High Speed 64 point FFT/IFFT module for IEEE standard 802.11

Joshi Ravindra Sharad
Assistant Professor
Dept : Electronics and Telecommunication Engineering ,
KC College of Engineering and Management Studies and Research. Thane , India.
email : joshi_ravindra_s@yahoo.co.in

Poornima Talwai
Associate Professor
Dept : Electronics.
Ramrao Adik Institute of Technology
Navi Mumbai, India
email : poornima . talwai@gmail.com

***Abstract*— In this paper a 64 point FFT/ IFFT module for IEEE standard 802.11.a. is presented. IEEE standard 802.11.a requires OFDM ie Orthogonal Frequency Divisional Multiplexing . It requires a 64 point FFT / IFFT module as one of its basic building blocks. As per the standards it is expected that the 64 point FFT/ IFFT computation should be done in 3.2 us. In this module mixed radix algorithm is being used. The 64 point FFT is decomposed into an 8x8. The eight point FFT is implemented using radix2. The arithmetic chosen is fixed point arithmetic as it helps in realizing smaller and simpler circuitry. Here a grid based approach is considered which makes the overall circuitry simple.**

***Keywords*— VHDL, FFT, Signal processing, Ramgrid.**

I. INTRODUCTION

FFT or Fast Fourier Transform finds wide range of applications in the Engineering field. FFT or Fast Fourier Transform is an algorithm by which we can reduce the mathematical computations which may be too large as in the case of DFT . The algorithm also called Cooley Tukey algorithm reduces the number of operations involved from N^2 to $N\log_2 N$ [10][11]. It finds many applications in communications. Various signal processing applications can be Spectrum Analyzers,. Harmonic analyzers, Sonography equipments, MPEG audio coding, MRI equipments, Accoustic Engineering, Direction finding systems, Radars, Medical Imaging etc.

FFT architectures

In FFT implementations as the no of points increases the no of two point butterflies to be computed also increases, similarly the number of complex multiplications to be performed also increase. Depending on the speed desired and resources available various types of architectures have been implemented in the past and are being implemented. The architectures can be broadly classified into software based architectures, DSP processor based architectures or application specific Integrated Processors or Algorithmic type architectures. some of the algorithmic type architectures are fully parallel FFT , Column FFT and Pipelined FFT, non pipelined. There are still many more type of architectures.Non pipelined type architectures generally do the processing in a sequential manner the data is stored in a RAM then a chunk of data for a operation is fetched an operation ie 2 point FFT or complex multiplication is performed on the data then the data is stored back in to the RAM as per the logic. These type of architectures utilize very less resources. the limitation being the speed. To increase the speed generally mixed radix algorithm based architectures can be broadly used as they decrease the total number of complex multiplications. The other type of architectures is the pipelined architectures generally used in real time systems in which when one data sample goes inside the system a processed data sample comes outside. The various types of pipelined architectures are described ahead,

R2MDC – Radix 2 Multipath delay commutator Architecture [18] In this architecture delay and commutator elements are used . Here the input sequence is broken into two parallel datastreams flowing foreward with correct distance. The distance is maintained with the help of delays and commutators. In this type of logic the utilization of butterflies and multipliers is 50%. **R2SDF** Radix- 2 single path Delay Feedback architecture[19][23] This type of architecture utilizes the feedback elements in a more efficient manner. . It requires less memory compared to R2MDC. **R4SDF** Radix – 4 Single Path delay feedback architecture [20] It is a radix -4 version of R2SDF employing (CORDIC) coordinate Rotational Digital computer iterations. It is highly complicated as it utilizes a radix -4 butterfly.**R4MDC** Radix -4 Multipath Delay Commutator architecture [18] . It suffers from low utilization of all components. **R4SDC** Radix – 4 single Path Delay Commutator architecture [21] This architecture improves the utilization of butterfly elements by modifying the butterfly elements. It uses a modified radix -4 algorithm with programmable ¼ radix – 4 butterflies. Design of butterfly and delay commutator elements becomes complicated due to programmability requirement. **$R2^2$ SDF** Radix 2^2 single path delay feed back architecture [22] In this type of architecture one radix -4 butterfly operation is broken into two radix – 2 butterfly operations with trivial multiplications of + or – 1 and + or – j. with feedback mechanism. The memory is fully utilized as with R2SDF and R4SDF. Most of the FFT architectures generally tend to be modifications of the above type of architectures. when designing circuitry for any of these architectures the commutator elements or feed back elements have to be designed separately for each stage again logic has to be established for proper transfer of data from one stage to another so each stage has to be designed seperately. Again a proper synchronization has to be made between all the stages.

