Advances in Modelling and Analysis B
Vol. 61, No. 3, September, 2018, pp. 123-126
Journal homepage: http://iieta.org/Journals/AMA/AMA_B

DSP implementation of the Discrete Fourier Transform using the CORDIC algorithm on fixed point

Youness Mehdaoui[1*], Rachid El Alami[2]

[1] Research Team in Electronics, Instrumentation and Measurements, USMS, Béni-mellal 23030, Morocco, Computer and Interdisciplinary Physics Laboratory, USMBA, Fez, Morocco, USMBA, Fez 30003, Morocco
[2] LESSI Laboratory, Department of Physics, Faculty of Sciences Dhar El Mehraz, Fez 30003, Morocco

Corresponding Author Email: youness.mehdaoui@gmail.com

https://doi.org/10.18280/ama_b.610303

Received: 16 July 2018
Accepted: 25 Auguet 2018

Keywords:
DFT, cordic, fixed point, dsp, time of processing

ABSTRACT

Fourier transform is a tool enabling the understanding and implementation of a large number of numerical methods for signal and image processing. This tool has many applications in domains such as vocal recognition, image quality improvement, digital transmission, the biomedical sector and astronomy.
This paper proposes to focus on the design methodology and experimental implementation of Discrete Fourier Transform (DFT). The interest of this work is an improvement which makes it possible to reduce the processing time of calculates the DFT while preserving the best performances by using the operator CORDIC and the fixed point, so this work is compared with the results found in the literatures.

1. INTRODUCTION

Discrete Fourier Transformation (DFT) is a mathematical tool for processing the digital signal, which is the discrete equivalent of the continuous Fourier transform that is used for analog signal processing. The calculation of the DFT of the complex sequences in the time domain will convert these sequences into frequency domain and the inverse procedure is done by the Inverse Discrete Fourier Transform [1].

The COordinate Rotation Digital Computer (CORDIC) algorithm [2-4] was originally created by J.E.Volder [2]. The algorithm approximates most functions based on trigonometry. It performs rotations without using multiplication operations. Another advantage of this algorithm is that it makes it possible to obtain a precision determined in advance by performing a given number of iterations.

In the DFT, to calculate the twiddle, we will use the sine and the cosine, the algorithm CORDIC implemented with fixed point will allow speed of calculating sinus and cosines, all this will allow us to have a reduced time with a better precision.

In this work an implementation of the DFT on a Digital Signal Processor (DSP) c64x+ and compare the results with what was found in [5], to come out with a conclusion of the utility of the use of specialized circuits like the DSP.

This paper is organized as follows: In section 2, we introduce the Discrete Fourier Transform, we will explain the algorithm CORDIC in section 3. The fixed point development is given in section 4. The methodology of the proposed implementation is presented in section 5 and 6. The results will be presented in section 7. We will end with a conclusion in section 8.

2. DISCRETE FOURIER TRANSFORM

Physical processes can be described in the time domain using the value of a quantity h as a function of time t, or in the frequency domain using its amplitude H as a function of its frequency f, we can then consider that h(t) and H(f) are two representations of the same function.

For a discrete and periodic signal the corresponding transform is called the Discrete Fourier Transform (DFT). We will focus exclusively on this type of transform in this paper.

For an input sequence x(n), the DFT of N points is defined as follows:

$$X(k) = \sum_{n=0}^{N-1} x(n).W_N^{nk} \quad (1)$$

where: $k = 0,1, \dots \dots N-1$

Where the integer n is the time index, the integer k is the frequency index and the complex number W_N^{nk} which corresponds to the nth root of the unit, commonly called twiddle factor, is defined as follows:

$$W_N^{nk} = \exp(\frac{-2i\pi nk}{N}) = \cos(\frac{2\pi nk}{N}) - i.\sin(\frac{2\pi nk}{N}) \quad (2)$$

The inverse DFT (IDFT) is expressed as follows:

$$x(n) = \frac{1}{N}\sum_{n=0}^{N-1} X(n).W_N^{-nk} \quad (3)$$

where: $k = 0,1, \dots \dots N-1$

We observe that N complex multiplications and N-1 complex additions are needed to compute a point, so we need N^2 complex multiplications and N^2-N complex additions to compute a DFT / IDFT of N samples.

