International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

FPGA Implementation of Prototype Filter Bank Multicarrier Modulator

Mohammed K. Al-Haddad[1], Hadi T. Ziboon[2]

[1] *Electronics and Communications Eng. Dpt., University of Baghdad, Baghdad, Iraq*
[2]*Electrical Eng. Dpt., University of Technology, Baghdad, Iraq*

E-mail address: mohammed.alhadad@coeng.uobaghdad., 30014@uotechnology.edu.iq

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Filter Bank Multicarrier (FBMC) is an emerging modulation technique that employs pulse shaping for improved spectral properties. However, the implementation complexity of FBMC is higher than the currently adopted modulation techniques like the Orthogonal Frequency-Division Multiplexing (OFDM). In this paper, we present a hardware implementation of a prototype FBMC modulator on Field-Programmable Gate Array (FPGA) platform. The approach employs the Canonical Signed Digit (CSD) in the implementation of the array multipliers used in the shaping filter and the twiddle factors in Inverse Fast Fourier Transform (IFFT). The implementation of the IFFT follows the Split Radix (SR) algorithm and the shaping filter is implemented by the Frequency Spreading (FS) method. The overall structure of the modulator is built as a pipeline structure for higher speed performance. The system is built using Very High Speed Integrated Circuits (VHSIC) Hardware Description Language (VHDL) code that was compiled and downloaded on DE2-115 FPGA kit using Quartus II software.

Keywords: Filter Bank Multicarrier, Pulse Shaping, FFT/IFFT, FPGA, VHDL

1. INTRODUCTION

Filter Bank Multicarrier (FBMC) is a Multicarrier Modulation (MCM) considered as a candidate to replace the Orthogonal Frequency-Division Multiplexing (OFDM) in future communications [1, 2, 5]. This is because OFDM employs basic rectangular pulse shape, which leads to high spectral leakage due to the sinc shaped spectrum. In addition, OFDM requires a Cyclic Prefix (CP) in order to perform channel equalization to combat the effect of multipath channel. On the other hand, FBMC employs a nontrivial pulse shaping to the modulated carriers; this pulse shaping has the effect of improved spectral properties and helps to combat the channel's time-frequency dispersive effect without the need for the CP [1]. The pulse shapes used in FBMC like PHYsical layer for DYnamic AccesS and cognitive radio (PHYDYAS) filter [2] and Hermite filter [3] are characterized by their locality [4]. In FBMC, the modulating symbols are real symbols unlike the case in OFDM where the modulating symbols are complex Quadrature Amplitude Modulation (QAM) symbols. Therefore, the modulating symbols in FBMC can be either Pulse Amplitude Modulation (PAM) or Offset QAM (OQAM) where the real and imaginary parts of the QAM symbols are transmitted consecutively. This requires the FBMC to transmit at double the rate of the OFDM. The desired properties of the FBMC produced by the pulse shaping comes at the expense of increased complexity. Unlike OFDM, The length of the shaping filter in FBMC is longer than the signaling interval leading to overlap between the transmitted symbols. The implementation of both OFDM and FBMC have the Fast Fourier Transform (FFT) and Inverse FFT (IFFT) as the main block but the pulse shaping and symbol overlap causes increased implementation complexity in FBMC over OFDM. The pulse shaping in FBMC can be implemented either by the Polyphase Network (PPN) [1, 2] technique or by the Frequency Spreading (FS) technique, which is applicable to the PHYDYAS filter case only [2].

The computational complexity of the FBMC has attracted the attention of many researchers. Bartzoudis *et al.* [5] analyzed the computational complexity of four types of MCM: CP-OFDM, FBMC, Universal Filtered Multi-Carrier (UFMC) and Generalized Frequency Division Multiplexing (GFDM) showing that CP-OFDM has the least operations per samples and the UFMC with the highest number of operations per sample. Dandach and Siohan [6] produced an

E-mail:author's email

algorithm to reduce the amount of computation in the modulator side to half for an arbitrary length shaping-filter but did not account for the computation of the phase factor associated with the arbitrary filter length. Kollar and Al-Amaireh [7] presented complexity reduction in the transmitter by considering that the input symbols to the FBMC modulator are real symbols. Taheri *et al.* [8] suggested performing the IFFT/FFT and pulse shaping for *B* symbols block and using fast circular convolution. Although there is reduction in the amount of computations, this technique requires IFFT/FFT block to be of *NB*/2 size instead of *N*, where *N* is the total number of subcarriers. Varga and Kollar [9] presented a review of the implantation aspects of FBMC discussing the different approaches like the FS, PPN and Recursive Discrete Fourier Transform (R-DFT) methods. The requirements and complexity of each approach is evaluated in terms of number of operations and hardware requirements. Keerthana and Rajaram [10] presented FPGA implementation of FS based FBMC modulator, the results were presented in terms of speed and power consumption of the individual system blocks not for the overall system. Geevarghese and Muthusamy [11] presented a pruning algorithm for the IFFT as it is the most computationally demanding block of the FBMC. The presented algorithm offered a reduction in the number of arithmetic operations. Shaheen and Zekry [12] suggested FPGA implementation of FBMC transceiver; their work was presented on the Register Transfer Level (RTL) especially the FFT and the PPN based shaping filter. Carvalho *et al*. [13] presented a block-level FPGA implementation of FBMC modulator with FS based pulse shaping and showed that the FS method requires less resource allocation than the PPN method. Nadal *et al*. [14] presented a pipelined implementation for FBMC modulator based on PPN method that support multiple shaping filter lengths.

