ISSN:0975-9646

Design of a Cordic Based Radix-4 FFT Processor

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

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

Abstract- **The digital signal processing has been dominated by microprocessors with enhancements such as special addressing modes and single cycle multiply-accumulate instructions. While these processors offer extreme flexibility with low cost, they are often not fast enough for most of DSP tasks. The arrival of reconfigurable logic computers offer hardware solutions for higher speeds at costs which are less than traditional software approach. Unfortunately, algorithms used for these micro-processor based systems cannot be mapped into hardware. In such hardware efficient algorithms there is a class of solutions for trigonometric functions that use shifts and adds in its operation. The trigonometric functions are found out using vector rotations, while other functions like square root are realized using an incremental expression. The trigonometric algorithm is called Coordinate Rotation Digital Computer known as CORDIC. This CORDIC algorithm is further used in radix-4 FFT for faster computation. While converting samples taken in real time into equivalent frequency domain samples FFT computation is used in which twiddle factor computation is done using CORDIC.**

Keywords- **FFT, Radix-4, CORDIC, Complex Multiplier.**

I. INTRODUCTION

FFT processor based on FPGA has widely used in signal processing, image processing and other fields. It has the characteristic with high parallel and throughput, and can meet some high real time computing by using the FPGA device. Compared with commonly FFT processor by DSP or PC processing, it has a certain advantage in speed.in this paper in-place computation is used and for complex multiplication CORDIC algorithm is used which enhance the speed of FFT computation.[1][7]

Using FFT, signals can be moved to the frequency domain where filtering and correlation can be performed with fewer operations. The design of a radix-4 FFT processor based on CORDIC algorithm presented in this paper. The processor is implemented in a FPGA that is characteristics of high effectiveness, low cost, short development cycle, and a satisfactory performance. The selection of the CORDIC algorithm for implementing basic butterfly operation for the FFT which excludes the need of storing the twiddle factors and angles saves a lot of hardware compared to other techniques. A dual port memory block and corresponding addressing scheme are used for in-place data access. The address generator block needed for fetching data from and writing outcomes into the dual port memory in proper sequence, is also combined within the chip which includes the controller as well.[2][7]

The proposed CORDIC algorithm reduces the need of storing the twiddle factors as well as angles, due to that significant area saving with no negative impact on its performance. The rest of the paper is arranged as follows. In section II Radix-4 FFT and fundamentals of CORDIC algorithm are described. Then in section III architecture of CORDIC based FFT is described. The experimental results are provided in section IV and finally paper is concluded in section V.[1][4]

II. FFT AND CORDIC ALGORITHM

A. Radix-4 FFT Algorithm-

The N-point discrete Fourier transform is defined by-

$$X(k) = \sum_{n=0}^{N-1} x(n).W_N^{nk}\ k = 0,1,\ldots.,N-1, W_N^{nk} = e^{-j\frac{2\pi}{N}nk} \tag{1}$$

The radix-4 decimation-in-time FFT algorithm is described briefly in the following equations. The N-point input sequence is split into four subsequences, x(4m), x(4m+1), x(4m+2), x(4m+3), where k = 0, 1, ... , N/4-1. Thus DFT of the radix-4 decimation-in-time can be expressed as:

$$\begin{aligned} X(k) = &\sum_{m=0}^{N/4-1} x(4m).W_N^{4mk} \\ &+ \sum_{m=0}^{N/4-1} x(4m+1).W_N^{(4m+1)k} \\ &+ \sum_{m=0}^{N/4-1} x(4m+2).W_N^{(4m+2)k} \\ &+ \sum_{m=0}^{N/4-1} x(4m+3).W_N^{(4m+3)k} \end{aligned} \tag{2}$$

Let $W_N^{4mk} = W_{N/4}^{mk}$, $W_N^{k} = W^{p}$, A = $\sum_{m=0}^{N/4-1} x(4m).W_{N/4}^{mk}$, B = $\sum_{m=0}^{N/4-1} x(4m+1).W_{N/4}^{mk}$, . C = $\sum_{m=0}^{N/4-1} x(4m+2).W_{N/4}^{mk}$, D = $\sum_{m=0}^{N/4-1} x(4m+3).W_{N/4}^{mk}$, then each Eq. (2) can be written as:

$$\begin{aligned} &X(k) = A + BW^{p} + CW^{2p} + DW^{3p} \\ &X(k + N/4) = A - j.BW^{p} - CW^{2p} + j.DW^{3p} \\ &X(k + N/2) = A - BW^{p} + CW^{2p} - DW^{3p} \\ &X(k + 3N/4) = A + BW^{p} - CW^{2p} - j.DW^{3p} \end{aligned} \tag{3}$$

