ISMSIT 2017
1st International Symposium on Multidisciplinary Studies and Innovative Technologies Proceedings
(November 2-4, 2017, Tokat, Turkey)
Symposium homepage: www.ismsitconf.org

Design of DFT Algorithm and Realization by Using FPGA

Kürşad UÇAR[1*],Hasan Erdinç KOÇER[2]

[1]*Department of Electric-Electronic Engineering, Selcuk University, Turkey*
[2] *Department of Electric-Electronic Engineering, Selcuk University, Turkey*
[1*]*(kucar@selcuk.edu.tr)*

Abstract – Field Programmable Gate Arrays (FPGAs) are configurable logic semiconductor devices which connect by programmable interconnections. They can be reprogrammed to desired application or functionality requirements. VHDL is a one of the hardware descriptive languages that use for changing the structure of the FPGA hardware. With DFT, discrete time data sets converts into a discrete-frequency representation, so this procedure provides a process of fast programming that often uses in signal processing. In this study, a design for processing DFT proposes. A 16 point DFT algorithm wrote using VHDL. The inputs of the design are 32 bit single precision floating numbers, these input points processed by Cooley-Tukey DFT algorithm and as a result 16 points complex numbers which are again in single precision floating point format obtained. During the processing of the data, Random access memory (RAM) element and pipeline registers are used for storing of the intermediate values. DFT operation conducted in three stages and each of the stage work parallel to other. Developed VHDL code realized physically by using Altera FPGA board.
Keywords – FPGA, VHDL, DFT, Pipeline

I. INTRODUCTION

In a research at University of Toledo, FPGA based high speed FFT Processor designed for wideband direction of arrival applications. [1] In this research parallel and pipelined Fast Fourier Transform (FFT) used for prediction of a wideband waveform, and selected algorithm was the Coherent Signal Subspace Method. The FFT algorithm developed in MATLAB using the Xilinx System Generator block-set to auto-generate VHDL code. A wideband DOA algorithm via the FFT separates the frequency components. The design of FFT algorithm has 4096 samples into the each 64 point FFT blocks, and processor is capable of 8GS/s throughput. The FFT processors which is using radix-4, operates in parallel with a quad-pipeline for maximum throughput. This research has presented, and the results indicate that for 8-bit and 10-bit FFT processors provided yield throughputs of 8.01GS/s and 8.02GS/s respectively. [1]

Another research which calls FPGA implementations of Fast Fourier transforms for real-time signal and image processing. [2] Fourier transforms use widely for signal and image processing.. The design wrote in Haldel-C language for rapid architecture design and suitability for the design of IP cores. The inputs and outputs have real and imaginary parts of two's complement form which gave as L-bit fixed-point number. Ahmed Saeed and others have implemented FFT and IFFT algorithms using FPGA in their work [6].

The butterfly operation is the most important part of the FFT algorithm. It takes data from memory blocks and after computing the FFT, it writes results back at the same memory in every clock cycle. The coefficient look-up tables which used for FFT twiddle factor computation realizes as LUT ROMs. 2D FFT operated in parallel for implementation on distributed-memory or shared-memory on multiprocessor systems. 1-D FFT is one of the elements of 2-D FFT and it includes the N point 1-D FFT core associated with related twiddle factor. The extended pipelining that has used in program's architecture provides the better system frequency. 2D FFT works in parallel and also affects the performance. This research has proposed and implemented, and it is a solution to the frequency-domain image filtering application. [2]

In this study, this calls the design of DFT algorithm and realization using FPGA carried out. The design developed in VHDL language for using signal processing in a Radar Image Processing Algorithm. The DFT algorithm is based on the Cooley- Tukey [3] mathematical method like previous studies for high-performance requirements. The design inputs and outputs are in 32-bit floating-point number format that includes 16 real and 16 imaginary numbers. The mathematical process consists of 3 separate DFT process steps, RAM and Pipeline Registers used to kept intermediate values. Same type of DFT blocks run in parallel to each other, and the results store different address of different RAMs to reduce the delay. DFT blocks which are containing different mathematical operations run in serial, even as parallel in their internal. The architectural design of the program developed on purpose to obtain the correct results as quickly without loss of data. In this way, the resulting values obtained at almost 170ns on FPGA.