In this paper, a prototype FBMC modulator with pipeline structure is presented. The (IFFT) is implemented using Split Radix (SR) algorithm [15] for reduced complexity. The shaping filter is the PHYDYAS filter and it is implemented using FS method. The multipliers used in the IFFT and the shaping filter are implemented as array multipliers which are suitable for multiplication by a constant multiplicand and fast performance. The proposed structure of the FBMC modulator is presented in details and can be generalized to any number of subcarriers. The system is implemented using VHDL code for efficient performance in terms number of components (logic elements) and speed. There were no Intellectual Property (IP) cores(*), Digital Signal Processing (DSP) blocks or third party Hardware Descriptive Language (HDL) code generators used in the modulator components. This provide high degree of flexibility to the system designer when implementing the individual system components. The remaining of this paper is organized as follows: Section 2 presents a brief theoretical review of FBMC. Section 3 describes the overall system structure. Section 4 describes the details of the multipliers implementation. Section 5 describes the pipeline structure of the system. The results of the compilation report are presented in Section 6 and Section 7 presents the conclusions.

2. THEORETICAL BACKGROUND

The general expression of FBMC signal is given by

$$x(t)=\sum_{n=-\infty}^{\infty}\sum_{k=0}^{N-1}a_{n,k}\,j^{(n+k)}\,p(t-nT/2)\,e^{j2\pi k(t-nT/2)/T} \quad (1)$$

where $a_{n,k}$ is the transmitted real symbol that is either a PAM symbol or OQAM symbol, n is the symbol time index, k is the subcarrier frequency index, T is the interval of transmitting two real FBMC symbols corresponding to one complex OFDM symbols. $p(t)$ is a unit energy pulse shaping filter with duration of KT; K is the overlapping factor. The shaping filter $p(t)$ satisfies certain conditions of orthogonality necessary for symbol recovery and locality necessary to combat the time-frequency dispersive effect of the multipath channel as discussed in [2, 16]. The term j^{n+k} is a phase factor that corresponds to the overall orthogonality of $x(t)$. In discrete time, the expression of the FBMC can be rewritten by substituting $t=mt_s$ where $t_s=T/N$ is the sampling time

$$x(m)=\sum_{n=-\infty}^{\infty}\sum_{k=0}^{N-1}a_{n,k}\,j^{(n+k)}\,p(m-nM)\,e^{j2\pi k(m-nM)/N} \quad (2)$$

where $M=N/2$ represents the FBMC signaling interval in terms of number of sampling interval t_s. The recovery of the transmitted symbols can be achieved by projecting $x(m)$ onto the basis function given by

$$p_{n,k}(m)=j^{(n+k)}\,p(m-nM)\,e^{j2\pi k(m-nM)/N} \quad (3)$$

therefore

$$a_{n,k}=\mathrm{Re}\left\{\sum_{m=nM}^{nM+L-1}x(m)\,p_{n,k}^{*}(m)\right\} \quad (4)$$

The summation inside the Re{} operator produces a complex quantity, the imaginary part represent the intrinsic interference caused by the symbols overlap. The role of the phase factor $j^{(n+k)}$ is to distinguish between the value of the recovered symbol $a_{n,k}$ and the intrinsic interference due to the symbols overlap by making the value of $a_{n,k}$ to appear as a real part and the intrinsic interference as imaginary part that will be discarded by the Re{} [17]. For the implementation of (1) and (4) using the PPN method the reader may refer to [2] and [18].

The PHYDYAS shaping filter is based on the concept of frequency sampling technique. The impulse response of the PHYDYAS filter is given by [2]

$$p(t)=1+2\sum_{k=1}^{K-1}P_k\cos(2\pi k/KT)=\sum_{k=-(K-1)}^{K-1}P_k\,e^{j2\pi k/KT} \quad (5)$$

(*) Except for the PLL which acts as a clock generator with user selectable clock frequency.

It is constructed from discrete sinusoids as in (5) making it periodic with period KT. The values of P_k are given in Table I.

TABLE I. PHYDYAS FILTER COEFFICIENTS IN FREQUENCY DOMAIN

K	PHYDYAS Filter Coefficients			
	$P0$	$P_1=P_{-1}$	$P_2=P_{-2}$	$P_3=P_{-3}$
2	1	$\sqrt{2}/2$	-	-
3	1	0.911438	0.411438	-
4	1	0.971960	$\sqrt{2}/2$	0.235147

Since the PHYDYAS filter has only $2K-1$ spectral components it can be implemented using FS because it requires less multiplications than the PPN implementation. In PPN method, the shaping filter of length $L=KN$ samples is multiplied in time domain by the K repetition of the N samples of the IFFT [19]. The IFFT input symbols are

$$c_{n,k} = j^{(n+k)} a_{n,k} \tag{6}$$

Let $b_{n,m}$ be the IFFT output symbols, i.e.

$$b_{n,m} = \mathrm{IFFT}[c_{n,k}] \tag{7}$$

In the FS method, this multiplication in time domain is equivalent to convolution in frequency domain. The frequency domain of $p(t)$ in (5) has $2K-1$ samples. Define $b_{n,m}^K$ as the K repetition of a block of N samples of $b_{n,m}$ defined in (7), $b_{n,m}^K$ will be the KN samples block that will be multiplied by $p(m)$, (see Fig. 1 in [19]). From the Fourier transform properties [20], it can be shown that the FFT$[b_{n,m}^K]$ is the K-sample expanded version of the $c_{n,m}$ =FFT$[b_{n,m}]$, i.e.,

$$\left(c_{n,k}\right)_K = \mathrm{FFT}[b_{n,m}^K] = \begin{cases} c_{n,k/K} & k \bmod K = 0 \\ 0 & k \bmod K \neq 0 \end{cases} \tag{8}$$

