INTERNATIONAL JOURNAL OF SCIENTIFIC PROGRESS AND RESEARCH (IJSPR) ISSN: 2349-4689
Issue 107, Volume 37, Number 02, 2017

Radix-2 DIT Fast Fourier Transforms using Folding and Pipeline Architecture

Pooja Barange[1] , Deepak Kumar[2]

[1]M.Tech. Scholar, [2]Assistant Professor,EC Department,

Vidhyapeeth Institute of Science & Technology, Bhopal, M.P., India

Abstract— A fast Fourier transform (FFT) is any fast algorithm for computing the DFT. The development of FFT algorithms had a tremendous impact on computational aspects of signal processing and applied science. The decimation-in-time (DIT) fast Fourier transform (FFT) very often has advantage over the decimation-in-frequency (DIF) FFT for most real-valued applications, like speech/image/video processing, biomedical signal processing, and time-series analysis, etc., since it does not require any output reordering. Upon comparison, the proposed algorithm based on unsigned, signed and complex multiplier is fast than previous algorithm. This all design and experiments were carried out on a Xilinx 6.2i Virtex-2p device family.

Index Terms— FFT, Decimation in Time, Decimation in Frequency, real Value data

I. INTRODUCTION

The discrete Fourier change (DFT) is a critical instrument in numerous branches of science and designing [1] and has been concentrated broadly [2]. For some commonsense applications, it is imperative to have a usage of the DFT that is as quick as could be expected under the circumstances. Previously, speed was the immediate result of astute calculations [2] that minimized the quantity of number juggling operations. On present day universally useful microchips, in any case, the execution of a system is generally dictated by convoluted collaborations of the code with the processor pipeline, and by the structure of the memory. Outlining for execution under these conditions requires a cozy learning of the PC design. In this paper, we address this issue by method for a novel versatile methodology, where the system itself adjusts the calculation to the points of interest of the equipment. We created FFTW, a versatile, superior usage of the Cooley-Tukey quick Fourier change (FFT) calculation [3], written in C. We have analyzed numerous C and Fortran usage of the DFT on a few machines, and our tests demonstrate that FFTW ordinarily yields altogether preferable execution over all other openly accessible DFT programming.

The FFT (Fast Fourier Transform) and its converse (IFFT) are the key parts of OFDM (Orthogonal Frequency Division Multiplexing) frameworks. As of late, the interest for long length, fast and low- control FFT has expanded in the OFDM applications. There are three sorts of primary configuration models for actualizing a FFT

frameworks, FFT and backwards FFT (IFFT) assume a vital part. The OFDM strategy, because of its adequacy in overcoming unfavorable channel impacts [1, 2] and additionally range usage, has turned out to be broadly embraced in wire line and remote correspondence principles. The OFDM procedure has been embraced in a few measures like computerized sound television (DAB) [3], advanced video TV physical (DVB-T) [4], lopsided computerized supporter line (ADSL) [5] and rapid advanced endorser line (VDSL) [6]. Accordingly, proficient and low-control VLSI usage of FFT processors is fundamental for fruitful arrangement of these OFDM-based frameworks. As per the guidelines of DAB, DVB-T, ADSL and VDSL, different FFT sizes are required, as appeared in Table 1. From this Table, plainly variable-length FFT equipment is a vital module in the minimal effort arrangement of the above correspondence frameworks. The Cooley – Tukey N-point FFT calculation requires O(Nlog N) calculations, which is a colossal sparing over direct calculation of the discrete Fourier change (DFT). In any case, equipment usage of the calculation is both computational serious, as far as number-crunching operations, and correspondence escalated, as far as information swapping. For ongoing preparing of FFT, O(log N) math operations are required per test cycle. High speed real-time processing can be accomplished in two different ways. In the application specific parallel or pipelined processor approach, the required operations are performed at the clock frequency equivalent to the sample frequency, and this approach usually consumes less power.

