruun-FFT [6] - also lends itself to a pipeline implementation.

3. BRUUN-FFT AND ITS PIPELINE IMPLEMENTATION

The pipeline implementation of the Bruun-FFT algorithm, which has been introduced in [7], is depicted in Fig. 3.

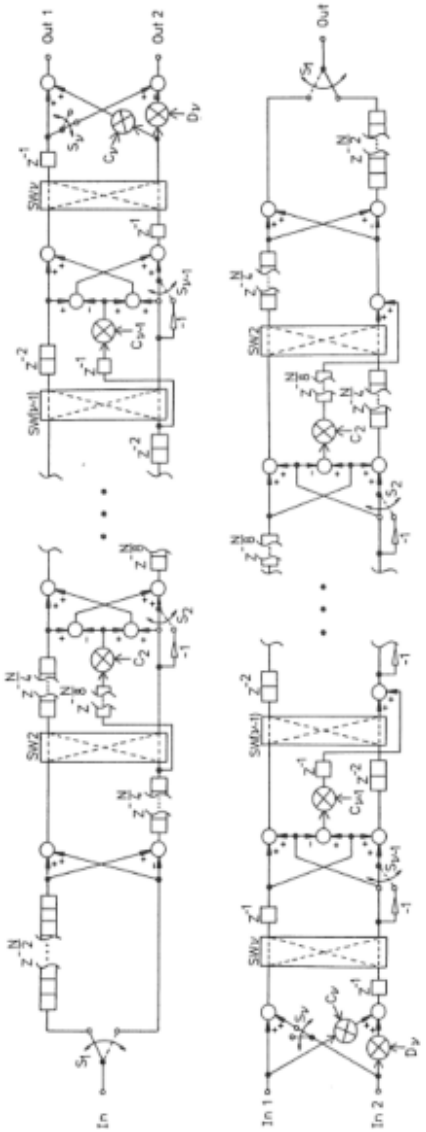

Figure 3: Bruun-pipeline for the transform length $N=2^{\nu}$.

Note that in contrast to the CT-pipeline, delay elements, twiddle-factor multipliers, adders, and switches, only have to deal with real-valued signals while most of its counterparts in the CT-FFT have to process complex values. The benefit is that a significantly lower number of HW-components are sufficient for the Bruun-FFT. The savings will be quantified in chapter 4.

4. VLSI IMPLEMENTATION EFFORT

The intent of this chapter is to estimate the chip area required for a full-fledged VLSI-implementation of both the CT- and Bruun-pipeline. In order to achieve this task four steps have to be taken:

1. The number of processing elements has to be computed.
2. An error analysis has to be performed so that the chip area estimation can be made on an equal basis in terms of computational accuracy.
3. A VLSI model has to be defined for each processing element including the interconnections between them.
4. The chip area for the entire pipeline has to be estimated by utilizing the information gained in the three previous steps.

4.1 NUMBER OF PROCESSING ELEMENTS FOR THE CT- PIPELINE

The counting of the processing elements is done by regarding the FFT-version of the pipeline. The IFFT-pipeline is dual to the FFT and hence exhibits exactly the same number of processing elements.
The number of real-valued non-trivial multipliers for the CT-pipeline amounts to

$$n_{mult} = 4 \cdot \sum_{k=0}^{\nu-3} k = 4 \cdot (\nu - 2)$$

assuming that a complex multiplication requires 4 real multiplications and two additions.

The number of real-valued non-trivial adders is

$$n_{add} = 6 \cdot (\nu - 1) + 2$$

considering that the first stage has a real input signal and hence requires only two real adders while all the following stages require complex additions and an additional two additions from the complex multiplication.

The number of real-valued delay elements sums up to

$$n_{delay} = \frac{N}{2} + 2 \cdot \frac{N}{4} + 2 \cdot \sum_{k=1}^{\nu-3} 2^k = \ldots = 2^{\nu} + 2^{\nu-1} - 4 .$$

Note that the first N delay elements in the FFT are real-valued.

Finally, the number of real-valued non-trivial twiddle factor coefficients to be stored is

$$n_{coeff} = \sum_{k=0}^{\nu-1} (2^k - 2) + 1 = \ldots = 2^{\nu} - 2 \cdot \nu$$

assuming that each twiddle factor multiplier has its own twiddle factor storage and considering all the symmetries of real and imaginary part of the coefficients.

