FPGA Implementation of Mixed Radix CORDIC FFT

Izhar Ansar Ahmed[1] Dr. M. S. Ali[2]
[1]M.E. Scholar [2]Principal & Professor
[1]Department of Electrical & Electronics Engineering
[1]Department of PG Studies, PRMCEAM, Badnera, Amravati [2]PRMCEAM, Badnera, Amravati

Abstract— In this Paper, the architecture and FPGA implementation of a Coordinate Rotation Digital Computer (CORDIC) pipeline Fast Fourier Transform (FFT) processor is presented. Fast Fourier Transforms (FFT) is highly efficient algorithm which uses Divide and Conquer approach for speedy calculation of Discrete Fourier transform (DFT) to obtain the frequency spectrum. CORDIC algorithm which is hardware efficient and avoids the use of conventional multiplication and accumulation (MAC) units but evaluates the trigonometric functions by the rotation of a complex vector by means of only add and shift operations. We have developed Fixed point FFT processors using VHDL language for implementation on Field Programmable Gate Array. A Mixed Radix 8 point DIF FFT/IFFT architecture with CORDIC Twiddle factor generation unit with use of pipeline implementation FFT processor has been developed using Xilinx XC3S500E Spartan-3E FPGA and simulated with maximum frequency of 157.359 MHz for 16 bit length 8 point FFT. Results show that the processor uses less number of LUTs and achieves Maximum Frequency.

Key words: FFT, CORDIC, FPGA, MIXED RADIX, PIPELINE FFT

I. INTRODUCTION

The Fast Fourier Transform (FFT) and its inverse (IFFT) are very important algorithms in signal processing, software-defined radio, and Orthogonal Frequency Division Multiplexing (OFDM). Implementation of the FFT/IFFT algorithm, with digital signal processors (DSP processors), requires a sequential algorithm [4]. This slows down the execution time. On the other hand, FPGA utilizes parallel processing system, putting the FPGA computing speed at a significant advantage over DSP processors The outputs of the shorter transforms are reused to compute many outputs, thus the total computational cost becomes less. Many designs have used pipelining and folding techniques to increase latency and speed.

Among these hardware-efficient algorithms, there is a class of iterative solutions for trigonometric and other functions that use only shifts and adds to perform. CORDIC, an acronym for COordinate Rotation Digital Computer. The trigonometric CORDIC algorithms were originally developed as a digital solution for real time navigation problems [2]. The CORDIC was introduced in 1956 by Jack E. Volder as a highly efficient, low-complexity, and robust technique to compute the elementary functions. The CORDIC algorithm has many applications including the 8087 math coprocessor and radar signal processors. CORDIC rotation has also been proposed for computing Discrete Fourier, Discrete Cosine, Discrete Hartley and Z-transforms, filtering, Singular Value Decomposition. The CORDIC is a digital computer for real time computation. CORDIC is extremely popular in hardware accelerators and also in SIMD realizations and almost all function calculators employ CORDIC. CORDIC is generally faster than other approaches when a hardware multiplier is unavailable or when the number of gates required to implement is to be minimized. [2] [9].

II. LITERATURE SURVEY

In literature review, we will discuss about earlier implementations and there results. Ahmed Saeed et al. implemented of radix-2^2 single-path delay feedback pipelined FFT/IFFT processor using hardware description language VHDL on an Xilinx XC5VSX35T and simulated up to 465MHz and exhibited execution time of 0.135μS for transformation length 256-point [4]. This results show that the processor achieves higher throughput and lower area and latency. Miguel A. Sanchez et al. presented an in-depth study of the implementation and characterization of fast Fourier transform (FFT) pipelined architectures suitable for broadband digital channelized receivers [13]. Trini Sansaloni et al. presented FFT Spectrum Analyzer Project for Teaching Digital Signal Processing with FPGA Devices [17].

Ray Andraka presented survey of CORDIC algorithms for FPGA based computers in which unrolled CORDIC paper is discussed [12].

A. Mixed Radix FFT

Mixed Radix FFT is special type of architecture where different stages of FFT are implemented by using different Radix stages so that numbers of stages are reduced. Mixed Radix FFT architecture minimizes computation cost and number of complex multiplication. While calculating FFT using Radix-2 method, it can be concluded that the even-numbered points and the odd-numbered points are computed independently. This leads to the possibility of using different computational methods for different independent parts of the algorithm which will reduce computational complexity. Structure of 8 point radix 4/radix 2 DIF FFT is as shown in figure 1.

