5

Fast Fourier Transform Processors: Implementing FFT and IFFT Cores for OFDM Communication Systems

A. Cortés, I. Vélez, M. Turrillas and J. F. Sevillano
TECNUN (Universidad de Navarra) and CEIT
Spain

1. Introduction

The terms Fast Fourier Transform (FFT) and Inverse Fast Fourier Transform (IFFT) are used to denote efficient and fast algorithms to compute the Discrete Fourier Transform (DFT) and the Inverse Discrete Fourier Transform (IDFT) respectively. The FFT/IFFT is widely used in many digital signal processing applications and the efficient implementation of the FFT/IFFT is a topic of continuous research.

During the last years, communication systems based on Orthogonal Frequency Division Multiplexing (OFDM) have been an important driver for the research in FFT/IFFT algorithms and their implementation. OFDM is a bandwidth efficient multiple access scheme for digital communications (Engels, 2002; Nee & Prasad, 2000). Many of nowadays most important wireless communication systems use this OFDM technique: Digital Audio Broadcasting (DAB) (*World DAB Forum*, n.d.), Digital Video Broadcasting (DVB) (ETS, 2004), Wireless Local Area Network (WLAN) (IEE, 1999), Wireless Metropolitan Area Network (WMAN) (IEE, 2003) and Multi Band –OFDM Ultra Wide Band (MB–OFDM UWB) (ECM, 2005). Moreover, this technique is also employed in important wired applications such as Asymmetric Digital Subscriber Line (ADSL) or Power Line Communication (PLC).

OFDM systems rely on the IFFT for an efficient implementation of the signal modulation on the transmitter side, whereas the FFT is used for efficient demodulation of the received signal. The FFT/IFFT becomes one of the most critical modules in OFDM transceivers. In fact, the most computationally intensive parts of an OFDM system are the IFFT in the transmitter and the Viterbi decoder in the receiver (Maharatna et al., 2004). The FFT is the second computationally intensive part in the receiver. Therefore, the implementation of the FFT and IFFT must be optimized to achieve the required throughput with the minimum penalty in area and power consumption. The demanding requirements of modern OFDM transceivers lead, in many cases, to the implementation of special–purpose hardware for the most critical parts of the transceiver. Thus, it is common to find the FFT/IFFT implemented as aVery Large Scale Integrated (VLSI) circuit. The techniques applied to the FFT can be applied to the IFFT as well. Moreover, the IFFT can be easily obtained by manipulating the output of a FFT processor. Therefore, the discussion in this chapter concentrates on the FFT without loss of generality.

Different kinds of FFT algorithms can be found in the literature; e.g.: (Good, 1958; Thomas, 1963), (Cooley & Tukey, 1965), (Rader, 1968b), (Rader, 1968a), (Bruun, 1978) and (Winograd, 1978). Among the different kinds of FFT algorithms, the algorithms based on the approach proposed by James W. Cooley and John W. Tukey in (Cooley & Tukey, 1965) are very popular in OFDM systems. These Cooley–Tukey (CT) algorithms present a very regular structure, which facilitates an efficient implementation. The computations of the FFT is divided into $log_r(N)$ stages, where N is the number of points of the FFT and r is called the radix of the algorithm. Within each stage, data shuffling and the so-called butterfly computation and twiddle factor multiplications are performed. Usually, the butterfly operations are all identical and the twiddle factor multiplications and data shuffling follow some kind of pattern. This regular structure makes them very attractive for VLSI circuit implementation.

Different hardware architectures have been used in the literature for the implementation of the CT algorithms. The FFT hardware architectures can be classified into three groups:

- Monoprocessor: A single hardware element is used to perform all the butterflies, twiddle factor multiplications and data shuffling of each stage. The same hardware is reused for all the stages.
- Parallel: The computation of the butterflies, twiddle factor multiplications and data shuffling within one stage is accelerated by using several processing elements. The same hardware elements are again reused for all the stages.
- Pipeline: A single hardware element is used to perform all the butterflies, twiddle factor multiplications and data shuffling of each stage. However, in contrast to former categories, a different hardware element is used to process each stage.

It is common in the literature to further classify the pipeline architectures according to the structure used for the shuffling into two basic types: Delay Commutator (DC) and Delay Feedback (DF). Also, according to the number of lines of data used in these pipeline architectures, they can be classified into Single–path (S) or Multiple–path (M) architectures.

Many different variations of CT algorithms have been proposed in the literature to improve different aspects of the implementation (memory resources, number of arithmetic operations, etc.) and their mapping to a specific hardware architecture. Table 1 summarizes the features of some FFT/IFFT processors for OFDM systems proposed in the literature. Proposals such as (Chang & Park, 2004; Serrá et al., 2004) employed a monoprocessor architecture to process the FFT. (Jiang et al., 2004; Lin, Liu & Lee, 2004) used parallel architectures and (Kuo et al., 2003) chose a cached memory monoprocessor architecture for the FFT processing in an OFDM system. However, the most widely used architectures for the FFT/IFFT processor in an OFDM system are pipeline architectures. (Cortés et al., 2007; He & Torkelson, 1998; Lee & Park, 2007; Lee et al., 2006; Turrillas et al., 2010; Wang et al., 2005) propose Single-Path Delay Feedback (SDF) architectures. (Lin et al., 2005; Liu et al., 2007) employ an Multi-Path Delay Feedback (MDF) architecture. (Bidet et al., 1995) proposes a Single-Path Delay Commutator (SDC) architecture and (Jung et al., 2005; Saberinia, 2006) chose an Multi-Path Delay Commutator (MDC) architecture. (Saberinia, 2006) called the MDC architecture as Buffered Multi-Path Delay Commutator (BRMDC) due to the buffers used in the input data. Analyzing the radix, r, of the algorithms employed in the literature, it can be observed that:

- For the monoprocessor architectures, radix 2 (Serrá et al., 2004) and radix 4 (Chang & Park, 2004) algorithms have been used.

Reference	FFT points	Architecture	Algorithm	Application
(Jung et al., 2005)	64	Pipeline–MDC	r–2 DIT	WLAN
(Maharatna et al., 2004)	64	Pipeline	r–2 DIT	WLAN
(Serrá et al., 2004)	64	Monoprocessor	r–2 DIT	WLAN
(Lin et al., 2005)	128	Pipeline–MRMDF	MR $2/2^3$ DIF	UWB
(Saberinia, 2006)	128	Pipeline–BRMDC	r–2 DIF	UWB
(Lee et al., 2006)	128	Pipeline–SDF	r–2^4 DIF	UWB
(Liu et al., 2007)	64/128	Pipeline 8-path DF	MR DIF	UWB
(Cortés et al., 2007)	128	Pipeline	r–2^4 DIF	UWB
(Bidet et al., 1995)	8192	Pipeline–SDC	r–4/2 DIF	DVB–T
(Lin, Liu & Lee, 2004)	8192	Parallel	r–2^3 DIT	DVB–T
(Wang et al., 2005)	2/8 K	Pipeline–SDF	r–4/2 DIF	DVB–T
(Lenart & Owal, 2006)	2/4/8 K	Pipeline–SDF	r–2^2 DIF	DVB–T/H
(Lee & Park, 2007)	8 K	Pipeline–SDF	BD	DVB–T
(He & Torkelson, 1998)	1024	Pipeline–SDF	r–2^2 DIF	OFDM
(Kuo et al., 2003)	64–2048	Cached memory	r–2 DIT	OFDM
(Chang & Park, 2004)	1024	Monoprocessor	r–4 DIF	OFDM
(Jiang et al., 2004)	64	Parallel	r–2 DIT	OFDM
(Lin, Lin, Chen & Chang, 2004)	64	Pipeline	r–2 DIF	MIMO
(Rudagi et al., 2010)	64	Pipeline	r–2 DIT	OFDM
(Yu et al., 2011)	64	Pipeline	r–2 DIF	OFDM
(Tsai et al., 2011)	64	Pipeline	MR	OFDM
(Turrillas et al., 2010)	32 K	Pipeline	r–2^k DIF	DVB–T2

Table 1. FFT proposals for OFDM systems

- For the parallel architectures, FFT algorithms such as radix 2 (Jiang et al., 2004) and radix 2^3 (Lin, Liu & Lee, 2004) have been used.
- For the pipeline architectures, different FFT algorithms such as radix 2 (Jung et al., 2005; Rudagi et al., 2010; Saberinia, 2006; Yu et al., 2011), mixed radix (MR) (Bidet et al., 1995; Lin, Lin, Chen & Chang, 2004; Lin et al., 2005; Liu et al., 2007; Tsai et al., 2011; Wang et al., 2005), radix 2^2 (Cortés et al., 2007; He & Torkelson, 1998; Lenart & Owal, 2006; Turrillas et al., 2010), radix 2^3 (Cortés et al., 2007; He & Torkelson, 1998; Turrillas et al., 2010), radix 2^4 (Cortés et al., 2007; Lee et al., 2006; Turrillas et al., 2010) and balanced decomposition (BD) (Lee & Park, 2007) which reduces the number of twiddle factors have been proposed.

When the length of the FFT is not very large, the fixed-point format is the most widely used number representation format due to the area-saving with respect to the floating-point representation. Thus, (Chang & Park, 2004; Cortés et al., 2007; He & Torkelson, 1998; Jiang et al., 2004; Jung et al., 2005; Kuo et al., 2003; Lee et al., 2006; Lin, Lin, Chen & Chang, 2004; Lin et al., 2005; Liu et al., 2007; Maharatna et al., 2004; Saberinia, 2006; Serrá et al., 2004) proposed fixed–point FFTs for $N < 8192$. Nevertheless, (Wang et al., 2005) also used the fixed–point format for large FFTs. However, when the length of the FFT increases, different types of non fixed-point formats have been used. (Bidet et al., 1995) proposed a 8K processor that uses Convergent block floating–point to improve the quantization error of the FFT design. The block floating–point implementation or the semi–floating point (SFlP) format achieve a more efficient FFT design in (Lee & Park, 2007; Lenart & Owal, 2006). In a different approach, (Turrillas et al., 2010) employed a variable datapath technique to process a 32K points FFT.

