Implementation of FFT Algorithm for OFDM Wireless LANs

Katta. Mangarao, Sirigineedi.Srividya

Abstract— The explosive growth of 802.11-based wireless LANs has attracted interest in providing higher data rates and greater system capacities. Among the IEEE 802.11 standards, the 802.11a standard based on OFDM (Orthogonal Frequency Division Multiplexing) modulation scheme can be defined to address high-speed and large-system-capacity challenges. Although DSPs have been used to implement the 802.11a standard, they can only support limited data rates due to the lack of global parallelism found at the application level. Hence, it is still a major challenge to develop a software implementation for the 802.11a standard on a DSP to meet the high-data-date requirements.

To meet these requirements this paper proposes the design of 256 point Radix 2^2 SDF FFT (Fast Fourier Transform) architecture for OFDM wireless LANs because FFT and IFFT (Inverse Fast Fourier Transform) are major hardware requirements in OFDM. The data transmission is by interfacing the QPSK modulator with IFFT at transmitter and QPSK demodulator with FFT, at receiver for easy of designing. The design has been coded in verilog. The simulations and synthesis is carried out by using Xilinx 9.2i simulator.

Keywords— OFDM,Radix 2^2 Algorithm, FFT, QPSK

1 INTRODUCTION

The OFDM modulation is cost effectively realized by the Inverse Fast Fourier Transform (IFFT) that enables the use of a large number of subcarriers— up to 1024 according to the Mobile WiMAX system profiles – to be accommodated within each OFDMA symbol. Before transmission, each OFDMA symbol is extended by its cyclic prefix at the transmitter. At the receiver end, cyclic prefix is discarded and OFDM demodulation is applied through the Fast Fourier Transform (FFT).

The Fast Fourier Transform (FFT) is most efficient algorithm to compute the Discrete Fourier Transform (DFT) and performs most important operations in modern digital signal processing and communication systems. The pipeline architecture of FFT is a special class of FFT algorithms which can compute the FFT in a sequential manner; it achieves real-time behaviour with nonstop processing when data is continually fed through the processor.

Classical implementation of the FFT/IFFT architecture, with digital signal processors (DSPs), requires a sequential algorithm. This increases the execution time. On the other side, the present programmable circuits, like an FPGA, uses a tens of thousands of lists and triggers during process, resulting of parallel processing system, puts the FPGA computing speed at a significant advantage over DSPs.

.This paper presents the implementation of a 256 point $\boldsymbol{Radix\ 2^2}$ singlepath delay feedback pipelined FFT/IFFT processor. And also focus on the QPSK for data constellation.

2 ARCHITECTURE AND DESIGN METHODOLOGY

2.1 $\boldsymbol{Radix\ 2^2}$ Decimation in Frequency FFT Algorithm

The notion of $\boldsymbol{Radix\ 2^2}$ algorithm is used to clearly reflect the structural relation with radix-2 algorithm and the identical computational requirement with radix-4 algorithm.

The Discrete Fourier Transform (DFT) of a sequence *x(n), n=0, 1,…, N-1* is defined as: *X(k), k=0, 1, …,N-1*

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N{}^{nk} \quad \text{, for } 0 \le K < N \qquad (1)$$

Where $W_N = e^{-j2\pi/N}$ denotes the primitive *Nth* root of unity.The input sequence x(n) and its DFT is X(k).A 3-dimensional linear index map is applied by.

$$n =< \frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3 >$$
$$K =< K_1 + 2K_2 + 4K_3 >$$

This yield

$$X(K_1 + 2K_2 + 4K_3) = \sum_{n_3=0}^{\frac{N}{4}-1} \sum_{n_2=0}^{1} \sum_{n_1=0}^{1} x\left(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3\right) W_N{}^{\left(\frac{N}{2}n_1+\frac{N}{4}n_2+n_3\right)(K_1+2K_2+4K_3)} \qquad (2)$$

$$X\,(K_1 + 2K_2 + 4K_3) = \sum_{n_3=0}^{\frac{N}{4}-1} \left[H(K_1, K_2, n_3) W_N{}^{n_3(K_1+2K_2)}\right] W_{\frac{N}{4}}{}^{n_3K_3} \qquad (3)$$

where n_1, n_2, n_3 are the index terms of the input sample n and K_1, K_2, K_3 are the index terms of the output sample k and where $H(K_1, K_2, n_3)$ is expressed in eqn.