Three are still many pipelind implementations . Our main objective is to achieve a speed of 3.2 us for a 64 point FFT/IFFT module. [24] have implemented a ASIC version.
We have implemented a 64 point FFT architecture in a different way . Here we utilize only one RAM and only one eight point FFT. The detailed description can be understood ahead.

II. MATHEMATICAL BACKGROUND

Mathematical Analysis of Mixed Radix Algorithm

The mathematical analysis of the system can be understood from the analysis of the mixed radix algorithm, in depth analysis of the algorithm can be obtained from Hayes [13]. Here the no of points N can be expressed as a product, of

$$N = N_1 \times N_2 \quad (1)$$

In equation (1) N is divided into an array of N_1 columns and N_2 rows .The index maps for n and k are defined in equation(2) and equation(3) the sequence becomes an array given by equation(4)

$$n = N_2 \times n_1 + n_2 \{0 \le n_1 \le N_1 - 1, 0 \le n_2 \le N_2 - 1\} \quad (2)$$

$$k = k_1 + N_1 \times k_2 \{0 \le k_1 \le N_1 - 1, 0 \le k_2 \le N_2 - 1\} \quad (3)$$

$$X = \begin{bmatrix} x(0) & x(N_2) & \cdots & x(N_2(N_1-1)) \\ x(1) & x(N_2+1) & \cdots & x(N_2(N_1-1)+1) \\ \vdots & \vdots & & \vdots \\ x(N_2-1) & x(2N_2-1) & \cdots & x(N_1N_2-1) \end{bmatrix} \quad (4)$$

The N point DFT can be expressed as given by equation (5)and equation (6).

$$X(k) = X(k_1 + N_1 k_2) \quad (5)$$

$$X(k) = \sum_{n_2=0}^{N_2-1} \sum_{n_1=0}^{N_1-1} x(N_2 n_1 + n_2) W_N^{(k_1+N_1k_2)(N_2n_1+n_2)} \quad .(6)$$

Simplifying equation(6) we get (7), (8)

$$X(k) = \sum_{n_2=0}^{N_2-1} \left[\left[\sum_{n_1=0}^{N_1-1} x(N_2 n_1 + n_2) W_{N_1}^{n_1 k_1} \right] W_N^{k_1 n_2} \right] W_{N_2}^{k_2 n_2} \quad (7)$$

$$G(n_2, k_1) = \sum_{n_1=0}^{N_1-1} x(N_2 n_1 + n_2) W_{N_1}^{n_1 k_1} \quad (8)$$

is the N_1 point DFT of the sequence $x(N_2n_1 + n_2)$ which is the row n_2 of the two dimensional array given in expression (4). After computing N_1 point DFT of each row of the array it produces another array consisting of complex numbers $G(n_2, k_1)$ here in place computation is done and the data is stored in the same row of the array. The next step in the evaluation of X(k) is to multiply by twiddle factors .

$$G_1(n_2, k_1) = W_N^{k_1 n_2} G(n_2, k_1) \quad (9)$$

We can prepare an array for the twiddle factors

$$W_N^{k_1 n_2}$$

$$G = \begin{bmatrix} G(0,0) & G(0,1) & \cdots & G(0,(N_1-1)) \\ G(1,0) & G(1,1) & \cdots & G(1,(N_1-1)) \\ \vdots & \vdots & & \vdots \\ G(N_2-1,0) & G(N_2-1,1) & \cdots & G(N_2-1,N_2-1) \end{bmatrix} \quad (10)$$

