Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Radix-4 FFT Architecture

Ajay S. Padekar
Electronics department,
BVDU, College of Engineering,
Pune, India

Prof. S. S. Belsare
Department of E & TC,
BVDU, College of Engineering,
Pune, India

Abstract- In this paper, we proposed design architecture of an efficient radix-4 FFT and CORDIC algorithm for FPGA implementation of FFT. Due to use of radix-4 speed gets doubled than radix-2 in FFT computation. For twiddle factor multiplication CORDIC algorithm is used, which will reduce the computation time and make process faster. The Co-ordinate Rotation Digital Computer Algorithm (CORDIC) offers the opportunity to calculate all the required functions in a rather simple and elegant fashion. The design architecture is written in VHDL code using Modelsim and XILINX ISE tools.

Keywords- FFT, Radix-4, CORDIC, Vector mode, Rotation mode

I. INTRODUCTION

The Fast Fourier Transform (FFT) and its inverse (IFFT) are very useful algorithms in signal processing. This paper explains the implementation and simulation of radix-4. Due to radix-4 and CORDIC, FFT can accomplish more computing speed than DSP's, and also achieve cost effective performance with less development time [1].

This paper discusses levels of computation in Butterfly structures. And for fast calculation Twiddle Factor is introduced in the mathematical operation. This twiddle factor have different methods to generate it, as Co-ordinate Rotation Digital Computer, (CORDIC) algorithm, pipelined-CORDIC algorithm, polynomial-based approach, ROM-based scheme, and the recursive function generators. In radix-2 the samples are multiple of two while in radix-4 it is multiple of 4. Hence radix-4 technique is much faster than Radix-2[1].

The Fourier transform when applied to digital (Discrete) rather than an analog (continuous) signal is called as Discrete Fourier Transform. FFT (Fast Fourier Transform) is a improved version of the DFT that is applied when the number of input samples in the signal is power of two. FFT computation requires N*log2(N) operations, whereas DFT requires N2 operations, So the FFT is more faster than DFT. Hence Fast Fourier Transform (FFT) becomes more important in applications which require fast signal processing. Using this transform time domain signal is converted into frequency domain; where filtering and correlation can be performed easily [2].

The FFT algorithm is chosen to consider the execution speed, hardware complexity, flexibility and precision for many applications. However, for real time systems the speed of the execution is the main concern. Several architectures have been developed like: single memory architecture, dual-memory architecture, cached memory architecture, array architecture, and pipelined architecture. Pipelined architectures are suitable for most of the application due to smaller latency with low power consumption [2].

The rest of this paper is organized as follows: Section II describes Radix-4 FFT algorithm next in Section III simulation result of Radix-4 is given next in Section IV Block diagram of CORDIC is discussed after that we conclude the paper in Section V.

II. RADIX-4 FFT ALGORITHM

Fig. 1 shows an example of Radix-4 decimation in time (DIT) the method used for N=16-points FFT algorithm. As shown in FFT flow-graph inputs are in normal order while the outputs are in digit-reversed order. At input side the samples are taken from time domain which are processed with radix-4 FFT and get equivalent components in frequency domain. The numbers over flow lines indicates the twiddle factor to be multiplied with the samples [5].

Figure 2 (a) and (b) shows the basic butterfly structure of radix-4 which have four inputs and four outputs; inputs are as x(n), x(n + n/4), x(n + n/2) and x(n + 3n/4) outputs are in digit reversed order X(k). FFTs in which N = r*k, and where the butterflies used in each stage are the same, are called radix-r algorithms.