$$H(K_1, K_2, n_3) = \left[x(n_3) + (-1)^{K_1}\, \mathrm{x}\left(n_3 + \frac{N}{2}\right)\right] + (-\mathrm{j})^{(K_1+2K_2)} \left[\mathrm{x}\left(n_3 + \frac{N}{4}\right) + (-1)^{K_1}\, \mathrm{x}\left(n_3 + \frac{3N}{4}\right)\right] \qquad (4)$$

After this simplification, a set of four DFTs of length *N/4* is obtained. Each term in equation (4) represents a Radix-2 butterfly (BFI), and the entire equation(4) also represents Radix-2 butterfly (BFII) with trivial multiplication by *–j*. *N=16*-point Radix 2^2 decimation in frequency (DIF) method used for the pipelined architecture is shown in Fig.1 and observe that the

inputs are in normal order whereas the outputs are in permuted (digit-reversed) order. The trivial multiplication by – j is represented with pentagons between the BFI and BFII.The full twiddle factor multipliers (TFM) are required to compute the multiplication by the twiddle factor $W_N^{n_3(K_1+2K_2)}$ after the two butterflies.

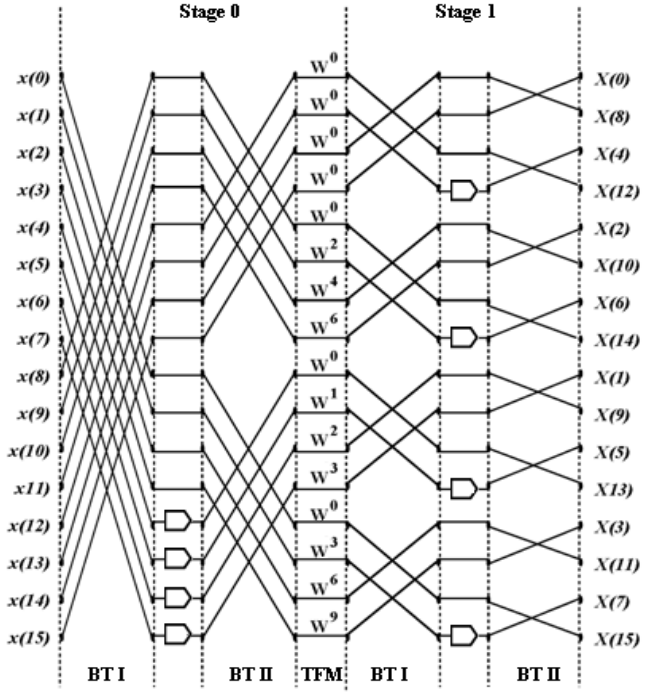

Fig.1 Flow graph of $\boldsymbol{Radix\ 2^2}$ DIF FFT algorithm

For N=16

The *N*-point FFT processor has $\log_4(N)$-stages with *i* is the stage number. A typical stage consists of BFI, BFII, delay-feedback, ROM, and TFM. A $\log_2(N)$ counter is used to control the processor. The formation of the last stage is different according to the size of FFT; if *N* is power of 2, the last stage is formed by BFI only. But if *N* is power of 4, the last stage formed by BFI and BFII.

2.2 BFI structure

The detailed structure of BFI is shown in Fig. 2. The input comes from the preceding component, TFM. The output fed to the next component, normally BFII. In first $N/2^{i+2}$ cycles, multiplexors direct the input data to the feedback registers until they are filled (position "0"). On next $N/2^{i+2}$ cycles, the multiplexors select the output of the adders/subtractors (position "1"), and it computes a 2- point DFT with incoming data and the data stored in the feedback registers.