For $0 \le k \le L-1$, the n^{th} L-length FBMC symbol can be written as

$$s_{n,m} = \mathrm{IFFT}\left[\left(c_{n,k}\right)_K \underset{NK}{*} P_k\right] \tag{9}$$

where the symbol $\underset{NK}{*}$ denotes the NK-point circular convolution. The term FS comes from (9) where every sample of $c_{n,k}$ is spread to $2K-1$ samples with overlapping of adjacent spread samples as illustrated in Fig. 1 [2].

The FS demodulator will be in the same manner but in reverse order, i.e., the KN-Point FFT and later the circular convolution with P_k coefficients as illustrated in Fig. 2.

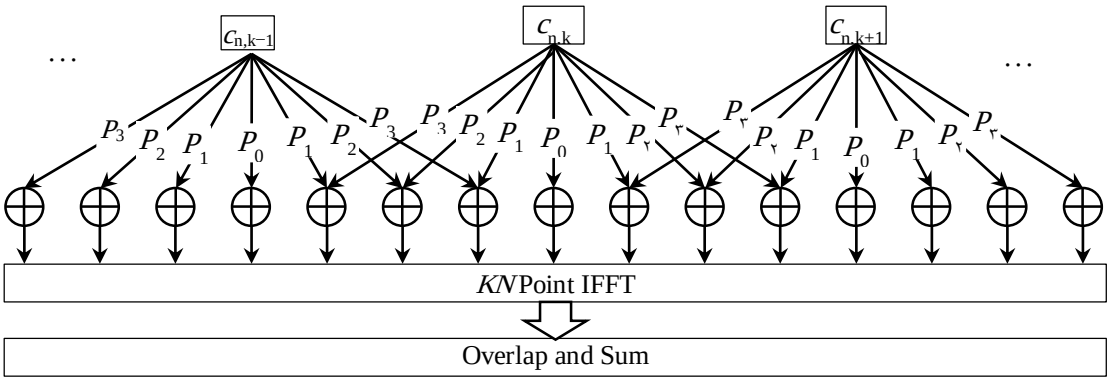


Figure 1. FS based FBMC modulator.

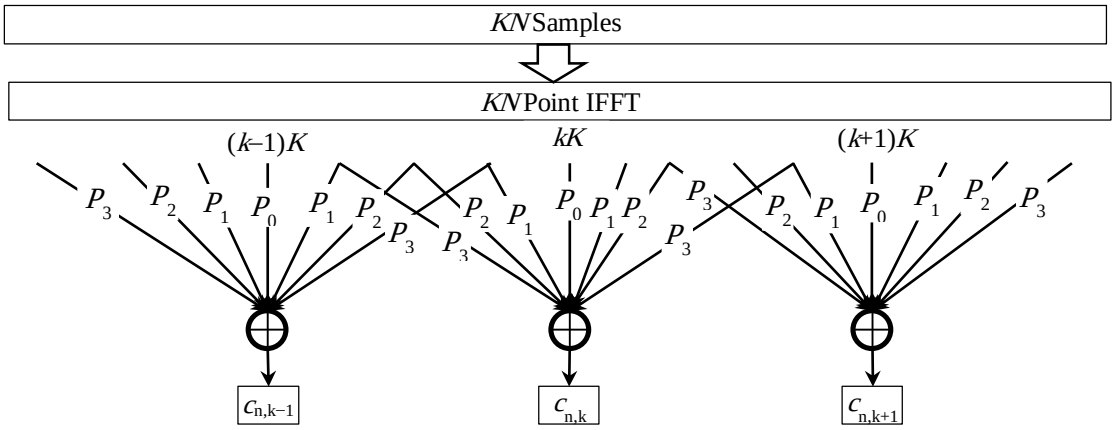


Figure 2. FS based FBMC demodulator.

3. System Description of the Prototype Modulator

The prototype FBMC modulator is implemented using the parameters N=16 and K=4, therefore, the number of IFFT points using FS method is L=NK=64. SR algorithm is used to implement the IFFT for reduced complexity. The system is structures as pipeline architecture including the internal stages of the IFFT. The system is implemented by VHDL to program the FPGA kit DE2-115. Fig. 3 shows the main components of the system which are described below.

A. The Timing Circuit

The main component of the timing circuit, shown in Fig. 4, is a 7-bit timer called the clock generator, in the general case the number of bits is $1+\log_2(NK)$. The input to the clock generator is a clock that we refer to as the system clock. The system clock is derived from the existing 50MHz board clock of the DE2-115 kit. The clock frequency can be scaled up or down, according to the final system design requirements, by a built-in Phase Locked Loop (PLL) circuit with 4 output clocks. The PLL outputs are predefined in the VHDL code and can be selected by the user through the multiplexer select inputs. The multiplexer's output will be the system clock driving the clock generator. The outputs of the timing circuit are

1- count_{0_5}: This is the lowest 6 bits of the clock generator used for addressing the L=64 samples after spreading.