A radix-r FFT uses N/r radix-r butterflies for each stage and has logrN stages. That's why in this case for 16 point radix 4 FFT requires k=2 stages. The signal flow graph of radix-4 DIT butterfly operation is illustrated in Figure 3.
The arithmetic kernel of radix-4 DIT FFT is the butterfly operation defined as-

$X_0 = P_0 + W_1P_1 + W_2P_2 + W_3P_3$
$X_1 = P_0 - jW_1P_1 - W_2P_2 + jW_3P_3$
$X_2 = P_0 - W_1P_1 + W_2P_2 - W_3P_3$
$X_3 = P_0 + jW_1P_1 - W_2P_2 - jW_3P_3$

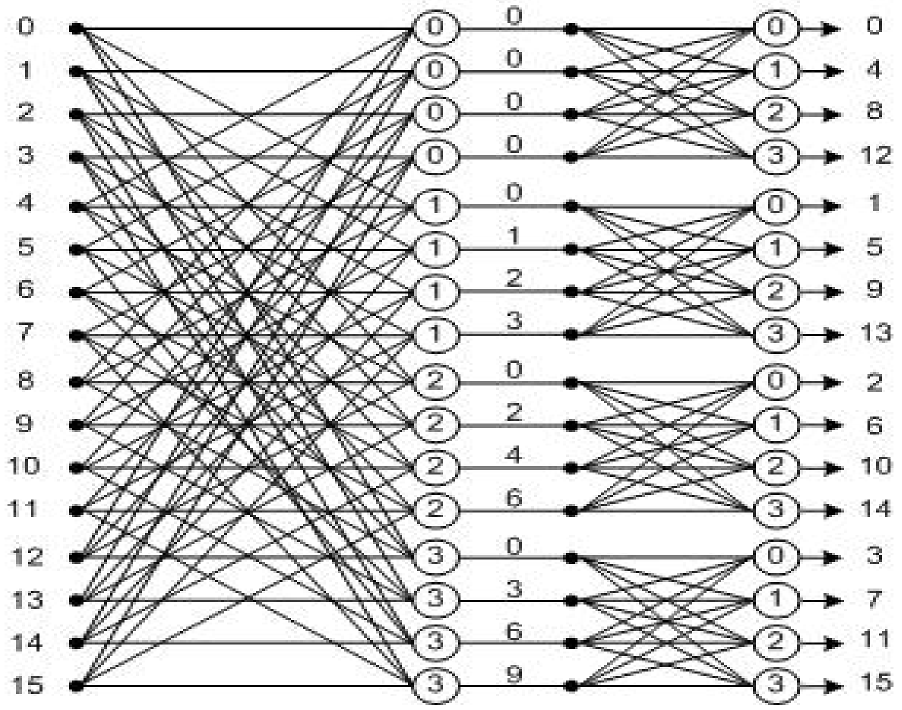


Figure 1: 16-point radix-4 DIT algorithm with input in normal order and output in digit-reversed order [7].

Radix-4 algorithms have a computational advantage over radix-2 algorithms because one radix-4 butterfly does the work of four radix-2 butterflies, and the radix-4 butterfly requires only three complex multipliers compared to four complex multipliers of four radix-2 butterflies.