B. Pipeline FFT

In this implementation we have used Radix 4 single delay feedback pipeline architecture as given in figure 2. Every FFT stage performs 4 point FFT and forward output to next 4 point FFT stage in this way radix 4 stage is implemented [8]. Radix-4 Single-path Delay Feedback (R4SDF) was proposed as a radix-4 version of Radix-2 Single Delay Feedback (R4SDF), employing CORDIC [18].

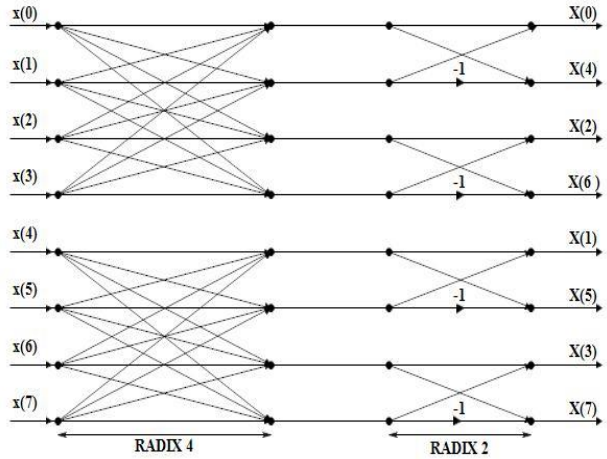

Fig. 1: 8 point radix 4/radix 2 DIF FFT

The utilization of multipliers has been increased to 75% due to the storage of 3 out of 4 radix-4 butterfly outputs. However, the utilization of the radix-4 butterfly, which is fairly complicated and contains at least 8 complex adders, is dropped to only 25%. It requires $\log_4 N$-1 multipliers, $\log_4 N$ full radix-4 butterflies. Hardware requirement of R4SDF for N point FFT is $\log_4 N$-1 multipliers and 8 $\log_4 N$ adders [10].

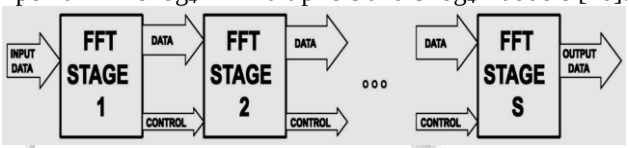

Fig. 2: Pipelined implementation of FFT

C. Cordic Algorithm

The CORDIC arithmetic processor chip is designed and implemented to perform various functions possible in rotation and vectoring mode of circular, linear, and hyperbolic coordinate systems. There are two computing modes in CORDIC, 1) Rotation mode 2) Vectoring mode. The rotation mode is used to perform the general rotation by a given angle θ. The vectoring mode computes unknown angle θ of a vector by performing a finite number of micro rotations [3] [5].

If a vector V with coordinates (x, y) is rotated through an angle ϕ as shown in figure 3 then the new vector V′ can be obtained with coordinates (x′, y′) where x′ and y′ can be obtained using x, y and ϕ. The desired angle ϕ is decomposed into several combinations of micro rotations. These micro rotations are based on standard which is referred as reference angle. The vector V can be represented as $\begin{bmatrix} x \\ y \end{bmatrix}$ and after rotation by an angle ϕ to the position V′= $\begin{bmatrix} x' \\ y' \end{bmatrix}$.

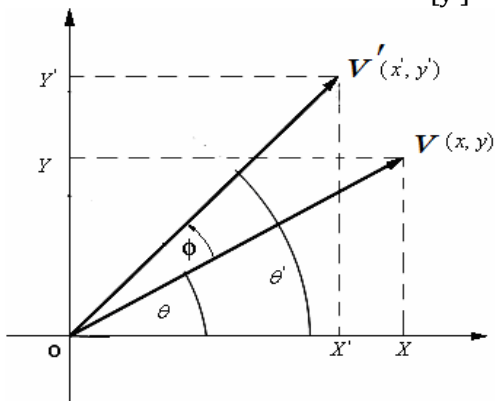

Fig. 3: Rotation of a vector V by the angle ϕ

After performing calculations we get equations as

$$x_{i+1} = x_i - y_i d_i 2^{-i}$$

$$y_{i+1} = y_i + x_i d_i 2^{-i}$$

$$z_{i+1} = z_i - d_i \tan^{-1}(2^{-i})$$

$$d_i = -1 \text{ if } z_i < 0$$

$$d_i = +1 \text{ otherwise}$$

III. PROPOSED APPROACH

We have implemented fixed point Pipeline CORDIC based Mixed radix 8 point DIF FFT and IFFT on FPGA and realized in VHDL language. In the table I, Elementary angles in phase magnitude and its hexadecimal conversion is given. The block diagram of proposed CORDIC FFT is as given in figure 4. The selector block is used for computing respective memory of input samples. Now when block Ram gets write Address signal from address generator block, it saves both address along with respective input samples. The 4 point FFT block has butterfly unit within it.