From Table 1, it can be seen that many different algorithms and architectures have been proposed for OFDM systems. The designer must select the most appropriate algorithm and the most efficient architecture for that algorithm, given the specifications of a certain OFDM system. This selection is a difficult task. There is not a clear algorithm/architecture winner. Therefore, the designer should explore different algorithms and architectures in the literature to find the optimal one for the specific OFDM application under development.

The typical way of expressing the FFT algorithms in the literature is by means of summations or flow graph notation. Examples of these representations can be found in (He & Torkelson, 1998; Lee & Park, 2007; Lee et al., 2006; Lin, Lin, Chen & Chang, 2004; Lin, Liu & Lee, 2004; Lin et al., 2005; Liu et al., 2007; Maharatna et al., 2004; Tsai et al., 2006). These representations do not help the designer to understand the algorithm fast, therefore, making it difficult to relate it to its HW resources. Additionally, sometimes there is a lack of a general expression for the algorithm/architecture, which makes it harder to adapt and evaluate for a different OFDM system. Therefore, these representations are not practical for design space exploration. What the designer needs for efficient design space exploration is a general expression for the different design parameters of the algorithms which is easy to understand and makes it fast to map to hardware resources.

In (Pease, 1968), a matrix notation to express the FFT is proposed. Different FFT algorithms are obtained combining a reduced set of operators to simplify the implementation of parallel processing in a special–purpose machine. In (Sloate, 1974), H. Sloate used the same approach as (Pease, 1968) to demonstrate how several FFT algorithms previously defined could be derived using the matricial expressions. Additionally, he analyzed some new algorithms and worked out how to relate the matricial expressions to their implementation. However, that notation is not generalized for pipeline architectures. Recently, (Cortés et al., 2009) generalized the above approach presenting a unified approach for radix r^k pipeline SDC/SDF FFT architectures. Radix r^k pipeline FFT architectures are very efficient architectures that are well suited for OFDM systems.

This chapter reviews the matricial representation of radix r^k pipeline SDF FFT architectures. Thus, a general expression in terms of the FFT design parameters that can be linked easily to hardware implementation resources is presented. This way, the designer of FFT/IFFT processors for OFDM systems is provided with the tools for efficient design space exploration. The design space exploration and the optimal architecture selection procedure is illustrated by means of a case study. The case study analyzes the FFT/IFFT processor in a WLAN IEEE 802.11a transceiver. The high level analysis proposed in (Cortés et al., 2009) is extended to implementation level to select the most efficient FFT/IFFT core in terms of area and power consumption.

2. A unified matricial approach for radix r^k FFT SDF pipeline architectures

This section presents the matricial representation of radix r^k Decimation In Frequency (DIF) SDF pipeline architectures. First, the DFT matrix factorization procedure that leads to the FFT algorithms is reviewed. This review is used to define the basic types of matrices needed for the FFT. Next, in order to simplify the notation and latter mapping of the matricial representation to hardware resources, some operators are defined. Then, the general expression for radix r^k FFT SDF pipeline architectures is presented and the mapping to hardware resources illustrated.

2.1 Review of the DFT matrix factorization

The DFT can be expressed matricially as,

$$\mathrm{DFT}(\mathbf{x}) = \mathbf{T}_N \cdot \mathbf{x} \tag{1}$$

where

$$\mathbf{x}^T = \{\, x_0 \; x_1 \; x_2 \; \cdots \; x_{N-1} \,\}$$

and $\mathbf{T}_N$ is the size N square matrix of coefficients given by

$$(\mathbf{T}_N)_{mn} = e^{-j\frac{mn2\pi}{N}}. \tag{2}$$

$$\mathbf{T}_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{-j\frac{2\pi}{N}} & e^{-j\frac{2\cdot 2\pi}{N}} & \cdots & e^{-j\frac{(N-1)2\pi}{N}} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & e^{-j\frac{(N-1)2\pi}{N}} & e^{-j\frac{2(N-1)2\pi}{N}} & \cdots & e^{-j\frac{(N-1)^2 2\pi}{N}} \end{bmatrix}$$

In order to factorize $\mathbf{T}_N$, (Sloate, 1974) defines

$$\mathbf{T}_N = \mathbf{P}_N^{(r)} \cdot \mathbf{T}_N^{(p)}, \tag{3}$$

where r is the radix of the algorithm, $p = N/r$ is a positive integer ($p \in \mathbb{N}^*$) and $\mathbf{P}_N^{(r)}$ is the stride permutation matrix. This matrix, $\mathbf{P}_N^{(r)}$, is defined by its effect on a vector:

$$\mathbf{P}_N^{(r)} \cdot \{\, x_0 \; x_1 \; \cdots \; x_{N-1} \,\}^T = \{\, \mathbf{y}_0 \; \mathbf{y}_1 \; \cdots \; \mathbf{y}_{p-1} \,\}^T$$

with $\mathbf{y}_i = \{\, x_i \; x_{i+p} \; x_{i+2p} \; \cdots \; x_{i+(r-1)p} \,\}$. The matrix $\mathbf{T}_N^{(p)}$ can be expressed as

$$\mathbf{T}_N^{(p)} = [\mathbf{I}_r \otimes \mathbf{T}_{N/r}] \cdot \mathbf{D}_N^{(r)} \cdot [\mathbf{T}_r \otimes \mathbf{I}_{N/r}] \tag{4}$$

where $\mathbf{I}_z$ is the identity matrix of size z. The symbol $\otimes$ represents the Kronecker product. Given an $m \times n$ matrix $\mathbf{A}$ and a matrix $\mathbf{B}$, the Kronecker product is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{0,0}\mathbf{B} & a_{0,1}\mathbf{B} & \cdots & a_{0,n-1}\mathbf{B} \\ a_{1,0}\mathbf{B} & a_{1,1}\mathbf{B} & \cdots & a_{1,n-1}\mathbf{B} \\ \vdots & \vdots & & \vdots \\ a_{m-1,0}\mathbf{B} & a_{m-1,1}\mathbf{B} & \cdots & a_{m-1,n-1}\mathbf{B} \end{bmatrix}. \tag{5}$$

$\mathbf{D}_N^{(r)}$ is a diagonal matrix given by

$$\mathbf{D}_N^{(r)} = quasidiag(\{\, \mathbf{I}_p \; \mathbf{K}_p^1 \; \mathbf{K}_p^2 \; \cdots \; \mathbf{K}_p^{r-1} \,\}), \tag{6}$$

with $\mathbf{K}_p = diag(\{\, 1 \; e^{-j\frac{2\pi}{N}} \; e^{-j\frac{2\cdot 2\pi}{N}} \; \ldots \; e^{-j\frac{(p-1)2\pi}{N}} \,\})$. Replacing (4) in (3),

$$\mathbf{T}_N = \mathbf{P}_N^{(r)} \cdot [\mathbf{I}_r \otimes \mathbf{T}_{N/r}] \cdot \mathbf{D}_N^{(r)} \cdot [\mathbf{T}_r \otimes \mathbf{I}_{N/r}]. \tag{7}$$

Equation (7) shows how to write a DFT matrix in terms of smaller DFT matrices. This process can be repeated recursively until a expression in terms of $\mathbf{T}_r$ is arrived. Equation (7) is a matricial representation of the decomposition technique used in the well known Cooley-Tukey FFT (Cooley & Tukey, 1965).

2.2 Definition of operators

In order to simplify the notation, nine operators are defined. Two different types of operators can be distinguished: the reordering operators and the arithmetic operators.

- Reordering operators
  - *Shuffling Operator* $\mathbf{S}^{(a)}$: Let N be divisible by r^a,

$$\mathbf{S}^{(a)} = \mathbf{I}_{r^a} \otimes \mathbf{P}^{(r)}_{N/r^a}. \tag{8}$$

  Note that,

$$(\mathbf{S}^{(a)})^{-1} = \mathbf{I}_{r^a} \otimes (\mathbf{P}^{(r)}_{N/r^a})^{-1}. \tag{9}$$

- Arithmetic operators
  - *Butterfly Operator* $\mathbf{B}$: Let N be divisible by r,

$$\mathbf{B} = \mathbf{I}_{N/r} \otimes \mathbf{T}_r. \tag{10}$$

  - *First Twiddle Factor Multiplier Operator* $\mathbf{M1}^{(a,b)}$:
    Let N be divisible by r^{a+b},

$$\mathbf{M1}^{(a,b)} = \begin{cases} \mathbf{I}_{r^a} \otimes (\prod_{l=1}^{b}[\mathbf{P}^{(r)}_{r^l} \otimes \mathbf{I}_{N/r^{a+l}}] \cdot \mathbf{D}^{(r^b)}_{N/r^a} \\ \cdot \prod_{l=1}^{b}[\mathbf{P}^{(r)}_{r^l} \otimes \mathbf{I}_{N/r^{a+l}}]), & if r^b < \frac{N}{r^a} \\ \mathbf{I}_N, & if r^b \geq \frac{N}{r^a}. \end{cases} \tag{11}$$

  - *Second Twiddle Factor Multiplier Operator* $\mathbf{M2}^{(a)}$:
    Let N be divisible by r^{a+2},

$$\mathbf{M2}^{(a)} = \mathbf{I}_{r^a} \otimes \mathbf{D}^{(r)}_{r^2} \otimes \mathbf{I}_{N/r^{a+2}}. \tag{12}$$

  - *Third Twiddle Factor Multiplier Operator* $\mathbf{M3}^{(a,b)}$: Let N be divisible by r^{a+b} and $b \geq 2$,

$$\mathbf{M3}^{(a,b)} = \mathbf{I}_{r^a} \otimes ([\mathbf{P}^{(r)}_{r^2} \otimes \mathbf{I}_{r^{(b-2)}}] \cdot \mathbf{D}^{(r^2)}_{r^b} \cdot [\mathbf{P}^{(r)}_{r^2} \otimes \mathbf{I}_{r^{(b-2)}}]) \otimes \mathbf{I}_{N/r^{a+b}}. \tag{13}$$