4.2 NUMBER OF PROCESSING ELEMENTS FOR THE BRUUN- PIPELINE

As for the CT-pipeline the number of processing elements for the Bruun-pipeline is given in the following:
the number of real-valued non-trivial multipliers amounts to

$$n_{mult} = \nu$$

The number of real-valued non-trivial adders is given by

$$n_{add} = 4 \cdot (\nu - 2) + 4 = 4 \cdot (\nu - 1)$$

The number of real-valued delay elements, assuming $\nu \geq 3$, is

$$n_{delay} = \sum_{k=1}^{\nu-1} 2^k + \sum_{k=1}^{\nu-3} 2^k + \sum_{k=1}^{\nu-2} 2^k \ldots = 2^{\nu} + 2^{\nu-1} + 2^{\nu-2} - 6$$

counting the series of the upper, the middle and the lower delay.

Finally, the number of real-valued non-trivial twiddle factor coefficients to be stored is

$$n_{coeff} = \sum_{k=0}^{\nu-1} (2^k - 1) + 2^{\nu} - 2 = \ldots = 2^{\nu+1} - \nu - 3$$

4.3 ERROR ANALYSIS

In an error analysis it is important that the model takes care of potential signal overflows. In general this is done by employing "safe scaling" [4], [5] which is applied at every stage of the FFT (or IFFT). Block floating point [4], [5] exhibits improved noise properties but unfortunately is not appropriate for a pipeline architecture since the latter starts computing the next stage before the previous one is finished. Block floating point, on the other hand, requires an FFT stage to be finished entirely before scaling can be applied. The error analysis of both the CT-FFT and Bruun-FFT using "safe scaling" can be taken from the literature [8]. By defining the so-called noise-to-signal ratio (NSR) as

$$NSR = \sqrt{\frac{N^{-1} \cdot \sum_{m=0}^{N-1} E[e_m]^2}{E\left[\left|F'_{N,m}\right|^2\right]}}$$

where $E[e_m]^2$ is the noise power at frequency bin $F'_{N,m}$ the ratio

$$\frac{NSR_{Bruun}}{NSR_{CT}} = \sqrt{\frac{55 \cdot N^2}{64 \cdot (67 - 22 \cdot N^{-1})}}$$

can be computed which holds for both two's complement and sign magnitude arithmetic . It can easily be seen that the error behaviour of the Bruun-FFT is inferior to the one of the CT-FFT so that in order to exhibit the same computational accuracy the number

$$\Delta b = \left\lceil 20 \cdot \log_{10}\left(\frac{NSR_{Bruun}}{NSR_{CT}}\right) / 6 \right\rceil$$

of additional bits must be used for the Bruun-FFT. The result can be seen in Table 2 for various transform lengths

Table 2: Number of additional bits Δb required by the Bruun-FFT to exhibit the same accuracy as the CT-FFT.

Exponent ν	Transform Length $N=2^{\nu}$	Δb
3	8	0
4	16	1
5	32	2
6	64	3
7	128	4
8	256	5
9	512	6
10	1024	7
11	2048	8
12	4096	9

4.4 FULL CUSTOM CHIP AREA ESTIMATION

In order to estimate the chip area consumed by a fully integrated FFT-pipeline, the SIEMENS VENUS-S VLSI-process [9] shrunk to 65nm has been employed. The estimated chip areas of some basic logic elements are listed in Table 3. Multipliers of b x b bits are some of the largest building blocks in an FFT-pipeline and consist of the basic elements given in Table 3. The area of a real-valued b x b multiplier can be approximated by

$$A_{mult} = k_{Booth} \cdot \left[(b-1)^2 \cdot (A_{fa} + A_g) + (2b-1) \cdot A_g\right] + k_{Booth} \cdot \left[(b-1) \cdot A_{fa}\right]$$

where the first term represents the carry-save array and the second term the ripple-carry stage. Factor k_{Booth} scales the formula to represent the area of a Booth-Code multiplier including the wiring between the basic elements and equals roughly 1.3 [10].
Memory constitutes another complex element in an FFT-pipeline. According to [10] the area for RAM storage may be computed by

$$A_{RAM} = k_{1,RAM} \cdot 2^{2R} + k_{2,RAM} \cdot R \cdot 2^{R}$$

if R is the number of 1-bit storage elements in the RAM and a quadratic chip-layout is assumed.