2- count_{2_5}: This is the highest 4 bits of count_{0_5} used to address the N=16 symbols before spreading.

3- c_5: This bit is used as a clocking signal between successive pipeline stages.

4- c_6: This bit is used as a select signal for even/odd values of the time index n. It controls the loading of the input symbols in the real/odd part of the input register.

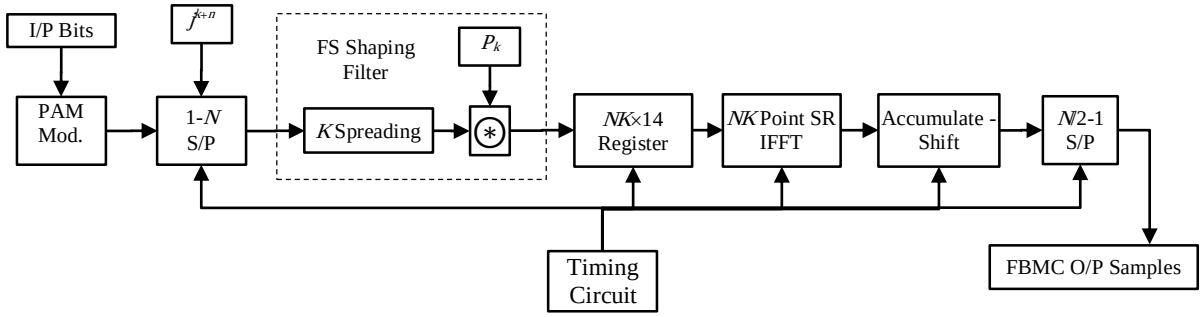


Figure 3. A block diagram of the main components of the FBMC prototype modulator.

(*) Except for the PLL which acts as a clock generator with user selectable clock frequency.

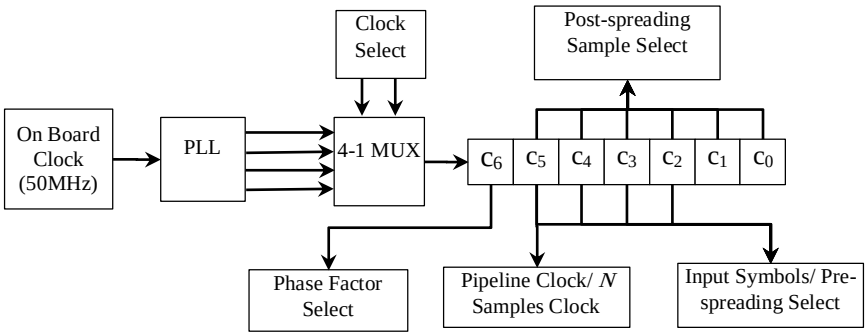


Figure 4. The Timing Circuit of the FBMC modulator.

B. The PAM Modulator

The 8PAM modulation is used to modulate input data bits. This is equivalent to 64QAM modulator in OFDM system. It is implemented as a simple look-up table that maps the input of 3 bits, representing unsigned integer, into the corresponding value of the 8PAM modulator. The output is a signed integer whose value is within the range of –7 to 7, therefore the 8PAM modulator output symbols have 4 bits and represent the input data symbols to the FBMC modulator.

C. Serial-to-Parallel Converter and Phase Factor

The serial input data symbols are mapped to a register for complex values of size $2N\times4$ (N=16). The symbols are mapped alternatingly to the real part register and the imaginary part register to account for the phase factor j^{n+k}, where the sum $n+k$ is performed as mod2 addition. When the value of n is even (c_6=0), the even indexed symbols are loaded into the even addresses of the real register and 0's are loaded to the odd addresses; the odd indexes symbols are loaded into the odd indexed addresses of the imaginary register and 0's are loaded in the even addresses. When the value of n is odd (c_6=1) the even indexed symbols are loaded into the even addresses of the imaginary register and 0's are loaded into the odd addresses; the odd indexed symbols are loaded into the odd indexed addresses of the real register and 0's are loaded into the even addresses; this is illustrated by Table II.

TABLE II. MAPPING OF THE SERIAL-TO-PARALLEL AND THE PHASE FACTOR TO THE INPUT REGISTER

k	Symbol	Even n (c_6=0)		Odd n (c_6=1)	
		Re.	*Im.*	*Re.*	*Im.*
0	b_0	b_0	0	0	b_0
1	b_1	0	b_1	b_1	0
2	b_2	b_2	0	0	b_2
3	b_3	0	b_3	b_3	0
4	b_4	b_4	0	0	b_4
5	b_5	0	b_5	b_5	0
6	b_6	b_6	0	0	b_6
7	b_7	0	b_7	b_7	0

D. Spreading and Convolution

According to the FS technique, the input symbols are K-times spread by inserting (K–1) 0's after each of the N symbols. The convolution is performed by multiplying each symbol by the coefficients P_k for $1\le k\le 2K-1$. The actual number of multiplications is K–1 because P_0=1 and the remaining $2K$–2 coefficients are pairs of equal values. The convolution of the spread symbols with the coefficients P_k is illustrated in Fig. 1 and can be expressed as

$$x_{4k+l}=\begin{cases} d_k & l=0 \\ P_l d_k + P_{K-l} d_{k+1} & 1\le l\le K-1 \end{cases} \quad 0\le k\le N-1 \qquad (10)$$

The multiplication is implemented with 13 bits precision; 12 bits for magnitude and 1 bit for sign. The addition is performed for only two overlapped samples and it may result in a carry bit. Therefore, the result of addition is one bit more than the input operands, i.e., the output of this stage has 14 bits precision. Since the multipliers and adders have different propagation delay, the output will not be valid synchronously, therefore, the outputs are loaded into $L\times14$ bits register by the pipeline clock c_5 making this register the output of the first stage of the pipeline. The following stages of the pipeline are inside the IFFT block as will be explained next.