i	$\tan\phi_i = 2^{-i}$	$\phi_i = \tan^{-1}(2^{-i})$	ϕ_i in radians	ϕ_i in hexadecimal
0	1	45^0	0.7854	020000
1	0.5	26.565^0	0.4636	012E40
2	0.25	14.036^0	0.2450	09FB4
3	0.125	7.125^0	0.1244	05111
4	0.0625	3.576^0	0.0624	028B1
5	0.03125	1.7876^0	0.0312	0145D
6	0.015625	0.8938^0	0.0156	0A2F
7	0.0078125	0.4469^0	0.0078	0518
8	0.0039062	0.2234^0	0.0039	028C
9	0.0019531	0.1117^0	0.0019	0146

Table I: Elementary angles in phase magnitude and its hexadecimal conversion

The entire model is made of address generation unit, control unit, block RAM unit, division unit, 4-point FFT butterfly unit, rotation angle unit and CORDIC twiddle factor generation unit. Address generator unit assigns addresses to different lines at different point of FFT and hence it is very important for working of FFT. Block RAM unit stores intermediate and final real and imaginary results. Multiplying factor unit, rotation unit and CORDIC angle generator unit are used for twiddle factor generation. 4 point FFT unit operate as Radix 4 butterfly unit gives output. We have implemented 4 stage pipeline of 4 point FFT architecture for computing Radix 4 – 16 point DIF FFT by using pipeline architecture we are using minimum area of FPGA for our implementation that is our proposed work uses less resources.

When a start signal is asserted, at the same time, both to 4 Point FFT and Rotation factor generator block, the FFT block sends a signal to CORDIC block for computing necessary twiddle factors consisting of sine-cosine terms. FFT block is controlled by Rotation factor generator block. Now when address generator block sends read address signal to RAM, it sends stored input data samples along with

memory path in FFT block. Finally this twiddle factors are applied to the output of the butterflies, and a bit reverse output is obtained.

When Radix 2/Radix 4 – 8 point DIF FFT is to be implemented then second stage of Radix 2 DIF FFT is used. While implementing IFFT divide by eight and sixteen operation is performed by shifting data right by 3 and 4 places respectively and padding (bit stuffing) zero toward left.

