International Journal of Engineering & Technology, 7 (3.29) (2018) 35-38

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Multi path pipelined architecture with twin parallel processing after second stage for high-speed FFT

G. Prasanna Kumar [1] *, Pushpa Kotipalli [2], B. T. krishna [1]

[1] JNTUK University College of Engineering, Kakinada
[2] Shri Vishnu Engineering College for Women, Bhimavaram
**Corresponding author E-mail: godiprasanna@gmail.com*

Abstract

This paper presents review on different pipelined FFT architectures and proposes a new pipelined FFT architecture with twin parallel processing after second stage. The proposed architecture follows a novel data flow path, Twiddle factor generation and multiplication is implemented by multiplier and shift registers. The first two stages are implemented by multipath pipelined form after that it follows twin parallel form. The twin parallel form consists of two pipelined units simultaneously generates FFT output values. This architecture reduces latency in a greater extent with a smaller cost of hardware. The proposed architecture compared with previous architectures. The proposed architecture is implemented for Radix-2 and Radix-22 DIF FFT. The throughput of proposed architecture is four.

Keywords: *FFT; Twiddle Factor; Pipelined; Radix-2; Radix-22; DIF; Latency.*

1. Introduction

2. Introduction and literature review

Discrete Fourier transform (DFT) is widely used in the field of communication, signal processing and image processing. The basic equation of DFT for an N point sequence is

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad 0 \le k \le N-1$$

The major computations for calculating DFT is additions and multiplications with twiddle factors, so direct computation of DFT requires computations proportional to N2. In the past years Cooley and Tukey proposed the fast Fourier transform (FFT) algorithm [1] to find DFT in an efficient manner, it
was popular in 1965. There are two main important algorithms of DFT are Decimation in frequency (DIF) and Decimation in time (DIT). In case of DIF FFT out sequence is decomposed into even and odd components whereas in DIT FFT input sequence is decomposed into even and odd samples.
The Decimation in frequency (DIF) FFT data flow graph for 8 sample values are shown below. The basic DFT equation can be written as

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

The output samples divided into even and odd samples according to the following equations

$$X[2k] = \sum_{n=0}^{\frac{N}{2}-1} \{x[n] + x[n+N/2]\} W_{N/2}^{kn}$$

$$X[2k+1] = \sum_{n=0}^{\frac{N}{2}-1} \{x[n] - x[n+N/2]\} W_{N/2}^{kn}$$

There are $N \log_2 N$ adders and $\frac{N}{2} \log_2 N$ multipliers

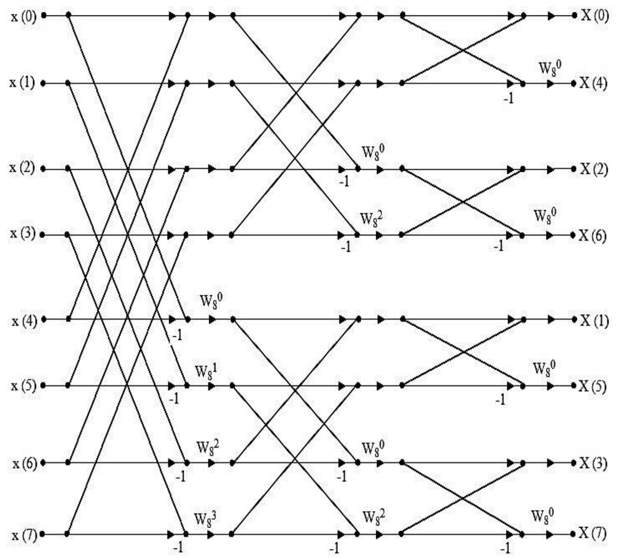

Fig. 1: Radix-2 DIF FFT Architecture.

The DIT FFT is decomposed based on the following equations

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} x[2n] w_N^{2nk} + w_N^k \sum_{n=0}^{\frac{N}{2}-1} x[2n+1]\, w_N^{2nk}$$

From this equation for computing N point DFT, first find the N/2 point DFT of even samples and N/2 point DFT of odd samples and add the results will form a butterfly structure.
The main important components in the realization of FFT is adders, multipliers, adders are required for each stage whereas multipliers are for multiplication with the twiddle factor.