II. Fast Fourier Transform

Before going further to examine on the FFT and IFFT outline, it regards clarify a bit on the quick Fourier change and reverse quick Fourier change operation. The quick Fourier change (FFT) and opposite quick Fourier change (IFFT) are gotten from the fundamental capacity which is called Discrete Fourier Transform (DFT). Using FFT/IFFT rather than DFT is that the calculation of the capacity can be made speedier where this is the fundamental criteria for usage in the computerized signal handling. In DFT the calculation for N-purpose of the DFT will figure one by one for every point. While for

FFT/IFFT, the calculation is done at the same time and this technique spares a considerable amount of time. The following is the condition (2.2) demonstrating the DFT and from here the condition is determined to get FFT/IFFT capacity.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi k/N} \qquad (1)$$

processor. One is the single-memory design. It makes them process component and one primary memory. Consequently, it involves a little zone. The second is the double memory design, which has two recollections. This design has a higher throughput than the single-

memory engineering since it can store butterfly yields and read butterfly inputs in the meantime. The quick Fourier change assumes an imperative part in numerous computerized signal handling (DSP) frameworks. Late advances in semiconductor preparing innovation have empowered the arrangement of devoted FFT processors in applications, for example, information transfers, discourse and picture handling. In particular, in the OFDM correspondence

X (k) represent the DFT frequency output at the *k*-the spectral point where *k* ranges from 0 to *N-1*. The quantity *N* represents the number of sample points in the DFT data frame.

The quantity x (n) represents the *n*th time sample, where *n* also ranges from 0 to *N-1*. In general equation, x (n) can be real or complex.

The DFT equation can be re-written equation (2) into:

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

The quantity W_N^{nk} is defined as in equation (3)

$$W_N^{nk} = e^{-j2\pi k/N} \qquad (3)$$

Here is the place the mystery lies amongst DFT and FFT/IFFT where the condition (2.4) capacity above is called Twiddle Factor. This component is ascertained and put in a table keeping in mind the end goal to make the calculation less demanding and can run at the same time. The Twiddle Factor table is relying upon the quantity of point use. Amid the calculation of IFFT, the variable does not to recalculate since it can allude to the Twiddle element table therefore it spare time since computation is done simultaneously.

III. INVERSE FAST FOURIER TRANSFORM

Opposite quick Fourier change (IFFT) is utilized to produce OFDM images. The information bits is speak to as the recurrence space and since IFFT change over sign from recurrence area to time space, it is utilized as a part of transmitter to handle the procedure.

Table 1: Twiddle factor for 8 point inverse fast Fourier transform IFFT (N=8)

n	W	Value
1	*W* -0 8	1
2	*W* -1 8	0.7071+j0.7071
3	*W* -2 8	j
4	*W* -3 8	-0.7071+j0.7071
5	*W* -4 8	-1
6	*W* 5 8	-0.7071-j0.7071
7	*W* -6 8	0
8	*W* -7 8	0.7071-j0.7071

IFFT is defined as the equation (4) below:

$$x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k) W_N^{-nk} \qquad (4)$$

Same FFT calculation can be utilized to discover IFFT capacity with the adjustments in specific properties. The progressions that actualize is by including a scaling component of 1/N and supplanting twiddle variable worth ( ) likewise can be utilized for the opposite quick Fourier change. The following is the table 1 demonstrates the estimations of twiddle component for IFFT.