E. The IFFT Block

The IFFT is the core of any multicarrier modulator like FBMC; in our case where the FS technique is adopted, the number of additions and multiplications is proportional to $L\log_2(L)$. Therefore, we will discuss the implementation of the IFFT block in more details. The SR algorithm is used in the implementation of the IFFT. The main calculations are the multiplications by the twiddle factors W_L^k defined by

$$W_L^k = e^{j2\pi k/L} = C_k + jS_k \quad (11)$$

Multiplying a twiddle factor W^k, where the subscript L is dropped for simplicity, by a complex number $Z=R+jX$ can be expressed as

$$W^k Z = (C_k - S_k)X + C_k(R-X) + j\left((C_k + S_k)R - C_k(R-X)\right) \quad (12)$$

In (12) the complex multiplication is performed by 3 real multiplications because the term $C_k(R–X)$ is repeated unlike the conventional way that requires 4 real multiplications. The number of real additions is also 3 because the terms $(C_k–S_k)$ and (C_k+S_k) are constants and the term $(R–X)$ is repeated.

The SR-FFT is a combination of radix 2 and radix 4 algorithms [15]. The computation of the even indexed points is made by radix 2 approach while the computation of the odd indexed points is made by radix 4 approach.

The butterfly takes an L-shape because the odd indexed points are calculated two stages ahead as shown in Fig. 5. The value of L in Fig. 5 is the size of the butterfly, which is equal to the FFT points at the first stage only and it is reduced by half as the butterfly progresses further in the next stages. The last stage is completed by all 2-points butterflies. The multipliers of the twiddle factors are designed with 16 bits precision but can accept multiplicand with any precision. The precision of the multiplications and additions is increase by one bit for every stage of the IFFT to account for the carry due to the addition and subtraction.

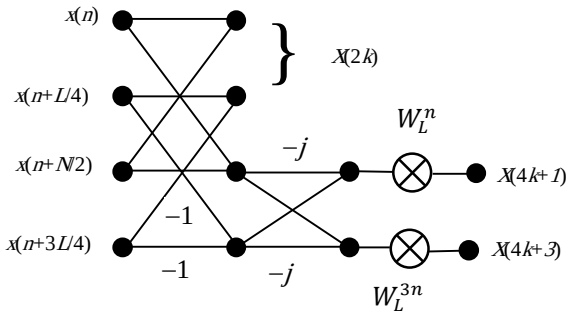


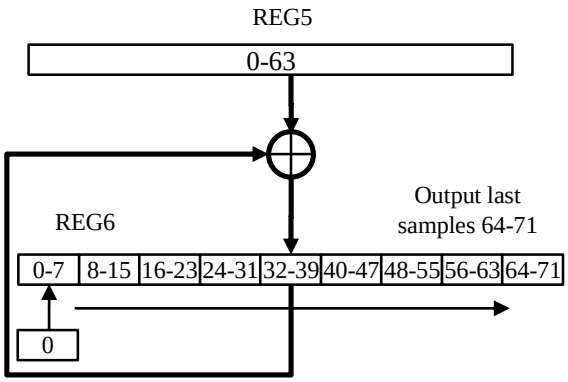
Figure 5. The butterfly structure of SR FFT.

F. Accumulate-Shift

The output of the IFFT block will be the individual FBMC samples each of length L=64 samples that will be loaded to a shift register that acts as an accumulator to account for the overlapping of the output FBMC symbols. The shifting is performed every $M=N/2$=8 samples as shown in Fig. 6.

1. Implementation of the Shaping Filter and the Twiddle Factors Multipliers

There are two general methods for multipliers implementations; both of them use the shift and add operations. The first method is the register-based multiplier; this method is more suitable for multiplication of two variable operands but it has higher delay because every shift operation requires a clock cycle, therefore it is also called sequential multiplier [21]. The second method is the array multiplier, in this method, one of the operands (the multiplier) is constant and the second operand (the multiplicand) is variable. The multiplier operand is hardwired by the multiplier's logic gates and the multiplicand is the only input to the multiplier.



REG6(8-71)=REG6(0-63)+REG5
REG6(0-7)=0

Figure 6. The accumulate and shfit register.

The method of Array Constant Multiplier (ACM) is faster than the register-based multiplier because the multiplier's delay is the propagation delay only and no clock is needed but at the expense of more logic gates than the register-based method. This method is common in DSP applications like FFT/IFFT and digital filters [22]. Beside the ACM, we will use the Canonical Signed Digit (CSD) [22, 23] in the representation of the multiplier instead of the conventional binary representation. The CSD is a ternary system where the digits take the values of 0 and ±1, the –1 is denoted by $\bar{1}$. The CSD is useful when there is a consecutive 1's in the binary representation of a number because a number of n consecutive 1's equals 2^n–1 which can be represented by 2 nonzero CSD digits. For example $(1111)_B=(1000\bar{1})_C$, where the subscripts B and C refer to the binary and CSD representations. This has the effect of reducing the number of 1's in the representation of the multiplier and hence reducing the number of adders used in the ACM implementation. The algorithm for converting a binary number to CSD can be described as:

1- Locate 3 or more consecutive 1's starting from the least significant bit.