The twiddle factors are multiplied after each stage of butterfly architecture in Radix-2 DIF FFT architecture. The flow of data is from left to right. Radix-2 FFT architectures generally twiddle factor multiplication is done after every stages so that more number of multipliers are required. Radix-22 DIF FFT architecture for every two stages one stage twiddle factor multiplication is done shown in fig 2 below.

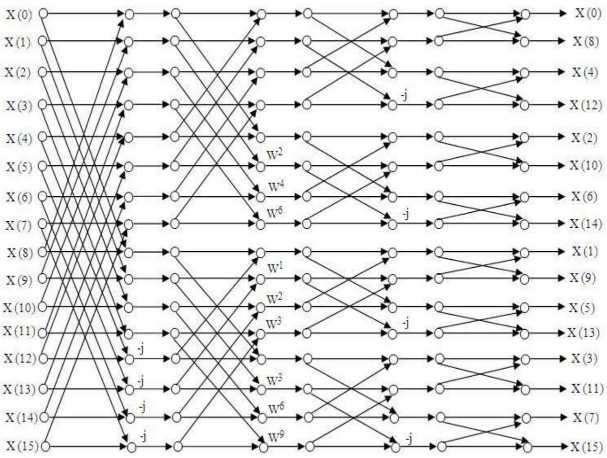

Fig. 2: Data Flow of Radix-22 DIF FFT for N=16.

For VLSI implementation of FFT pipelined architectures are very essential. The efficient pipelined architectures reduces hardware and effectively utilizes the hardware during this process Latency plays very important role. Latency is how many clock cycles are required after pumping all data values to the pipelined architectures. This paper proposes a new FFT architecture with twin parallel processing after second stage which reduces Latency in a big extent due to increase in hardware components.

The rest of the paper is organized as follows. Section II reviews the different pipelined FFT architectures. Section III provides proposed twin parallel architecture followings from second stage Section IV describes the simulation results and comparisons with recent work that has been done, section V provides the conclusion of the paper.

3. Different pipelined FFT architectures

Radix-2 Multipath Delay Commutator (R2MDC):

Radix -2 multipath delay commutator is one of the popular approaches for pipelined implementation of FFT [2]. it uses single butterfly elements and commutators between the stages. Commutator and multiplexer switches corresponding input to butterfly.

Radix-4 Multipath Delay Commutator (R4MDC):

Radix-4 multipath delay commutator is one of pipelined implementation of FFT [2] similar to R2MDC. The number of stages are less

Radix 4 single path delay Feedback (R4SDF):

Radix 4 single path delay feedback FFT architecture [3] is one of the feedback architecture uses shift register similar to R2SDF but in radix-4 format. This architecture utilizes the multipliers in more efficient manner.

Radix 2 single path delay Feedback (R2SDF):

Radix 2 single path delay feedback FFT architecture effectively utilizes memory. The architecture consists of shift registers to store the data output from the butterflies.

Radix 4 single path delay commutator (R4SDC):

Radix 4 single path delay commutator is one of the simplified architecture for FFT [5]. It has only complex adder with programmable control by commutator for each and with an advantage of higher multiplier utilization and less memory requirement.These are the standard Pipelined FFT architectures each architecture has its own data flow path. Recently Pipelined FFT architecture with reduced latency is very familiar, architecture [14] proposes Pipelined Architectures for Real-Valued FFT and Hermitian-Symmetric IFFT, and this reduces the latency in great extent and architecture [15] concentrates more on Latency of the data flow

By using the twin data Streams.