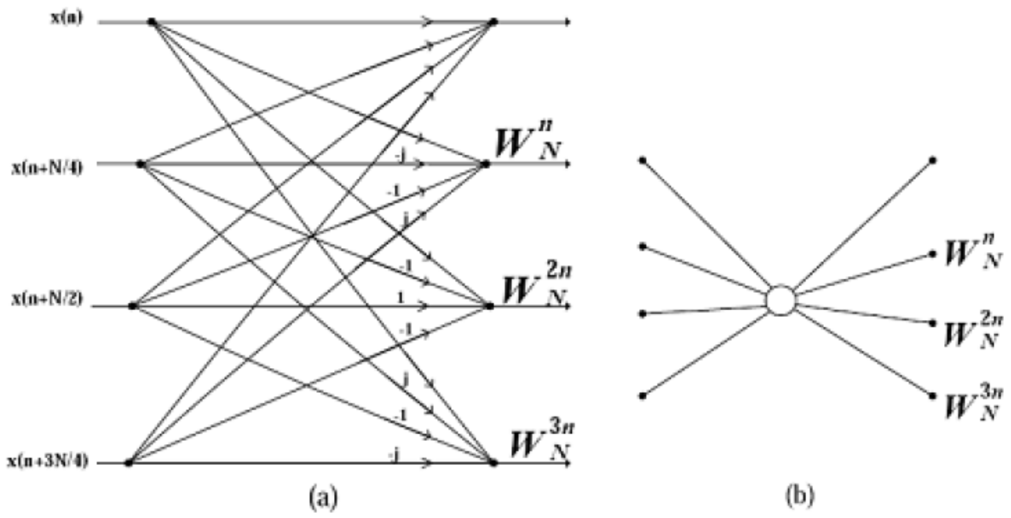


Figure 2: The basic butterfly for radix-4 FFT algorithm [1].

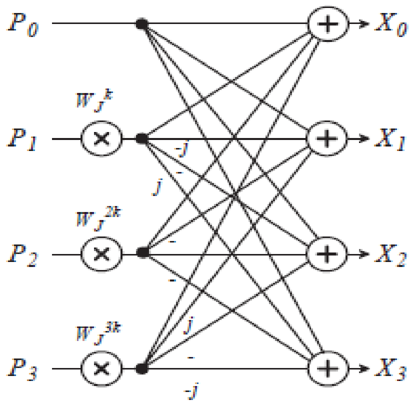


Figure 3: Butterfly structure to show operation [7].

III. CORDIC

The block diagram of CORDIC-based FFT processor is shown in Figure 4. The entire model consist of address generation block, control unit block, dual port RAM block, 4-point butterfly unit block and CORDIC twiddle factor generation block. This model is depicted by fixing the parameter, sampling points and accuracy to meet the actual specifications [2].

The FFT processor discussed here is based on radix-4 DIT algorithm in which the in-place computation is used to get an efficient use of the memories. To perform in-place computation simultaneously, a dual-port RAM has been used. For the data storage, reading and writing control unit involves the timing control to make the corresponding data and rotating factor values pass into the butterfly and CORDIC computing block in sequence. Counter is used to generate data and addresses of the 'twiddle factor'. The address generation logic is very easy and does not force limit on the throughput of the system [2].

There are two modes of CORDIC one is vector mode and the other is rotation mode. Set of trigonometric functions are performed by vector and rotation mode. The basic idea of CORDIC algorithm is to decompose a rotation into a set of micro-rotations. By decomposing the CORDIC algorithm into a sequence of operational stages the pipelined CORDIC arithmetic unit can be obtained.

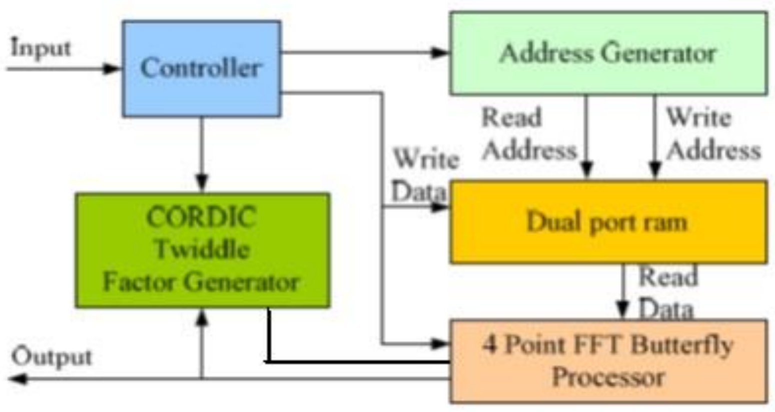


Figure 4: CORDIC based FFT architecture [2].

In this paper, one uses x and y to perform as the real part and imaginary part of input and output vectors. In rotation mode, co-ordinate components of the original vector are computed from given co-ordinate components of vector and an angle of rotation after rotation by a given angle. In the vector mode, magnitude and angular argument of the original vector are computed from given co-ordinate components of a vector.