The direct calculation of the DFT is inefficient with increasing the size of the signal to be transformed.

3. CORDIC OPERATOR

3.1 CORDIC Algorithm

The CORDIC algorithm makes it possible to perform calculations such as vector rotations or cartesian-polar and polar cartesian coordinates changes in the plane Euclidean.

We can cite, for example, applications where the CORDIC algorithm is used: single-sideband modulation, discrete, direct and fast Fourier transform [6-7], calculate arcsine function [19], frequency filtering (Gray-Marke trellis, orthogonal filters) and wave filters [8]), adaptive modeling of non-stationary processes (optimal recursive filtering, Kalman filter [9]). He is also involved in the resolution of a large number of linear algebra problems like the orthogonal algorithms of Givens [10], Fadeeva, singular value decomposition [11], QR and Cholesky decomposition.

3.2 Principle of the CORDIC algorithm

The CORDIC algorithm is based on trigonometric function calculations; its principle is to perform rotations on a base vector for a given angle.

Suppose the rotation of the vector V(x, y) by an angle φ as illustrated in figure 1

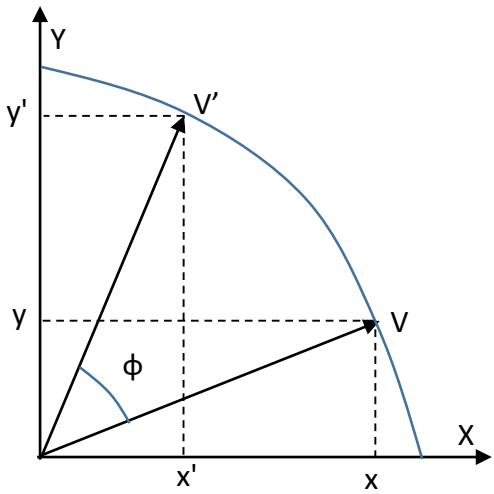

Figure 1. Rotation of the vector V in the Cartesian plane

The coordinates of the vector V' are expressed according to the equations:

$$x' = x\cos\phi - y\sin\phi$$
$$y' = y\cos\phi + x\sin\phi \quad (4)$$

If we restrict the angles of rotation $\tan^{-1}(\pm 2^{-i})$ where$i = 0,1,2,3,\ldots$, we then obtain φ by a series of successive elementary rotations of the order of:

$$\theta_{i+1} = \theta_i - d_i.\tan^{-1}(2^{-i}) \quad (5)$$

where: $d_i = \pm 1$

The index d_i indicates the direction of rotation of the angle for each iteration, this index is determined at each iteration according to the result of a comparison.

Each iterative vector $V_{i+1}(x_{i+1}, y_{i+1})$ is represented by:

$$x_{i+1} = K_i\left[x_i - y_i d_i\left(2^{-i}\right)\right]$$
$$y_{i+1} = K_i\left[y_i - x_i d_i\left(2^{-i}\right)\right] \quad (6)$$

Where: $K_i = \cos(\tan^{-1}(2^{-i})) = (1 + 2^{-2i})^{1/2}$

Since for a relatively high number of iterations, the product tends towards a constant result, it is possible for us to apply it later in the algorithm. In fact, for a given sequence of elementary rotations, the factors K_i can be grouped together and applied at one time. Thus we obtain a set of simplified and specific equations for calculating the mathematical operations sought:

$$x_{i+1} = x_i - d_i dy_i$$
$$y_{i+1} = y_i - d_i dx_i \quad (7)$$

where:

$$dy_i = y_i 2^{-i} \quad (8)$$

$$dx_i = x_i 2^{-i} \quad (9)$$

$$d\theta_i = \tan^{-1}(2^{-i}) \quad (10)$$

$$d_i = \pm 1 \quad (11)$$

according to the sign of θ_i or y_i.