4. Proposed twin parallel architecture after second stage

The proposed architecture for Radix-22 DIF-FFT is shown in figure 3. In which first two stages are designed by multipath delayed architecture, from the second stage onwards splits the structure into two parallel form which raises the throughput into four. In this architecture F0, F1, F2, F3, F4, F5, F6, F7, F8 are the First in First out (FIFO) memory units which consist of Read(R) and Write (W) signals. These signals are controlled by a controller. In DIF-FFT data elements are taken in Normal order and output are generated in bit reversal order. Beginning from the first stage which consists of Butterfly and a FIFO, at beginning FIFO writes first eight data elements by control write signal from the controller and from the next cycle FIFO starts reading by Read enable, after eight clock cycles from begging butterfly1 generates two outputs. BF is the butterfly unit in the figure. The butterfly unit at beginning generates all the sixteen data output elements in this eight values are stored in FIFO's after first stage. After generating first eight data elements. The FIFO's after second stage stores running data of Twiddle factors are multiplied after second stage of pipelined architecture by twiddle factor generation unit which consists of multiplier and shift register unit. During the first eight output values from stage 1 the twin parallel architecture has two parallel units for the first unit the twiddle factor coefficients are {0, 0,0,0} and for second unit twiddle factors are {0,4,2,6} butterfly II after 2 clock cycle's reads to the next twin parallel unit which generates four outputs in a clock cycle. For the 16 Point FFT, upper unit generates X(0) , X(8) and lower unit generates X(2) , X(10) . During the next clock cycle X(4) ,X(12) and X(6),X(14) depending on selection of multiplexer and FIFO read and write signals which is under controlled by a controller. The remaining eight output values are generated by taking data flow values stored in FIFO after first stage .theses values read from FIFO to butterfly by selecting selection signals of multiplexers. Similar control signals are generated by the controller after next clock cycle to obtain remaining eight output values.

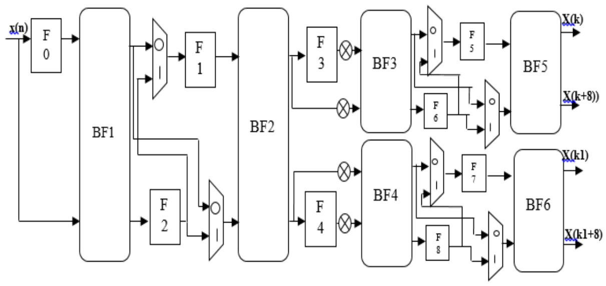

Fig. 3: Radix 22 DIF- FFT twin Parallel Processing after Second Stage for N=16.

The proposed twin parallel architecture after second stage up to stage 2 of the architecture hardware components are less after stage 2 onward hardware components are doubled due to this parallel units data flow in two parallel ways as a result throughput increases to 4.

5. Control unit

The control unit controls signals of FIFO's, Multiplexers and twiddle factor multiplication. It generates FIFO read and write signals selection signals of the mux.

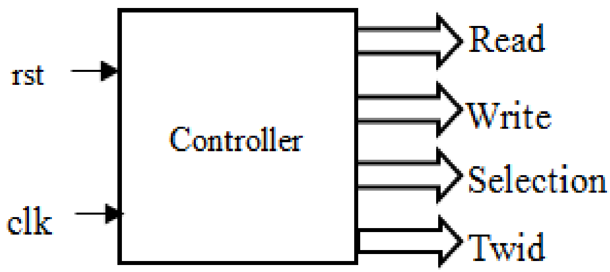

Fig. 4: Controller.

Figure 4 shows the controller for the figure shown, for a given input of reset and clock controller generates read, write, selection of multiplexer and twiddle factor signals.
The timing diagram for the controller for the figure is shown in figure 5

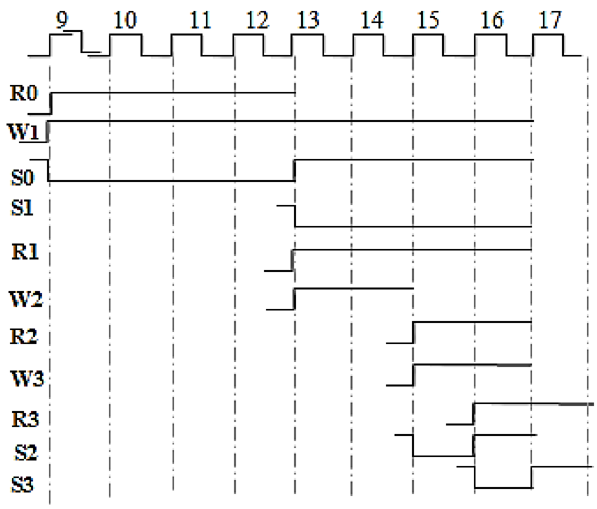

Fig. 5: Timing Diagram of Controller for N=16.

