ISSN: 0975-8585

Research Journal of Pharmaceutical, Biological and Chemical Sciences

Radix 4 and Radix 8 Butterfly Units for OFDM applications.

Sharmila Hemanandh[1*], and A Sivasubramanian[2]

[1]Department of Electronics and Communication Engineering, Sathyabama University, Chennai-600119, India.
[2] School of Electronics, VIT University, Chennai – 600 127, India

ABSTRACT

Fast Fourier Transform (FFT) is the basic building block of (Orthogonal Frequency Division Multiplexing) OFDM system. The overall efficiency of the OFDM system depends on the rate at which significant volume of data is processed by the FFT. This paper proposes a customized floating point butterfly unit used in the design of FFT which is the arithmetic kernel of OFDM system. Customized floating point data format with reduced number of bits compared to the IEEE 754 standard can be used. This in turn reduces the overall power and area of the system. The dynamic range offered by the proposed floating point format satisfies the dynamic range requirements of 16QAM and 64QAM. This paper focuses on the development of customized floating point unit. Then the results of the conventional radix 4 and radix 8 butterfly units are compared with the butterfly units designed using the proposed fused floating point unit. Considerable saving in area is achieved in the proposed radix 4 and radix 8 butterfly units.

Keywords: 16QAM, 64QAM, Customized floating point unit, FFT, OFDM

**Corresponding author*

INTRODUCTION

Fast Fourier Transform reduces the arithmetic complexity in calculating the discrete fourier transform (DFT)[1]. FFT is one of the basic algorithms widely used in signal processing, industrial automation, scientific computing and biomedical systems. High speed data transmission is a basic necessity in wireless broad band communication systems. IFFT/FFT is used to perform modulation/demodulation in OFDM systems. The overall performance of the OFDM system mainly depends on IFFT/ FFT block [2]. Moreover, the type of arithmetic used plays an important role in determining the computational complexity of the FFT.

Applications such as multimedia and terrestrial video broadcasting use data modulation schemes like BPSK, QPSK, 16-QAM, 64-QAM, and 256 QAM [3]. The input digital data are converted to complex signals and then mapped on to the channel using IFFT block. The complex signals are represented using 32 bits with 16 bits for real part and 16 bits for imaginary part. The IEEE 754 standard specifies single precision (32 bit) representation [4] where 8 bits are used to represent exponent and 23 bits are used to represent the mantissa. The dynamic range offered by 32 bit representation is very high in the order of 1.4 x 10^-45 to 3.4 x 10^38. Accuracy can still be maintained with reduced bit width [5]. This paper proposes a highly efficient fused floating point unit with 5 bits for exponent and 10 bits for mantissa. Radix 4 and radix 8 FFT algorithms are implemented using the proposed floating point unit to measure the complexity reduction when compared to the conventional single precision and fixed point units.

FFT ALGORITHM

The N point DFT of a signal x(n) is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \qquad 0 \le k \le N-1$$

$$W_N = e^{-j2\pi/N}$$

FFT is a computationally efficient algorithm [6] which exploits the symmetry and periodicity properties of the twiddle factor.

$$\text{Symmetry property}: W_N^{k+N/2} = -W_N^k$$

$$\text{Periodicit y property}: W_N^{k+N} = W_N^k$$

The most popular radix 2 method of computing the FFT algorithm was proposed by Cooley and Turkey. In radix 2 algorithm the length of the sequence is always expressed in powers of 2. Based on divide and conquer approach many algorithms such as radix 4, radix 8, and split radix are developed to reduce the computational complexity.

RADIX 4 FFT ALGORITHM

The computational complexity of radix 2 algorithm can be reduced further by using radix 4 algorithm. Radix 4 algorithm [7] exploits the fact that the multiplication of the input with the twiddle factors especially +j and –j is carried out without the use of a complex multiplier. In radix 4 algorithm the N point DFT is divided into four N/4 point DFT. The basic butterfly diagram of radix 4 decimation in frequency FFT is shown in Fig.1. The radix 4 butterfly has 4 inputs x(n), x(n+N/4), x(n + N/2), and x(n + 3N/4). The N point DFT is computed as the sum of the outputs of the four N/4 point DFTs. The four N/4 point DFTs together represent the N point DFT. Radix 4 algorithm requires only 75% of the complex multiplication utilized in the computation of radix 2 algorithm [8].