Then we calculate:

$$x' = A_n x \quad (12)$$

where the constant A_n depends only on the sequence of elementary rotations given by:

$$A_n = \prod_{i=0}^{n} K_i \quad (13)$$

We notice that the main interest of this constant is that it does not depend on θ but only on the number of stages. For an increasing number of stages, this constant tends to the value equal to 0.607252935.

3.3 Cosine and sine

In the paper [12], it is shown that the sine and cosine of the input angle can simultaneously be calculated by CORDIC in rotation mode.

$$X_n = A_n.X_0\cos Z_0$$
$$Y_n = A_n.X_0\sin Z_0 \quad (14)$$

by defining $X_0 = 1/A_n$, the rotation produces a scaled sine and cosine of the angle Z_o.

4. FIXED-POINT DEVELOPMENT

The diagram below illustrates a typical development scenario in use today:

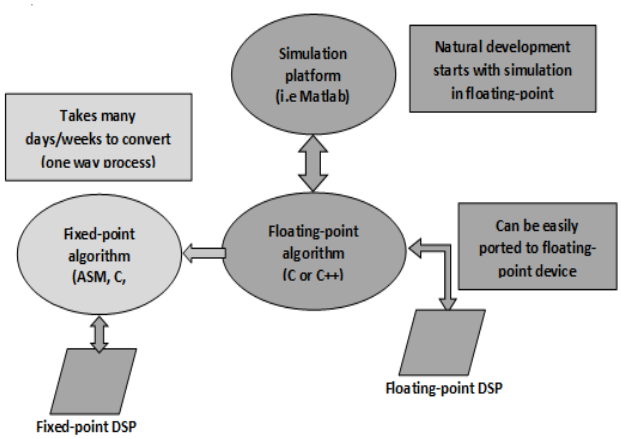

Figure 2. The dilemma of fixed-point development

The design may initially start with a simulation (i.e. MatLab) of a control algorithm, which typically would be written in floating-point math (C or C++). Existing methodologies [13, 14] achieve a floating-to-fixed-point transformation leading to an ANSI-C code with integer data types. This algorithm can be easily ported to a floating-point device. However, because of the commercial reality of cost constraints, most likely a 16-bit or 32-bit fixed-point device would be used in many target systems.

The effort and skill involved in converting a floating-point algorithm to function using a 16-bit or 32-bit fixed-point device is quite significant. A great deal of time (many days or weeks) would be needed for reformatting, scaling and coding the problem. Additionally, the final implementation typically has little resemblance to the original algorithm [15, 18].

For digital signal processors (DSPs), the methodology aim is to define the optimized fixed point specification which minimizes the code execution time and leads to sufficient accuracy [16], some experiments [17] can represent up to 30% of the global implementation time.

5. THE PROPOSED IMPLEMENTATION OF THE CORDIC ALGORITHM

The figure 3 shows the proposed implementation of CORDIC which will be integrated in the DFT; we will name this proposition algorithm by CORFAST.

6. THE PROPOSED IMPLEMENTATION OF THE DFT

In this section the proposed DFT is presented, it is implemented on a fixed-point DSP and its performance will be evaluated and compared with the results in [5].

The flow diagram for DFT Computation is shown in Figure 4. In the butterfly calculation part of the flowchart, the custom CORFAST is used.