2.3 Matricial representation of radix r^k pipeline FFT architecture

The decomposition procedure given in equation (7) can be applied recursively to devise the r^k SDF pipeline architectures. Let $N = r^{(kn_k+l)}$ with $\{k, n_k\} \in \mathbb{N}^*$ and $l \in \{0, 1, \ldots, k-1\}$, the matricial representation of r^k DIF SDF pipeline architectures is given by

$$\mathbf{T}_N = \begin{cases} \mathbf{Q}_N \cdot \mathbf{T}^{(p_k)}_N, & for\ \text{l=0} \\ \mathbf{Q}_N \cdot \mathbf{H}^{(l,k,n_k)} \cdot \mathbf{T}^{(p_k)}_N, & for\ \text{l>0} \end{cases} \tag{14}$$

$$\mathbf{T}^{(p_k)}_N = \prod_{m=0}^{n_k-1} \mathbf{H}^{(k,k,n_k-m-1)} \tag{15}$$

$$\mathbf{Q}_N = \prod_{i=0}^{n_1-1} \mathbf{S}^{(i)}, \tag{16}$$

where $p_k = r^{k(n_k-1)}$. The term $\mathbf{H}^{(b,k,i)}$ represents the i^{th} stage of a r^k FFT algorithm. When $b = 1$, $\mathbf{H}^{(1,k,i)}$ is given by,

$$\mathbf{H}^{(1,k,i)} = \mathbf{M1}^{(k\cdot i,k)} \cdot (\mathbf{S}^{(k\cdot i)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i)}. \tag{17}$$

When $b = 2$, $\mathbf{H}^{(2,k,i)}$ reduces to,

$$\mathbf{H}^{(2,k,i)} = \mathbf{M1}^{(k\cdot i,k)} \cdot (\mathbf{S}^{(k\cdot i+1)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+1)} \cdot \mathbf{V}^{(2,k\cdot i)}, \tag{18}$$

where $\mathbf{V}^{(b,k\cdot i)}$ is given by

$$\mathbf{V}^{(b,k\cdot i)} = \mathbf{M2}^{(k\cdot i+b-2)} \cdot (\mathbf{S}^{(k\cdot i)+b-2})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-2)}. \tag{19}$$

When b is even and $b > 2$, $\mathbf{H}^{(b,k,i)}$ is given by,

$$\mathbf{H}^{(b,k,i)} = \mathbf{M1}^{(k\cdot i,k)} \cdot (\mathbf{S}^{(k\cdot i+b-1)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-1)} \cdot \mathbf{V}^{(b,k\cdot i)} \cdot \left[\prod_{m=0}^{M=\frac{b}{2}-2} \mathbf{G}^{(b,k\cdot i,m)}) \right], \tag{20}$$

where $\mathbf{V}^{(b,k\cdot i)}$ is given by (19) and $\mathbf{G}^{(b,k\cdot i,m)}$, by (22).

When b is odd and $b > 1$, $\mathbf{H}^{(b,k,i)}$ is given by,

$$\mathbf{H}^{(b,k,i)} = \mathbf{M1}^{(k\cdot i,k)} \cdot (\mathbf{S}^{(k\cdot i+b-1)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-1)} \cdot \left[\prod_{m=0}^{M=\frac{b-1}{2}-1} \mathbf{G}^{(b,k\cdot i,m)} \right], \tag{21}$$

where $\mathbf{G}^{(b,k\cdot i,m)}$ is given by (22).

$$\mathbf{G}^{(b,k\cdot i,m)} = \begin{cases} \mathbf{M3}^{(k\cdot i+2(M-m),b-2(M-m))} \\ \cdot (\mathbf{S}^{(k\cdot i+b-2m-3)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-2m-3)} \\ \cdot \mathbf{M2}^{(k\cdot i+b-2m-4)} \\ \cdot (\mathbf{S}^{(k\cdot i+b-2m-4)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-2m-4)}, \ if \text{ b even and b>2} \\ \mathbf{M3}^{(k\cdot i+2(M-m),b-2(M-m))} \\ \cdot (\mathbf{S}^{(k\cdot i+b-2m-2)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-2m-2)} \\ \cdot \mathbf{M2}^{(k\cdot i+b-2m-3)} \\ \cdot (\mathbf{S}^{(k\cdot i+b-2m-3)})^{-1} \cdot \mathbf{B} \cdot \mathbf{S}^{(k\cdot i+b-2m-3)}, \ if \text{ b odd and b>1 .} \end{cases} \tag{22}$$

When $N = r^{kn_k+l}$ with $l > 0$, (14) means that, after n_k stages with r^k structure, an additional processing stage with the structure of a r^l algorithm is required.

The above matricial representation is general in terms of N, r and k. The Decimation In Time (DIT) version of the architecture can be easily obtained by transposing the expressions.

2.4 Mapping to pipeline architectures

The structure of typical stages r^k pipeline SDF architecture is depicted in Figure 1. It can be observed in the first row of Figure 1 that the output of one stage is connected to the input of the next stage. Within each stage, there is a sequence of hardware processing elements which can perform the computations defined by the operators presented in Section 2.2 as illustrated in the second and third rows of Figure 1. The hardware required to implement the computations demanded by each operator is discussed in (Cortés et al., 2009). It is important to note that the shuffling operators surrounding the butterfly operators can be merged in the single device with feedback typical of the SDF architecture. To illustrate this, in the fourth row of Figure 1 the implementation of radix 2^k algorithms is presented.

Each stage $\mathbf{H}^{(k,k,i)}$ consists of k butterfly operators $\mathbf{B}$, with their corresponding shuffling and unshuffling $\mathbf{S}$ and $(\mathbf{S})^{-1}$ terms. The hardware that implements the arithmetic of the butterfly operators $\mathbf{B}$ is the same in all the stages of the FFT processor. The implementation of the butterfly depends on the value of r. The length of each of the delay lines used for the shuffling and unshuffling corresponding to the first butterfly of the first stage is N/r. The length of each of these delay lines is reduced by a factor of r from one butterfly to the next one as is shown in the fourth row of Figure 1. The number of delay commutator structures used for the shuffling and unshuffling around a butterfly unit is the same as the number of butterfly units.

Operators **M1**, **M2** and **M3** mainly translate to complex multipliers (C.M.) and Look-Up Tables (LUTs) to store the twiddle factors (T.F.). A stage of the form $\mathbf{H}^{(k,k,i)}$ has one twiddle factor multiplier operator **M1**. The number of twiddle factors stored in the LUT used to implement **M1** is N for the first stage and it reduces by a factor of r^k from one stage to the next. When $N = r^{kn_k}$, no complex multiplier is needed to implement the twiddle factor multiplications given by $\mathbf{M1}^{(k(n_k-1),k)}$ of the last stage $\mathbf{H}^{(k,k,n_k-1)}$, because $\mathbf{M1}^{(k(n_k-1),k)} = \mathbf{I}_N$. The final processing given by terms of the form $\mathbf{H}^{(l,k,n_k)}$ when $N = r^{kn_k+l}$, with $l \neq 0$, does not actually have a operator **M1**; i.e.: $\mathbf{M1}^{(kn_k,k)} = \mathbf{I}_N$.

For $k > 1$, twiddle factor multiplier operators **M2** appear. A stage $\mathbf{H}^{(b,k,i)}$ has $b/2$ operators **M2** when b is even and $(b-1)/2$ when b is odd. The number of twiddle factors stored in the LUT used to implement **M2** is always r^2.

For $k > 2$, twiddle factor multiplier operators **M3** appear. A stage $\mathbf{H}^{(b,k,i)}$ has $b/2-1$ operators **M3** when b is even and $(b-1)/2$ when b is odd. The number of twiddle factors stored in the LUT used for the implementation of the first **M3** within a stage $\mathbf{H}^{(b,k,i)}$ is r^b and it reduces by a factor of r^2 from one **M3** to the next one within the same stage.

2.5 Hardware resources of r^k pipeline architectures

In (Cortés et al., 2009), the complexity of the r^k algorithms in terms of area was analyzed. In this section, some of their conclusions are summarized.

For a given N, the designer can vary the parameters r and k to achieve different implementations. These parameters influence the reordering operators and the arithmetic operators.

The overall amount of memory words used for the delay lines due to the reordering operators depends neither on r nor k. In an SDF architecture, the amount of memory is $N-1$ words.

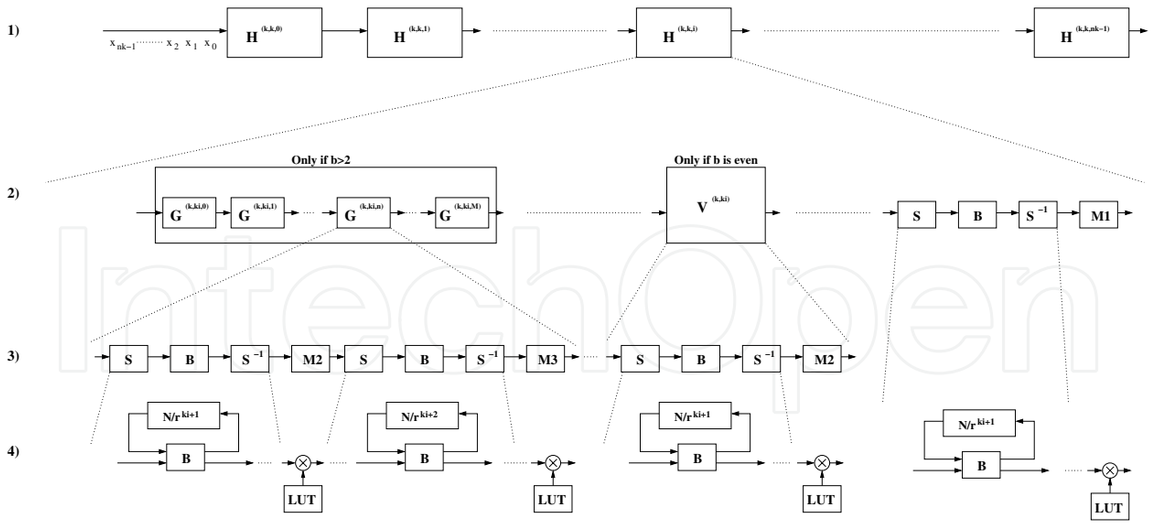


Fig. 1. Structure of typical stages of r^k pipeline SDF architectures

Therefore, the overall amount of memory words is fixed by the number of points of the FFT. $N-1$ is the minimum amount of memory reported in the literature for a pipeline FFT architecture. Modifying the values of r or k does not increase the amount of memory words used for the reordering operators.