Table 3: Chip areas for various computational elements and some constants in μm^2 assuming a 65nm process.

Element	Chip area
1-bit adder	A_{fa} = 429.53 μm^2
1-bit shifter	A_{sh} = 51.00 μm^2
AND-gate	A_g = 107.13 μm^2
constants	in μm^2
$k_{1,RAM}$	29.97 μm^2
$k_{2,RAM}$	398.44 μm^2
$k_{1,ROM}$	2.75 μm^2
$k_{2,ROM}$	287.11 μm^2

The area for ROM storage basically follows the same equation with just the constants k_1 and k_2 changed as shown in Table 3.

Now all information is available to estimate the total chip area for an FFT- or IFFT-pipeline. Figure 4 shows the estimated chip area A_{CT} for the CT-pipeline (connected rectangles) and A_{Bruun} for the Bruun-pipeline (connected diamonds). A wordlength of b=16 for both data and coefficients is assumed and constitutes a reasonable value for fixed scaling FFT-pipelines in an OFDM context [11]. Figure 4 reveals that the Bruun-pipeline exhibits a lower chip area consumption for a large span of transform lengths $N=2^n$.

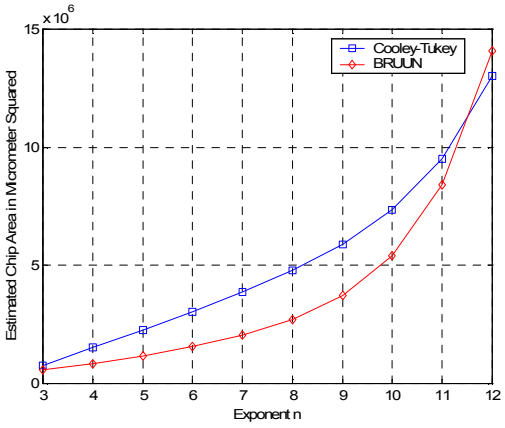

Figure 4: Estimated chip area of the Cooley-Tukey- and the Bruun-Pipeline for a 65nm-process, 16-bit wordlength equivalent.

Figure 5 depicts the relative chip area savings r calculated by

$$r = \frac{A_{CT} - A_{Bruun}}{A_{CT}}$$

and shows a relative savings of almost 50% for $N=2^6=64$ and b=16. The contribution of the various pipeline elements for this transform length is summarized in Figure 6 exhibiting that most of the chip area is required for the multipliers. As the number of multipliers is significantly lower in the Bruun-pipeline compared to the CT-pipeline the reduction of the rquired chip area is substantial. For larger transform lengths the advantage of the Bruun pipeline concerning multiplier count looses impact since the area of the delay elements becomes more influential as depicted in Figure 7. Yet, for the example of b=16 the Bruun-pipeline is still advantageous in terms of chip area until N reaches the value $2^{12}=4096$.

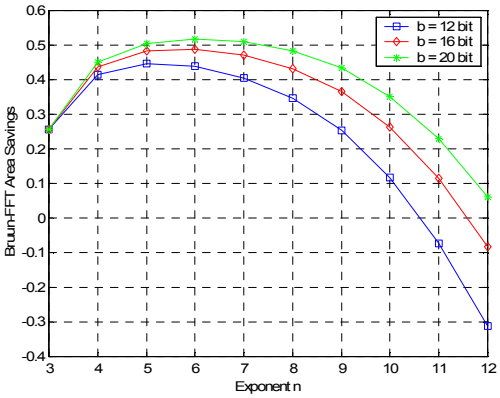

Figure 5: Area savings of Bruun-Pipeline over Cooley-Tukey-Pipeline depending on the FFT-size, and wordlength b.

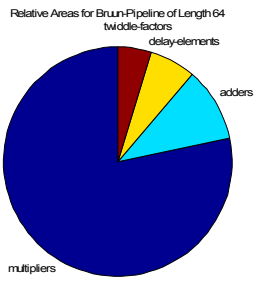

Figure 6: Relative Areas for Bruun-Pipeline of Length N=64, b=16.