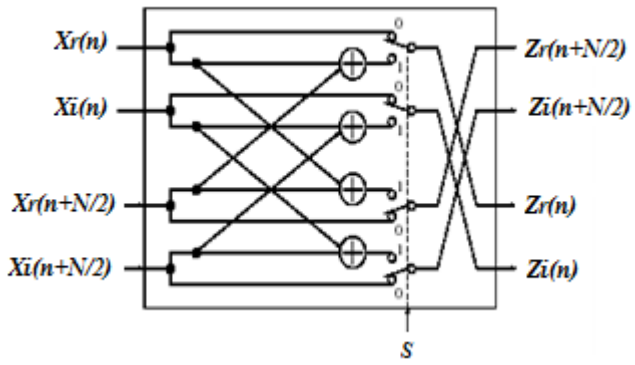

Fig: 2. BFI for $R2^2$ SDF FFT

2.3 BFII structure

The detailed structure of BFII is shown in Fig. 3. The input comes from the previous component, BFI. The output fed to the next component, normally TFM. In first $N/2^{i+2}$ cycles, multiplexors direct the input data to the feedback registers until they are filled (position "0"). In next $N/2^{i+2}$ cycles, the multiplexors select the output of the adders/subtractors (position "1"), The a 2-point DFT with incoming data computes by butterfly and the data stored in the feedback registers.

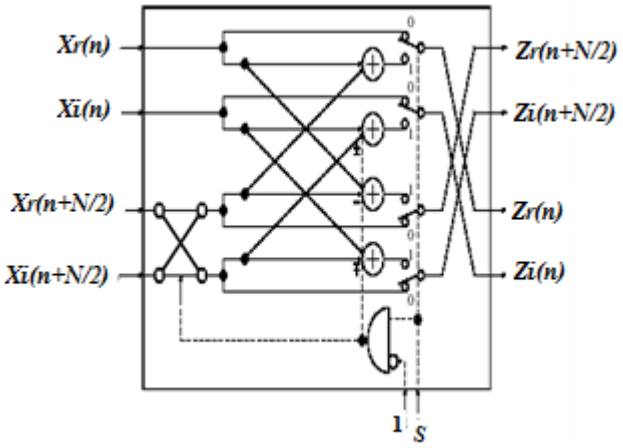

Fig: 3. BFII for $R2^2$ SDF FFT

The multiplication by –*j* involves real-imaginary swapping and inversion of sign. The multiplexors are used to swapping the real-imaginary terms and the sign inversion is handled by switching the adding-subtracting operations by mean of multiplexing. When there is a need for multiplication by –*j*, all multiplexors switches to position "1", the real-imaginary data are swapped and the adding-subtracting operations are switched.

2.4 TFM Structure

A six clock cycle fully-pipelined complex-multiplier has been implemented to multiply the twiddle factor by the output of BFII. According to Equation (5), the algorithm of multiplying the twiddle factor (c + js) by BFII output (Zr + jZi) uses four multipliers, one adder, and one subtractor.

$$(Z_r + jZ_i).(c + js) = (Z_r.c - Z_i.s) + j(Z_i.c + Z_r.s) \ldots (5)$$

Twiddle factor generator is a key component in IFFT/FFT computation. There are many popular generation methods for

twiddle factor generation. The twiddle factors are generated using MATLAB according to equation (6), converted to fixed point, and then stored in ROM.The twiddle factors at the i^{th} stage ,with $i = 0,1,\ldots,(\log_4 N) - 2$ is given by $W_i = \{u_x\}; x = 0,1,\ldots,{}^{N}/_{2^{2i}}$ with $u_x = e^{-j2\pi v/N}$

$$v = \begin{cases} 0, & 0 \le x < a \\ 2^{2i+1}.(\mathrm{x} - \mathrm{a}), & a \le x < 2a \\ 2^{2i}.(\mathrm{x} - 2\mathrm{a}), & 2a \le x < 3a \\ 3.2^{2i}(\mathrm{x} - 3\mathrm{a}), & 3a \le x < 4a \end{cases} \quad (6)$$

$$\text{With,} \quad a = \frac{N}{2^{2i+1}}$$

2.5 Delay-Feedback Structure

In order to reuse the presented hardware, the delay feedback is used. This mechanism provides a solution where the first input to the butterfly is delayed until the second input is reached, after that the calculations can proceed. This is achieved by accepting part of the data stream into the butterfly elements, but an alternative of computing on the block, it sends to a feedback delay line by mean of multiplexers. By the time, the data presented again at the input of the butterfly. The delay-feedback mechanism is implemented by First-in First-out shift register. The feedback delay at the i^{th} stage is given in equation.(7).