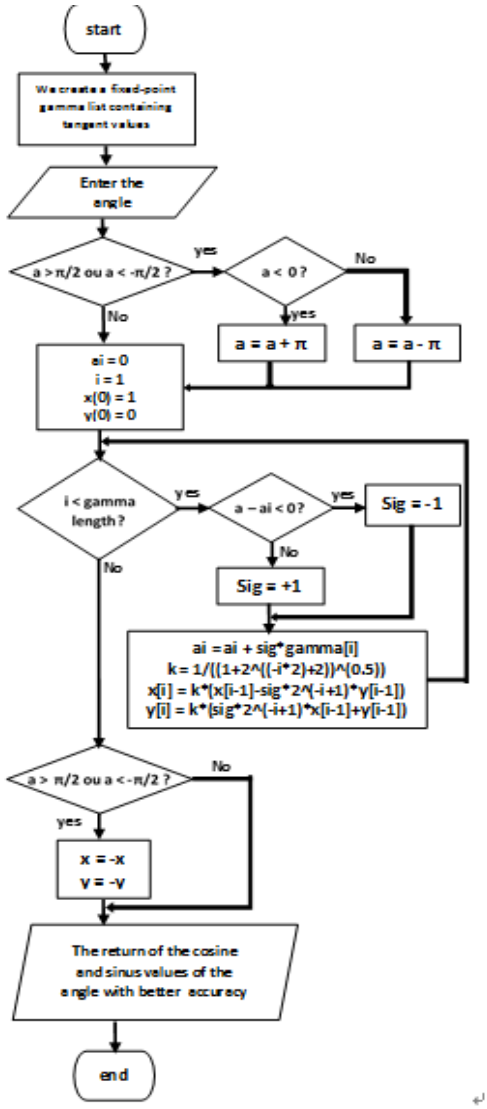

Figure 3. The proposed implementation by CORFAST

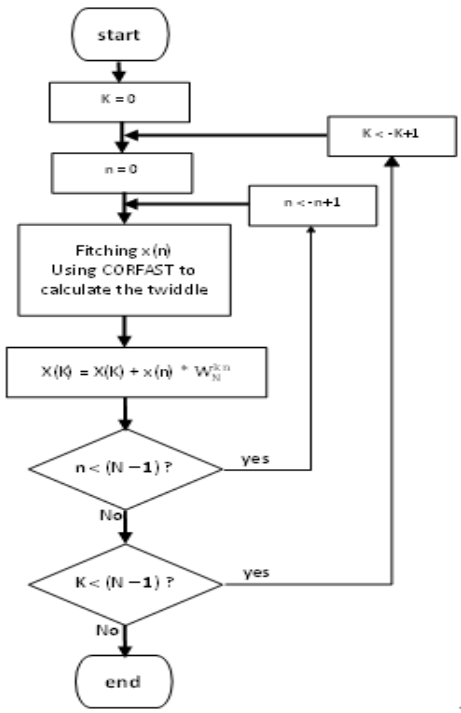

Figure 4. Diagram for DFT Computation using CORFAST

7. DSP IMPLEMENTATION

Using the Code Composer Studio software to do the simulations, this software uses the internal hardware of the DSP C64x+ very efficiently. The algorithms are implemented using DSP processor C64x+ and tested for different input data lengths. The following results are obtained for 10, 12 and 20 point DFT length; the clock cycle is equal to 1 GHz (1ns). We will compare the results found with the results in the paper [5]. Table 1 summarizes the resul1ts.

Table 1. The cycle's number, time taken and their ratios

Number of input sequences	DFT This work		DFT [5]		Ratio
	Benchmark (cycles)	Time taken (in ns)	Benchmark (cycles)	Time taken (in ns)	
10	372394	372394	24179	483580	1.3
12	556878	556878	28999	579980	1.04
20	1540534	1540534	96509	1930180	1.25

8. CONCLUSION

In this work we implemented a DFT using a DSP which is specialized in this kind of application, we can conclude that our implementation is faster (with a ratio of 1.3, 1.04, 1.25 for a number of sequences 10, 12, 20 respectively) compared work [5]. The results found leads us to conclude that the use of specialized circuits like the DSP will give better results than the use of circuits like FPGA which makes the implementation very expensive at the time level, architecture complexity.

REFERENCES

[1] Proakis JG, Manolakis DG. (1996). Digital signal processing, principles, algorithms and applications. Prentice Hall India Publication 459-462.