II. MATERIALS AND METHOD

The discrete Fourier transform (DFT) maps a finite number of equally spaced samples of a function into combination of complex coefficient sinusoids, ordered by their frequencies. [4]. By applying the DFT to N number of complex numbers, results again represented by N number complex numbers:

$$C_k[i] = cos\left(\frac{j2k\pi}{N}\right) \quad (1)$$

$$S_k[i] = sin\left(\frac{j2k\pi}{N}\right) \quad (2)$$

Equations 1 and 2 are the DFT's base functions. Input signal can be written by using the base functions as:

$$x[i] = \sum_{k=0}^{\frac{N}{2}} Re(x[k]) \left(cos \frac{j2\pi k}{N}\right) +$$

$$\sum_{k=0}^{\frac{N}{2}} Im(x[k]) \left(sin \frac{j2\pi k}{N}\right) \quad (3)$$

x[k] represents the numbers on complex plane where k indicates order of input numbers. *Re(x[k])* is real part and *Im(x[k])* is imaginary part of complex number. To calculate the DFT of any input signal like in equation 3 by using equivalent of cosines and sinus on complex plane, the results can be written as:

$$X[k] = \sum_{n=0}^{N-1} \mathrm{x[n]}\, e^{\frac{-j2\pi kn}{N}} \quad (4)$$

With equation 4, DFT applies to input signals and numbers are transferred to complex plane.

The DFT algorithm that developed by Cooley and Tukey [5] decreases number of operations to *Nlog(2N)* where *N* is the number of inputs. This algorithm takes the advantage of periodic and symmetrical coefficients. In general FFT algorithm Twiddle Factor (TF) expressed as:

$$W_N^{nk} = e^{\frac{-j2\pi kn}{N}} = \cos\left(\frac{2\pi kn}{N}\right) - \mathrm{jsin}\left(\frac{2\pi kn}{N}\right) \quad (5)$$

Due to periodic and symmetrical properties of Cooley- Tukey DFT algorithm, TF can be expressed as:

$$W^k = W^{k+N} \quad (6)$$

$$W^k = W^{k+N/2} \quad (7)$$

X(*k*) is the output of DFT operations, when $e^{\frac{-2j\pi kn}{N}}$ replaced with W_N^{nk} in equation 4, equation 8 obtained.

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad (8)$$

The *x*(*n*) is an array that consists of *N* elements, when numbers of elements divided by 2, the equation 8 can be rewritten as:

$$X(k) = \sum_{n=0}^{\left(\frac{N}{2}\right)-1} x(n) W_N^{nk} + \sum_{n=N/2}^{N-1} x(n)\, W_N^{nk} \quad (9)$$

When *n*+*N*/2 is wrote instead of *n* in the second sum in equation 9. The equation becomes

$$X(k) = \sum_{n=0}^{\left(\frac{N}{2}\right)-1} x(n) W_N^{nk} + W^{kN/2} \sum_{n=0}^{(N/2)-1} x\left(n + \frac{N}{2}\right) W_N^{nk} \quad (10)$$

From the trigonometric rules, it gave as:

$$W^{\frac{kN}{2}} = e^{-jk\pi} = (e^{-jk})^k = (cos\pi - jsin\pi)^k = (-1)^k \quad (11)$$

So, for the *k*'s odd values the TF coefficient is -1 and for *k*'s even values the TF coefficient is 1. Rewriting equations 10 by replacing $k = 2k$ for *k*'s even values and $k = 2k + 1$ for *k*'s odd values, gives:

$$X(2k) = \sum_{n=0}^{\left(\frac{N}{2}\right)-1} \left[x(n) + x\left(n + \frac{N}{2}\right)\right] W^{2nk},$$

$$k = 0,1,\ldots,\left(\frac{N}{2}\right) - 1 \quad (12)$$

$$X(2k+1) = \sum_{n=0}^{\left(\frac{N}{2}\right)-1} \left[x(n) - x\left(n + \frac{N}{2}\right)\right] W^n W^{2nk},$$

$$k = 0,1,\ldots,\left(\frac{N}{2}\right) - 1 \quad (13)$$

Equations 12 and 13 for odd and even values are showed as DFT blocks in Figure 1.