Figure 1: Radix-2 Decimation in Time Domain FFT Algorithm

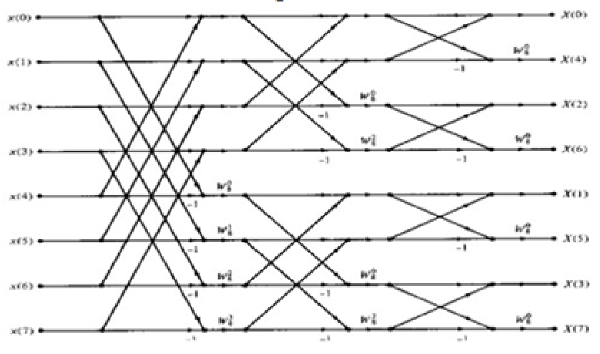


Figure 2: Radix-2 Decimation in Frequency Domain FFT Algorithm

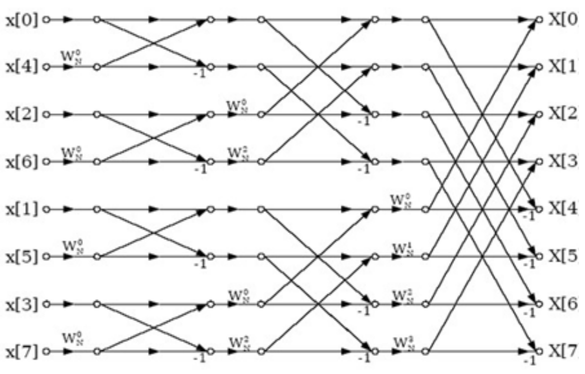


Figure 1: Radix-2 Decimation in Time Domain FFT Algorithm

IV. PROPOSED METHOD

The flow chart of the proposed methodology is shown in figure 3. In this paper we are used three techniques i.e. unsigned multiplier, signed multiplier and complex multiplier. In this figure the two signed and unsigned bit multiplier (i.e. multiplier n-bit, multiplicand m-bit) and final output of the multiplier is n+m bits. All the multiplier is design into two parts i.e. partial product generator and multi operand addition'

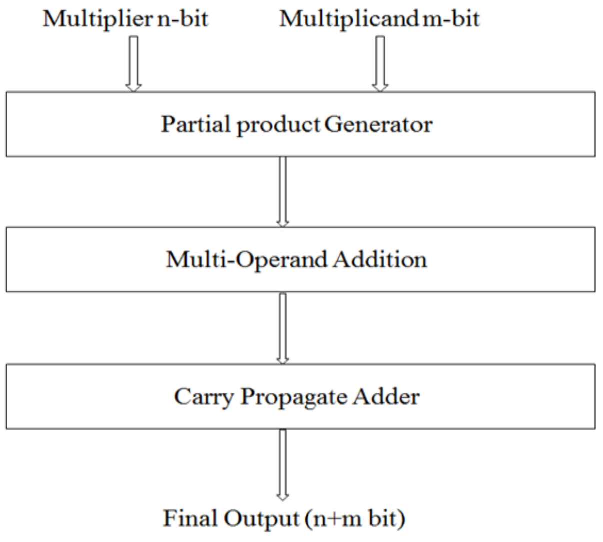
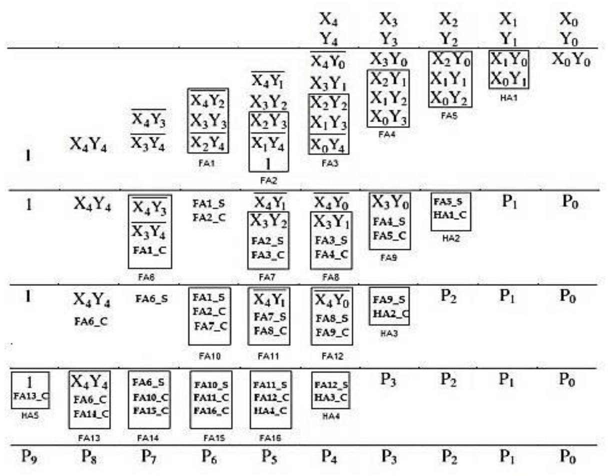


