Novel Algorithms for 2-D FFT and its Inverse for Image Compression

Anitha T G
Dept. of Electronics and Communication
Vinayaka Mission University
Salem, India
anitharajmitts@gmail.com

S. Ramachandran
Dept. of Electronics and Communication
SJB Institute of Technology
Bengaluru, India
ramachandr@gmail.com

Abstract— **High performance Fast Fourier Transform and Inverse Fast Fourier Transform are indispensable algorithms in the field of Digital Signal Processing. They are widely used in different areas of applications such as bio signal data compression, radars, image processing, voice processing etc. FFT algorithm is computationally intensive and need to be processed in real time for most applications. This paper presents novel algorithms for 2-D FFT and IFFT so that they may be realized in hardware. The algorithms have been developed to suit VLSI realization, where the processing speed is of paramount importance. The FFT and IFFT algorithms have been coded in MATLAB and successfully tested for 2D color images. The reconstructed images are indistinguishable from the original as can be seen from the results presented. The reconstructed quality of the images is better than 35 dB.**

Index Terms— **Digital Signal Processing, Discrete Fourier Transform, Fast Fourier Transform, Inverse Fast Fourier Transform, MATLAB.**

INTRODUCTION

The field of Digital Signal Processing (DSP) has grown enormously in the past decade and is playing a significant role in driving the technology. The DSP applications reach out to our everyday life such as medicine, surveillance, authentication and many more areas. In all these applications, Discrete Fourier has been widely used for efficient implementations. The Fast Fourier Transform is an efficient algorithm to compute the Discrete Fourier Transform (DFT) fast and its inverse. The evaluation of both produces same results but FFT is much faster. The DFT involves N^2 complex multiplications and N(N-1) complex additions, where N is 8 for 8x8 pixels block of an image. As the value of N increases, the number complex calculations also increase resulting in high processing time.

A Fast Fourier Algorithm (FFT) was discovered in 1965 by Cooley and Tukey, which reduced the number of calculations drastically and paved the way for real time processing of discrete signals which revolutionized the field of Digital Signal Processing [1]. Most of FFT algorithms decompose the overall *N*-point DFT into successively smaller and smaller expressions popularly known as butterfly structure. The FFT algorithms achieve its computational efficiency through divide and conquer strategy. The fundamental principle of FFT algorithm is that of dividing the computation of DFT sequence of length N into successively smaller DFTs by exploiting the Symmetry property and Periodicity property of DFT. The FFT reduces the number complex multiplications to (N/2) $\log_2$N and additions to N $\log_2$ N. Thus there is a large reduction in the calculation which makes real time processing a reality. Further, the butterfly structure makes hardware realizations difficult since regularity of expressions are broken.

This Paper presents a novel algorithm that can overcome the irregularity encountered in the butterfly structural implementations in VLSI realization. In contrast to the butterfly structure, the proposed algorithm is highly regular and hence suitable for VLSI implementations. The proposed design incorporates a high degree of pipelining and massive parallelism and hence offers high throughputs. Section I presents the basics of Discrete Fourier Transforms. The regular FFT/IFFT algorithm outlined earlier is proposed in Section II.The quantitative and qualitative analysis of the algorithm is summarized and tabulated in Section III successively.

A 1-D Discrete Fourier Transform Pair

One dimensional signal is described using function that depends on one independent variable with reference to time. Audio signals, radar signals and biomedical signals are examples of one dimensional signal. A transform changes one domain value such as time in to frequency components and makes the signal processing far more easily than the time domain components.

The Discrete Fourier Transform for a 1D signal is given by:

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

for n = 0, 1, 2, ……, N-1.

978-1-4673-4862-1/13/$31.00 ©2013 IEEE

The Inverse Fourier Transform is given by

$$x(n) = 1/N \sum_{k=0}^{N-1} X(k) W_N^{-nk} \quad (2)$$

for k = 0, 1, 2, ……., N-1.

B 2-D Discrete Fourier Transform Pair

Two dimensional signals is described using function that depends on two variables. Images are examples of two dimensional signals. The Discrete Fourier Transform for a 2D signal is given by the expression:

$$F(u,v) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(x,y) W_N^{(ux+vy)} \quad (3)$$

for k = 0, 1, 2, ……..N-1.

The Inverse Fourier Transform is given by

$$f(x,y) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} F(u,v) W_N^{-(ux+vy)} \quad (4)$$

for n = 0, 1, 2, ……., N-1.

Fast Fourier transform algorithms resorts to a variety of tricks for reducing the time required to compute a DFT [2]. Several methods for computing FFT and IFFT are discussed in Ref. [1]. Decimation In Time (DIT) divides a sequence x (n) in to odd and even sequences in successive stages to realize the whole DFT sequence. Likewise, Decimation In Frequency (DIF) divides the number of frequency components X (k) in to odd and even components to get the DFT in frequency domain [3]. FFT helps to transform the signal from time domain to frequency domain, where filtering and correlation can be performed with fewer operations [4].