Increasing the value of r for fixed values of N and k will reduce the number of butterflies and the overall number of complex multipliers. However, the butterflies become more complex. Regarding the number of twiddle factors, the number of twiddle factors due to **M1** are reduced, while the number of twiddle factors due to **M2** and **M3** increase. For small values of k, there is a reduction of the overall number of twiddle factors; while for large values of k, there is an increase in the overall number of twiddle factors. Thus, the benefit of increasing the value of r depends on the value of k.

If the values of N and r are fixed, the benefits of increasing k can be studied. The overall number of butterflies is $\log_r N$, and thus, increasing k does not increase either the number of butterflies or their area. Increasing the value of k reduces the number of complex multipliers due to **M1**. Once **M2** and **M3** appear in an architecture, the number of complex multipliers due to each one remains approximately constant. Thus, an important benefit of increasing the value of k is that it reduces the overall number of complex multipliers. When k increases, the number of twiddle factors due to **M1** is reduced. For $\lceil (\log_r N)/2 \rceil \leq k < \log_r N$, one single **M1** with N twiddle factors appears. Finally, when $k = \log_r N$, $\mathbf{M1} = \mathbf{I}_N$ and no hardware is needed to implement any **M1**. The number of twiddle factors due to **M2** remains constant with k. The number of twiddle factors due to **M3** increases with k. Adding the contributions of **M1**, **M2** and **M3**, the overall number of twiddle factors first reduces with k and then it starts to increase. It can be concluded that an optimum value of k exists for each value of N and r. Usually this optimum value is near $k = \lceil (\log_r N)/2 \rceil$.

From the general expressions, the designer has to look for those values of r and k that result in an optimum single-path FFT pipeline architecture for a given application. This search can be easily performed thanks to the close link between the proposed matricial representation and implementation. No other notation allows such an exploration within the pipeline architectures.

3. Case study: WLAN IEEE 802.11a

In this section, a case study is analyzed applying the proposed design space exploration in order to achieve the most efficient algorithm/architecture for an OFDM system. This design space exploration does not only analyze the hardware complexity of the algorithms as in (Cortés et al., 2009). Additionally, an implementation level analysis is carried out also taking into account the power consumption which is very important for wireless devices.

The main objective of this design space exploration is to select the most efficient radix r^k pipeline SDF FFT architecture for the OFDM application. An FFT for the WLAN IEEE 802.11a standard has been selected as the case study. In this case, the length of the FFT/IFFT is 64 points. According to (Cortés et al., 2009), the optimum value of k is near 3. Therefore, the following analysis concentrates on $r = 2^2$, $r = 2^3$ and $r = 2^4$ algorithms in order to perform the implementation level analysis and to study the silicon area and power consumption results.

The proposed design space exploration can be divided into four steps. At first, the OFDM system described by the standard is analyzed to extract the specifications. Once the OFDM specifications are known, the designer focuses on the system and on the FFT/IFFT specifications to determine the parameters needed for the FFT/IFFT design. The next step is the implementation level analysis of different FFT/IFFT algorithms and architectures in order to select the most efficient core for the WLAN system according to a given criterion. This step is composed of two main analysis: the Error Vector Magnitude (EVM) analysis in the transmitter and the Carrier-To-Noise Ratio (CNR) analysis in the receiver. After these analysis, the data bitwidth (*dbw*) and the twiddle factors bitwidth (*tbw*) are chosen so as not to degrade the system performance. Finally, the layout of the most efficient FFT/IFFT core for WLAN 802.11a in terms of area and power consumption is shown.

3.1 Study of the IEEE 802.11a standard

Table 2 shows the parameters specified in IEEE 802.11a standard for a WLAN system (IEE, 1999). $N_c = 64$ sub-carriers are used. $N_p = 4$ sub-carriers are used as pilot tones to make the coherent detection more robust against frequency offsets and phase noise. These pilot tones are always in sub-carriers -21, -7, 7 and 21 and they are modulated in Binary Phase-Shift Keying (BPSK). $N_d = 48$ tones are employed as data sub-carriers and the rest of sub-carriers are zero. The length of the guard interval must be longer than the delay spread of the channel.

$\mathbf{N}_c$	$\mathbf{N}_d$	$\mathbf{N}_p$	$\mathbf{N}_{gi}$	**R (Mbps)**	**BW (MHz)**	$\mathbf{t}_{sym}$ **(μs)**	**EVM (dB)**	$\mathbf{N}_f$	$\mathbf{L}_p$	H/N_o **(dB)**
64	48	4	16	54	20	4	-25	20	16	40

Table 2. OFDM Parameters for a WLAN IEEE 802.11a system

Considering an indoor environment, the necessary guard interval in a WLAN system is 0.8 μs. Therefore, the last $N_{gi} = 16$ data must be copied at the beginning of the OFDM symbol. The maximum data rate is $R = 54$ Mbps. Additionally, the bandwidth of the system is $BW = 20$ MHz. The standard also determines that the OFDM symbol period is $t_{sym} = 4\mu s$. In IEEE 802.11a, data can be modulated in BPSK, Quadrature Phase-Shift Keying (QPSK), 16-Quadrature Amplitude Modulation (QAM) or 64-QAM.

The EVM is defined as:

$$\text{EVM} = \sqrt{(I - I_o)^2 + (Q - Q_o)^2}, \tag{23}$$

where $I + jQ$ is the measured symbol and $I_o + jQ_o$ is the transmitted symbol. Figure 2 illustrates the computation of the EVM for a 16-QAM constellation. The IEEE 802.11a specifies

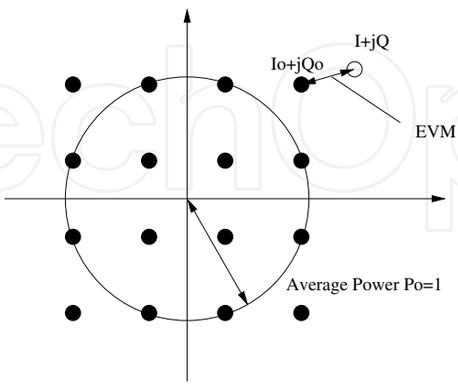


Fig. 2. EVM in the 16-QAM constellation

a mean value E_{rms} which has to be measured after transmitting $N_f = 20$ frames of $L_p = 16$ OFDM symbols with an average power P_o,

$$E_{\text{rms}} = \frac{1}{20}\sum_{i=1}^{20}\sqrt{\frac{\sum_{j=1}^{16}\left[\sum_{k=1}^{(48+4)}\text{EVM}(i,j,k)\right]}{(48+4)16\cdot P_o}}, \tag{24}$$

where $\text{EVM}(i,j,k)$ is the magnitude of the error vector for the k^{th} sub–carrier of the j^{th} OFDM symbol of the i^{th} frame. The EVM is the quality constraint that must not be degraded in the transmitter. The EVM value in Table 2 is the minimum EVM in a WLAN system when the system transmits data modulated in 64–QAM with the maximum data rate of 54 Mbps.

The IEEE 802.11a standard specifies the spectral mask of the output signal in the transmitter. Figure 3 presents the spectral mask of the IEEE 802.11a transmitter output where this height is $H/N_o = 40dB$. The quality constraint selected for the receiver is the non–degradation of the CNR when the input signal to the receiver has the spectral mask specified by the standard. This is a first approach that produces an overconstrained core. EVM and H/N_o specifications are used to determine the allowed quantization error.

3.2 System analysis and FFT/IFFT specifications

The goal of this analysis is to determine the parameters of Table 3. Table 3 presents five important system specifications which influence significantly the design of the FFT/IFFT: the length of the FFT/IFFT N, the system clock frequency f_{clk}, the value of K_{tx} in the transmitter, the value of K_{AGC} and the CNR in the receiver. As the OFDM system supports 64 sub-carriers, the FFT core must be designed to perform 64-points FFT; i.e.: $N = 64$. As shown in Table 2, the OFDM symbol period (t_{sym}) in a WLAN system is equal to 4 μs. Following the analysis of (Velez, 2005), $f_{clk} = 60$ MHz is selected, higher than the maximum data rate R and multiple of the BW.

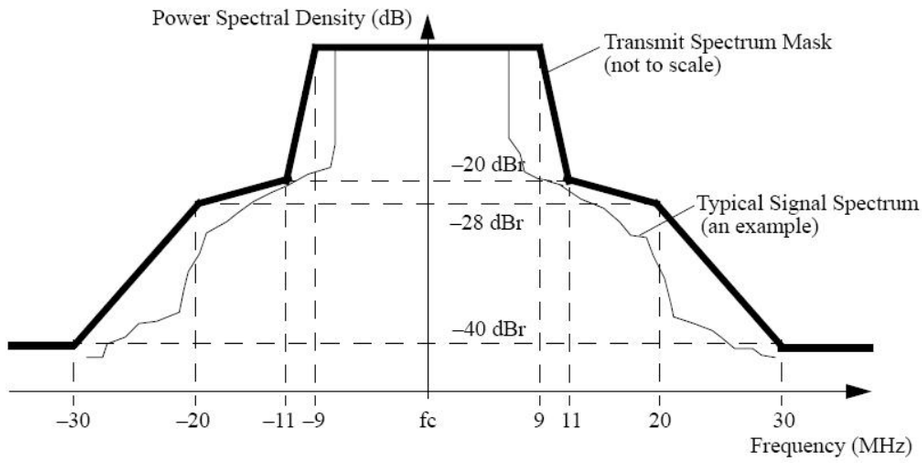


Fig. 3. Spectral mask of a transmitter output

N	Length of the FFT/IFFT
$\mathbf{f}_{clk}$	System clock frequency
$\mathbf{K}_{tx}$	Maximum gain to improve the DAC performance
$\mathbf{K}_{AGC}$	Maximum gain to improve the FFT performance
$\mathbf{CNR}_{max}$	Maximum carrier–to–Noise Ratio

Table 3. System specifications

3.2.1 k_{tx} in the transmitter

In order to select K_{tx}, an analysis of the IFFT output data must be carried out for each OFDM system. For this analysis, the modulation where the data reach the highest values in the I and Q data must be used.

In this analysis, K_{tx} is estimated by means of simulations. The transmission of the number of frames determined by the standard is simulated for different values of K_{tx} to calculate the EVM. Then, the largest K_{tx} that does not degrade the EVM can be selected. This way, the whole dynamic range of the Digital-to-Analog Converter (DAC) is exploited.