$$\lambda = \frac{N}{2^{2(i+1)}} \quad (7)$$

3 Radix 2^2 FFT ARCHITECTURE

An implementation of the R2^2 SDF architecture for N=256 is shown in Fig.4, observe the connection of the data-path to R2SDF and the reduced number of multipliers. This uses two types of butterflies, one is same as that in R2SDF, the other contains the logic to implement the trivial twiddle factor multiplication, as shown in Fig.2 & 3 respectively. The synchronization control of the processor is very simple due to the spatial regularity of Radix 2^2 algorithm,. A $\log_2(N)$ bit binary counter serves two purposes: address counter for twiddle factor reading in each stages and synchronization controller. With the help of the butterfly structures shown in Fig.2&3, the scheduled operation of the R 2^2 SDF processor in Fig. 4 is as follows. The 2-to-1 multiplexors in the first butterfly module change to '0'th position, and the butterfly is idle on first *N/2* cycles, The input data from left is shifted to the shift registers until they are filled. The multiplexors switches to '1'st position, on next *N/2* cycles, the butterfly computes a 2-point DFT with incoming data and the data stored in the shift registers.

$$Z1(n) = x(n) + \mathrm{x}\left(n + \frac{N}{2}\right)$$

$$Z1\left(n + \frac{N}{2}\right) = x(n) - \mathrm{x}\left(n + \frac{N}{2}\right), 0 \le n < \frac{N}{2}$$

The twiddle factors are applied at the butterfly output Z1(n), and Z1(n + N/2) is sent back to the shift registers to be "multiplied" in still next N/2 cycles when the first half of the next frame of time sequence is loaded. The operation of the second butterfly is same as that of the first one and the trivial twiddle factor multiplication has been implemented by real-imaginary swapping with a commutator and controlled

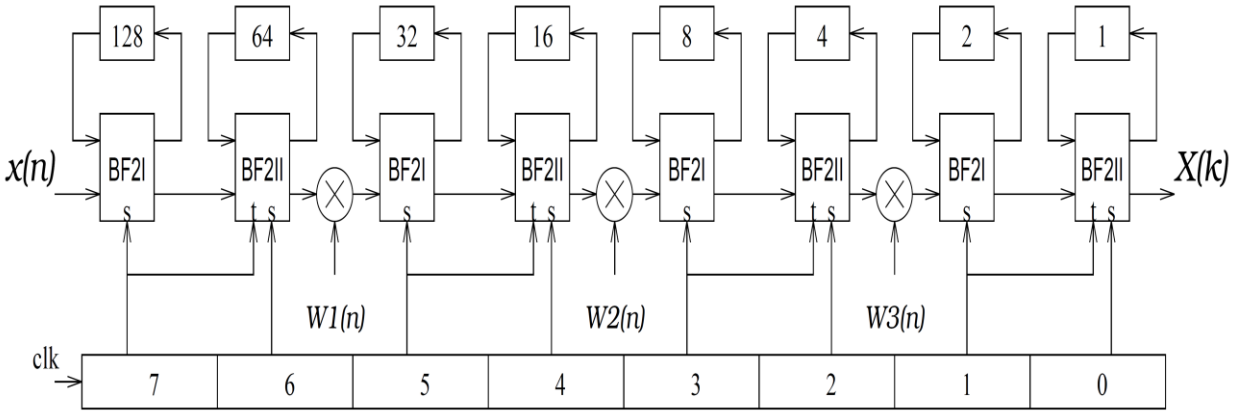

Fig.4 Radix- 2^2 SDF pipeline FFT architecture for N=256

add/subtract operations, as in Fig.2&3,it requires two bit control signal from the synchronizing counter. The data then goes through a full complex multiplier, with the utility of 75%, The complete DFT transform result streams out to the right, in bit-reversed order after N-1 clock cycles,. The next frame of transform can be computed without pausing due to the pipelined processing of each stage.