In this figure R0 is the read enable of FIFO F0, w1 is write enable of FIFO F1 and F2 after first stage. During the first eight clock cycles F0 write the data from x(0) to x(7) after next cycle on wards controller enables read R0 it reads first input value of x(0) and other side running data is x(8) after x(7) ,so x(0) and x(8) processed through butterfly unit which consists of two complex adders and generates two output values these are stored in two FIFO's of F1 and F2 and S0,S1 is the selection signals for the multiplexers after first butterfly unit. During the four clock cycles from 9 to 12 S0 made to zero it writes from one end butterfly unit and other end F2 writes the data. After completion of 12 clock cycle the generated output is ready to process from butterfly after first stage, FIFO read signal for F1 and F2 is R1 is enabled. The data in F1 is reads into butterfly unit at that time s1 is '0' selects running data processed through butterfly in second stage generates two output values stored in FIFO's of F3 and F4 by enabling write signal w2 for two clock cycles. in the same duration from 13 to 16 signal S0 is '1' that is data in FIFO F2 is reads into F1 , the running sum is stored in FIFO F2. This structure is designed for effective utilization of storage. After first butterfly stage 8 storage spaces are only there. Similar structure for last stage but it has two butterfly units, so it will give four outputs in a clock cycle. During the16th cycle butterfly 5 generates X (0),X(8) and butterfly 6 generates X(2),X(10). Next cycle it generates another [4] output values. To generate remaining 8 output values enable S1 to '1'. then the last 8 output values of first stage butterfly processed in similar fashion as already shown for getting another 8 values it will takes further 5 clock cycles. The overall latency for N=16 is [6] for N=32 latency is 14 and latency for N=64 is 30. Commonly for N an point FFT latency is $\frac{N}{2} - 2$.

Butterfly Unit:
Butterfly unit is radix-2 simple unit consists of two complex multipliers shown in figure 6 below

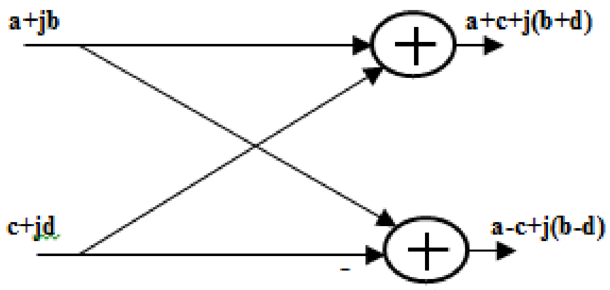

Fig. 6: Simple Two Point Butterfly.

Twiddle-factor multiplication Unit:
The twiddle factor multiplier unit is constructed by a multiplier and shift registers, here multiplier one input is running data stream and another input is twiddle factor coefficient in decimal fixed point representation. Initially conversation from fractional form to decimal by multiplying with the number 10 if only one decimal value is considered if two decimal points considered multiply with the number 100 .

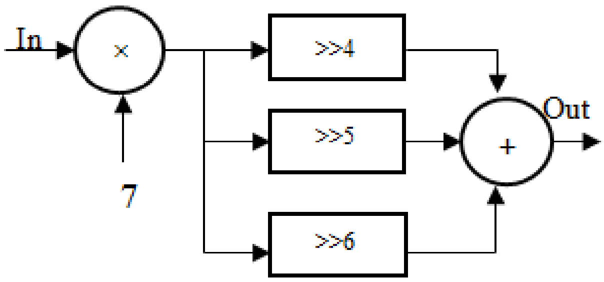

Fig. 7: Twiddle Factor Multiplier Unit.

In figure 7, shift register generates division operation.
Radix-2 N=8 FFT there are four twiddle factor are For w_8^0 and w_8^2 no multipliers are required because $w_8^0 = 1$ and $w_8^2 = -j$ multiplication by –j with a complex number results real part becomes negative of imaginary and imaginary part becomes real part, for $w_8^1 = 0.707 + j0.707$ and $w_8^3 = -0.707 - j0.707$ multiplication with 0.707 is required .The incoming data requires multiplication with 0.7 that is after truncation, for that there is technique first multiplication by seventy and divide the resultant by 10. Division can be done by shifting operation.