The next step is to do point to point multiplication between $G(n_2, k_1)$ array and twiddle factor array and store the data back in the original location to realize the desired result.The final step is to compute N_2 point DFT of the columns of the array $G_1(n_2, k_1)$

$$X(k_1 + N_1 K_2) = \sum_{n_2=0}^{N_2-1} G_1(n_2, k_1) W_{N_2}^{k_2 n_2} \quad (11)$$

The DFT coefficients are than read row wise from the two dimensional array

$$X(k) = X(K_1 + N_1 k_2) \quad (12)$$

To perform IDFT a complex conjugate of the twiddle factors is taken in all the stages and then divide by N operation is performed on all the data points in the last stage.In our module we are decomposing 64 points into 8x8. Instead of DFT we are using FFT. The 8 point FFT is further decomposed using a radix of 2.

III SYSTEM ARCHITECTURE

A. MAIN SYSTEM

The sytem block diagram is shown in figure 1 it consists of a RAMGRID, and modules M1, M2, M3, M4 and a control unit. The various signals are a input bus of 36 bits for giving

input data. A output bus of 36 bit to receive output data. The data input and out put is in complex format. The clk signal is present here we utilize a clk of 200MHz. To write data to the RAM we have ramwen signal, to read data from the RAM we have the ramren signal. The gpr signal helps us to identify whether the ram is going to be accessible externally as sequential RAM or as a RAMGRID internally. The adrkey is mapped to the RAM for feeding the data in natural order as well as retrieving the data in natural order. There are status signal for input and output. In depth description of each module is provided ahead.

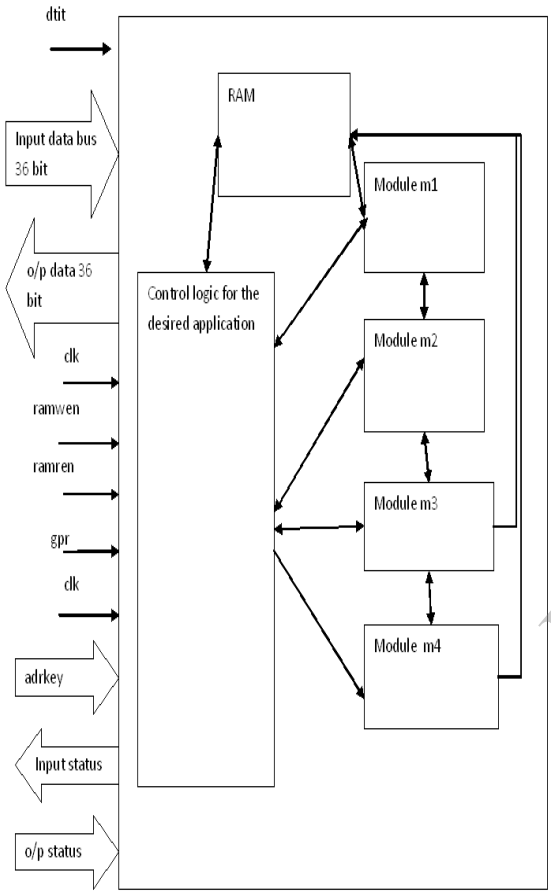

Fig 1 Main System

B. RAMGRID

The RAMGRID is organised as a set of registers. These can work as set of rows or set of columns, there are control signals r/c, rcnum, gp, ramren, ramwren, etc. When gpr = 1 then RAM (RAMGRID) is accessed as a general purpose RAM array with every register given an addresses. If gp = 0 then RAM is accessed as an array of rows and column registers accessible row wise or column wise. The row or column can be accessed by the rcnum signal ie row or column number. When r/c = 1 the entire RAM is accesed in the form of rows there are 8 separate input buses connected to the input of registers and there are 8 seperate output buses connected to the o/p of registers. Depending on the row no given that respective row is connected to input bus or o/p bus depending on read or write operation. If r/c = 0 the the entire RAM is accesed in the form of columns when the column no is specified the respective registers in the column are available for read or write operation.the ramgrid is shown in figure 2