4 IMPLEMENTATION OF THE PROPOSED FFT IN OFDM COMMUNICATION SYSTEM

The fundamental principle of the OFDM system is to decompose the high rate data stream (bandwidth=*W*) into *N* lower rate data streams and then to transmit them simultaneously over a large number of subcarriers. The IFFT is used for modulation and the FFT are used for demodulation. The data accomplishes the result of first level of radix-4 DFT. Further processing repeats this pattern with the distance of the input data decreases by half at each consecutive butterfly stages. constellations on the orthogonal subcarriers in OFDM system is done by QPSK modulation for easy of designing. The transmitter and receiver blocks contain the FFT and IFFT modules. The FFT processor must finish the transform within the time to serve the purpose in the OFDM system. This FFT architecture effectively fits into the system

5 QUADRATURE PHASE SHIFT KEYED (QPSK) MODULATION

An OFDM carrier signal is the sum of a number of orthogonal sub-carriers, with message data on each subcarrier being independently modulated commonly using some type of quadrature phaseshift keying (QPSK).This composite message signal is typically used to modulate a main RF carrier. $s[n]$ is a serial stream of input binary digits. At transmitter these are first demultiplexed into N parallel streams by inverse multiplexing, and each one mapped to a (possibly complex) symbol stream using some modulation constellation (QPSK). Note that the constellations may be different, so some streams may carry a higher bit-rate than others.

The implementation of QPSK involves changing the phase of the transmitted waveform. Each finite phase change represents the unique digital data. A PM waveform can be generated by using the digital data to change the phase of a signal while its frequency and amplitude remains constant. A QPSK modulated carrier undergoes four distinct changes in

phase that are represented as symbols and can take on the values of $\pi/4$, $3\pi/4$, $5\pi/4$, and $7\pi/4$. Each symbol represents two binary bits of data.

6 IMPLEMENTATION IN VERILOG

VERILOG Hardware Description Language (VERILOG) was introduced by Gateway Design Automation in 1984 as a proprietary hardware description and simulation language. Logic-circuit structures created by VERILOG synthesis tools directly from VERILOG behavioral descriptions, and target them into a preferred technology for realization. By using VERILOG, It is easy to design, simulate, and synthesize anything form a simple combinational circuit to a complete microprocessor based system on a chip. It started out as documentation and modeling language, allows the behavior of digital-system designs to be precisely specified, simulated and language specification allows multiple modules to be stored in a single text file. All these features of VERILOG will help better in simulation and synthesis of proposed architecture. The OFDM Transmitter and Receiver presented above has been fully coded in VERILOG Hardware Description Language (VERILOG). Once the design is coded in VERILOG, the simulations and synthesis report is carried out by using Modelsim XEIII 6.2c compiler and the Xilinx Foundation ISA Environment 9.2i.

7 RESULTS

The main characteristics and resource requirements of several pipeline FFT architectures are given in Table.1. Resource utilization percentage measures the computational efficiency how often the resources are in an idle state versus an active state. As shown in the table.1,by observing that the radix-4 Single-Path Delay-Commutator (R4SDC) and radix-2^2 Single Path Delay Feedback (R 2^2 SDF) architectures provide the highest Computational efficiency and were selected for implementation. The R4SDC architecture is interesting due to the computational efficiency of its addition; however the controller design is complex. The R2^2 SDF architecture has a Simple controller but a less efficient addition scheme. These designs are both radix-4 and scalable to an arbitrary FFT size N (N is a power of 4).