2- Replace the least significant 1 by $\bar{1}$, the remaining 1's by 0's and the 0 to the left of the original 1's by1.

(*) Except for the PLL which acts as a clock generator with user selectable clock frequency.

3- Converting $(\ldots 11\ldots)_B$ to $(\ldots 10\bar{1}\ldots)_C$ does not reduce the number of nonzero digits but the pattern $(\ldots 0110110\ldots)_B$ can be converted to $(\ldots 100\bar{1}0\bar{1}\ldots)_C$ by applying CSD to the right 11 and to produce $0111\bar{1}0\bar{1}0$.

4- Repeat 1-3 to the left until all bits are completed.

For example, a 10-bit representation of a multiplier a=0.71 in binary is $(1011010111)_B$ and in CSD is $(1100\bar{1}0\bar{1}00\bar{1})_C$, note that the number of 1's in the binary representation is 7 while in CSD the number of ±1's is 5. In binary ACM realization of the multiplier a is obtained by shift-and-add of the multiplicand, say b, for each corresponding 1 in the binary representation of the multiplier a. While in CSD ACM realization the shift and add of b is used for each 1 and the shift and add of the 2's complement of b for each $\bar{1}$, in total there will be less shift and add in case of the CSD realization.

From (11), it can be seen that the multiplication by the twiddle factor requires the real multiplication by the cosine and sine of $2\pi k/L$, where L is the number of IFFT points and $0 \le k \le L/2$. The realization of (12) requires three ACM multipliers one for cosine and the other two are of the sum and difference of sine and cosine. In order to reduce the design effort, the creation of the ACM's C_k, (C_k+S_k) and (C_k-S_k) was made for $1 \le k \le L/4 \le 1$, and the symmetry of the cosine and sine was used for higher values of k. All created ACM's are for positive values including (C_k-S_k), and when a negative value is needed for (C_k-S_k), the corresponding sign in (12) is changed accordingly. The values of k=0 and $k=L/4$ produce the so called trivial multipliers of 1 and j which does not correspond to an ACM. The value of $k=L/8$ is also a special case where $C_{L/8}=S_{L/8}$ and (12) is used in its reduced form

$$W^{L/8}Z=(C_{L/8}+jC_{L/8})(R+jX)=C_{L/8}(R-X)+jC_{L/8}(X+R) \quad (13)$$

In this case, one multiplier is designed and instantiated twice.

2. THE PIPELINE STRUCTURE

One way of IFFT implementation is to use in-place calculations to reduce the sizes of memory. In this approach, the IFFT block will not be able to process new input symbols until the previous input symbols are completely processed through all the stages and the output is passed to the next step. This approach saves memory size at the expense of speed, because the rate of the input data symbols will be limited by the speed of the overall FFT/IFFT block. Alternatively, a pipeline structure can by employed such that the output of each stage is composed of registers of size L and a corresponding combinational logic circuit representing the twiddle factors and shaping filter. The loading of each set of registers is controlled by a common timing clock, which is c_5 in this case. The period of c_5 should be chosen to be larger than the highest delay caused by any of the pipeline stages.

In this approach, the rate of the input data symbols will be higher than the in-place calculations approach because the delay will be for one stage only instead of all the stages. Fig. 7 shows a simplified diagram illustrating the data flow in the pipeline structure. There are 7 register stages in the implemented prototype.

1- REG0 (4×16): In this register, the output of the 8PAM modulator representing the real symbols are serially loaded using the address lines c_2-c_5 taking into account the phase factor. The stored values are the inputs to the shaping filter stage.

2- REG1 (14×64): In this register, the output of the shaping filter multipliers and adders are loaded in parallel using the clock c_5. The stored values are the inputs to stage 0 of the IFFT.

3- REG2 (17×64): Since stage 0 of the IFFT has only adders and subtractors with propagation delay much less than the propagation delay of the twiddle factors used in stage 1, these two stages are combine in one stage of combinational circuit and the output is loaded in REG2. The stored values are the inputs to stage 2 of the IFFT.

4- REG3 (18×64): In this register, the output of the stage 2 of the IFFT is loaded. The stored values are the inputs to stage 3 of the IFFT.

5- REG4 (19×64): In this register, the output of the stage 3 of the IFFT is loaded. The stored values are the inputs to stage 4 of the IFFT.

6- REG5 (21×64): Stage 4 and 5 of the IFFT is combined in one combinational circuit because stage 5 has only adders and subtractors. Since stage 5 is the last stage in the IFFT, the bit reversal is considered in the loading of its outputs in REG5.

7- REG6 (22×64): This register is used for the post-IFFT block, which is overlap and add as illustrated in Fig. 6.

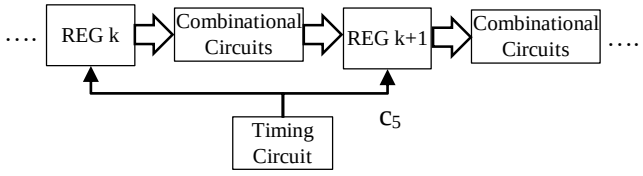


Figure 7. The pipeline structure of the of the implemented prototype.

The number of bits of the registers is increased for each stage to account for the carry that arises from the additions and subtractions. It is possible to truncate the least significant bits in the intermediate stages to reduce the complexity but this would increase the quantization noise that can propagate through computation of the successive stages. At the final stage of REG6 it is possible to truncate the least significant bits and output the desired number of bits which is 16 in our case.