Figure 3: Flow Chart of the proposed Methodology

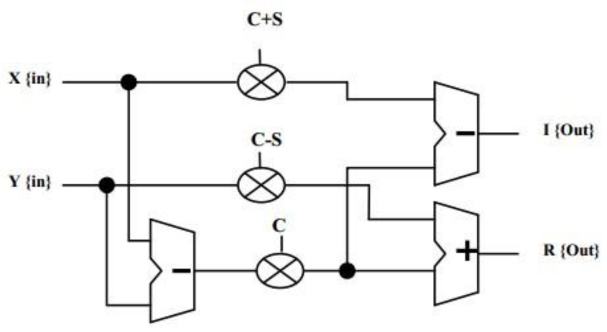
Array Multiplier:-

$$\begin{array}{ccccccc} & & & A_3 & A_2 & A_1 & A_0 \\ & & & B_3 & B_2 & B_1 & B_0 \quad \times \\ \hline & & & A_3 \bullet B_0 & A_2 \bullet B_0 & A_1 \bullet B_0 & A_0 \bullet B_0 \\ & & A_3 \bullet B_1 & A_2 \bullet B_1 & A_1 \bullet B_1 & A_0 \bullet B_1 & \\ & A_3 \bullet B_2 & A_2 \bullet B_2 & A_1 \bullet B_2 & A_0 \bullet B_2 & & \\ A_3 \bullet B_3 & A_2 \bullet B_3 & A_1 \bullet B_3 & A_0 \bullet B_3 & & & \quad + \\ \hline P_6 & P_5 & P_4 & P_3 & P_2 & P_1 & P_0 \end{array}$$

Signed Multiplier:-



Complex Multiplier:-



The FFT algorithms are classified into two broad categories, namely, the decimation-in-time (DIT) and the decimation-infrequency (DIF) algorithms. The proposed radix-2 DIT algorithm is shown in figure 3.

A+WB

A _______________

DIT butterfly includes an augmentation took after by increases. As appeared in Table I the calculation time of fixed-point augmentation took after by an expansion am not as much as that of expansion took after by an increase. The DIT-based FT butterfly in this manner includes less engendering delay than that of DIF-based RFFT butterfly

albeit both these butterflies include the same number of multipliers and adders. In this manner, the decision of DIT calculation to determine FT structure has preference over DIF calculation. In this paper, we exhibit efficient engineering for the DIT radix-2 RFFT calculation

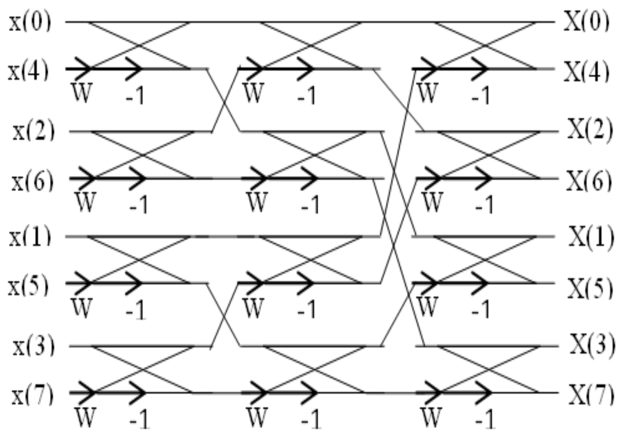


Figure 4: Proposed 8-point Radix-2 DIT

This calculation deteriorates an arrangement of DFT into four little DFTs of 1/4 lengths in a recursive way and their yields are utilized to control a few different yields by which the expense of calculation will be lessened. The input data is disintegrated into four small sequences of x (4n + i) where n = 0, 1... N/4-1 and i = 0, 1, 2, 3.

Different Size of FFT using Radix-2					
Size	N=8	N=16	N=32	N=64	N=128
Eun Ji Kim et al.	3569	8643	24892	50271	760000
Proposed Design	2832	7552	18880	45312	105728