Architecture	Complex multipliers	Complex adders	Memory size	Control logic	Comp. Utilization	
					add/sub	Multiplier
R2SDF	$\log_2 N - 2$	$2\log_2 N$	$N-1$	simple	50%	50%
R4SDF	$\log_4 N - 1$	$8\log_4 N$	$N-1$	medium	25%	75%
R4SDC	$\log_4 N - 1$	$3\log_4 N$	$2N-2$	complex	100%	75%
R2MDC	$\log_2 N - 2$	$2\log_2 N$	$3N/2-2$	simple	50%	50%
R4MDC	$3(\log_4 N - 1)$	$8\log_4 N$	$5N/2-4$	medium	25%	25%
(Proposed)	$\log_4 N - 1$	$4\log_4 N$	$N-1$	simple	75%	75%

Tabl:1Hardware resource requirement comparison of pipeline FFT architectures.

The RTL schematic diagram for QPSK modulator is shown in fig.5.

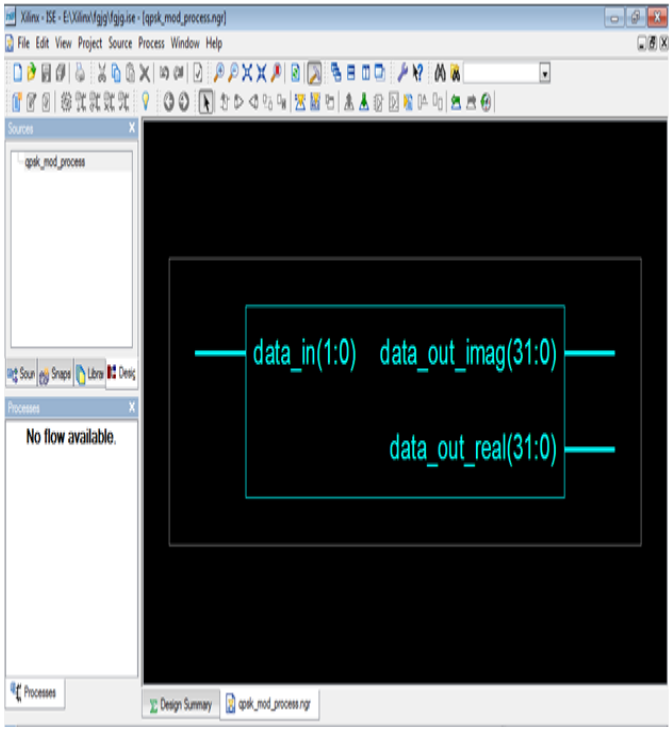

Fig.5 RTL schematic diagram for QPSK modulator

The RTL schematic diagram for FFT is shown in fig.6

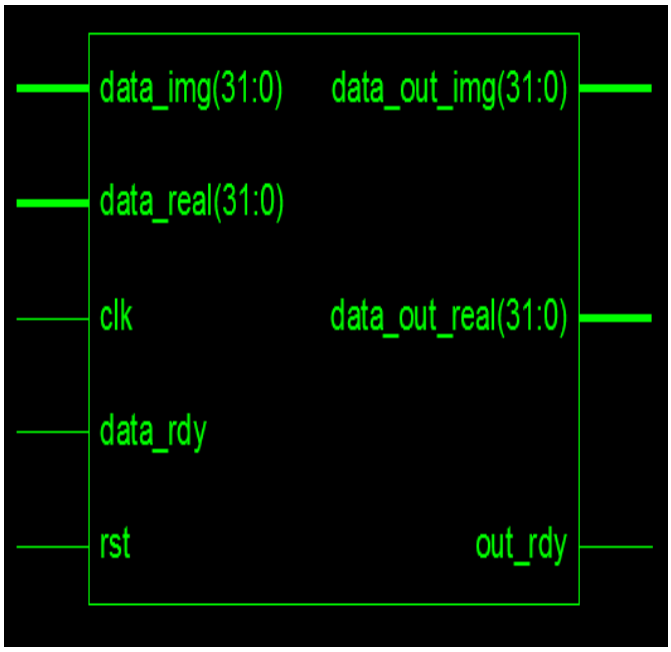

Fig.6 RTL schematic diagram for FFT

Table:2 Timing summary

Minimum period	4.784ns (Maximum Frequency: 209.03MHz)
Minimum input arrival time before clock	9.570ns
Maximum output required time after clock	4.283ns

The timing summary for FFT is given in table:2.The minimum time required to process the sample with Radix 2^2 FFT is 4.784ns and the maximum operating frequency is 209.03 MHz.