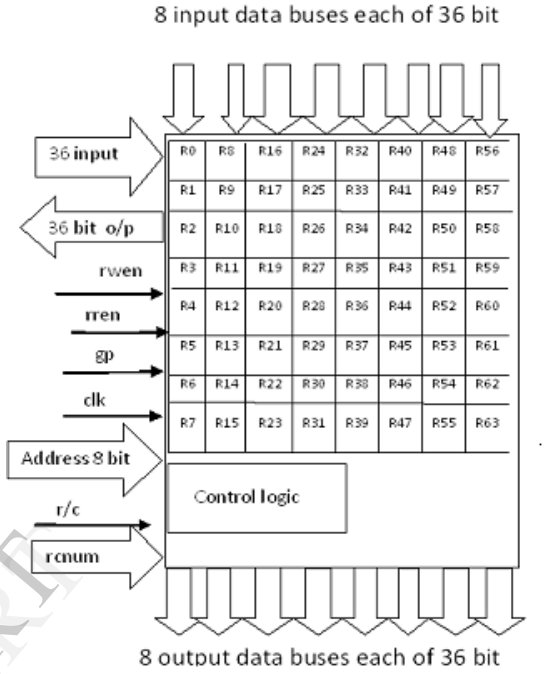

Fig 2. RAMGRID

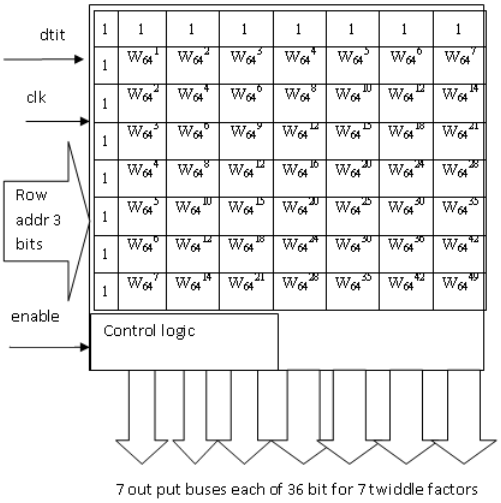

Fig 3. Split Bank Twiddle factor ROM

C. SPLIT BANK TWIDDLE FACTOR ROM

In this module the ROM is split into 8 banks so that the coefficients for 8 elements of the row can be obtained simultaneously just by specifying the row number shown in the figure 3. If DITFFT = 1 one set of twiddle factors is

obtained across the output data buses , if it is IDITFFT the complex conjugate set of twiddle factors is obtained.

D. OTHER MODULES

There are four modules M1, M2, M3, M4, module M1, M2 , M3 together help us to realize an 8 point FFT operation . module M4 helps us to perform twiddle factor multiplications on an entire row of registers. The external interfaces of all the four modules viz M1, M2, M3, M4 are all of the same type there is a difference in their internal structures. They operate in a pipelined manner with overlapped states decreasing the overall processing time.

M1 module In this module there are two radix 2 four point FFT blocks the input data across 8 buses are given to the 8 respective inputs by appropriate reordering the o/p is obtained across the 8 o/p buses.

M2 module For M2 module we have two complex multipliers for performing multiplication by twiddle factors as well as there is logic for doing multiplication by –j or +j .

M3 module For M3 module contains four radix 2, 2 point FFT blocks the 8 input points are fed across 8 input data bus then they are appropriately reordered and fed to the 2 point FFT modules the data is obtained across the o/p data buses .

M4 module Module M4 this module is a optional module and comes into play only in case of row operations the o/p from M3 is fed to this module . Here there are 7 complex multipliers. One input to the complex multipliers is from the split bank twiddle factor ROM the other input is the input data to be multiplied by. The entire row of data can be multiplied by the twiddle factor coefficients for the respective row so 7 complex multiplications can be performed in a single shot. The four modules M1, M2, M3, M4 can be in one of the four states

00 loading, 01 ---processing, 10 ----unloading(transfer) 11—halt/ wait state. .