Figure 4 shows the system model used to analyze the EVM degradation with the increase of the value of K_{tx}. The system model is composed of a floating–point IFFT in the transmitter and a floating–point FFT in the receiver. The effect of the clipping in the DAC is emulated by a *limiter* located after the multiplication by K_{tx}. This *limiter* is responsible for saturating the amplified IFFT output according to the integer bits of the data representation. Two integer bits are considered for both the input and output data in the IFFT, since the maximum value of the input data is 1.08 when the modulation is 64–QAM. In order to improve the dynamic range of the IFFT output, the data are amplified by a factor K_{tx}. A factor K_{tx}, which is a power of two, is preferred to simplify the hardware implementation. In (Velez, 2005), the overflow probability of the different types of modulations in WLAN is analyzed carrying out Monte Carlo simulations with $3 \leq K_{tx} \leq 10$. It is concluded that BPSK modulation is the most sensitive to the clipping at the DAC. (Velez, 2005) also concluded that the most suitable value

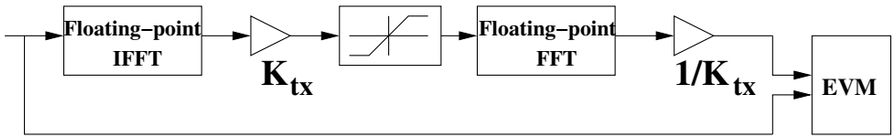


Fig. 4. System model to study the effect of K_{tx} in EVM

for K_{tx} is $K_{tx} = 4$ so as not to degrade the system performance. Additionally, $K_{tx} = 4$ is implemented as a simple shift in a real implementation.

Figure 5 shows the effect of the factor K_{tx} on the EVM in the transmitter, that is, how the EVM is degraded by the clipping produced in the IFFT. If a factor $K_{tx} = 8$ were selected, the

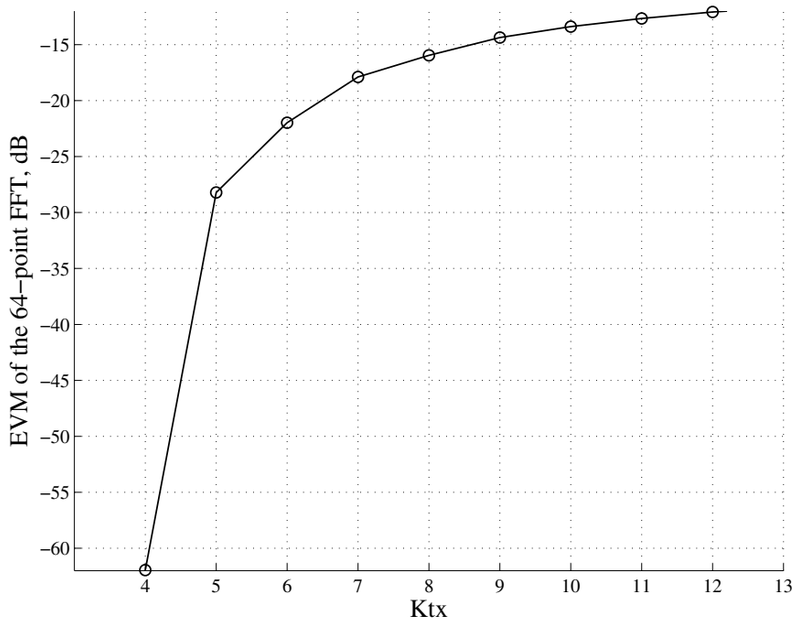


Fig. 5. Analysis of the effect of the K_{tx} in EVM

EVM value would not meet the standard specification. It can be observed that $K_{tx} = 4$ is a conservative decision, since the EVM specified by the standard for a BPSK modulation with a data rate of 9 Mbps is -8 dB, and the EVM obtained with $K_{tx} = 4$ is -61.95 dB.

3.2.2 k_{AGC} and CNR_{max} in the receiver

An Automatic Gain Control (AGC) module is a block found in many electronic devices. The AGC is commonly used to dynamically adjust the gain of the receiver amplifier to keep the received signal at the desired power level.

In this analysis, the effect of the AGC is modeled as a gain K_{AGC}. For the analysis, the value of K_{AGC} that does not degrade the CNR, is selected. This value is determined by means of simulations.

Figure 6 presents the system model to determine this factor K_{AGC}. This model is composed by a floating-point IFFT at the transmitter and a floating-point FFT at the receiver. The channel is modeled as as an Additive White Gaussian Noise (AWGN) channel.

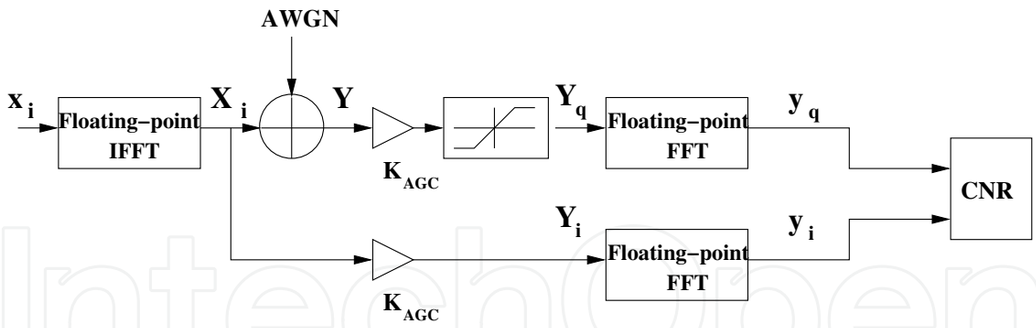


Fig. 6. System model to estimate the CNR for different values of K_{AGC}

Before applying the AWGN channel, the power of the output signal of the IFFT, X_i, is normalized to 1 dividing the signal by the factor λ, which is defined as

$$\lambda = \sqrt{\frac{N_d + N_p}{N_c^2}}. \tag{25}$$

In Figure 6, Y is the signal after applying the AWGN channel. At the receiver, this signal is multiplied by the gain K_{AGC} and a *limiter* is applied. This *limiter* is responsible for limiting the amplified signal according to the number of integer bits of the data representation. Once the *limiter* is applied, Y_q is the input of the floating-point FFT and y_q is the output of the floating-point FFT. y_i is the output of the floating-point FFT without noise and without applying the *limiter*. For the simulations, it is necessary to determine the variance of the noise to be applied to the transmitted signal so that the signal meets the spectral mask with the height H/N_o given by the standard. The Signal-To-Noise Ratio (SNR) of the transmitter output can be calculated from the height of the spectral mask H/N_o given by the standard. The power of the output signal of the transmitter can be written as,

$$P_o = \frac{(N_d + N_p) \cdot BW \cdot (H - N_o)}{N_c}, \tag{26}$$

where it is assumed that the power of the data sub-carriers is equal to the power of the pilot sub-carriers. The noise power is given by:

$$P_n = N_o \cdot BW. \tag{27}$$

Therefore,

$$\text{SNR} = \frac{P_o}{P_n} = (\frac{H}{N_o} - 1) \cdot \frac{(N_d + N_p)}{N_c}. \tag{28}$$

Then, the variance σ^2 of the noise in the channel is found to be,

$$\sigma^2 = \frac{1}{(\frac{H}{N_o} - 1) \cdot \frac{(N_d + N_p)}{N_c}}. \tag{29}$$

In order to select the value of the factor K_{AGC}, the CNR is used as a figure of merit. CNR is defined as,

$$\text{CNR} = \frac{\hat{P}_i}{\hat{P}_n}, \tag{30}$$

where $\hat{P}_i$ is the power of y_i and $\hat{P}_n$ is the power of the difference between y_i and y_q.

The gain K_{AGC} is selected so that the CNR is not degraded. Factor K_{AGC} fixes the power of the signal Y_q; i.e.: P_{Y_q}. The figures to analyze the effect of K_{AGC} on CNR plot the CNR versus P_{Y_q}, which is a more representative value.

The CNR is obtained by means of Monte Carlo simulations. This method allows the designer to estimate a measure of the performance of a communication system and the quality of the estimation itself (Sevillano, 2004). The looseness of our estimation in order to achieve a confidence interval equal to 0.95 is defined as,

$$\gamma = \frac{2 \cdot \sqrt{\hat{var}(C\hat{N}R)}}{C\hat{N}R} \tag{31}$$

Therefore, it can be said that the probability of CNR belonging to the interval $[(1-\gamma)C\hat{N}R, (1+\gamma)C\hat{N}R]$ is 0.95. The Monte Carlo simulations are stopped when $\gamma < 10^{-2}$.

It is assumed that the signal arrives with the maximum quality $H/N_o = 40$ dB. Therefore, applying (28), the SNR of the transmitted signal is of 39.09 dB.

Figure 7 shows the $(CNR)_{dB}$ versus the power of the input signal of the FFT P_{Y_q}. The signal works without degradation with $P_{Y_q} < -4$ dB, which corresponds to $K_{AGC} = 6$ in the simulations. For this value of K_{AGC}, $(CNR_{max})_{dB}$ is 39.1 dB.

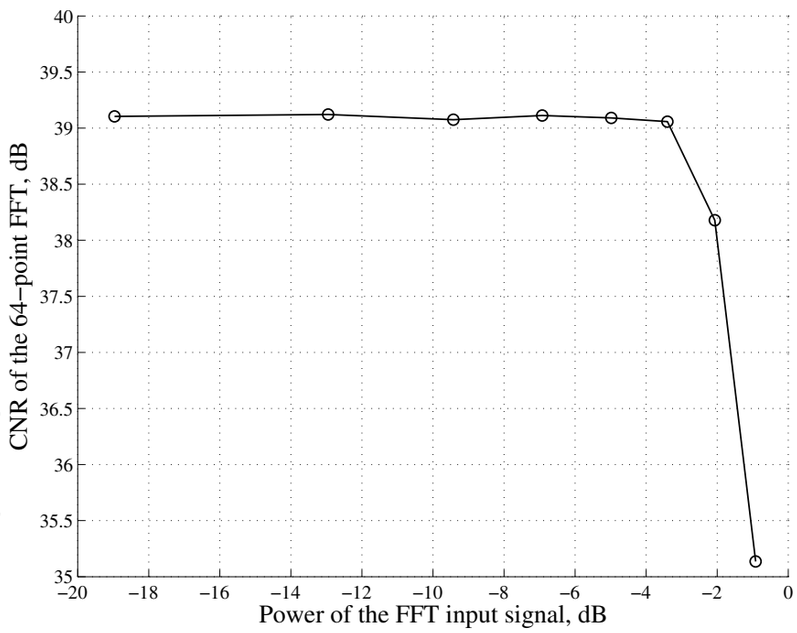


Fig. 7. $(CNR)_{dB}$ vs the power of the input signal in the FFT (P_{Y_q})

3.2.3 Summary of results

In this step, the length of the FFT N, the system clock frequency, f_{clk}, the gain factor in the transmitter, K_{tx}, and the gain factor in the AGC, K_{AGC} and the CNR_{max} have been estimated. Table 4 presents the necessary OFDM parameters for the next step.