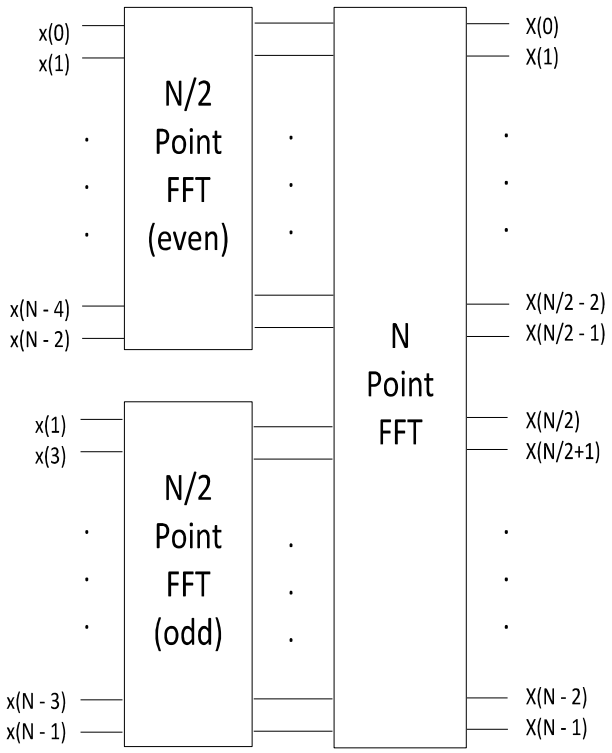

Fig. 1 Even and odd DFT block diagram

In this study, sixteen complex numbers are used as inputs imaginary part where each part is a real number with width of 32 bit Single-Precision Floating Point (SPFP). This means in this study 32 entries of 32 bit SPFP number used. For 16 inputs the Cooley-Tukey DFT algorithm has 4 steps from 16 = 2^4. The 2 input and 4 input DFT operations combined as first step, so the whole operation executed in 3 steps. According to these 3 steps the process showed in Figure 2.

A. DTF Step 1

When the numeric inputs are applied, these numbers are taken by first four DFT4 blocks. DFT4 blocks are the first combined step of 16 inputs DFT algorithm. Each DFT4 block has one clock input, one flag input, one flag output, and eight SPFP inputs and outputs for four real and imaginary parts of these numbers. Inputs are sending to DFT blocks shown in Figure 2 with the order shown in Figure 3. When DFT4 START flag raises, the inputs takes by DFT4 blocks. The mathematical operations that perform at DFT4 blocks showed in Figure 4. In this and following similar figures, black dots show addition of inputs multiply by coefficients given in the figures. During these operations four Altera SPFP addition and four Altera SPFP subtraction blocks are spent. At end of