The equations for the radix 4 algorithm is derived as follows

$$X(4k) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + x\left(n+\frac{N}{4}\right) + x\left(n+\frac{N}{2}\right) + x\left(n+\frac{3N}{4}\right)\right] W_{\frac{N}{4}}^{nk}$$

$$X(4k+1) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) - jx\left(n+\frac{N}{4}\right) - x\left(n+\frac{N}{2}\right) + jx\left(n+\frac{3N}{4}\right)\right] W_{\frac{N}{4}}^{nk} W_N^{nk}$$

$$X(4k+2) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) - x\left(n+\frac{N}{4}\right) + x\left(n+\frac{N}{2}\right) - x\left(n+\frac{3N}{4}\right)\right] W_{\frac{N}{4}}^{nk} W_N^{2nk}$$

$$X(4k+3) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + jx\left(n+\frac{N}{4}\right) - x\left(n+\frac{N}{2}\right) - jx\left(n+\frac{3N}{4}\right)\right] W_{\frac{N}{4}}^{nk} W_N^{3nk}$$

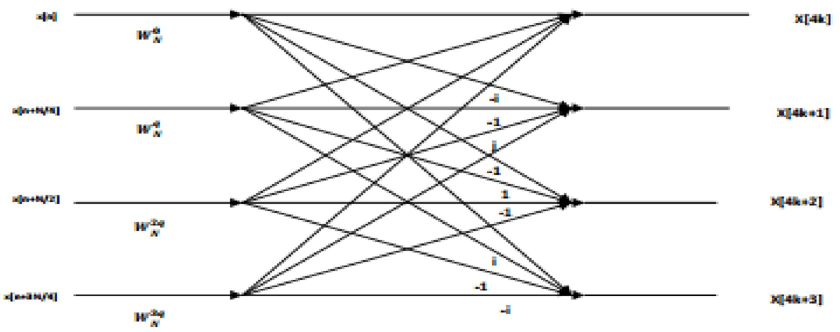

Fig 1- Radix 4 Butterfly Unit

RADIX 8 ALGORITHM

Radix 8 algorithms further reduce the computational complexity of radix 2 and radix 4 algorithms. The basic butterfly diagram of radix 8 algorithm has 8 inputs. The radix 8 algorithm [9] rearranges the N point DFT into eight N/8 point DFTs. The output of the N/8 point FFT is reused which greatly reduces the computational complexity of the algorithm. The output of the butterfly is computed as the sum of the outputs of all the N/8 point DFTs. Fig.2 shows the butterfly diagram of radix 8 algorithm. The equations for radix 8 algorithm are derived as

$$X(n) = \sum_{n=0}^{\frac{N}{8}} [[(x(n) + x\left(n+\frac{N}{2}\right)) + \left(x\left(n+\frac{N}{4}\right) + x\left(n+\frac{3N}{4}\right)\right)] + [(x\left(n+\frac{N}{8}\right) + x(n + \frac{5N}{8})) + (x\left(n+\frac{3N}{8}\right) + x(n+\frac{7N}{8}))]]W^{8nk}W^{0}$$

$$X(n+N/8) = \sum_{n=0}^{\frac{N}{8}} [[\left(x(n) - x\left(n+\frac{N}{2}\right)\right) - j\left(x\left(n+\frac{N}{4}\right) - x\left(n+\frac{3N}{4}\right)\right)] + W^{N/8}[(x\left(n + \frac{N}{8}\right) - x(n+\frac{5N}{8})) - j(x\left(n+\frac{3N}{8}\right) - x(n+\frac{7N}{8}))]]W^{8nk}W^{q}$$

$$X(n+N/4) = \sum_{n=0}^{\frac{N}{8}} [\left[\left(x(n) + x\left(n+\frac{N}{2}\right)\right) - \left(x\left(n+\frac{N}{4}\right) + x\left(n+\frac{3N}{4}\right)\right)\right] - j[(x\left(n+\frac{N}{8}\right) - x(n+\frac{5N}{8})) + j(x\left(n+\frac{3N}{8}\right) - x(n+\frac{7N}{8}))]]W^{8nk}W^{2q}$$

$$X(n+3N/8) = \sum_{n=0}^{\frac{N}{8}} [[\left(x(n) - x\left(n+\frac{N}{2}\right)\right) + j\left(x\left(n+\frac{N}{4}\right) - x\left(n+\frac{3N}{4}\right)\right)] + W^{3N/8}[(x\left(n+\frac{N}{8}\right) - x(n+\frac{5N}{8})) + j(x\left(n+\frac{3N}{8}\right) - x(n + \frac{7N}{8}))]]W^{8nk}W^{3q}$$