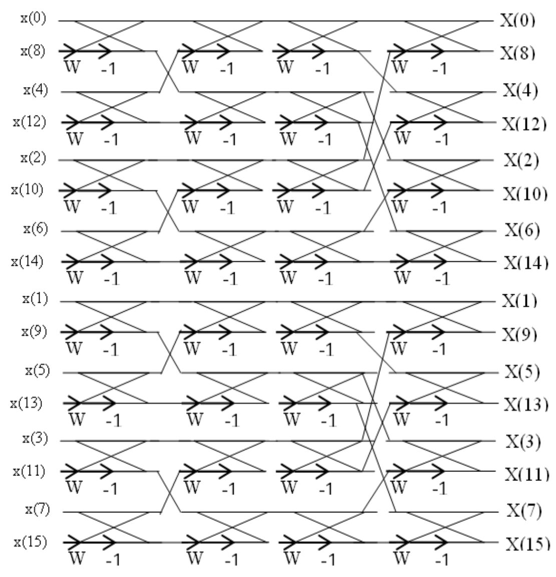


Figure 5: Proposed 16-point Radix-2 DIT Algorithm

V. DELAY AND AREA METHODOLOGY

The deferral and zone assessment philosophy considers all doors to be comprised of AND, OR, and Inverter (AOI), each having delay equivalent to 1 unit and territory equivalent to 1 unit. We then include the quantity of doors in the longest way of a rationale piece that adds to the most extreme deferral. The zone assessment is finished by checking the aggregate number of AOI doors required for every rationale square.

Table III: Delay and Area calculate in basic block of FFT

Adder Block	Del	Area
XO	3	5
2:1 MUX	3	4
Half Adder	3	6
Full Adder	6	13

Table IV: Delay and Area Count of the different types of FFT

Table V: Comparison result for existing algorithm and

Architecture	TMA	TAM	TAM-TMA	% Differenc
8-bit				
Pramod Kumar	8.345	8.967	0.622	6.9 %
Proposed Design	8.048	8.343	0.295	3.5 %
16-bit				
Pramod Kumar	9.453	10.321	0.868	8.4 %
Proposed Design	8.653	9.240	0.587	6.3 %
32-bit				
Pramod Kumar	14.532	14.982	0.45	3.0 %
Proposed Design	13.377	14.010	0.633	4.5 %

VI. SIMULATION RESULTS

All the planning and examination in regards to calculation that we have specified in this paper is being created on Xilinx 14.2i redesigned variant. Xilinx 14.2i has couple of the striking elements, for example, low memory necessity, quick investigating, and minimal effort. The most recent arrival of ISETM (Integrated Software Environment) outline device gives the low memory necessity rough 27 rate low. ISE 6.1i that gives propelled apparatuses like savvy incorporate innovation with better utilization of their figuring equipment gives quicker planning conclusion and higher nature of results for a superior time to outlining arrangement. ISE 14.2i Xilinx instruments grants more prominent adaptability for outlines which influence implanted processors. Likewise included is the most current arrival of the chip scope Pro

Serial IO Tool pack, giving streamlined troubleshooting of fast serial IO plans for Virtex-7 FX and Virtex-7 LXT and SXT FPGAs. With the assistance of this instrument we can create in the territory of correspondence and additionally in the region of sign preparing and VLSI low power outlining.

Radix-2 DIT Algorithm for N=8			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
P. K. Meher et al. [1]	-	-	16.895
Proposed Design (unsigned)	96	192	15.548
Proposed Design (signed)	221	387	15.137
Radix-2 DIT Algorithm for N=16			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
P. K. Meher et al. [1]	-	-	21.054
Proposed Design (unsigned)	256	512	19.26
Proposed Design (signed)	573	1000	18.927
Radix-2 DIT Algorithm for N=32			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
P. K. Meher et al. [1]	-	-	24.421
Proposed Design (unsigned)	640	1280	22.147
Proposed Design (signed)	1410	2462	22.806
Radix-2 DIT Algorithm for N=64			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
P. K. Meher et al. [1]	-	-	29.421
Proposed Design (unsigned)	1536	3072	26.319
Proposed Design (signed)	2431	4063	22.785
Radix-2 DIT Algorithm for N=128			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
P. K. Meher et al. [1]	-	-	36.421
Proposed Design (unsigned)	2739	5042	32.895
Proposed Design (signed)	3390	6042	33.806