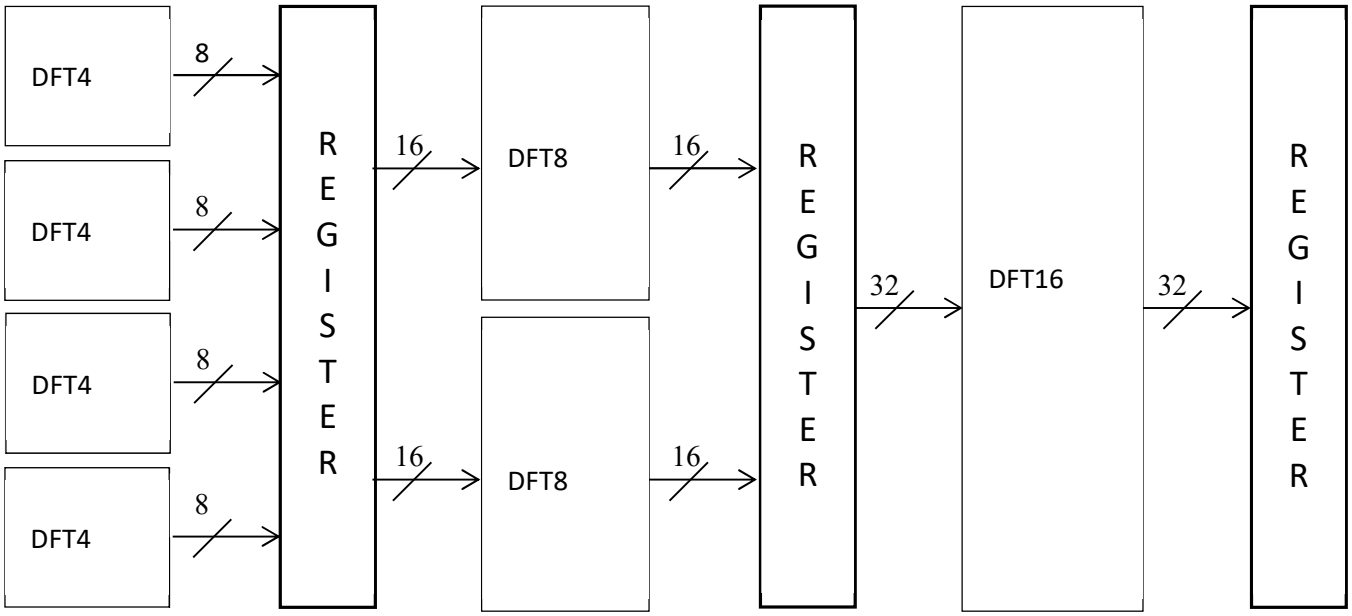

Fig. 2 The DFT process

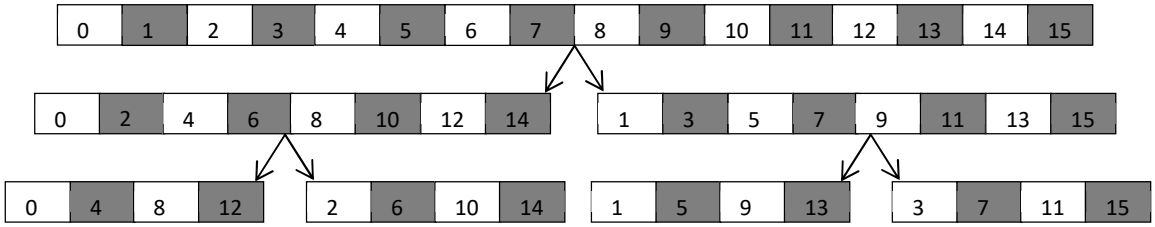

Fig. 3 Inputs of each DFT step

mathematical operations DFT4 FINISH flag raises that shows data is ready to write on pipeline registers. So; the first combined step has finished after writing the results to pipeline registers.

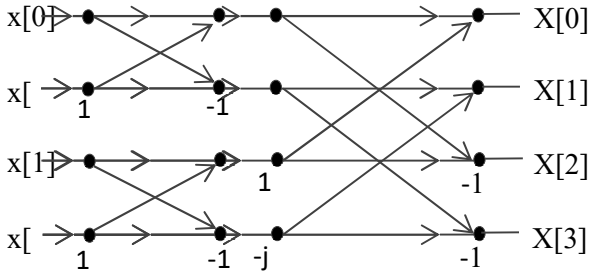

Fig. 4 DFT4 block's mathematical operations

B. DTF Step 2

Two DFT8 blocks requires for 16 points DFT. All the results of DFT4 blocks use in these two DFT8 blocks. 16 SPFP numbers are used in each DFT8 block. Using 16 input ports increase calculation speed of the DFT, because the inputs takes in 1 clock. The mathematical operations for DFT8 blocks showed in Figure 5. Mathematical operations designed to be processed in parallel in order to ensure performance. Six Altera SPFP addition blocks, six Altera SPFP subtraction blocks and four Altera SPFP multiplication blocks use for operations in each DFT8 block. At the end of operation, DFT8 FINISH flag raises. A DFT8 block uses sixteen 32 bit ports for sending the results out of the block. These results wrote to another pipeline register until taken from next step. Thus, the DFT's second step is over.