3. RESULTS AND COMPILATION REPORTS

The FBMC prototype modulator described in the previous section was entirely implemented using VHDL code and compiled by Quartus II version 14.0. The compilation report is shown in Table III.

TABLE III. COMPILATION REPORT AND POWER CONSUMPTION OF FBMC PROTOTYPE MODULATOR.

Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition
Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition
Revision Name	FBMC
Top-level Entity Name	FBMC
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	49,914/114,480 (44%)
Total combinational functions	49,258/114,480 (43%)
Dedicated logic registers	13,364/114,480 (12%)
Total registers	13364
Total pins	35/529 (7%)
Total virtual pins	0
Total memory bits	0/3,981,312 (0%)
Embedded Multiplier 9-bit elements	0/532 (0%)
Total PLLs	1/4 (25%)
Dynamic Power Consumption	20.59 mW

The system speed can be analyzed in terms of the pipeline clock c_5, which directly affect the bit rate. Since N symbols are passed for every clock cycle of c_5 where each symbol is carrying N_b bits/symbol, the bit rate R can be expressed in terms frequency f_{pl} of the pipeline clock c_5

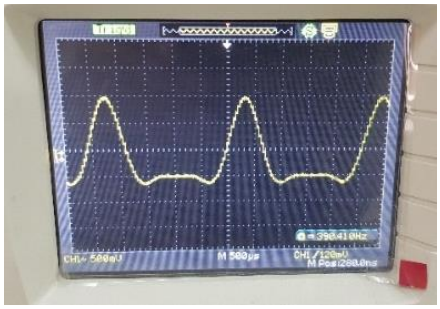
$$R=N_bNf_{pl} \tag{14}$$

Since there are 7 registers in the modulator each is loaded every clock period of c_5, therefore, the latency will be 7 clock periods of c5 or 7L clock periods of the system clock frequency, hence, the modulator latency can calculated as $7/f_{pl}$. According to the chosen system parameters, N=16 and N_b=3, the Time Quest Analysis provides the maximum frequency of c_5 for different hardware specifications. The maximum frequency of c_5 and the corresponding bit rate is shown in Table IV.

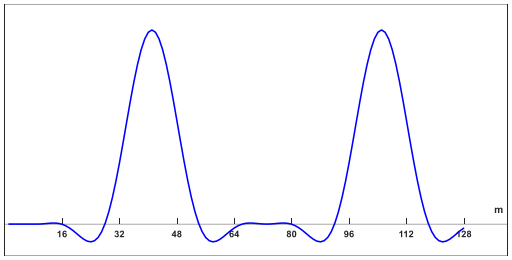
TABLE IV. THE MAXIMUM FREQUENCY AND BIT RATE SPECIFICATIONS OF THE IMPLEMENTED SYSTEM

Hardware Specifications	f_{max} (MHz)	R_{max} (Gbps)	Latency (ns)
Slow 1200mV 85°C	37.64	1.8	186
Slow 1200mV 0°C	41.79	2	168
Fast 1200mV 85°C	76.62	3.68	91.4

Fig. 8 and Fig. 9 show a comparison between the FBMC waveforms generated by the implemented modulator and the waveforms generated by a Matlab code. The figure shows clear resemblance between the waveforms generated by the hardware modulator and the waveforms generated by software.

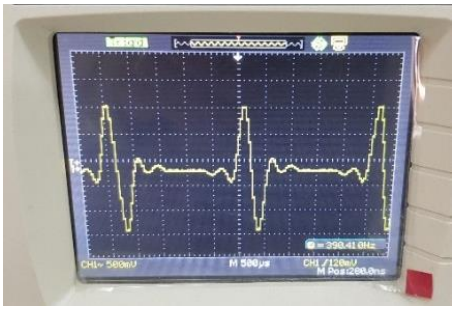


(a)

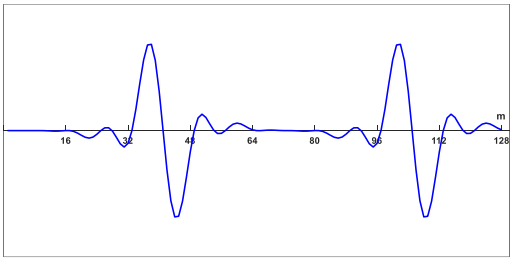


(b)

Figure 8. The 0th FBMC subcarrier generated by (a) FPGA, (b) Matlab



(a)



(b)

Figure 9. The 1st FBMC subcarrier generated by (a) FPGA, (b) Matlab

(*) Except for the PLL which acts as a clock generator with user selectable clock frequency.

4. CONCLUSIONS

In this paper, A FBMC modulator is implemented on FPGA kit DE2-115 using VHDL code. The PHYDYAS shaping filter is used for the FBMC and it is realized by FS method. The number of subcarriers is 16 and the symbol modulation level is 8PAM. The modulator is built as a pipeline architecture, the timing analysis shows that the pipeline clock can operate at 37.6MHz to 76.62MHz, depending on the hardware specifications, which represents the symbol rate of each subcarrier making the modulator to operate at bit rate of 1.8Gbps to 3.68Gbps with system latency of 91.4ns to 189ns. All of the modulator components are built as VHDL entities and no built-in components or intellectual property cores are used. The modulator can be easily extended to higher number of subcarriers and can be easily modified to build a FBMC demodulator as it contains the same components as the modulator.