Eq.(3) can be expressed as Eq.(4) in a matrix form according to the characteristic of W_N^{nk}.

$$\begin{bmatrix} A' \\ B' \\ C' \\ D' \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} . \begin{bmatrix} A \\ \mathrm{BW}^{p} \\ CW^{2p} \\ DW^{3p} \end{bmatrix} = Q . \begin{bmatrix} A \\ \mathrm{BW}^{p} \\ CW^{2p} \\ DW^{3p} \end{bmatrix}$$

Where

A’ = X(k), B’ = X(k + N/4), C’ = X(k + N/2), D’ = X(k + 3N/4)

Thus, a radix-4 butterfly computing unit can be viewed as the input sequences first multiplied by a rotating factor, then to a Q matrix left, which makes the total number of multiplications for the radix-4 FFT much less than that for the radix-2 FFT.

B. CORDIC algorithm

CORDIC algorithm was developed by J. E. Volder. It is an iterative algorithm in which to calculate the rotation of a vector only additions and shifts are taken. Fig. 1 shows an example of rotation of a vector V$_i$. Equations (5) to (8) explain the steps for calculating the rotation.

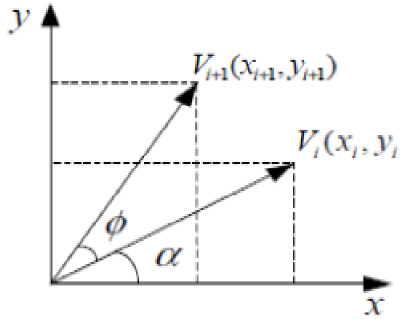


Figure 1: Schematic diagram of the transformation of V$_i$ (x$_i$, y$_i$) to V$_{i+1}$ (x$_{i+1}$, y$_{i+1}$)

$$x_{i+1} = r\cos(\propto + \emptyset)$$
$$= r(\cos\propto\cos\emptyset - \sin\propto\sin\emptyset) \qquad (4)$$
$$= x_i\cos\emptyset - y_i\sin\emptyset$$
$$y_{i+1} = r\sin(\propto + \emptyset)$$
$$= r(\sin\propto\cos\emptyset + \cos\propto\sin\emptyset) \qquad (5)$$
$$= y_i\cos\emptyset + x_i\sin\emptyset$$

Let each rotation angle Ø be equal to arctan 2 $^{-i}$, then:

$$x_{i+1} = \cos\emptyset\,(x_i - y_i.2^{-i} \qquad (6)$$
$$y_{i+1} = \cos\emptyset\,(y_i + x_i.2^{-i} \qquad (7)$$

Since Ø = arctan 2^{-i}, cos Ø can be simplified to a constant with fixed number of repetitions:

$$x_{i+1} = K(x_i - y_i.2^{-i}) \qquad (8)$$
$$y_{i+1} = K(y_i + x_i.2^{-i}) \qquad (9)$$

Where K = cos (arctan (2^{-i})) and $d_i = \pm 1$. Product of K 's can be represented by factor K which may be used as a single constant multiplication either at the start or end of the iterations. Then, (8) and (9) can be simplified as:

$$x_{i+1} = (x_i - y_i.2^{-i} \qquad (10)$$
$$y_{i+1} = (y_i + x_i.2^{-i} \qquad (11)$$

d_i defines the direction of rotation and the sequence of all d_i 's gives the final vector. d_i is given by:

$$d_i = -1 \quad \text{if } z_i < 0$$
$$= +1 \text{ if } z_i < 0 \qquad (12)$$

where z_i is called as angle accumulator and given by

$$z_i = z_i - d_i . \arctan(2^{-i}) \qquad (13)$$

All operations described by Eqs. (10) - (13) can be implemented by only additions and shifts. Therefore, CORDIC algorithm does not require dedicated multipliers.

As shown in Eq.(1), the key operation of the FFT is $x(n).W_N^{nk}$, where $W_N^{nk} = e^{-j\frac{2\pi}{N}nk}$ is just the so-called “twiddle factor”). This is equivalent to "Rotate $x(n)$ by angle $-j\frac{2\pi}{N}nk$” operation which can be easily calculated by the CORDIC algorithm. Without normal complex multiplication, CORDIC based butterfly can be very fast.

III. Architecture Of CORDIC-Based Radix-4 FFT

The overall structure of the proposed CORDIC-based FFT processor model is shown in Figure 7. The entire model is made of the address generation unit, the control unit, the dual port RAM unit, the 4-point butterfly unit and the CORDIC twiddle factor generation unit. This model is characterized by setting the parameter, sampling points and the accuracy to meet the actual needs.