Example: 10*0.7=7

First 8-bit representation of 10 is 00001010
Multiplication 10 with 7 results 70 binary format 01000110 right shift by four ,five and six units results 00000100, 00000010 and 00000001 addition of these three number results 00000111 which is equivalent to 7 so the multiplier unit in fig 8 gives the approximate and multiplication result with 0.7.
Multiplexers switching
The multiplexer switching unit after each stage is designed for effective utilization of memory and pipelined implementation

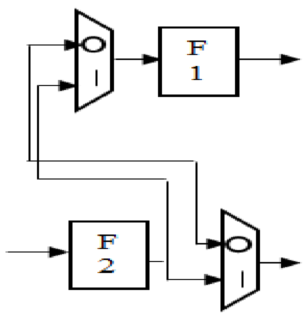

Fig. 8: Multiplexer Switching Unit.

The two multiplexers are having two selection lines of s1 and s2 and storage capacity of each FIFO after first stage computations is

four. So to store and propagate the values a switching mechanism is developed.
Initially , s1 is '0' stores four values F1 and F2 fills 4 data values after fourth cycle F1 pumps data to next butterfly and F2 reads data to F1 and writes running values simultaneously for effective utilization of hardware.

6. Simulation results and comparisons

The simulation results for twin parallel processing after the second stage with RTL Schematics are shown in Figure 9

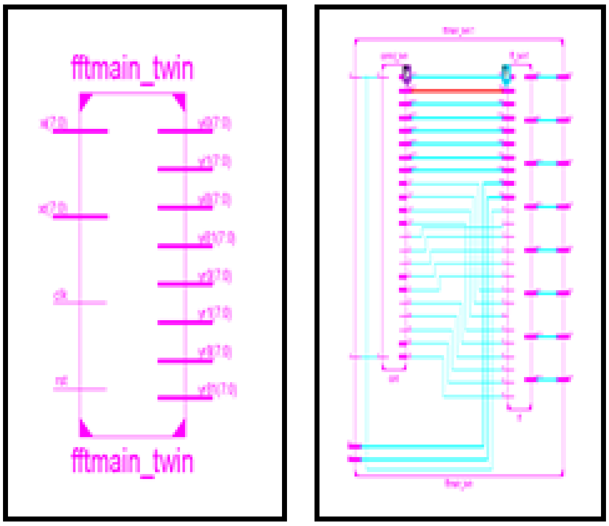

Fig. 9: RTL Schematics of Proposed Method for N=16.

Table 1: Comparison of Hardware Components for N Point FFT

Architecture	Multipliers	Adders	Latency
Radix-2 [11]	$log_2 N - 3$	$4log_2 N$	N
Radix-2³[14]	$log_8 N - 1$	$2log_2 N$	$3\frac{N}{2}$-1
Radix-2 [15]	$log_2 N - 3$	$2log_2 N$	$3\frac{N}{2}$-6
Radix-2³[15]	$2(log_8 N - 1)$	$2log_2 N$	$3\frac{N}{2}$-6
Radix-2 [16]	$4log_4 N - 4$	$4log_4 N + 4$	$3\frac{N}{4} - 1$
Proposed Radix-2	$2log_2 N - 4)$	$4(log_2 N - 1)$	$\frac{N}{2} - 2$
Proposed Radix-2²	$4(log_4 N - 1)$	$4(log_2 N - 1)$	$\frac{N}{2} - 2$

The proposed method is compared with several existing architectures from [x] [y] [z]. The latency of the proposed architecture is low because of twin data flow streams as we are used in this architecture but a small cost of hardware. The number of complex adders for both Radix-2 and radix22 is same for N=16 number of butterfly units are 6 and each butterfly consists of 2 adders. Total 12 complex adders are required, similarly for N=32 8 butterfly units are required and total 16 complex adders are required. Commonly we can represent for N data values 〖4(log_2〗N-1) complex adders are required. The complex multipliers for the proposed architecture with Radix-2 N=16 is 3, commonly for N values 〖2log_2〗N-4 are required and for Radix-22 N=16 is 4, for N data values is 〖4(log_4〗N-1). Table I represents all the comparisons of hardware complexity of the proposed architecture with several previous architectures.