E. ARITHMETIC DETAILS

In our design we are using the **Fixed point arithmetic** [5],[6]. This leads to a considerable saving in the number of logical cells and chip area it also tends to be faster than the floating point arithmetic. A new data format is chosen. In this format there are 18 bits for real and 18 bits for imaginary. Using this type of data format helps us to be in tune with the 18X18 multipliers. Due to this only 1 18X18 multiplier is required for a fixed point multiplication .In this system only four 18X18 multipliers are required for implementing one complex multiplier so activity is faster. The input data which may be of 8 bits or 10 bits has to be converted to our new data format before proceeding for processing. Here two cases were analyzed

Case1 Format for data Here D35 to D18 represents the real portion and D17 to D0 represents the imaginary portion.Here the data is considered to be of 10 bits with no padding at the Least significant bits and 7 bits for overflow, 1 sign bit. D35 is the sign bit for real D17 is the sign bit for imaginary. The input data of 10 bits for real portion is inserted at locations D18 to D27 and D0 to D9 for imaginary all other locations are padded with zeroes.

Format for twiddle factors.Here D35 to D18 represents the real portion and D17 to D0 represents the imaginary portion. Here there is 1 sign bit and 17 bit fraction .D35 is sign bit for real, D17 is sign bit for imaginary. D18 to D34 is the fractional portion of the real part of twiddle factor. D0 to D16 is the fractional portion of imaginary part of the twiddle factor.

Case 2 In this case the input data is considered to be of 8 bits with padding of two zeroes at the Least significant bits and 7 bits for overflow both in the real portion and in the imaginary portion. Here again D35 and D17 are sign bits for real and imaginary part respectively. Input data of 8 bits is placed at locations D2 to D9 for imaginary and D20 to D27 for real. All other locations are padded with zeroes. The format for twiddle factors is the same as the above case.

In the chosen dataformat fixed point adders , fixed point substractors and fixed point multipliers were designed using VHDL and tested.

F. SYSTEM WORKING

The data is fed from the external system into the RAM (RAMGRID) in serial order one point at a time. Then it is analyzed for DITFFT or IDITFFT accordingly signal is given to all the modules. Then r/c is made = 1 the entire ram array works like a grid operating row wise the data is taken one row at a time and fed to module M1 after the data from the row is processed by this module it is transferred to the next module M2 in M2 module twiddle factor multiplications are performed on the data.. In the meanwhile the module M1 is loaded with data from row 2 after the data in M2 is processed it is shifted to module M3 where four radix 2 2 point FFT modules perform 2 point FFT computations , the data from this module is then fed to M4 where twiddle factor multiplications for the row are performed. The flow for row operations is RAM –> M1-> M2-> M3 –> M4 -> RAM Since all the four modules are in different state and operating on a different data of row this leads to overlap or pipelining and increasing the speed of operation. This method of pipelining also helps us to decrease the resources, which would have been enormous if the desired speed is to be achieved.. A similar processing sequence can be constructed for value r/c = 0 here the entire ram array works like a column grid. Here also we can follow the same sequence as above with the exception that M4 module is disconnected here and data from M3 after processing is fed to the RAM on the column from which it was fetched. For column operations the M4 module is disconnected the data flows from RAM –>M1-→M2→-M3-→RAM. The flow of data can be understood from figure 4 . After the processing is complete the data is fetched from the RAM by the external system the addresskey is hardware mapped so that data is given in natural order as well as retrieved in natural order. In case of IDITFFT another set of twiddle factors is used in all the modules. To perform divide by 64 operation by hardware mapping of the databuses the divide by 64 operation is performed.

IV TESTING

A. Validation

A C ++ program was used for conversion of twiddle factors from decimal format to fixed point format. To verify the results a standard test data was generated using a C++ program the data was converted to fixed point format .The data was given as an input to the module designed by us . the output was in fixed point format. Another C++ program was used to convert the o/p data to decimal. The test data was given as input to the MATLAB FFT command the results were verified. The module was tested for logic.