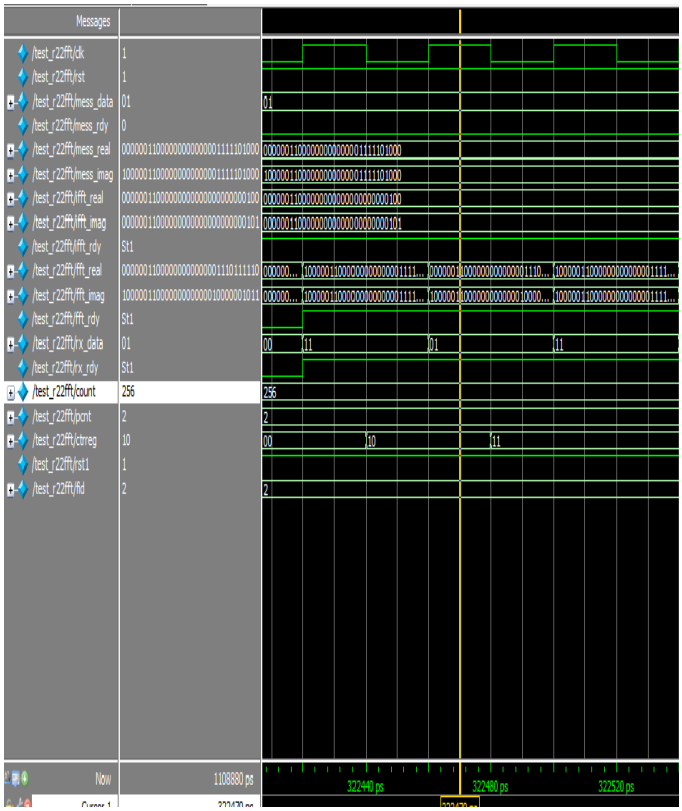

Fig.7. Simulated waveform showing that data transmission in OFDM System

The Fig.7 shows the simulated wave form of the OFDM communication system. When mess_ready is 'Low' there is no data transmission. When it is 'High', then the input data constellation is done and applied to the IFFT at the transmitter. When Rx_data is 'High' then the data is received by FFT at the receiver. Pcnt (packet count) indicates that the no. of packets is sent to the receiver.

8 CONCLUSIONS AND FUTURE WORK

This paper describes the design of Radix2^2 single-path delay feedback pipelined FFT processor with N=256 points to provide the higher data rates in OFDM. The proposed architecture has three main advantages (1) Fewer butterfly iteration to reduce power consumption, (2) Pipeline of Radix2^2 butterfly to speed up clock frequency, (3) Even distribution of memory access to make utilization efficiency of components. In summary, the speed performance of this design easily satisfies most application requirements of mobile WiMAX 802.16e, 802.11-based wireless LANs those uses OFDM modulated wireless communication system.

The implemented design gives an easy way to increase the number of points of FFT as well as IFFT by imposing simple modification. Future work can includes the development of complete OFDM system and upgrade it to a multiple input multiple outputs (MIMO) system.

REFERENCES

[1] Chi-Hong Su and Jen-Ming Wu, "*Reconfigurable FFT Design for Low Power OFDM Communication Systems"*, Proc. of the 2006 IEEE 10th Intern. Symposium on Consumer Electronics (ISCE'06), 28 June - 1 July *2006*, Russia, pp.1-4, 2006.

[2] J.-Y. Oh and M.-S. Lim, "New radix-2 to the 4th power pipeline FFT processor," *IEICE Transactions on Electronics*, vol. E88-C, no. 8, pp. 1740–1746, 2005.

[3] K. Harikrishna, T. Rama Rao and Vladimir A. Labay, "*A RADIX-2 Pipeline FFT for Ultra Wide Band Technology*",International Conference on Computer & Network Technology (ICCNT), conference proceedings published by World Scientific Press, Singapore. Chennai, India, Jul 24-28, 2009.