$$X(n+N/2)=\sum_{n=0}^{\frac{N}{8}}\Big[\Big[\Big(x(n)+x\left(n+\frac{N}{2}\right)\Big)+\Big(x\left(n+\frac{N}{4}\right)+x\left(n+\frac{3N}{4}\right)\Big)\Big]-[(x\left(n+\frac{N}{8}\right)$$
$$-x(n+\frac{5N}{8}))+(x\left(n+\frac{3N}{8}\right)+x(n+\frac{7N}{8}))]]W^{8nk}W^{4q}$$

$$X(n+5N/8)=\sum_{n=0}^{\frac{N}{8}}[[\Big(x(n)-x\left(n+\frac{N}{2}\right)\Big)-j\Big(x\left(n+\frac{N}{4}\right)-x\left(n+\frac{3N}{4}\right)\Big)]+W^{N/8}[(x\Big(n$$
$$+\frac{N}{8}\Big)-x(n+\frac{5N}{8}))-j(x\left(n+\frac{3N}{8}\right)-x(n+\frac{7N}{8}))]]W^{8nk}W^{5q}$$

$$X(n+3N/4)=\sum_{n=0}^{\frac{N}{8}}[[\Big(x(n)+x\left(n+\frac{N}{2}\right)\Big)-\Big(x\left(n+\frac{N}{4}\right)-x\left(n+\frac{3N}{4}\right)\Big)]+j[(x\left(n+\frac{N}{8}\right)$$
$$+x(n+\frac{5N}{8}))-(x\left(n+\frac{3N}{8}\right)+x(n+\frac{7N}{8}))]]W^{8nk}W^{6q}$$

$$X(n+7N/8)=\sum_{n=0}^{\frac{N}{8}}\Big[\Big[\Big(x(n)-x\left(n+\frac{N}{2}\right)\Big)+j\Big(x\left(n+\frac{N}{4}\right)-x\left(n+\frac{3N}{4}\right)\Big)\Big]$$
$$-W^{3N/8}[(x\left(n+\frac{N}{8}\right)-x(n+\frac{5N}{8}))+j(x\left(n+\frac{3N}{8}\right)-x(n$$
$$+\frac{7N}{8}))]]W^{8nk}W^{7q}$$

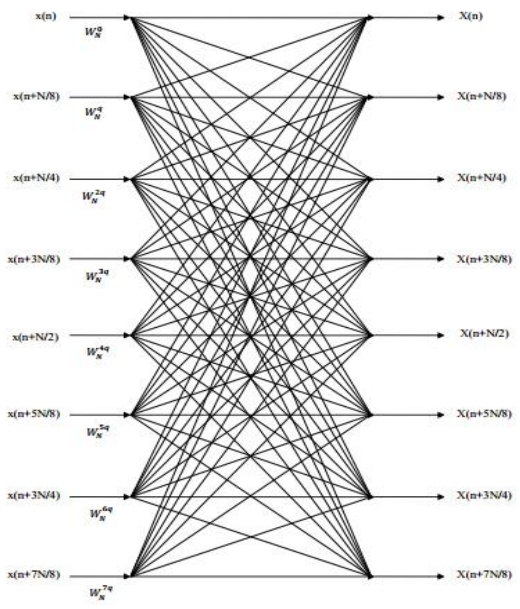

Fig.2 – Radix 8 Butterfly Unit

This paper proposes radix 4 and radix 8 butterfly unit designed using customized floating point unit and the results are compared with the conventional radix 4 and radix 8 algorithms designed using the IEEE compliant floating point unit.

QUADRATURE AMPLITUDE MODULATION

The FFT/IFFT algorithm is the most computationally intensive module in OFDM transceivers. At the transmitter the IFFT converts the complex data points received from the mapper into the same number of points in time domain. The range of input to the IFFT depends on the type of modulation. In general Quadrature Amplitude Modulation (QAM) [10] is a higher order form of modulation widely used in digital data radio communications. The types of QAM available are 16QAM, 32QAM, 64QAM, 128QAM, and 256QAM. The choice of a specific modulation type depends on the number of bits to be transmitted per symbol. 16QAM and 64QAM are widely used in wireless teleconferencing and digital terrestrial video broadcasting due to its high data rate requirements. In 16QAM 4 bits are used to represent a symbol where two bits are used for the inphase component (I) and two bits for the quadrature component (Q). Similarly 64 QAM uses 6 input bits per symbol, 3 bits for I and 3 bits for Q. ~~The~~ Tables 1 and 2 summarize the I and Q values of 16 QAM and 64 QAM respectively.