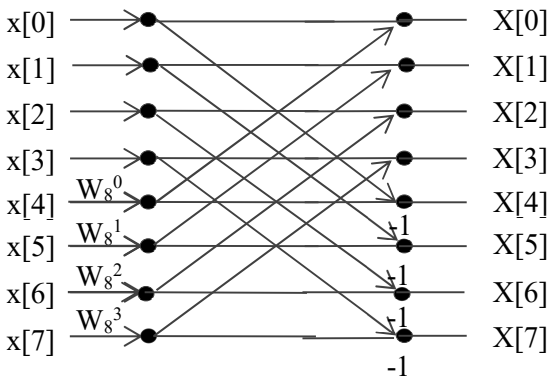

Fig. 5 DFT8 Block's mathematical operations

C. DTF Step 3

This is the last step of calculation. Last step is similar to previous steps has one flag input and one flag output that indicates step start and finish. There are 32 ports, used for numerical inputs and outputs. DFT8 results are that store in pipeline registers, take by this block when the DFT16 START flag becomes high. There are total of fourteen Altera SPFP addition and subtraction blocks and twenty Altera SPFP multiplication blocks, used for the necessary mathematical operation in this block. The mathematical operations performed by this block are shown in Figure 6. When the DFT16 FINISH flag is raised, results are sent to the outputs. These numbers are also become final results of DTF calculation. Thereby DFT process is ended for 16 numbers.

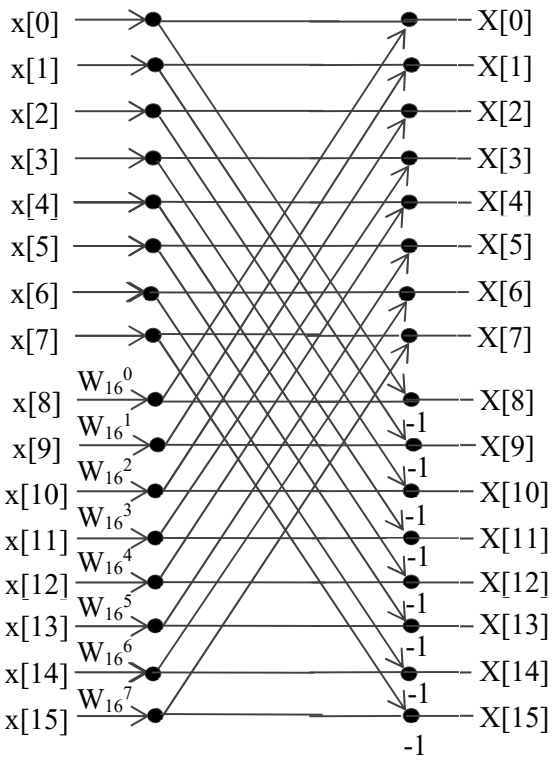

Fig. 6 DFT16 Block's mathematical operations

III. RESULTS

The Altera DE2-115 FPGA board used while processing DFT. Clock frequency was 50 MHz. At the end of the DFT algorithm 16 numbers obtained as a result. These numbers also composed from 2 parts, real and imaginary part. That means 32 single precision floating point number has to be given as input. Like inputs 32 single precision floating point numbers obtained at the outputs. During the processing DFT each steps take 17, 22 and 22 clock cycles respectively. So, the first results come out at the end of 61 clock cycles. Due to pipelined structure of the design, after first result, new results will be available after 22 cycle periods. The utilization of FPGA obtained from Altera synthesis software Quartus II is given in Table 1 and Table 2.

The correctness of the design compared with MatLab software. Same numbers processed by both MatLab and proposed design. For the given input numbers MatLab and proposed design results gave in Table 3. Form the results, we can conclude that almost same results with correctness of 0.0001 obtained.