[4] S. He and M. Torkelson, "*A new approach to pipeline FFT processor"*, 10th Int. Parallel Processing Symp. (IPPS'96), p.766–770, 1996.

[5] M.S Minallah and G. Raja, "*Real Time FFT Processor Implementation", 2nd International Conference on Emerging Technologies*, IEEE—ICET 2006. Peshawar, Pakistan 13-14 November, Pages: 192-195, 2006.

[6] Ahmed Saeed, M. Elbably, G. Abdelfadeel, and M. I. Eladawy **"***Efficient FPGA implementation of FFT/IFFT Processor"*. Issue 3, Volume 3, 2009.

[7] Zhijian Sun, Xuemei Liu, and Zhongxing Ji, **"***The Design of Radix-4 FFT by FPGA,***"** International Symposium on Intelligent Information Technology Application Workshops, 2008, pp.765-768.

[8] Y. Wang, Y. Tang, Y. Jiang, J. G. Chung, S. S. Song and M. S. Lim, **"***Novel memory reference reduction methods for FFT implementation on DSP processors," IEEE Trans Signal Processing*, vol. 55, no. 5, pp.2338-2349, May 2007.

[9] Ainhoa Cortés, Igone Vélez, Juan F. Sevillano, and Andoni Irizar, "*An Approach to Simplify the Design of IFFT/FFT Cores for OFDM Systems,"* IEEE Transactions on Consumer Electronics, No. 1, February 2006, pp. 26-32.

[10] R. Prasad, "*OFDM for Wireless Communications Systems"*, London: Artech House Inc., 2004.

[11] T. Sansaloni, A. P´erez-Pascual, V. Torres, and J. Valls, "*Efficient pipeline FFT processors for WLAN MIMO-OFDM systems," Electronics Letters*, vol. 41, no. 19, pp. 1043–1044, 2005.

[12] S. Sukhsawas and K. Benkrid, "*A high-level implementationof a high performance pipeline FFT on Virtex-E FPGAs,"* in*Proceedings of the IEEE Computer Society Annual Symposiumon VLSI (ISVLSI '04)*, pp. 229–232, February 2004.

[13] Zhou and D. Hwang, "*Implementations and optimizations of pipeline FFTs on Xilinx FPGAs*," in Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConFig '08), pp. 325–330, 2008.

[14] P. Jackson, C. Chan, C. Rader, J. Scalera, and M. Vai. "*A systolic FFT architecture for real time FPGA systems"* In High Performance Embedded Computing Conference (HPEC04), Sept. 2004.

[15] J.C.Chih and S.G. Chen," An Efficient FFT Twiddle Factor Generator ", submitted to Eusipco-2004.

[16] J. Mar, Y. Lin, T. Lung, and T. Wei, "*Realization of OFDM Modulator and Demodulator for DSRC Vehicular Communication System Using FPGA Chip*," International Symposium on Intelligent Signal Processing and Communications (ISPACS'06), 2006, pp. 477-480

[17] Zhijian Sun, Xuemei Liu, and Zhongxing Ji, "*The Design of Radix-4 FFT by FPGA,"* International Symposium on Intelligent Information Technology Application Workshops, 2008, pp.765-768

[18] C. Lin, Y. Yu, and L. Van, "*A low-power 64-point FFT/IFFT design for IEEE 802.11a WLAN application"* in Proc. International Symposium on circuit and systems, 2006, pp. 4523-4526

[19] Ainhoa Cortés, Igone Vélez, Juan F. Sevillano, and Andoni Irizar, "An Approach to Simplify the Design of IFFT/FFT Cores for OFDM Systems," IEEE Transactions on Consumer Electronics, Vol.52, No. 1, FEBRUARY 2006, pp. 26-32.

AUTHORS

- KATTA.MANGARAO is currently pursuing masters degree program in "Digital Electronics and Communication Systems" in "Chaitanya Institute of

Technology and Sciences", Kakinada, JNTUK, AP, India
PH-7306203010,
Email: charankumar.katta@gmail.com