N_{gi}	BW (MHz)	t_{sym} (μs)	EVM (dB)	N_f	L_p	f_{clk} (MHz)	N	K_{tx}	K_{AGC}	CNR_{max} (dB)
16	20	4	-25	20	16	60	64	4	6	39.1

Table 4. Specifications for the architecture selection for WLAN systems

3.3 Selection of algorithm/architectures

A high throughput FFT core is needed to fulfil the required specifications. Pipeline architectures are well suited to achieve small silicon area, high throughput, short processing time and reduced power consumption. Table 5 presents the hardware complexity of the candidate pipeline-SDF architectures. The pipeline-SDF radix 2^2 DIF architecture is composed

Architec.	r	Normal mults.	Constant mults.	ROM (regs.)	RAM (regs.)	adds /subs
pipeline–SDF	2^2	8	0	80	192	28
pipeline–SDF	2^3	4	4	64	192	40
pipeline–SDF	2^4	4	4	64	192	35

Table 5. Hardware complexity of candidate FFT architectures working at 60 MHz for WLAN systems

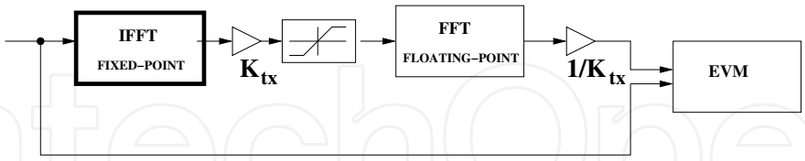
of six butterflies and two complex multipliers. The pipeline-SDF radix 2^3 DIF architecture is composed also of six butterflies, but one complex multiplier and two constants multipliers by one constant are used. The pipeline-SDF radix 2^4 DIF architecture is formed by six butterflies, one complex multiplier and one constant multiplier by two constants. Therefore, the three algorithms have the same number of multipliers taking into account normal and constant multipliers. The radix 2^3 and 2^4 DIF algorithms reduce ROM with respect to the radix 2^2 DIF one, but they add control logic.

To sum up, at this point it is not clear which is the most efficient architecture for WLAN. Therefore, it is necessary to make an implementation level analysis. The hardware complexity of radix 2^2, 2^3 and 2^4 DIF algorithms is compared to search the optimum design that meets the system specifications. The EVM constrains the word-length of the IFFT in the transmitter. The word-length in the receiver is constrained by the CNR. If the transmitter and the receiver are implemented in the same chip, the highest word-lengths must be chosen. The figures of merit to select the algorithm and architecture are the area and the power consumption estimated for an Application-Specific Integrated Circuit (ASIC) technology. Thus, this selection process can be stated as the problem of finding the FFT/IFFT processor which minimizes the AP cost function subjected to the constraints given by the specifications. The AP criterion trades off area and power consumption and can be used as a measure of the efficiency of the core. The area and power results presented in the following sections have been calculated for a TSMC 90 nm 6 ML technology with a clock frequency of 60 MHz working at 1.0V and a temperature of 25^oC. The area results have been estimated multiplying the cell area by a factor of 2.

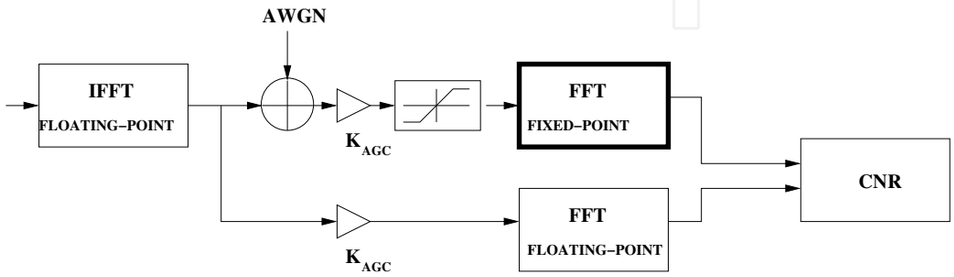
3.3.1 EVM analysis

In order to analyze the effect of the IFFT quantization error on the EVM during transmission, an ideal reception is considered. The system model is composed of a fixed-point IFFT at the

transmitter and a floating-point FFT at the receiver as can be seen in Figure 8(a). The EVM value is employed to select the values of *dbw* and *tbw* needed in the transmitter. An EVM margin of -10 dB is used in order to select a conservative word–length. Thus, a margin is left for other sources of error, such as the error produced by the analog processing. In order to



(a) EVM model



(b) CNR model

Fig. 8. System models to study the effect of quantization error on EVM and CNR

obtain the EVM results in Figures 9 and 10 the transmission of $N_f = 20$ frames of $L_p = 16$ OFDM symbols modulated in 64-QAM has been simulated. The pipeline-SDF with the radix 2^2, 2^3 and 2^4 DIF algorithms have been studied to find the most efficient implementation. For radix 2^3 and 2^4 algorithms, the bitwidth of the constant multiplier is assumed to be equal to the bitwidth of the normal multiplier.

Figure 9 shows the EVM and the area results of the pipeline–SDF DIF 64–point FFT/IFFT core for different algorithms. In order to guarantee an EVM of at least -35 dB, the radix 2^2 algorithm requires a (*dbw*,*tbw*) of (12,8), whereas the radix 2^4 and 2^3 algorithms need (12,7). For the given EVM, the radix 2^4 algorithm achieves the smallest core with $(dbw, tbw) = (12, 7)$.

The EVM and the power results of the pipeline–SDF DIF 64–point FFT/IFFT core are presented in Figure 10. It can be observed that the radix 2^2 algorithm requires less power consumption for the same (dbw, tbw) than the rest of algorithms, whereas the radix 2^3 algorithms consume more than the others.

Table 6 compares the area and the power consumption for the different algorithms to obtain an EVM of at least -35 dB. The power consumption in Table 6 has been normalized. (Kuo et al.,

Algorithm	**dbw**	**tbw**	**Area (mm^2)**	**P$_{norm}$**	**AP**
SDF r2^2	12	8	0.0993	0.0211	0.00210
SDF r2^3	12	7	0.0998	0.0233	0.00232
SDF r2^4	12	7	0.0960	0.0222	0.00214

Table 6. AP comparison for WLAN system with $EVM \leq -35dB$

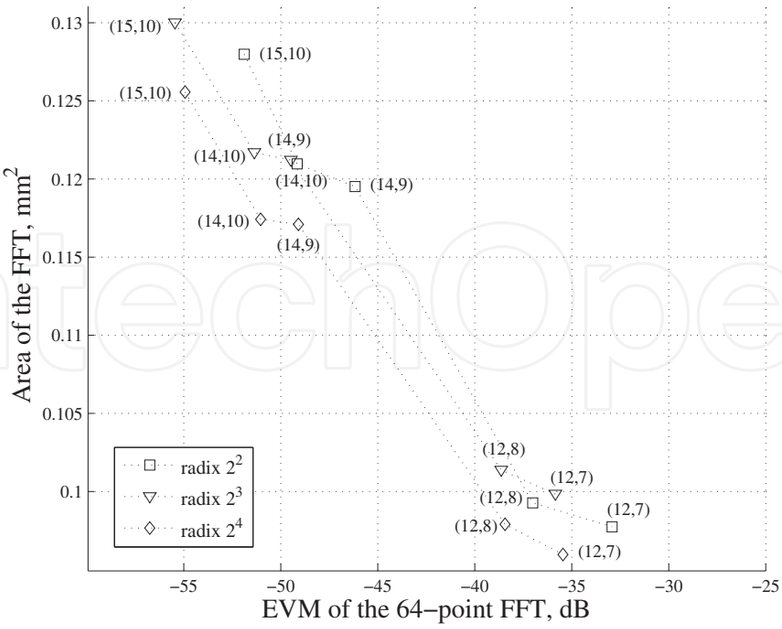


Fig. 9. EVM and area of the pipeline–SDF DIF 64–point FFT/IFFT core

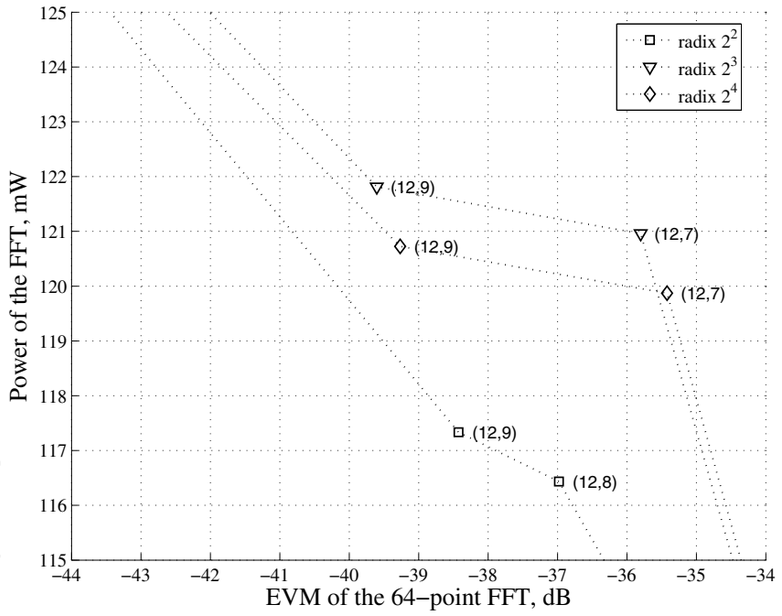


Fig. 10. EVM and power of the pipeline–SDF DIF 64–point FFT/IFFT core

2003) proposes a normalization of the power consumption as,

$$P_{norm} = \frac{P}{V_{DD}^2 \cdot N \cdot f_{clk}} \cdot 1000, \tag{32}$$

where P is the power consumption with a voltage of V_{DD} and working at f_{clk}. By slightly modifying the expression given by (Kuo et al., 2003) as follows,

$$P_{norm} = \frac{P}{V_{DD}^2 \cdot f_{clk}} \cdot 10^9. \tag{33}$$

The $r2^2$ algorithm needs larger bitwidths to achieve the target EVM. This extra bit in tbw increases the area needed. The $r2^3$ and $r2^4$ algorithms can achieve the required EVM with smaller tbw. The $r2^4$ algorithm with $dbw = 12$ and $tbw = 7$ achieves the most area–efficient implementation that fulfills the EVM specification. Nevertheless, the power consumption of the $r2^4$ algorithm is higher than the power consumption of $r2^2$ algorithm. In fact, the $r2^2$ algorithm is the most power–efficient design. In order to achieve a trade–off, the parameter $AP = Area \cdot P_{norm}$ is employed since it takes into account the area and the power consumption. The AP parameter trades off the area and power consumption of the core and, thus, it measures the efficiency of the design. For WLAN transmitter, it can be observed in Table 6 that the most efficient cores are the ones using $r2^2$ and $r2^4$ algorithms.