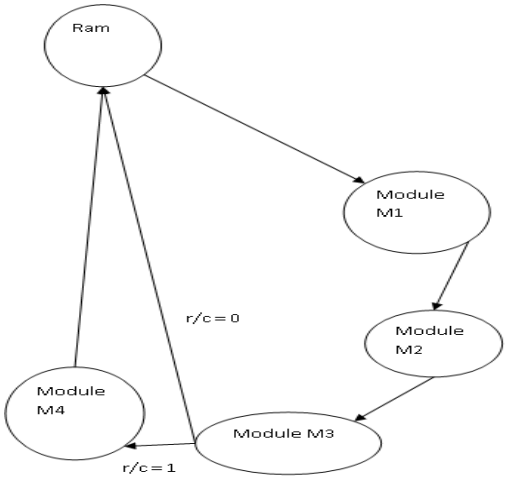

Fig 4 . State diagram for internal data flow

B. Test Results

A clock frequency of 200MHz was chosen The module was tested with a standard data input and results were verified The required time was verified on Isim simulator. **Loading time into the RAM is 960 ns**, **unloading time from the RAM is 960 ns. Actual processing time 1280 ns. Total actual time taken 3.277us which is near to the required processing time of 3.2us.** The logic takes **32** states for **row operatons 32** states **for column operations** total of **64 states for processing** Each state takes a time of **20 ns** so **1280 ns is the time for processing operations**. On reducing the no of states or processing time per state using high speed devices the speed can be further increased.

C. Synthesis report .

The module was syntehsized for a Spartan 6 LX45 FPGA the timing results and synthesis results were verified on ISIM simulator VHDL was used as the programming language.Xilinx ISE Webpack 13.1 was used for design synthesis and simulation. . The synrhesis report is given in table1

V COMPARISON

Module described in paper [24] is an ASIC version ours is a FPGA version. We can think of increasing the speed when we put it to ASIC version.Our architecture is a simple architecture. Similar architectures have been implemented using [26] [25] ours is a VHDL based architecture and structure is easy to implement.It is higly pipelined. There are 8 buses across which data flows. Similarly the modules m1, m2, m3, m4 operate on adifferent set of data in pipelined manner increasing the speed.We are using fixed point arithmetic so design of elements is easy. If overflow is to be neglected or reduced the bit per point can be reduced by 12 bits per point this can give 24 bits per point a still compact design can be derived from the basic architecture.If this is done the speed will increase forther.If we want higher precision then the data can be inserted midway ie keeping some bits for LSB trimming error , and some bits for overflow error. None of the existing designs take into account these considerations. For error control. Only one 8 point FFT in decomposed manner is utilized in our design so logic area is compact none of the designs do it this way. The concept of using only one RAM for data storage can be extended for radix 4 so 4x4x4 can also be performed The concept can be elaborated for higher no of points also.

Table 1 Synthesis Table

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	8495	54576	15%
Number of Slice LUTs	12573	27288	46%
Number of fully used LUT-FF pairs	4524	16544	27%
Number of bonded IOBs	89	218	44%
Number of BUFG/BUFGCTRLs	4	16	25%
Number of DSP48A1s	36	58	62%

VI CONCLUSION :

The logic utilizing the RAM GRID developed for 64 point DITFFT/IDITFFT can be used for high speed computations . similar analogous logic can be developed for larger no of points like 128 and 256 and still larger number of points . This type of logic utilizes less resources and gives more speed because of the multiple levels of pipelining and

processing involved and can be of great use in designing large calculation high speed computation circuitry.