[2] Volder JE. (1959). The CORDIC trigonometric computing technique. IRE Transactions on Electronic Computers (3): 330-334. http://dx.doi.org/10.1109/TEC.1959.5222693

[3] Andraka R. (1998). A survey of CORDIC algorithms for FPGA based computers. Proc. of the 1998 CM/SIGDA Sixth International Symposium on FPGAs, Monterey, CA. 191-200. http://dx.doi.org/10.1145/275107.275139

[4] Parhami B. (2010). Computer Arithmetic. Oxford University Press, 361-371.

[5] Debaprasad De, Gaurav Kumar K, rchisman Ghosh, Anurup Saha. (2017). FPGA implementation of discrete fourier transform using CORDIC algorithm. Advances in Modelling and Analysis B 60(2): 332-337. http://dx.doi.org/10.18280/ama_b.600205

[6] Despain AM. (1979). Very fast fourier transform algorithms hardware for implementation. IEEE Transactions on Computers C-28(5): 333–341. http://dx.doi.org/10.1109/TC.1979.1675363

[7] Despain AM. (1974). Fourier transform computers using CORDIC iterations. IEEE Transactions on Computers C-23(10): 993–1001. http://dx.doi.org/10.1109/T-C.1974.223800

[8] RAO SK. (1984). Orthogonal digital filters for VLSI implementation. IEEE Transactions on circuits and systems CAS 31(11). http://dx.doi.org/10.1109/TCS.1984.1085452

[9] Sung TY, Hu YH. (1986). VLSI Implementation of real-time Kalman filter, Acoustics, Speech, and Signal Processing. IEEE International Conference on ICASSP 86(11): 2223–2226. http://dx.doi.org/10.1109/ICASSP.1986.1169136

[10] Ahmed HM, Delosme JM, Morf M. (1982). Highly concurrent computing structures for matrix arithmetic and signal processing. Computer 15(1): 65–82. http://dx.doi.org/10.1109/MC.1982.1653828.

[11] Joseph R, Franklin C, Luk T. (1988). CORDIC Arithmetic for an SVD Processor. J. Parallel Distrib. Comput. 5(3): 271–290. http://doi.org/10.1109/ARITH.1987.6158686

[12] Andraka R. (1998). A survey of CORDIC algorithms for FPGA based computers. ACM 0 89791978-5/98/01. http://dx.doi.org/10.1145/275107.275139

[13] Kum KI, Kang J, Sung W. (2000). AUTOSCALER for C: An optimizing floating-point to integer C program converter for fixed-point digital signal processors. IEEE Transactions on Circuits and Syst—Part II 47(9): 840–848. http://doi.org/10.1109/82.868453

[14] Willems M, Bursgens V, Meyr H. (1997). FRIDGE: floating point programming of fixed-point digital signal processors. In Proceeding of 8th International Conference on Signal Processing Applications and Technology (ICSPAT '97), SanDiego, Calif, USA.

[15] DSPArithmeticTutorial. (2008). Texas Instrument.

[16] Menard D, Chillet D, Sentieys O. (2006). Floating-to-Fixed-Point Conversion for Digital Signal Processors. EURASIP Journal on Applied Signal Processing 1–19. http://dx.doi.org/10.1155/ASP/2006/96421

[17] Grotker T, Multhaup E, Mauss O. (1996). Evaluation of HW/SW tradeoffs using behavioral synthesis. In Proceeding of 7th International Conference on Signal Processing Applications and Technology(ICSPAT'96), 781–785, Boston, Mass, USA.

[18] Mehdaoui Y, Mrabti M. (2010). A faster MC-CDMA system using a DSP implementation of the FFT, 5th International Symposium On I/V Communications and Mobile Network, Rabat, Morocco. http://dx.doi.org/10.1109/ISVC.2010.5656245

[19] Anurup Saha, Archisman Ghosh, K. Gaurav Kumar. (2017). FPGA implementation of arcsine function using CORDIC algorithm. Advances in Modelling and Analysis A 54(2): 197-202. http://doi.org/10.18280/ama_a.540205