A Radix sorting is a fast and stable algorithm that sorts keys, say, numerical digits. The processing of keys begins at the Least significant Digit and proceeds to the Most Significant Digit. In DFT applications, the radix-n algorithm divides a DFT of size N in to n interleaved DFTs of size N/n with each recursive stages. Radix 2 first divides the DFT of size N in to two interleaved DFTs of size N/2 and then computes the DFT of even indexed inputs X_{2m} and odd indexed input X_{2m+1}. These results are combined to produce the DFT of the sequence [5]. The implementation of Radix 2 algorithm is centered on conjugating the twiddle factors of the corresponding forward FFT. It has single path delay feedback pipelined processor [6]. The Radix 2 algorithms are simplest FFT algorithms. The Radix 2 DIT FFT can be applied recursively to the two length N/2 DFTs to save computation and it has to be successively applied to reach DFTs of length-2 [5]. Floating point arithmetic has been virtually impossible to implement on FPGA based systems due to the inherent complexity of floating point arithmetic. With the introduction of high level languages such as VHDL, rapid prototyping of floating point formats has become possible making such complex structures more feasible to implement [7]. For this a VHDL library has to be formed which includes addition, subtraction, multiplication and division modules. This situation inevitably utilizes large amount of FPGA resources.

Most of the research to date for the implementation of FFT algorithms has been performed using general purpose processors, Digital Signal Processors (DSPs) and dedicated FFT Processors [3]. However, as Field Programmable Gate Arrays (FPGAs) have grown in capacity, improved in performance, and decreased in cost, they have become a viable solution for performing computationally intensive tasks, with the ability to tackle applications for custom chips and programmable DSP devices [8].

I. Proposed Regular, FFT Algorithm for Image Processing

Twiddle Factor is a key component in FFT/IFFT computation. It is any of the trigonometric constant coefficients that are multiplied by the data in the course of the FFT algorithm. They are the coefficients used to combine results from a previous stage to form inputs to the next stage.

Twiddle factor term was apparently coined by Gentleman and Sande in 1963. It is the root of unity complex multiplication constants in the butterfly operations and also it describes the rotating vector which rotates in increments according to the number of samples.

The twiddle factor is expressed as

$$W_N = e^{-j(2\pi nk)/N} \quad (5)$$

By Euler's formula, the exponential term can also be written as

$$e^{-j\theta} = \cos\theta - j\sin\theta \quad (6)$$

The exponential term (j2πnk/N) of the DFT equation may be written using the Euler's formula:

$$e^{-j\theta} = \cos(2\pi nk/N) - j\sin(2\pi nk/N) \quad (7)$$

The RHS of the Euler's equation is expressed as (m*n) matrix forms for various values of (u, x) and (v, y) as Sine and Cosine terms. The matrices and their transposes may be stored as lookup tables on ROMs and accessed block by block for hardware implementation [9].

As we have observed in the FFT equation, it consists of complex additions, complex multiplications and twiddle factors irregularities to compute the FFT. Also in hardware realization, computation speed is of utmost importance and hence the algorithm should be amenable for parallelism and pipelining to speed up the computation.

The irregularities of the twiddle factors in FFT can also be overcome by the Cosine and Sine transforms of the signal. The FFT for 2D signal is obtained by adding the Sine and Cosine transforms of the input signal. The transforms in this algorithm is obtained in matrix form by varying the values of u and x from 0 to 7. The transposed Sine and Cosine matrices are obtained by varying the values of v and y from 0 to 7.
The required Cosine matrix and Sine matrix for FFT and IFFT are obtained as shown in Eq. (8) and Eq. (9).

$$C(u+1,x+1) = \cos((pi/4)*(u*x)) \quad (8)$$

$$S(u+1,x+1) = \sin((pi/4)*(u*x)) \quad (9)$$

The proposed, regular, 2-D FFT can be obtained by subtracting the Cosine and Sine transform as per Eq. 10:

$$F(u,v) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \{F(u,v)(\cos(2\pi(ux+vy)/N) + j\sin(2\pi(ux+vy)/N))\} \quad (10)$$

for k = 0, 1, 2…., N-1.

Similarly, the IFFT for 2-D is computed by subtracting the Cosine and Sine transform of the FFT of the signal and may be expressed as:

$$f(x,y) = (1/MN)\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \{F(u,v)(\cos(2\pi(ux+vy)/N) - j\sin(2\pi(ux+vy)/N))\}$$

for n = 0, 1, 2…., N-1 (11)

A. Proposed Regular FFT/IFFT Algorithm

The proposed FFT/IFFT algorithm is also follows:

1. Read the input file from the host system.
2. Evaluate the (8x8) Cosine and Sine matrix using Equations 8 and 9.
3. The image is accessed as (8x8) block successively, and the Cosine and Sine transforms are obtained.
4. Compute the FFT of the image using Eq. 10.
5. Verify the obtained FFT values with built in FFT functional values of MATLAB.
6. Compute the IFFT of the processed image in step 4 using the Eq. 11.
7. Verify visually the reconstructed image with the original image.
8. Calculate the Power Signal to Noise Ratio (PSNR) of the processed signals for validation. Note: A PSNR value of 35 dB and above implies the reconstructed image is indistinguishable from the original image.