REFERENCES

[1] R. Nissel, S. Schwarz, and M. Rupp "Filter Bank Multicarrier Modulation Schemes for Future Mobile Communications," IEEE Journal on Selected Areas in Communications Vol. 35, No. 8, 2017.

[2] M. Bellanger, "FBMC physical layer: A primer," PHYDYAS FP7 Project Document, January 2010.

[3] R. Haas and J. Belfiore, "A time-frequency well-localized pulse for multiple carrier transmission," Wireless Personal Communication, Vol. 5, No. 1, pp. 1–18, July 1997.

[4] B. Boashash, Time-Frequency Signal Analysis, and Processing: A Comprehensive Reference, Elsevier Science, 2nd Ed., 2015.

[5] N. Bartzoudis, V. Berg, L. G. Baltar, O. Font, K. Roth, M. Payaró, M. Färber, "Complexity and Implementation Aspects of Filter Bank Multicarrier. A Potential Technology Enabler of Next Generation Radio Systems," ETSI Workshop on Future Radio Technologies – Air Interfaces, 2016, France.

[6] Y. Dandach and P. Siohan, "FBMC/OQAM Modulators with Half Complexity," IEEE Global Telecommunications Conference - GLOBECOM 2011, Nepal.

[7] Z. Kollar and H. Al-Amaireh, "FBMC Transmitters with Reduced Complexity", Radioengineering, Vol. 27, No. 4, December 2018.

[8] S. Taheri, M. Ghoraishi, P. Xiao, and L. Zhang, "Efficient Implementation of Filter Bank Multicarrier Systems Using Circular Fast Convolution," IEEE Open Access, Volume 5, 2017.

[9] L. Varga and Z. Kollar, "Low Complexity FBMC Transceiver for FPGA Implementation," IEEE 23rd International Conference Radioelektronika (RADIOELEKTRONIKA), 2013, Pardubice, Czech Republic.

[10] R. Keerthana and S. Rajaram, "FPGA implementation of FBMC baseband modulator for 5G wireless communication," 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, Kerala, India, India

[11] A. C. Geevarghese and M. Muthusamy, "FPGA implementation of IFFT architecture with enhanced pruning algorithm for low power application," Esevier Microprocessors and Microsystems Vol. 71, Nov. 2019.

[12] I. A. Shaheen and A. Zekry, "Design and Implementation of FBMC/OQAM Transceiver for 5G Wireless Communication system," IEEE International Conference on Promising Electronic Technologies (ICPET), 2019, Gaza, Palestine.

[13] M. Carvalho, M. Lopes Ferreira and J. C. Ferreira "FPGA-based Implementation of a Frequency Spreading FBMC-OQAM Baseband Modulator," IEEE 24th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2017, Batumi, Georgia.

[14] J. Nadal, C. Abdel Nour, and A. Baghdadi, "Low-Complexity Pipelined Architecture for FBMC/OQAM Transmitter," IEEE Transactions on Circuits and Systems—II: Express Briefs, Vol. 63, No. 1, January 2016.

[15] P. Duhamel and H. Hollmann, "Implementation of Split-Radix FFT Algorithms for Complex, Real, and Real Symmetric Data," in ICASSP'85, vol. 10, Apr. 1985.

[16] J. Du, and S. Signell "Time Frequency Localization of Pulse Shaping Filters in OFDM/OQAM Systems," IEEE 6th International Conference on Information, Communications & Signal Processing, Singapore, pp. 1-5, 2007.

[17] J. Du and S. Signell, "Classic OFDM Systems and Pulse Shaping OFDM/OQAM Systems," Technical Report (KTH - Royal Institute of Technology), pp. 1–32, Feb. 2007.

[18] P. Siohan, C. Siclet, and N. Lacaille, "Analysis and Design of OFDM/OQAM Systems Based on Filter Bank Theory," IEEE Transactions on Signal Processing, Vol. 50, No. 5, pp. 1170–1183, May 2002.

[19] L. Li, Y. Wang, L. Ding, "Time Synchronization Sequence with Weighted Conjugate Symmetry Property for FBMC-OQAM Systems," IEEE MILCOM, 2018, Oct., CA, USA.

[20] J. G. Proakis, Dimitris K Manolakis, *Digital Signal Processing* 4th Ed. Prentice-Hall, 2006.

[21] M. Lu, Arithmetic and Logic in Computer Systems, 2004, John Wiley & Sons.

[22] S. A. Khan, Digital Design of Signal Processing Systems a Practical Approach, 2011 John Wiley & Sons.

[23] G. K. Kumar and B. Narayanam, "Low Power Implementation of FIR Filter for De-Noising the EOG Signal," International Journal of Computing and Digital Systems, Vol. 9, No. 5, 2020.

Mohammed K. Al-Haddad is a faculty member of the Electronics and Communication Engineering Department in the University of Baghdad. He received his M. Sc. Degree in Electronics and Communications from the University of Baghdad in 1998. His research interests are digital signal processing, multicarrier communications and channel coding.

Prof. Dr. Hadi T. Ziboon is a faculty member of the Electrical Engineering Department in the University of Technology in Iraq. He received his Ph. D. degree in electronics and communications from the University of Aston in UK in 1984. His research interests are mobile communications, radar, digital signal processing, cognitive radio and software defined radio.

(*) Except for the PLL which acts as a clock generator with user selectable clock frequency.