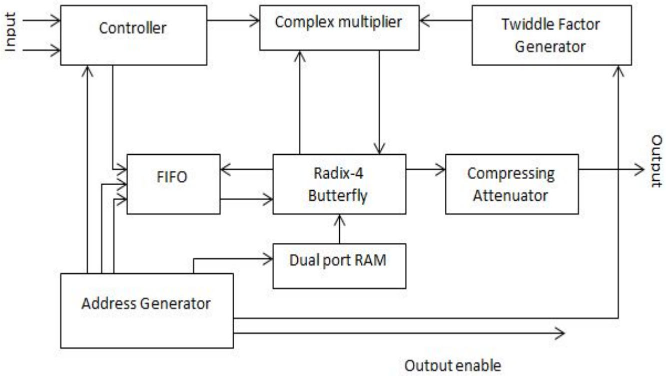


Figure 2: Architecture of CORDIC-based FFT processor.

The architecture of the FFT processor is shown in Figure.2. Data is inputted into FIFO through Controller. Address producer outputs read/write address and other read/write control signal to the FIFO, and it is also in charge of calculating of 16-dot FFT operation. Radix-4 butterfly unit completes 4-dot DFT. The angle of twiddle factor calculated by the twiddle factor producer was inputted into complex multiplier to perform complex multiplication. The inputted complex data is expended to 16-bit through complex multiplication and radix-4 butterfly. The result of each stage of butterfly is compressed to 14-bit complex data by a compressing attenuator. The advantages with this architecture are high data throughput, small area and a relatively simple control.

The FFT processor presented here is based on radix-4 DIT algorithm in which the in-place computation is utilized to achieve an efficient use of the memories. To perform these operations concurrently, a dual-port RAM has been employed. The control unit involves the timing control of the data storage, reading and writing to make the corresponding data and rotating factor coefficient flow into the butterfly and CORDIC computing unit in sequence in FFT operation. Data and addresses of the ‘twiddle factor’ can be easily generated by the counter. The address generation logic is very simple and does not limit the throughput of the system[2].

The CORDIC processor performs the vector rotation to compute a set of trigonometric functions. The principle idea of the CORDIC algorithm is to decompose a rotation into a sequence of micro-rotations. The pipelined CORDIC arithmetic unit can be obtained by decomposing the CORDIC algorithm into a sequence of operational stages[2].

The CORDIC method can be implemented in two different modes, known as the rotation mode and the vector mode. In this paper, one uses x and y to represent the real part and imaginary part of input and output vectors. In the rotation mode, the co-ordinate components of a vector and an angle of rotation are given and the co-ordinate components of the original vector are computed after rotation by a given angle In the vector mode, the co-ordinate components of a vector are given and the magnitude and angular argument of the original vector are computed.

IV. Results And Discussion

After the completion of this design architecture, any hardware description language can be used to implement it and checked for functionality correctness. In this Paper, the design architecture is written in VHDL HDL language using XILINX ISE tools. And FFT computation decimation in time model is selected.

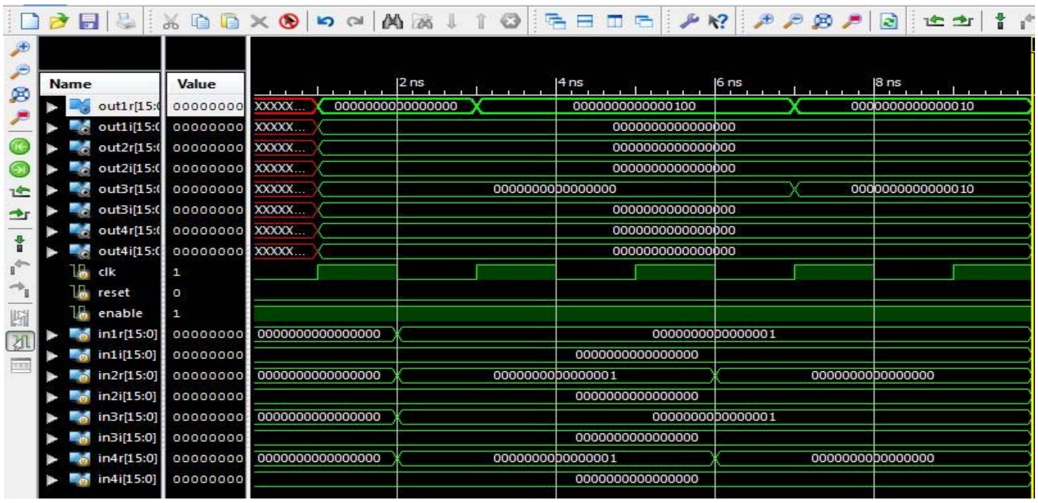
LOW LEVEL RADIX-4

Butterfly module is the main concern in code. As shown in Figure 3 in1r, in1i, in2r, in2i, in3r, in3i, in4r, in4i are the inputs to radix-4 module which are in terms of real and imaginary values. When these inputs are given to radix-4 module it computes FFT of respective inputs. Outputs are complex values and can be given in terms of out1r, out1i, out2r, out2i, out3r, out3i, out4r, out4i.