Table 2: Synthesized Results

Type	Reg	Luts	DSP48E	Power(mw)
Radix-2	119	311	8	251
Radix-2²	90	317	16	122

Table II represents the synthesized results of proposed architecture for N=16 on vertex-6 XC6VLX75T device.

7. Conclusion

This paper has presented a new architecture for implementation of FFT based on multipath twin parallel units after second stage of FFT flow graph. The proposed architecture has combination of serial as well as parallel architecture. This proposed approach reduces latency of overall process compared to existing methods. A new data flow path for FFT implementation is proposed in this paper. The proposed architecture uses more hardware after second stage and it has effectively utilizes the hardware. The throughput of proposed architecture is the number 4.

References

[1] J. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comput.Vol. 19, Pp. 297–301, Apr. 1965.

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

[3] E.H. Wold and A.M. Despain. Pipeline and Parallel-Pipeline FFT Processors for VLSI Implementation. IEEE Trans. Com- Put. C-33(5):414-426, May 1984.

[4] A.M. Despain. Fourier Transform Computer Using CORDIC Iterations. IEEE Trans. Comput., C-23(10):993-1001, Oct. 1974.

[5] G. Bi and E. V. Jones. A Pipelined FFT Processor for Word- Sequential Data. IEEE Trans. Acoust., Speech, Signal Processing, 37(12):1982-1985, Dec. 1989.

[6] W.C. Yeh and C.-W. Jen, "High-Speed And Low-Power Split-Radix FFT," IEEE Trans. Signal Process., Vol. 51, No. 3, Pp. 864–874, Mar. 2003.

[7] Y.-T. Lin, P.-Y. Tsai, And T. D. Chiueh, "Low-Power Variable-Length Fast Fourier Transform Processor," Computers And Digital Techniques, IEE Proceedings -, Vol. 152, No. 4, Pp. 499–506, 2005.

[8] L. Yang, K. Zhang, H. Liu, J. Huang, And S. Huang, "An Efficient Locally Pipelined FFT Processor," IEEE Trans. Circuits Syst. II, Exp. Briefs, Vol. 53, No. 7, Pp. 585–589, Jul. 2006.

[9] M. Shin and H. Lee, "A High-Speed Four Parallel Radix- FFT/IFFT Processor For UWB Applications," In Proc. IEEE ISCAS, 2008, Pp. 960–963.

[10] Y. Voronenko And M. Püschel, "Algebraic Signal Processing Theory: Cooley-Tukey Type Algorithms For Real Dfts," IEEE Trans. Signal Process., Vol. 57, No. 1, Pp. 205–222, Jan. 2009.

[11] M. Garrido, K. K. Parhi, and J. Grajal, "A Pipelined FFT Architecture for Real-Valued Signals," IEEE Trans. Circuits Syst. I, Reg. Papers, Vol. 56, No. 12, Pp. 2634–2643, Dec. 2009.

[12] S. Langemeyer, P. Pirsch, And H. Blume, "A FPGA Architecture For Real-Time Processing Of Variable-Length FFTS," In 2011 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP), 2011, Pp. 1705–1708.

[13] M. Ayinala, M. Brown, And K. Parhi, "Pipelined Parallel FFT Architectures Via Folding Transformation," IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 20, No. 6, Pp. 1068–1081, 2012.

[14] M. Ayinala and K. K. Parhi, "FFT Architectures for Real-Valued Signals Based On Radix-23 and Radix-24 Algorithms," IEEE Trans. Circuits Syst. I, Reg. Papers, Doi:10.1109/TCSI.2013.224625.

[15] Sayed Ahmad Salehi, Rasoul Amirfattahi, and Keshab K. Parhi "Pipelined Architectures for Real-Valued FFT AND Hermitian-Symmetric IFFT with Real Datapaths" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, VOL. 60, NO. 8, AUGUST 2013.

[16] Antony Xavier Glittas, Mathini Sellathurai, And Gopalakrishnan Lakshminarayanan A Normal I/O Order Radix-2 FFT Architecture To Process Twin Data Streams For MIMO IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION(VLSI) SYSTEMS, VOL.26, No.6,June 2016.