dataR <= "000" & rtemp0(7 downto 3); -- divide by 8

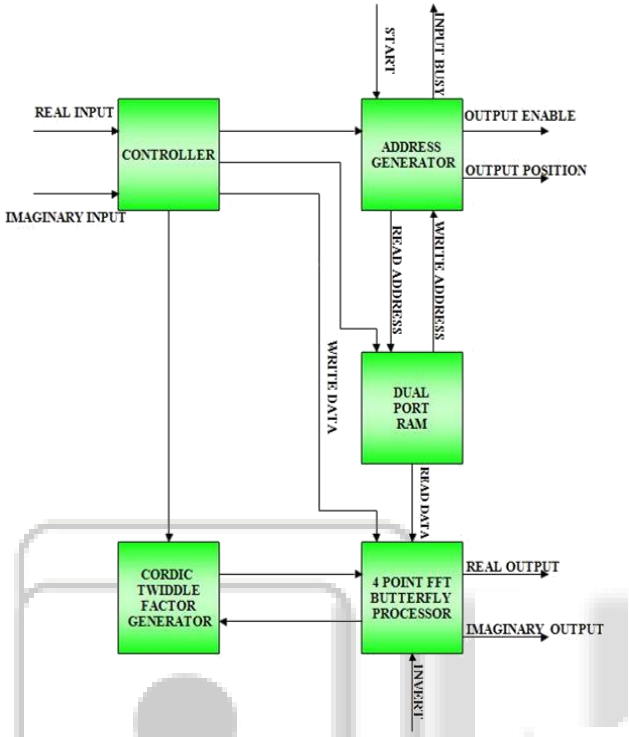

Fig. 4: Block diagram of proposed CORDIC FFT

In figure 5 FFT Processor is shown. Clock, Reset, start, Invert, Real data input (15 : 0) and Imaginary data input (15 : 0) are input signals. Input busy, Output data enable, Output Position (4 : 0), Real data output (17 : 0) and Imaginary data output (17 : 0) are output signals. State machine structure has been created for applying inputs in different states. Invert signal is used for FFT and IFFT implementation by giving 0 and 1 input respectively. State machines are commonly used in designing the control logic for ASIC or FPGA devices. They represented a very easy and understandable method for specifying sequence of actions over time.

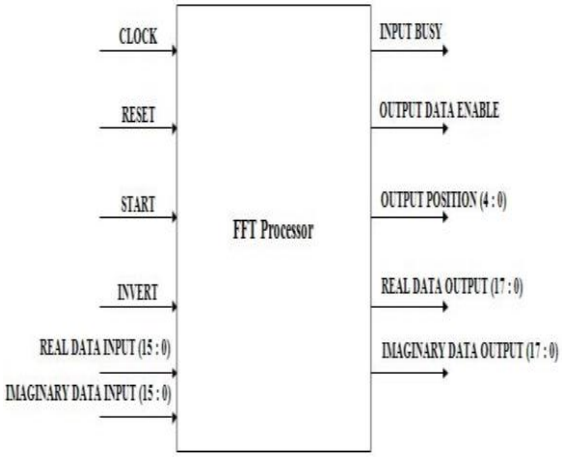

Fig. 5: FFT Processor

IV. RESULTS AND DISCUSSION

We have used ModelSim SE PLUS 6.3f for VHDL simulation, Xilinx ISE 14.7 design suite for FPGA synthesis, Xilinx XC3S500E for FPGA implementation and MATLAB R2012a for verification. Clock signal is forced in ModelSim for simulation and Start signal for starting FFT operation. Reset signal for clearing all input and data real and data imaginary for giving input. Output position signal is used for indicating address of output data to show bit reversal in ModelSim simulation. Output data enable signal indicates when output is available after all calculations are done.

The ModelSim simulation is shown in figure 6 for FFT[1, 2, 3, 4, 1, 2, 3, 4] input and output can be seen in ModelSim from rtemp0 to rtemp7 for real output and itemp0 to itemp7 for imaginary output in bit reverse pattern and verification is performed by MATLAB as in figure 7.