We have given two different set of inputs one is after one clock cycle as [1 1 1 1] which produces output as shown in Figure 3 and another set after two clock cycles as [1 0 1 0] it can be shown as in Figure 3.

CORDIC OUTPUT

Figure 4 shows the output of CORDIC module in this figure we can observe that CORDIC module takes iin and qin as an input and gives iout and qout as output after computation. Amplitudes are defined for angle computation and every set of input is processed in tckhalf cycle and this result is forwarded to Dual port RAM for storage for later use. In each cycle one single input having real and imaginary term is multiplied with the twiddle factor. And final output will be derived.



Name	Value
out1r[15:(	00000000
out1i[15:(	00000000
out2r[15:(	00000000
out2i[15:(	00000000
out3r[15:(	00000000
out3i[15:(	00000000
out4r[15:(	00000000
out4i[15:(	00000000
clk	1
reset	0
enable	1
in1r[15:0]	00000000
in1i[15:0]	00000000
in2r[15:0]	00000000
in2i[15:0]	00000000
in3r[15:0]	00000000
in3i[15:0]	00000000
in4r[15:0]	00000000
in4i[15:0]	00000000

Figure 3: Output of 4-point Radix-4 FFT algorithm on Xilinx ISE design suit 13.2

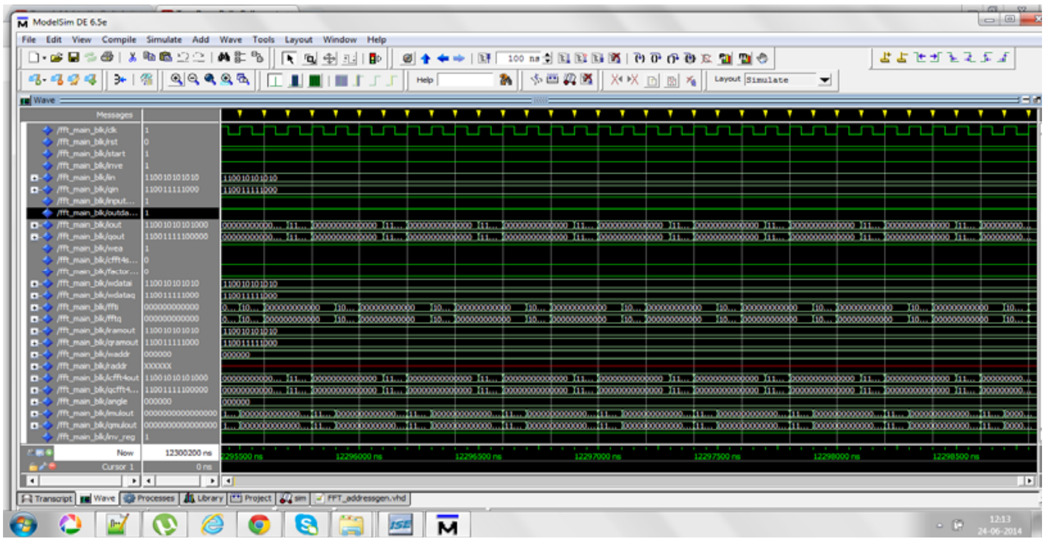


Figure 4 Output of CORDIC Module in Xilinx ISE design suit 13.2

V. Conclusion

As FFT takes values in time domain and generates equivalent sample values in frequency domain, 16 values of samples are taken as an input to the design which is in time domain and, using Decimation in Time method for FFT computation output is generated in terms of frequency samples. Due to in-place computation memory requirement is reduced and using CORDIC for complex twiddle factor generation and multiplication the speed of the computation gets improved with less complexity another factor which helps to improve speed is radix-4 algorithm which computes the output in half the stages required for radix-2 algorithm.

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.

[7] Ajay S. Padekar; Prof. S. S. Belsare; "*Radix-4 FFT Architecture*"; International Journal of Advanced Research in Computer Science and Software Engineering; Volume 4, Issue 5, May 2014 ISSN: 2277 128X.