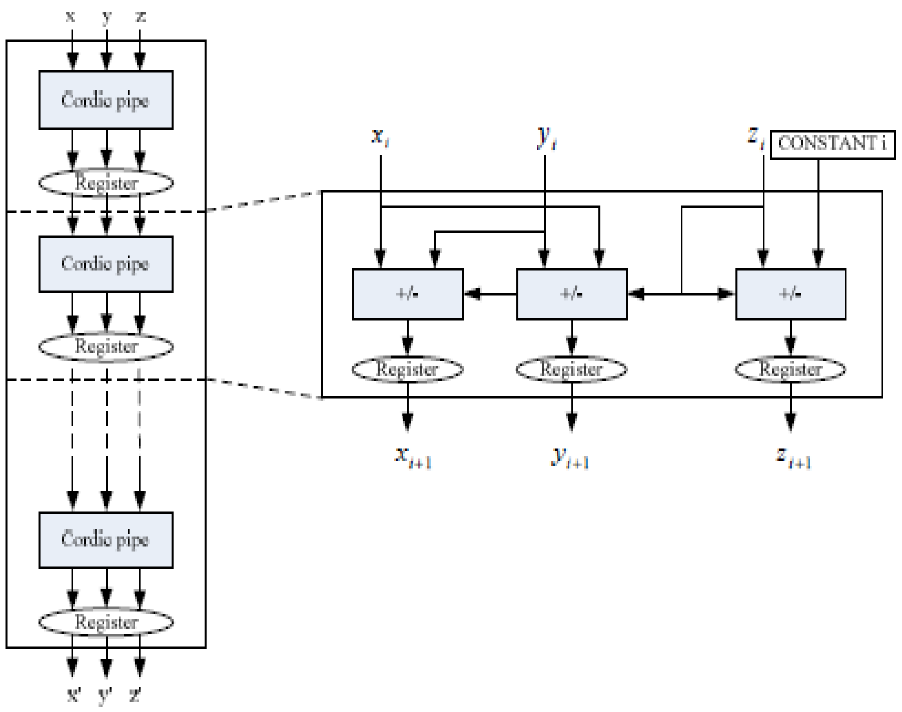
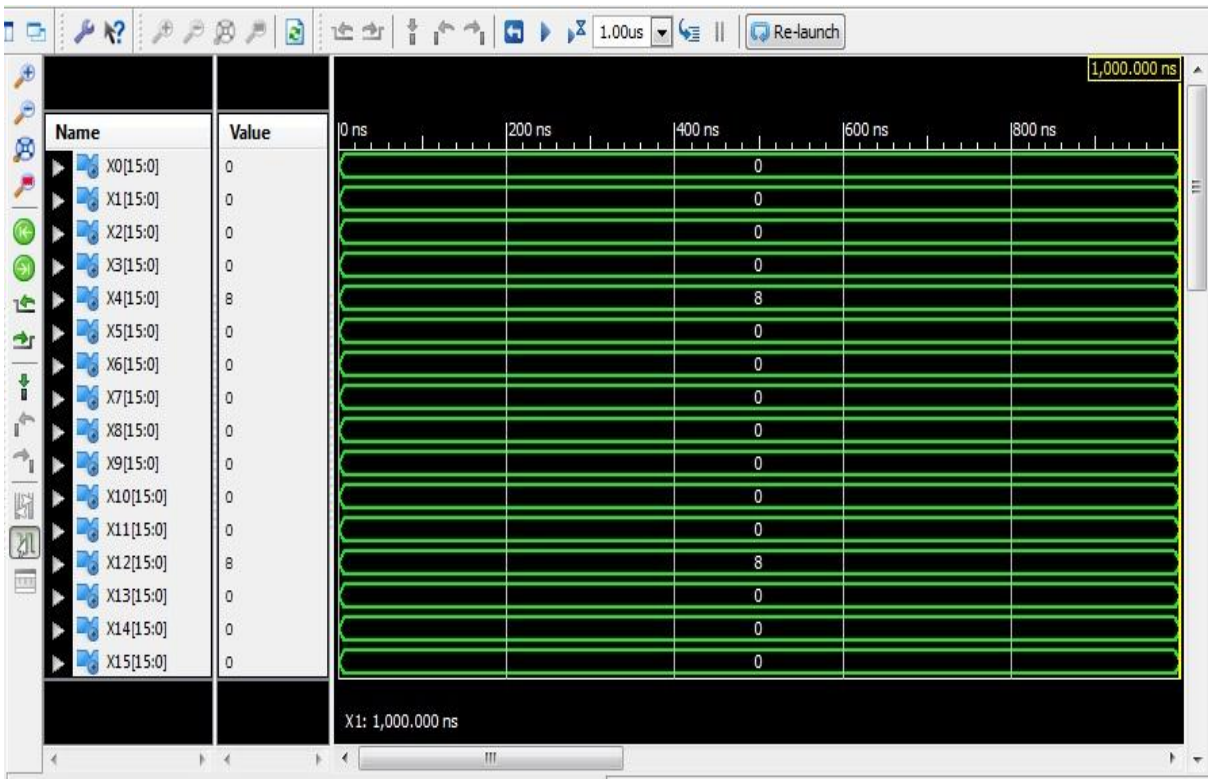