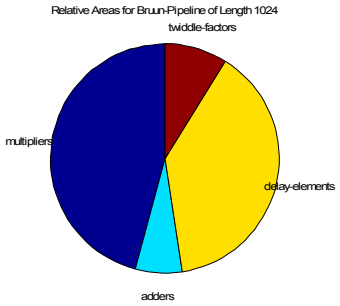

Figure 7: Relative Areas for Bruun-Pipeline of Length N=1024, b=16.

4.5 FPGA IMPLEMENTATION OF FFT-PIPELINES

The area advantage of the Bruun-pipeline is even larger if the pipeline is implemented on modern FPGAs like the Virtex 6 ® family manufactured by Xilinx ®. In the Virtex 6 ® family the mulipliers in a so-called DSP slice are 25x18-bit devices and hence the CT-pipeline cannot fully play off its reduced wordlength requirement. Figure 8 shows the result of the area savings calculation assuming that the multipliers have a fixed size of 25x18 bits while the data wordlength stays at 16 bits. For $N=2^7=128$ the area savings amount to almost 60%.

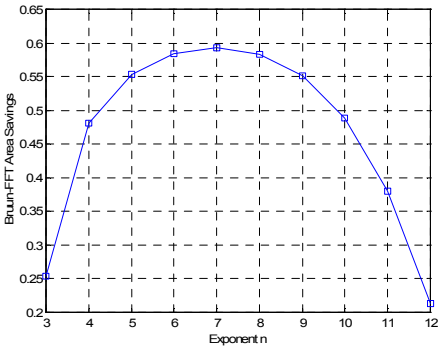

Figure 8: Area savings of Bruun-Pipeline over Cooley-Tukey-Pipeline depending on the FFT-size, assuming a fixed size 25x18-bit multiplier.

5. CONCLUSION

It has been demonstrated that OFDM systems call for a so-called "real-valued" FFT/IFFT which can efficiently be implemented by means of a pipeline architecture employing the Bruun-FFT algorithm. Compared to the commonly used Cooley-Tukey-Pipeline the Bruun-FFT-Pipeline offers a significant reduction in required chip area. For a full custom design this reduction can be up to 50% for a general wordlength of b=16 bit, depending on the transform length N. For FPGA-implementations where the multipliers generally come in fixed size slices the area advantage for the Bruun-FFT is potentially even greater reaching up to almost 60% in the given example.

6. REFERENCES

[1] Bahai, A.R. S., Saltzberg, B. R., and Ergen, M., Multi-Carrier Digital Communications Theory and Applications of OFDM, Springer, 2004.

[2] Rauwerda, G. K., Heysters, P. M., Smit, and Gerard J.M., *An OFDM receiver implemented on the coarse-grain reconfigurable Montium processor*, 9th Intl. OFDM-Workshop 2004, Dresden, pp. 1 - 5.

[3] Posega, R., *Advanced OFDM Systems for Terrestrial Multimedia Links*, PhD Diss., EPFL, Lausanne, 2005..

[4] Rabiner, L.R., and Gold, B., Theory ans Application of Digital Signal Processing, Prentice-Hall, Englewood Cliffs, 1975.

[5] Mitra, S., K, and Kaiser, J.F., Handbook for Digital Signal Processing, John Wiley, NY, 1993.

[6] Storn, R., *On the Bruun Algorithm and its Inverse*, Frequenz 46 (1992) 3-4, pp. 110 – 116.

[7] Storn, R., *A Novel Radix-2 Pipeline Architecture for the Computation of the DFT*, in IEEE Proc. ISCAS 1988, pp. 1899-1902.

[8] Storn, R., *Some Results in Fixed Point Error Analysis of the Bruun-FFT Algorithm*, IEEE Trans. Signal Proc., Vol. 41, No. 7, July 1993, pp. 2371 – 2375.

[9] VENUS-S Semicustom Design System, Zellenkatalog Standardzellen ACMOS3, ZTI DES V 312, Siemens-AG, Muenchen, 1987.

[10] Storn, R., *Algorithmen und Architekturen der diskreten Fouriertransformation zur schnellen Faltung reeller Signale*, PhD diss., Univ. Stuttgart, Germany, 1990.

[11] Yuan Chen, Yu-Wei Lin, and Chen-Yi Lee, *A Block Scaling FFT/IFFT Processor for WiMAX Applications*, IEEE Asian Solid-State Circuits Conf. ASSCC 2006, pp. 203 - 206.