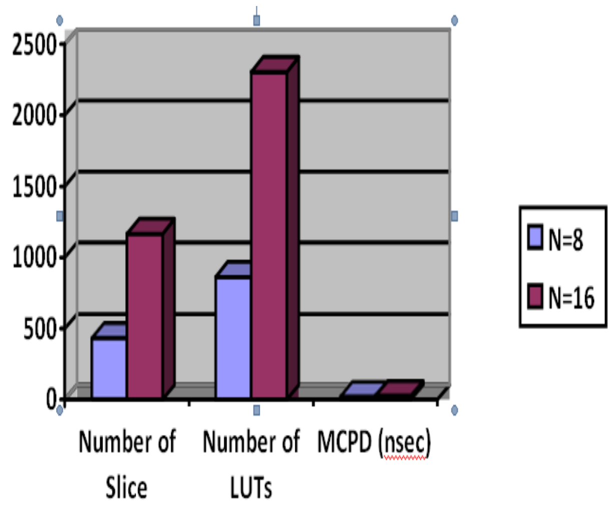


Figure 9: Bar graph of the complex number for N=8 and N=16

Radix-2 DIT Algorithm for N=8 and N=16			
Parameter	Number of Slice	Number of LUTs	MCPD (nsec)
N=8	438	864	24.948
N=16	1168	2304	30.687

VII. CONCLUSION

The prime objective is to construct a FFT in order to have low power consumption and lesser area. The parameters (i) power consumption (ii) Area occupancy were given due consideration for comparing the proposed circuit with other FFTs. The circuits were simulated using Model-Sim 6.3c and synthesized with Xilinx ISE 6.2i.The performance of various 64 point FFT such as Radix-2, Radix-4, split Radix, mixed -radix 4-2, R2MDC and the proposed modified R2MDC were carried out and their performance were analyzed with respect to the number of CLB slices, utilization factor and Power consumption

REFERENCES

[1] Charles. Roth Jr., ―Digital Systems Design using VHDL‖, Thomson Brooks/Cole, 7th reprint, 2005.

[2] S. S. Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A, ―Implementation of Vedic multiplier for Digital Signal Processing‖, International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011, Proceedings published by

International Joural of Computer Applications® (IJCA), pp.1-6.

[3] Himanshu Thapaliyal and M.B Srinivas, ―VLSI Implementation of RSA Encryption System Using Ancient Indian Vedic Mathematics‖, Center for VLSI and Embedded System Technologies, International Institute of Information Technology Hyderabad, India.

[4] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, ―Vedic Mathematics: Sixteen simple Mathematical Formulae from the Veda‖, Delhi (2011).

[5] Harpreet Singh Dhillon and Abhijit Mitra, ―A Reduced-bit Multiplication Algorithm for Digital Arithmetic‖, International Journal of Computational and Mathematical Sciences, Febrauary 2008, pp.64- 69.

[6] Sumit Vaidya and Depak Dandekar. ―Delay-power performance comparison of multipliers in VLSI circuit design‖. International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010

[7] Pramod Kumar Mehe, Basant Kumar Mohanty, Sujit Kumar Patel, Soumya Ganguly, and Thambipillai Srikanthan, "Efficient VLSI Architecture for Decimation-in-Time Fast Fourier Transform of Real-Valued Data", IEEE Transactions on Circuits And Systems—I: Regular Papers, Vol. 62, No. 12, December 2015

[8] M. Ayinala, Y. Lao, and K. K. Parhi, ―An in-place FFT architecture for real-valued signals,‖ IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 60, no. 10, pp. 652–656, Oct. 2013

[9] Shashank Mittal, Md. Zafar Ali Khan and M.B. Srinivas, ―Area Efficient High Speed Architecture of Bruun's FFT for Software Defined Radio‖, 1930- 529X/07/$25.00 © 2007 IEEE

[10] B. G. Jo and M. H. Sunwoo, ―New continuous-flow mixed-radix (CFMR) FFT processor using novel in-place strategy,‖ IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 5, pp. 911–919, May 2005