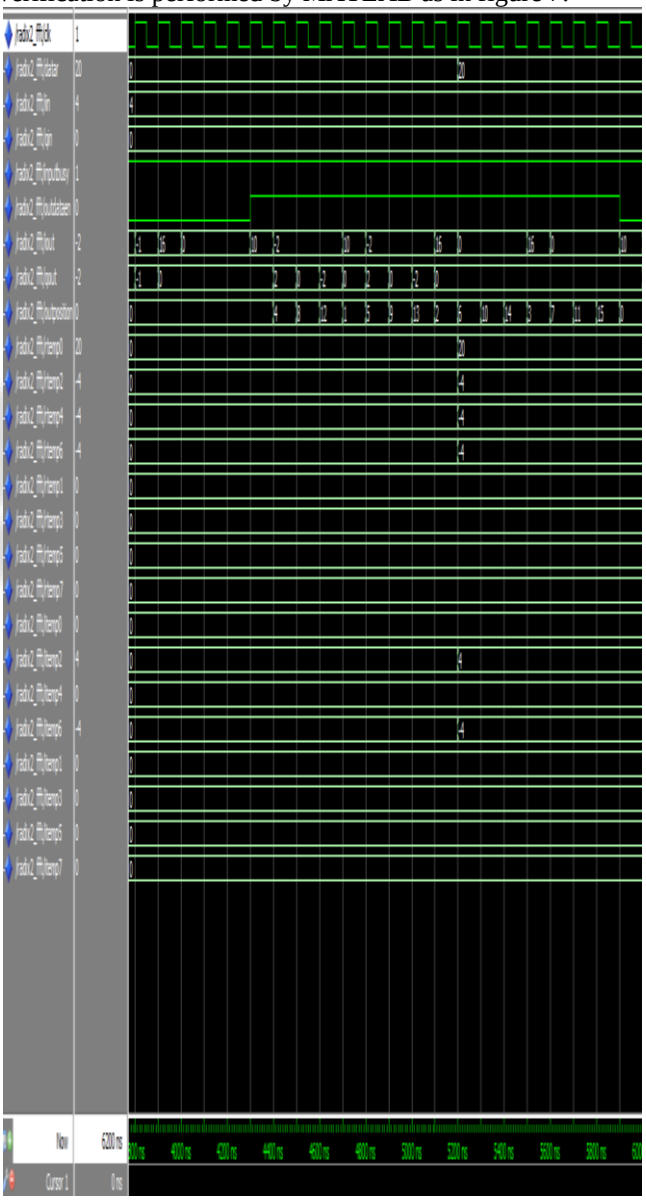

Fig. 6: ModelSim simulation of FFT(1,2,3,4,1,2,3,4)

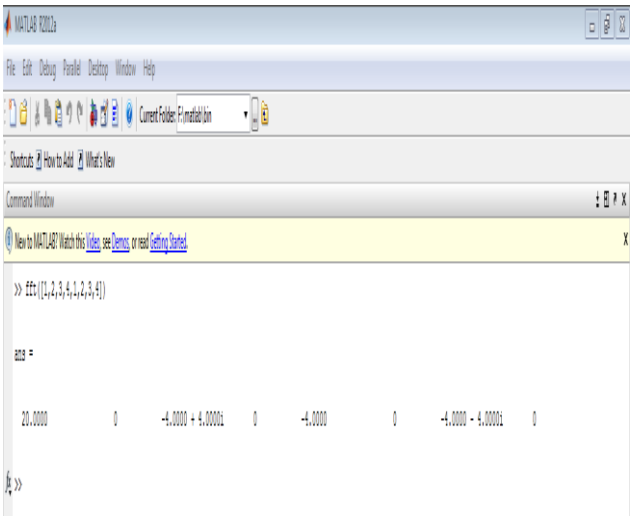

Fig. 7: MATLAB output of FFT(1, 2, 3, 4, 1, 2, 3, 4)

Fig. 8: Outputs on FPGA kit of FFT(1, 2, 3, 4, 1, 2, 3,4)

Figure 8 shows output on FPGA kit and we have verified our results. In Table II Comparison table we have compared our results with reference [15] and reference [16], we can see that we have achieved maximum frequency compared to other results on Spartan 3E kit. Numbers of look up tables required are much less in our proposed work than in earlier works.

Implementation	No. of 4 input LUTs used	No. of occupied slices used	No. of Bonded IOBs used	No. of BUFG - MUXs Used	Maximum Frequency (MHz)
Reference [15]	487	-	36	12	-
Reference [16]	1202	147	7	1	140.308
Mixed-Radix FFT/IFFT Proposed Work	959	619	9	1	157.359

Table II: Comparison Table

V. CONCLUSION

The CORDIC algorithm is a powerful and widely used tool for digital signal processing applications and can be implemented using PDPs (Programmable Digital Processors). But a large amount of data processing is required because of complex computations. This affects the cost, speed and flexibility of the DSP systems. So, the implementation of DIF FFT using CORDIC algorithm on FPGA is the need of the day as the FPGAs can give enhanced speed at low cost with a lot of flexibility. This is due to the fact that the hardware implementation of a lot of multipliers can be done on FPGA which are limited in case of PDPs. In this dissertation the CORDIC based Pipeline Radix2/Radix4 – 8 point DIF FFT/IFFT is simulated using Modelsim which is then used for simulation of Fast Fourier Transform. Then the implementation is done on XILINX Spartan 3E FPGA. The results of FPGA are verified by MATLAB.

It can be concluded that the designed RTL model for CORDIC based Pipeline Radix2/Radix4 – 8 point DIF FFT/IFFT and Radix 4 – 16 point DIF FFT/IFFT function is accurate and can work for many applications.

VI. FUTURE SCOPE

As there will be always some improvement that can be done, same goes to this project. The future scope should include the implementation of Floating point CORDIC FFT processor with more number of points, implementation of radix 2^2 architecture, CORDIC based implementation and simulation of Discrete Cosine Transform, Discrete Hartley Transform and Singular Value Decomposition. The above proposed architectures can be used in various signal processing, image processing and communication engineering applications

ACKNOWLEDGEMENT

I would like to express my gratitude to the following people for their support and guidance for helping me to complete this work. I would like to thank my respected M.E. guide and Principal Dr. M. S. Ali, who provided me constructive criticism and a positive feedback during this project work and for providing me necessary facilities. I am indebted to Dr. N. V. Thakur, Dean (PG Studies, PRMCEAM, Badnera) of

M.E. Department and other teaching non-teaching staff for their help. Without them and their co-operation completion of this project work would have been inevitable and their presence behind me is totally indispensable. My sincere thanks to the staff of "Electrical and Electronics Engineering" and "Electronics and Telecommunication Engineering". I am also thankful to my parents and my sister whose best wishes are always with me.