Table 1. Flow summary

Device	Cyclone IV E
Timing Models	Final
Total logic elements	81,185 / 114,480 (71 %)
Total registers	42189
Total pins	38 / 529 (7 %)
Total virtual pins	0
Total memory bits	3,824 / 3,981,312 (< 1 %)
Embedded Multiplier 9-bit elements	196 / 532 (37 %)
Total PLLs	0 / 4 (0 %)

Table 2. DSP block usage summary

Simple Multipliers(9-bit)	28
Simple Multipliers(18-bit)	84
Embedded Multiplier Blocks	--
Embedded Multiplier 9-bit elements	196
Signed Embedded Multipliers	0
Unsigned Embedded Multipliers	112
Mixed Sign Embedded Multipliers	0
Variable Sign Embedded Multipliers	0
Dedicated Input Shift Register Chains	0

Table 3. Numerical comparison of results with Matlab.

Inputs	MatLab results	DFT results
2	37.0000 +14.0000i	37.0000 +14.0000i
3	1.6069 - 6.8242i	1.6069 - 6.8242i
4	-8.5355 - 7.9497i	-8.5355 - 7.9497i
4	-2.4294 - 0.5727i	-2.4294 - 0.5727i
2+i	-4.0000 - 3.0000i	-4.0000 - 3.0000i
1+3i	5.0782 - 0.1778i	5.0782 - 0.1778i
4+3i	-1.3640 - 1.7071i	-1.3640 - 1.7071i
3+i	0.8267 + 0.1838i	0.8267 + 0.1838i
1	1.0000	1.0000
3	0.3931 - 0.8326i	0.3931 - 0.8326i
3	-1.4645 + 1.9497i	-1.4645 + 1.9497i
1	7.2578 - 0.2557i	7.2578 - 0.2557i
1+i	-10.0000 - 3.0000i	-10.0000 - 3.0000i
2+2i	-3.0782 + 3.8347i	-3.0782 + 3.8347i
2+2i	11.3640 - 0.2929i	11.3640 - 0.2929i
1+i	-1.6551 + 4.6447i	-1.6551 + 4.6447i

IV. DISCUSSION

This should explore the significance of the results of the work, not repeat them. The results should be drawn together, compared with prior work and/or theory and interpreted to present a clear step forward in scientific understanding. Combined Results and Discussion sections comprising a list of results and individual interpretations in isolation are particularly discouraged.

V. CONCLUSION

The design and implementation of Coley- Tukey pipelined DFT processor on an FPGA has been presented. The description was made by VHDL in ALTERA DE2-115 card. The results from the VHDL described architecture are validated against the same algorithm in Matlab. The data and twiddle factor word length were chosen to 32 bit SPFP.

REFERENCES

[1] I.S. Uzun, “FPGA implementations of fast fourier transforms for real-time signal and image processing”, Field-Programmable Technology (FPT).In: IEEE 2003 International Conference on; Proceedings., pp.102-109, 2003.

[2] M. Jamali, J. Downey, N. Wilikins, C. R. Rehm, J. S. Tipping, Development of a FPGA-based high speed FFT processor for wideband Direction of Arrival applications," Radar Conference, 2009 IEEE, pp.1-4, 2009.

[3] J. W. Cooley, and J. W. Tukey., An Algorithm for Computation of Complex Fourier Series, Mathematics of Computation, 1965.

[4] Brigham, E. Oran, The fast Fourier transform and its applications. Englewood Cliffs, N.J.: Prentice Hall, 1988.

[5] T. Ayhan, FFT algoritmalarinin fpga üzerinde gerçeklenmesi. Istanbul Technical University, Istanbul, Turkey, 2008.

[6] A. Saeed, M. Elbably, G. Abdelfadeel, M. I. Eladawy , “Efficient FPGA implementation of FFT/IFFT Processor”, International Journal of Circuits, Systems and Signal Processing, ResearcGate, Vol .3, pp. 103-110, January 2009.