TABLE – 1 16 QAM ENCODING

Input to16 QAM (b0b1)	Output from 16 QAM (I component)
00	-3
01	-1
10	1
11	3

Input to 16 QAM (b2b3)	Output from 16 QAM (Q componenet)
00	-3
01	-1
10	1
11	3

Input to64 QAM (b0b1b2)	Output from 64 QAM (Q component)
000	-7
001	-5
010	-3
011	-1
100	1
101	3
110	5
111	7

TABLE – 2 64 QAM Encoding

Input to64 QAM (b0b1b2)	Output from 64 QAM (I component)
00	-7
001	-5
010	-3
011	-1
100	1
101	3
110	5
111	7

In the best case maximum constellation order used is 64-QAM with maximum possible FFT length of 2048. Using Matlab it was verified that the maximum possible dynamic range of a 2048 point FFT for the modulation types 16QAM and 64QAM would be 6.144e+003 + 6.144e+003j and 1.4336e+004 + 1.4336e+004j respectively.

CUSTOMIZED FLOATING POINT UNIT

The widely adopted IEEE single precision floating point format allows 16 bit to represent the real part and 16 bit to represent the imaginary part. The dynamic range offered by the single precision floating point format with 8 bit exponent was calculated to be 1.4 x 10^-45 to 3.4 x 10^38. This paper proposes to reduce the number of bits in the exponent and mantissa field of the IEEE 754 standard that determines the dynamic range and precision respectively. A customized floating unit with 16 bits (1 bit sign + 5 bit exponent + 10 bit mantissa) is proposed to achieve comparable dynamic range and precision required for QAM. The 5 bit exponent offers a dynamic from 2^{-15} to 2^{16}[11]. The dynamic range offered by the 5 bit exponent covers the entire range of values from the output of FFT with input from 16QAM or 64QAM. Using the customized fused floating point unit radix 4 and radix 8 butterfly units are coded in Verilog and synthesized using Quartus II.

RESULTS

The appropriate word length that offers the dynamic range required for 16QAM and 64QAM was analyzed and derived using Matlab. Based on the analysis radix 4 and radix 8 butterfly units for FFT computation in OFDM were designed using the customized fused floating point unit. The proposed design for radix 4 and radix 8 butterfly units were realized in Verilog HDL to verify its functionality. The design was synthesized using Quartus II. The synthesis results listed in Table.3 confirm that the proposed butterfly units reduce the memory requirements of FFT. The proposed butterfly units exhibit better performance in all aspects when compared with IEEE 754 compliant butterfly units used in OFDM.

Table – 3 FPGA implementation results for radix 4 and radix 8 butterfly

Algorithm		Total Logic Elements	Total Combinational functions	Logic Registers
Radix 4	Without fused unit	3312	3185	1154
	Customized with fused unit	1217	1175	537
Radix 8	Without fused unit	6255	6246	2617
	Customized with fused unit	5285	5271	2109

REFERENCES

[1] Xueqin Zhang, Kai Shen, Chengguang Xu 2013; ISBN NO. 978-1-4799-3380-8.

[2] Harikrishna K, Rama Rao T, Valadimir A. Labay IEEE, 2010; pp. 117 – 119, ISBN NO. 978-1-4244-5725-0

[3] Yiyan Wu Y. Zou 1995; 41, 392 – 399.

[4] Tejaswini H.N, Ravishankar C.V, IJARECE, 2015; 4(6), 1748 – 1752.

[5] Radu Suciu, Peter Reusens IEEE, 2009; 3 - 20.

[6] Renu Bala, Shamim Aktar, 2014; 102 (15), 22 - 25.

[7] Jiang Wang and Leif Arne Ronningen, 2014; Vol. 2 (1), 101 - 103.

[8] Adriana Bonilla R, Roberto J, Vega L, Karlo G, Lenzi e Luís G.P, Meloni simpósio brasileiro de telecomunicacoes 2012; 12, 13-16.

[9] Shushan Qiao, Yong Hei, Bin Wu, Yumei Zhou, IEEE, 2007.

[10] Govinda Mutyala Rao, T Bibhudendra, Acharya Sarat, Kumar Patra IEEE, 2008; 1-4.

[11] Fang Fang, Tsuhan Chen, Rob A. Rutenbar, EURASIP J Adv Signal Process, 2002; 9, 879 – 892.