REFERENCES

[1] James W. Cooley and John W. Tukey, "An Algorithm for Machine Calculation of Complex Fourier Series," Mathematic of Computation, vol. 19, pp. 297 – 301, 1965.
[2] J. E. Volder, "The CORDIC trigonometric computing technique," IRE Trans. Electronic Computing, Volume EC-8, pp 330 - 334, 1959.
[3] S. He and M. Torkelson, "A new approach to pipeline FFT processor", in Proceedings of the 10th International Parallel Processing Symposium (IPPS '96)", pp. 766–770, Honolulu, Hawaii, USA, April 1996.
[4] S. Ahmed, M. Elbably, G. Abdelfadeel, and M. Eladawy, "Efficient FPGA implementation of FFT/IFFT Processor", International Journal of Circuits, Systems and Signal Processing, pp. 103-110, 2009.
[5] P. Duhamel and H. Hollmann, "Split-radix FFT algorithm," Electron. Lett., vol. 20, no. 1, pp. 14 – 16, Jan. 1984.
[6] P. Duhamel and M. Vetterli, "Fast Fourier Transforms: A Tutorial Review and A State of The Art," IEEE Signal Processing Society, vol. 4, no. 19, pp. 259 – 299, 1990.
[7] B. Lakshmi and A. S. Dhar, ''CORDIC Architectures: A Survey,'' Hindawi publication, volume 2010.
[8] Bin Zhou,Yingning Peng and David Hwang, "Pipeline FFT Architectures Optimized for FPGAs," Hindawi publication, volume 2009.
[9] Roberto Sarmiento, F´elix Tobajas, Valent´ın de Armas, Roberto Esper-Cha´ın, Jos´e F. L´opez, Juan A. Montiel-Nelson and Antonio N´u˜nez, "A CORDIC Processor for FFT Computation and Its Implementation Using Gallium Arsenide Technology," IEEE Transactions on VLSI systems, Vol. 6, no. 1, March 1998.
[10] Jack Volder, "The Birth of CORDIC," Journal of VLSI Signal Processing, pp. 101–105, 2000.
[11] Douglas Jones, "Decimation in Frequency Radix 2 FFT," Connexions module: m12018.
[12] R. Andraka, "Survey of CORDIC algorithms for FPGA based computers," Proceedings of the 1998 ACM/SIGDA sixth international symposium on FPGAs, pp 191-200, Monterey, California, Feb.22-24, 1998.
[13] Miguel A. Sanchez et al., "Implementing FFT-Based Digital Channelized Receivers on FPGA Platforms," IEEE transactions on aerospace and electronic systems, 2008.
[14] Trini Sansaloni et al., "FFT Spectrum Analyzer Project for Teaching Digital Signal Processing With FPGA," IEEE, 2007.
[15] Remya Ramachandran and Vanmathi K., "Simulation of Radix 2 FFT using Xilinx," IJASTR, Vol.1, January 2014.
[16] Adriana Bonilla, "Design and implementation of FFT algorithm in FPGA," SBDT, 2012.
[17] Walther J.S, "Unified algorithm for elementary functions", Spring Joint Computer Conference, pp 379 - 385, 1971.
[18] Keshab K. Parhi, "VLSI Digital Signal Processing Systems: Design and Implementation," Wiley, 1999.
[19] Uwe Meyer-Baese, "Digital Signal Processing with Field Programmable Gate Arrays," Springer, 2007.
[20] J. G. Proakis and D. G. Manolakis, "Digital signal processing principles, algorithms and applications," Prentice Hall, Delhi, 2010.
[21] Spartan-3E Starter Kit Board User Guide UG230 (v1.0) March 9, 2006.
[22] Fast Fourier Transform v3.2 Xilinx Core, DS260 Aug. 2005.
[23] ISE In-Depth Tutorial UG695 (v 12.3) September 21, 2010.