II. Experimental Results and Comparative Study

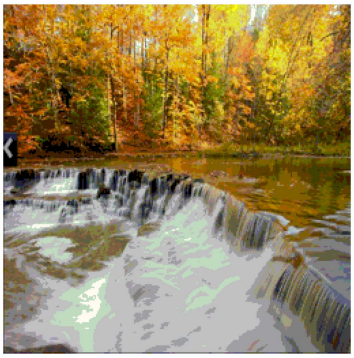
Due to the importance and use of FFT in many signal and Image processing applications, a novel FFT and IFFT algorithm has been presented in this paper. This algorithm exploits the Regularity, Parallelism and Pipelined architecture for the implementation.

The FFT and IFFT are coded in MATLAB in order to establish the correct working of the proposed algorithm. It also serves as a reference for checking the hardware results after the algorithm is implemented on FPGA.

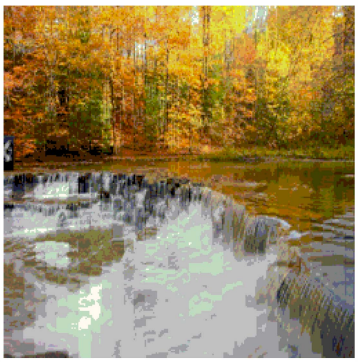
The algorithm is designed using MATLAB (R2008a) and applied on to various images. MATLAB is considered as industry standard for this field of research [9]. The reconstructed images are of good quality with PSNR value

Table 1. Reconstructed Quality of Sample Images using the Proposed FFT/IFFT Algorithms

2-D (color images)			
Input File	**File Type**	**File Size (pixels)**	**PSNR (dB)**
Dolphins	bmp	532x327	39
Luvre-Paris	bmp	805x531	39
Seminar	jpg	1312x1000	38
Volcano	bmp	528x361	38
Tree	bmp	483x812	35
Cleveland	bmp	806x550	35



a



b

Fig. 1 Reconstructed image Cleveland: a Original b Reconstructed

a

b

Fig. 2 Reconstructed image Luvre Paris: a Original b Reconstructed

better than 35 dB. Table 3 shows the output quality of the proposed Regular FFT/IFFT algorithm. Figures 1 and 2 show the sample reconstructed images.

III. Conclusion

A novel algorithm has been presented in this paper to compute FFT and IFFT of the image signals. The proposed algorithm has been coded in Matlab, and the reconstructed images are good with a PSNR value of 35 dB or above. The algorithm can be implemented on FPGA for high throughputs. Presently the hardware architectures are being designed. These architectures are also being coded in Verilog conforming to RTL coding guidelines.

References

[1] Alan V. Oppenheim, Ronald W. Schafer, John R. Buck, Discrete –Time Signal Processing, Prentice Hall, Second Edition, pp. 646-652, 1999.

[2] L. Rabiner B. Gold, *Theory and Application of Digital Signal Processing*, Prentice-Hall, 1975.

[3] Sneha N. Kherde, Meghana Hasamnis, "Efficient Design and Implementation of FFT", International Journal of Engineering Science and Technology (IJEST), ISSN : 0975-5462, NCICT Special Issue Feb. 2011.

[4] The Design and Implementation of FFT Algorithm Based on the Xilinx FPGA IP Core, The 2nd International Conference on Computer Application and System Modeling (2012).

[5] Shaminder Kaur, "FPGA Implementation of OFDM Transceiver using FFT Algorithm", International Journal of Engineering Science and Technology (IJEST) Vol. 4, No. 04 April 2012.

[6] Asmita Haveliya, "Design and simulation of 32-point FFT using Radix-2 algoritm for FPGA implementation", 2012 Second International Conference on Advanced Computing & Communication Technologies.

[7] Ahmed Saeed, M. Elbably, G. Abdelfadeel, M. I. Eladawy, "Efficient FPGA implementation of FFT/IFFT Processor", International Journal of Circuits, Systems and Signal Processing, Volume 3, 2009.

[8] Nabeel Shirazi, M. Athansa and A.Lynn abbott, "Implementation of a 2-D Fast Fourier Trasform on FPGA based Custom Computing Machine", Proceedings of 12th Reconfigurable Architecture Workshop, Denver 2005.

[9] Taoufik SAIDANI, Yahia SAID, Mohamed ATRI Rached Tourki, "Real Time FPGA acceleration for Discrete Wavelet Transform of the 5/3 Filter for JPEG 2000", SETIT 2012, IEEE, pp. 21-24, March 2012.

[10] I.S.Uzun, A.Amira A.Bouridane, FPGA Implementations of Fast Fourier Transforms for Real-Time Signal and Image Processing.

[11] S. Ramachandran, "Digital VLSI Systems Design", Springer 2007.