REFERENCES

[1] Clare Huggett, Kaushik Maharatna, Kolin Paul " On the Implementation of 128 - pt FFT/IFFT for High – Performance WPAN" Circuits and Systems 2005 IEEE international Symposium May 2005 vol 6 pp 5513 – 5516.
[2] Ahmed Saeed, M. Elbably, G. Abdelfadeel, and M. I. Eladawy " Efficient FPGA Implementation of FFT/IFFT processor." International Journal of Circuits Systems and signal Processing Issue 3 Volume 3 2009 pp 103 – 110.
[3] J Greg, Nash, Centar "A High Performance scalable FFT" WCN IEEE 2007 , Wireless Communication And networking conference March 2007 pp 2367 – 2372.
[4] http://www.atmel.com/dyn/resources/prod_documents/doc1132.pdf
[5]http://darcy.rsgc.on.ca/ACES/ICE4M/FixedPoint/FixedPointRepresentationFractionalMath.pdf.
[6] http://www.digitalsignallabs.com/fp.pdf Fixed Point arithmetic an Introduction. Randy Yates
[7] http://www.ipcores.com/
[8] Sundance multiprocessor technology ltd at site www.sundance.com.
[9] Datasheet DS260, xft_ds260 on www.xilinx.com.
[10] Digital Signal Processing Principles , Algorithms and Applications by John G Proakis, Dimitris G Manolakis..PHI Edition IV Chapter 8 Pages 511 to 536
[11] Analog and Digital Signal Processing by Ashok Ambardar.
[12] Mentor Graphics VHDL Reference Manual.
[13]Digital Signal Processing by M H HAYES. Tata MacGraw Hill Chapter 7 Pages 7.1 to 7.12
[14] k. Maharatna, E . Grass and U . Jagdhold , "A NOVEL 64-POINT FFT/IFFT PROCESSOR FOR IEEE 802.11(A) STANDARD" 2003 IEEE International conference on Accoustics, Speech, Signal Processing. Issue date 6- 10 April 2003. 0-7803-7663-3/03©2003 IEEE PP II321 – II324. Vol2.
[15] Digital Signal Processing M H Hayes . Tata MacGraw Hill Chapter 11 Pages 11.3 to 11.8
[16] http://www.dspguide.com/ The Scientists and Engineers Guide to Digital Signal Processing by Stewen W Smith Chapter 4 Pages 1 to 6.
[17] http://www.bores.com/courses/intro/chips/6_data.htm
[18] L. R. Rabiner and B. Gold, "Theoryand Application of Digital SignalProcessing," Prentice-Hall, 1975.
[19] E. H. Wold and A. M. Despain,"Pipeline and Parallel-pipeline FFTProcessors for VLSI implementation,"IEEE Trans. Computers, C-33(5), pp.414-426, 1984.
[20] A. M. Despain, "Fourier Transform Computer Using CORDIC Iterations,"IEEE Trans. Computers, C-23(10), pp.993-1001, 1974.
[21] G. Bi and E. V. Jones, "A Pipeline FFT Processor for Word-sequential Data,"IEEE Trans. Acoust. Speech, Signal Processing, Vol. 37(12), pp. 1982-1985, 1989.
[22] S. He and M. Torkelson, "A NewApproach to Pipeline FFT Processor,"The 10th International ParallelProcessing Symposium(IPPS), pp. 766 - 770, 1996.
[23] H. L. Groginsky and G. A. Works, "A Pipeline Fast FourierTransform," IEEE trans. on Computers, Vol. C–19(11), pp.1015–1019, 1970.
[24] K. Maharatna, E. Grass and U. Jagdhold" A NOVEL 64-POINT FFT/IFFT PROCESSOR FOR IEEE 802.11(A) STANDARD" ICASSP 2003ppII – 321 to II - 324
[25]Hang Liu, Hanho Lee"High Speed Four Parallel 64 point Radix 2^4 MDF FFT/IFFT processor for MIMO –OFDM Systems." The 23rd International Technical Conference on Circuits/Systems Computers and Communications ITC – CSCC 2008 pp 1469 - 1472
[26] T.TIRUMALA KOTESWARA RAO, S. SARATH CHANDRA" Implementation of 64-Point FFT Processor Based on Radix-2 Using Verilog" IJERT vol2 ISSUE 10 OCT 2013pp2811 – 2815.