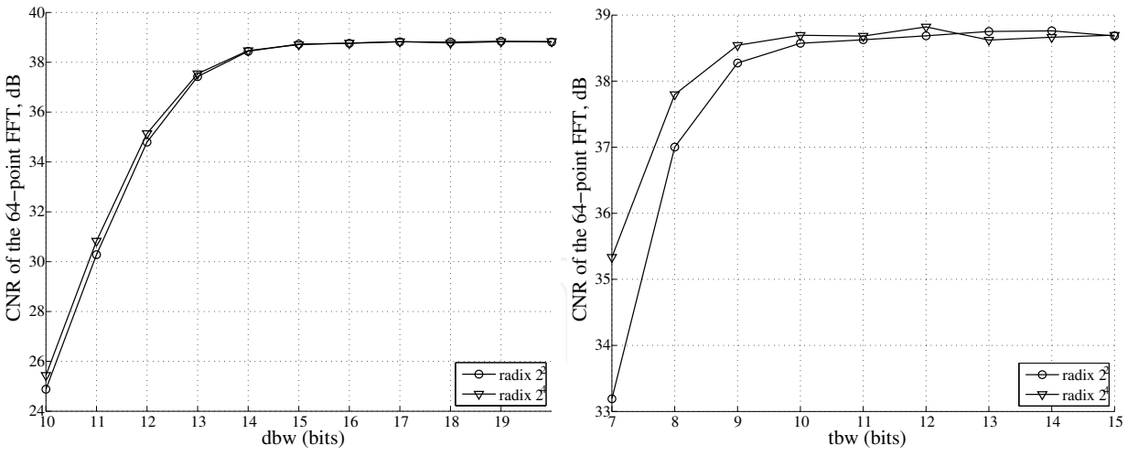
3.3.2 CNR analysis

After the EVM analysis, the $r2^3$ algorithm is discarded. Therefore, the CNR analysis focuses on the $r2^2$ and $r2^4$ algorithms. At this point, the area and power results of the FFT/IFFT core for different bitwidth configurations are already known. Then, the CNR analysis is used to select the dbw and tbw which fulfills the CNR specification.

In order to analyze the effect of the FFT quantization, the simulation model is formed by a floating–point IFFT at the transmitter and a fixed–point FFT at the receiver as is shown in Figure 8(b). First, only data are quantized. In this case, a figure of the CNR versus dbw can be used in order to select the dbw which does not degrade the CNR. Once dbw is selected, the twiddle factors are also quantized and a figure of the CNR versus tbw is shown in order to choose the tbw which does not degrade the CNR. This analysis is done for the candidates in order to determine the necessary dbw, tbw to comply with the CNR specification.

As an example, figures of $(CNR)_{dB}$ versus dbw and tbw are given for two algorithms. Figure 11(a) shows the $(CNR)_{dB}$ obtained versus the dbw parameter. In this case, the twiddle factors are not quantized. From the figure, a data bitwidth of $dbw = 15$ is selected to avoid degrading the $(CNR)_{dB}$ for both radix 2^2 and radix 2^4 algorithms. Once the data bitwidth is chosen, the twiddle factors are quantized. Figure 11(b) presents the $(CNR)_{dB}$ versus tbw where $dbw = 15$. In this case, a twiddle factor bitwidth of $tbw = 10$ is selected for both radix 2^2 and radix 2^4 algorithms. It can be observed that increasing more tbw does not improve the performance of the core.

Table 7 summarizes the (dbw, tbw) needed by the FFT to comply with the CNR requirement. Comparing the bitwidths needed by the IFFT in the transmitter to comply with the EVM and the ones needed by the FFT in the receiver to comply with the CNR, it can be said that the CNR is a much more restrictive specification. Therefore, the (dbw, tbw) selected for the FFT are the bitwidths used in the FFT/IFFT core. Table 7 presents the AP results of the FFT algorithms with the necessary bitwidths (dbw, tbw) to comply with the specifications. Taking into account the AP, the most efficient core for a WLAN system is the pipeline–SDF radix 2^2 DIF architecture.



(a) dbw (b) tbw ($dbw = 15$)

Fig. 11. Selection of dbw and tbw of the pipeline–SDF DIF 64–point FFT/IFFT core for radix 2^2 and radix 2^4 algorithms

Algorithm	**dbw**	**tbw**	**Area (mm^2)**	**P_{norm}**	**AP**
SDF r2^2	15	10	0.1284	0.0277	0.00356
SDF r2^4	15	10	0.1256	0.0287	0.00360

Table 7. AP comparison for WLAN system with $(CNR)_{dB} \geq 38.32$ dB

To sum up, the parameters of the final implementation of the pipeline–SDF radix 2^2 DIF architecture are shown in Table 8. As mentioned before, the system clock frequency is 60 MHz. Working at this frequency, the processing time of the chosen core is 0.67 μs. The data format is 2.13, whereas the format of the twiddle factors is 1.9. The EVM is -51.92 dB and the $(CNR)_{dB}$ is 38.57 dB. The estimated silicon area of the pipeline–SDF radix 2^2 DIF FFT/IFFT core is 0.1284 mm^2 and the power consumption estimation P_{norm} is 0.0277. It can be observed that the EVM complies with the specification in Table 4, and the CNR is close to the CNR_{max}.

N	**dbw**	**tbw**	**f_{clk}**	**t_{proc}**	**CNR**	**EVM**
64	15	10	60 MHz	0.67 μs	38.57 dB	-51.92 dB

Table 8. Core Parameters for the pipeline–SDF radix 2^2 DIF architecture

3.4 Layout of the FFT/IFFT core in an ASIC

Figure 12 shows the layout of the 64 complex–point FFT/IFFT fabricated in a 90 nm TSMC technology, 6–ML CMOS process. The core size is 0.1362 mm^2. In the previous section, the core area was estimated to be 0.1284 mm^2. It can be said that the area estimation is accurate enough. In order to present a comparison with the proposals found in the literature, the area of the cores is normalized as (B.M.Baas, 1999) using the equation:

$$A_{norm} = \frac{A}{(T_a/T_b)^2}, \tag{34}$$

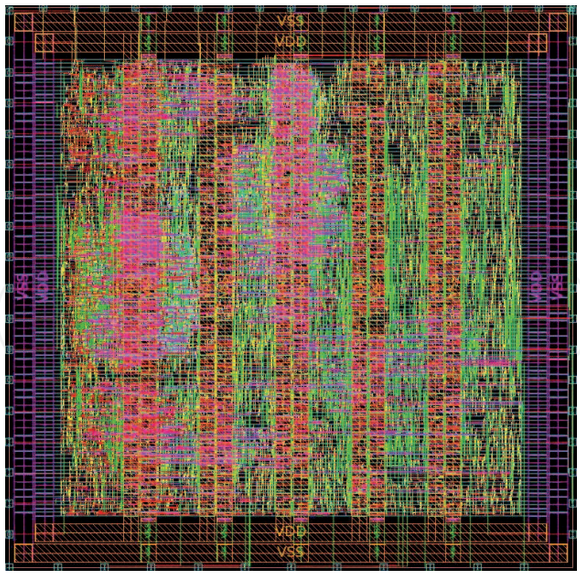


Fig. 12. Layout of the 64 complex–point FFT/IFFT fabricated in a 90 nm technology, 6–ML CMOS process. The core size is $0.370 \times 0.368\ mm^2$

.

where T_a is the anchor of the transistor of the technology actually used, A is the occupied area for T_a and T_b is the technology for which the area is normalized.

In order to assess the quality of the FFT/IFFT core, Table 9 makes a comparison with other 64 complex–point FFT/IFFT cores found in the literature. In Table 9, the area (A_{norm}) and the power (P_{norm}) have been normalized to a 90 nm technology using (34) and (33). The AP parameter indicates that our core is the most efficient one for a WLAN application. In fact, the presented core requires the smallest normalized power (P_{norm}).

	dbw	**tech.** (μ**m**)	$\mathbf{f}_{clk}$ **(MHz)**	$\mathbf{A}_{norm}$ **(mm^2)**	$\mathbf{t}_{proc}$ (μ**s**)	$\mathbf{P}_{norm}$	**AP**
(Yu et al., 2011)	16	0.18	80	0.2209	–	0.0378	0.00835
(Tsai et al., 2011)	14.8	0.09	394	0.102	–	0.091	0.00928
Proposal	15	0.09	60	0.1362	0.67	0.0277	0.00377

Table 9. Comparison of fixed–point FFT processors for WLAN systems

4. Conclusions

Many different FFT/IFFT algorithms and architectures have been proposed in the literature for OFDM systems as has been presented in Section 1. Additionally, the usual FFT notations do not facilitate to perform a general analysis for the FFT/IFFT algorithm and architecture selection.