Figure 5: CORDIC unit architecture [2].

IV. SIMULATION RESULTS

In this paper VHDL code is developed for radix-4 algorithm for that decimation in time model is selected. Butterfly module is the main concern in code. As an input 16 sampled values of time domain signal each of 16 bit binary value are taken. Input is S = [1, 0, -1, 0, 1, 0, -1, 0, 1, 0, -1, 0, 1, 0, -1, 0]. As shown in Figure [6] simulation of radix-4 FFT gives certain finite values which are in frequency domain. Simulation is done in Xilinx ISE design suit 13.2. Clock is taken as an input for FFT calculation; on positive edge of the clock result is simulated. Test bench module is developed for verification of the result. X0[15:0] is nothing but 16 bit output.



Name	Value
X0[15:0]	0
X1[15:0]	0
X2[15:0]	0
X3[15:0]	0
X4[15:0]	8
X5[15:0]	0
X6[15:0]	0
X7[15:0]	0
X8[15:0]	0
X9[15:0]	0
X10[15:0]	0
X11[15:0]	0
X12[15:0]	8
X13[15:0]	0
X14[15:0]	0
X15[15:0]	0

X1: 1,000.000 ns

Figure 6: Result of 16-point radix-4 FFT in Xilinx ISE design suit 13.2

V. CONCLUSION

In this paper Radix-4 FFT processor architecture is studied and simulated in VHDL using Xilinx software. Radix-4 algorithms have an advantage over radix-2 algorithms because a single radix-4 butterfly does the work of four radix-2 butterflies which will reduce the steps of computation and end up with more speed in processing the data. For generation of twiddle factor CORDIC algorithm is used. CORDIC is useful when number of inputs go on increasing it will reduce the complexity in computation as number of input samples gets increased.

REFERENCES

[1] 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.

[2] Ren-Xi Gong, Jiong-Quan Wei and Dan Sun, Ling-Ling Xie, Peng-Fei Shu and Xiao-Bi Meng; "*FPGA Implementation of a CORDIC-based Radix-4 FFT Processor for Real-Time Harmonic Analyzer*"; 2011 eventh International Conference on Natural Computation.

[3] Oppenheim Alan V., Schafer Ronald W., with Buck John R.; "*Discrete Time Signal Processing*" pp. 629-661.

[4] Proakis John G., Manolakis Dimitris G.; "*Digital Signal Processing*" pp. 448-493.

[5] Shi Jiangi; Tian Yinghui; Wang Mingxing; Yang Zhe; "*A Novel design of 1024-point pipelined FFT processor based on CORDIC algorithm*"; Intelligent System Design And engineering Application (ISDEA) 2012 second International Conference on Digital Object Identifier.

[6] Long Pang; Bocheng Zhu; He Chen Electronics; "*Design and Realization of small point FFT processor based on twiddle factor classification*" Communications and Control (ICECC) 2011 International Conference On Digital Object Identifier.