In Section 3, a design space exploration among different algorithms has been carried out. This search is hard to perform if general expressions are not available for the different algorithms in a unified way and if a mapping to the implementation can not be easily established. In this chapter, the matricial notation summarized in Section 2 is used as a tool to help the designer

in this search. The OFDM parameters obtained from the IEEE 802.11a standard analysis have been employed as constraints for the optimization problem. The AP parameter, which trades off area and power consumption, has been used as a measure of the efficiency of the core. Finally, a pipeline–SDF radix 2^2 DIF FFT/IFFT processor has been proposed, since it achieves the minimum of the AP cost function.

To sum up, it can be concluded that there is no unique FFT/IFFT algorithm, architecture and implementation that is optimal for all OFDM systems. Therefore, it is recommended to perform a search across the algorithm, architecture and implementation dimensions for each OFDM system. The matricial notation is presented in this chapter as a unified and compact representation that can help the designer in this search. This search is feasible, and the FFT/IFFT cores implemented using this approach present a great efficiency.

5. References

Bidet, E., Castelain, D., Joanblanq, C. & Senn, P. (1995). A Fast Single-Chip Implementation of 8192 Complex Point FFT, *IEEE Journal of Solid State Circuits* 30(3): 300–305.

B.M.Baas (1999). A Low-Power, High Performance, 1024-point FFT Processor, *IEEE Journal of Solid-State Circuits* 34(3): 380–387.

Bruun, G. (1978). z-Transform DFT Filters and FFTs, *IEEE Transactions on Acoustics, Speech and Signal Processing* 26(1): 56–63.

Chang, Y.-S. & Park, S.-C. (2004). An Enhanced Memory Assignment Scheme for Memory-Based FFT Processor, *IEEE Transactions Fundamentals* E87-A(11): 3020–3024.

Cooley, J. W. & Tukey, J. (1965). An Algorithm For Machine Calculation of Complex Fourier Series, *Mathematics of Computation* 19: 297–301.

Cortés, A., Vélez, I. & Sevillano, J. F. (2009). Radix r^k FFTs: Matricial Representation and SDC/SDF Pipeline Implementation, *IEEE Transactions on Signal Processing* 57(7): 2824–2839.

Cortés, A., Vélez, I., Irizar, A. & Sevillano, J. F. (2007). Area efficient IFFT/FFT core for MB-OFDM UWB, *Electronic Letters* 43(11).

ECM (2005). *Standard ECMA-368, High rate ultra wideband PHY and MAC standard*.

Engels, M. (ed.) (2002). *Wireless OFDM Systems. How to make them work?*, Kluwer Academic.

ETS (2004). *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television*.

Good, I. J. (1958). The Interaction Algorithm and Practical Fourier Analysis, *Journal of the Royal Statistical Society* 20(2): 361–372.

He, S. & Torkelson, M. (1998). Designing Pipeline FFT Processor for OFDM (de)Modulation, *URSI International Symposium on Signals, Systems, and Electronics* 29: 257–262.

IEE (1999). *Part11: Wireless LAN Medium Access Control and Physical Layer specifications. High-speed Physical Layer in the 5 GHz Band*.

IEE (2003). *Part 16: Air interface for fixed broadband wireless access systems amendment 2: Medium Access Control modifications and additional physical layer specifications for 2-11 GHz*.

Jiang, M., Yang, B., Fu, Y., Jiang, A. & an Wang, X. (2004). Design Of FFT processor with Low Power Complex Multiplier for OFDM-based High-speed Wireless Applications, *International Symposium on Communications and Information Technologies* pp. 639–641.

Jung, Y., Yoon, H. & Kim, J. (2005). New Efficient FFT Algorithm and Pipeline Implementation Results for OFDM/DMT Applications, *IEEE Transactions on Consumer Electronics* 49(1): 14–20.

Kuo, J.-C., Wen, C.-H., Lin, C.-H. & Wu, A.-Y. (2003). VLSI Design of a Variable-Length FFT/IFFT Processor for OFDM-Based Communication Systems, *EURASIP Journal on Applied Signal Processing* pp. 1306–1316.

Lee, H.-Y. & Park, I.-C. (2007). Balanced Binary-Tree Decomposition for Area-Efficient Pipelined FFT Processing, *IEEE Transactions on Circuits and Systems-I* 54(4): 889–900.

Lee, J., Lee, H., Cho, S.-I. & Choi, S.-S. (2006). A High-speed, Low-complexity Radix-2 FFT Processor for MB-OFDM UWB Systems, *IEEE International Symposium on Circuits and Systems* pp. 4719–4722.

Lenart, T. & Owal, V. (2006). Architectures for Dynamic Data Scaling in 2/4/8k Pipeline FFT Cores, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 14(11): 1286–1290.

Lin, H.-L., Lin, H., Chen, Y.-C. & Chang, R. C. (2004). A Novel Pipelined Fast Fourier Transform Architecture for Double Rate OFDM Systems, *IEEE Workshop on Signal Processing Systems Design and Implementation* pp. 7–11.

Lin, Y.-W., Liu, H.-Y. & Lee, C.-Y. (2004). A Dynamic Scaling FFT Processor for DVB-T Applications, *IEEE Journal of Solid-State Circuits* 39(1): 2005–2013.

Lin, Y.-W., Liu, H.-Y. & Lee, C.-Y. (2005). A 1-GS/s FFT/IFFT Processor for UWB Applications, *IEEE Journal of Solid-State Circuits* 40(8): 1726–1735.

Liu, L., Ren, J., Wang, X. & Ye, F. (2007). Design of Low-Power, 1GS/s Throughput FFT Processor for MIMO-OFDM UWB Communication System, *IEEE International Symposium on Circuits and Systems* pp. 2594–2597.

Maharatna, K., Grass, E. & Jagdhold, U. (2004). A 64-Point Fourier Transform Chip for High-Speed Wireless LAN Application Using OFDM, *IEEE Journal of Solid-State Circuits* 39(3): 484–492.

Nee, R. V. & Prasad, R. (2000). *OFDM for Wireless Multimedia Communications*, Artech House.

Pease, M. (1968). An Adaptation of the Fast Fourier Transform for Parallel Processing, *Journal of the Association of Computing Machines* 15(2): 252–264.

Rader, C. M. (1968a). A Linear Filtering Approach to the Computation of the Discrete Fourier Transform, *Northeast Electronics Research and Engineering Meeting Record* 10: 218–219.

Rader, C. M. (1968b). Discrete Fourier transforms when the Number of Data Samples is Prime, *Proceedings of the IEEE* 56: 1107Ű–1108.

Rudagi, J. M., Lobo, R., Patil, P. & Biraj, N. (2010). An Efficient 64-point Pipelined FFT Engine, *2010 International Conference on Advances in Recent Technologies in Communication and Computing* pp. 204–208.

Saberinia, E. (2006). Implementation of a Multi-band Pulsed-OFDM Transceiver, *Journal of VLSI Signal Processing* 43: 73–88.

Serrá, M., Martí, P. & Carrabina, J. (2004). IFFT/FFT core architecture with an Identical Stage Structure for Wireless LAN Communications, *IEEE 5th Workshop on Signal Processing Advances in Wireless Communications* pp. 606–610.

Sevillano, J. F. (2004). *Diseño de un Demodulador QPSK para Sistemas de Radiodifusión Digital por Satélite*, PhD thesis, Campus tecnológico de la Universidad de Navarra.

Sloate, H. (1974). Matrix Representation for Sorting and the FFT, *IEEE Transactions on Circuits and Systems* CAS-21(1): 109–116.

Thomas, L. H. (1963). Using a Computer to Solve Problems in Physics, *Applications of Digital Computers* .

Tsai, P.-Y., Chen, C.-W. & Huang, M.-Y. (2011). Automatic IP Generation of FFT/IFFT Processors with Word-length Optimization for MIMO-OFDM Systems, *EURASIP Journal on Advances in Signal Processing* 2011(ID-136319).

Tsai, T. H., Peng, C. C. & Chen, T.-M. (2006). Design of a FFT/IFFT Soft IP Generator using on OFDM Communication System, *WSEAS Transactions on Circuits and Systems* 5(8): 1173–1180.

Turrillas, M., Cortés, A., Sevillano, J. F., Vélez, I., Oria, C., Irizar, A. & Baena, V. (2010). Comparison of area–efficient FFT algorithms for DVB-T2 receivers, *Electronic Letters* 46(15).

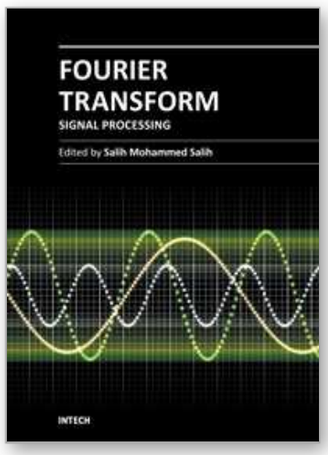
Velez, I. (2005). *Metodología de Diseño de Sistemas Digitales para Telecomunicaciones basada en C/C++: Aplicación a WLAN 802.11a*, PhD thesis, Campus tecnológico de la Universidad de Navarra.

Wang, C. C., Huang, J. M. & Cheng, H. C. (2005). A 2K/8K mode Small-Area FFT Processor for OFDM Demodulation of DVB-T Receivers, *IEEE Transactions on Consumer Electronics* pp. 28–32.

Winograd, S. (1978). On Computing the Discrete Fourier Transform, *Mathematics of Computation* 32: 175–199.

World DAB Forum (n.d.). `http://www.worlddab.org/`.

Yu, C., Yen, M.-H. & Hsiung, P.-A. (2011). A Low-Power 64-point Pipeline FFT/IFFT Processor for OFDM Applications, *IEEE Transactions on Consumer Electronics* 57(1): 40–45.



Fourier Transform - Signal Processing

Edited by Dr Salih Salih

ISBN 978-953-51-0453-7
Hard cover, 354 pages
Publisher InTech
Published online 11, April, 2012
Published in print edition April, 2012

The field of signal processing has seen explosive growth during the past decades; almost all textbooks on signal processing have a section devoted to the Fourier transform theory. For this reason, this book focuses on the Fourier transform applications in signal processing techniques. The book chapters are related to DFT, FFT, OFDM, estimation techniques and the image processing techqniques. It is hoped that this book will provide the background, references and the incentive to encourage further research and results in this area as well as provide tools for practical applications. It provides an applications-oriented to signal processing written primarily for electrical engineers, communication engineers, signal processing engineers, mathematicians and graduate students will also find it useful as a reference for their research activities.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen