FPGA Implementation of Discrete Fourier Transform Core Using NEDA

Abhishek Mankar, N Prasad, and Sukadev Meher
Dept. of Electronics and Communication Engineering
National Institute of Technology, Rourkela, India - 769008

Abstract – **Transforms like Discrete Fourier Transform (DFT) are a major block in communication systems such as OFDM, etc. This paper reports architecture of a DFT core using new distributed arithmetic (NEDA) algorithm. The advantage of the proposed architecture is that the entire transform can be implemented using adder/subtractors and shifters only, thus minimising the hardware requirement compared to other architectures. The proposed design is implemented for 16 – bit data path (12 – bit for comparison) considering both integer representation as well as fixed point representation, thus increasing the scope of usage. The proposed design is mapped on to Xilinx XC2VP30-7FF896 FPGA, which is fabricated using 130 nm process technology. The hardware utilization of the proposed design on the mapped FPGA is 295 slices, 478 4-input LUTs and 304 slice flip flops. The maximum on board frequency of operation of the proposed design is 79.339 MHz. The proposed design has 72.27% improvement in area, 10.31% improvement in both maximum clock frequency and throughput when compared to other designs.**

Index Terms – **Discrete Fourier Transform (DFT), new distributed arithmetic (NEDA), FPGA, DSP.**

I. INTRODUCTION

Transforms such as Discrete Fourier Transform (DFT) are a major block in many communication systems like OFDM, etc. DFT is also considered as one of the major tools to perform frequency analysis of discrete time signals. A discrete time sequence can be represented by samples of its spectrum in the frequency domain, using DFT. The mathematical representation of the transform is shown below.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{(-\frac{j2\pi kn}{N})} \quad \text{n = 0, 1 … N-1} \qquad (1)$$

The inverse of DFT is given as below.

$$x(n) = (1/N) \sum_{k=0}^{N-1} X(k) e^{(\frac{j2\pi kn}{N})} \quad \text{k = 0, 1 … N-1} \qquad (2)$$

As seen from equations (1) and (2), the forward and inverse transforms can be implemented using same kernel with few modifications, thus reducing the requirement of hardware. Many efficient ways have been put up for direct implementation of DFT due to its computational complexity. Fast Fourier Transform (FFT) is one of the most efficient and common ways to implement DFT. Reduced computational complexity and low latency are two driving factors for implementing DFT using FFT. Other FFT architectures that have been developed are based on radix – 4, radix – 2, hybrid, split radix.

If an N – point DFT is implemented directly, the requirement of arithmetic operations is of the order of O (N^2) that is N^2 multiplications and N (N-1) additions. The order of arithmetic operations for FFT is O (NlogN) that is NlogN additions and (N/2)logN multiplications. Thus FFT implementation of DFT is preferred. Depending on inputs being real or complex, the structures of adders and multipliers are built.

Distributed arithmetic (DA) [1] has become one of the most efficient tools in VLSI implementation of digital signal processing (DSP) architectures. It efficiently computes inner products of vectors, which is a key requirement in many DSP systems. One of the key computational blocks in DSP is multiply/accumulate (MAC), which is implemented by a standard adder unit and a multiplier. Using DA, MAC unit can be implemented by pre computing all possible products and using a ROM to store them. The con of using DA is in its exponential increase of the size of ROM with increase in internal precision and number of inputs. An approach to overcome this drawback is by distributing the coefficients to inputs. One of such examples is new distributed arithmetic (NEDA) [2]. As in [2], it can be used to implement any transform that is based on fourier basis. This approach helps in finding out the redundancy in computing vector inner product thus reducing the number of computational blocks, especially adders.

Many architecture models for DFT, based on DA, are implemented in [3] – [5]. Discrete Hartley Transform (DHT), considered as sister transform of DFT, also was implemented using DA [6]. The disadvantage of all these implementations is that they use ROM or RAM which makes the designs with increased architecture. Other approach in implementing DFT is based on employing CORDIC units [7] – [8]. Even this has cons as the structure of a CORDIC unit is comparable to that of a multiplier thus making it more hardware complex.

In this paper we proposed a NEDA based architecture for DFT where coefficients of cosine are distributed over the inputs, in the inner product. We have carefully chosen the coefficients for which the distribution has to be done, such further reducing the redundancy in the inner product. The architecture thus obtained is an efficient one as it is being implemented using only adder/subtractors and shifters.

The rest of the paper is arranged as follows. Section II briefs the discussion on distributed arithmetic including NEDA. Section III introduces the proposed architecture. Section IV shows the results and discussion. Section V concludes the paper.

II. OVERVIEW OF NEW DISTRIBUTED ARITHMETIC

Inner product implementation calculates the sum of products as following.

$$Z = \sum_{i=1}^{k} C_i X_i \tag{3}$$

where C_i are the fixed coefficients and X_i are considered as input samples. Equation (3) can be written as a matrix product as

$$Z = [C_1\ C_2\ \cdots\ C_k] \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_k \end{bmatrix} \tag{4}$$

C_i and X_i both are considered in two's complement format. Thus C_i can be expressed as

$$C_i = -C_i^M 2^M + \sum_{k=N}^{M-1} C_i^k\, 2^k \tag{5}$$

where $C_i^k = 0$ or 1, k = N, N+1, …, M and C_i^M is the sign bit and C_i^N is the least significant bit (LSB). Substituting equation (5) in equation (4) results in the following matrix product which is modelled according to the required design.

$$Z = [-2^0 \quad 2^{-1} \quad \cdots \quad 2^{-12}] \begin{bmatrix} C_1^0 & \cdots & C_k^0 \\ \vdots & \ddots & \vdots \\ C_1^{12} & \cdots & C_k^{12} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_k \end{bmatrix} \tag{6}$$

as said before, $C_i^k = 0$ or 1. For the proposed design, the value of k is taken as 8 since the design being proposed contains 8 input samples.

Equation (6) can be re-written as follows.

$$Z = [-2^0 \quad 2^{-1} \quad \cdots \quad 2^{-12}] \begin{bmatrix} W_0 \\ W_1 \\ \vdots \\ W_{12} \end{bmatrix} \tag{7}$$

where W_i is the partial sum of the value Z. As the matrix containing C is a sparse one, the elements of W matrix contain only additions. Thus, the matrix product is converted to only additions and shifts, which are carried out at last for obtaining Z. This facilitates architecture with reduced hardware and minimum computational blocks.

The cosine values that are used in the proposed design are described using NEDA as follows:

$$Angle = \cos\frac{\pi}{4} \tag{8}$$

$$Angle = [-2^0 \quad 2^{-1} \quad \cdots \quad 2^{-12}] \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{9}$$

III. PROPOSED ARCHITECTURE

The design of proposed DFT core is divided into two stages. First stage divides the computation based on real and imaginary coefficients, carries out the computation based on NEDA. Second stage integrates the two sub modules to make the design of the core complete.

The division of the core into real and imaginary coefficients is based on the following change made to equation (1).

$$X(k) = \sum_{n=0}^{7} x(n) \cos\left(\frac{2\pi kn}{N}\right) - j \sum_{n=0}^{7} x(n) \sin\left(\frac{2\pi kn}{N}\right) \tag{10}$$

Equation (10) is re written as follows.

$$X(k) = X_C(k) - jX_S(k) \tag{11}$$

where,

$$X_C(k) = \sum_{n=0}^{7} x(n) \cos\left(\frac{2\pi kn}{N}\right)$$

$$X_S(k) = \sum_{n=0}^{7} x(n) \sin\left(\frac{2\pi kn}{N}\right) \tag{12}$$

The set of equations for computing real and imaginary coefficients is given in equation (13). By exploring the property of symmetry of the transform, the remaining coefficients are calculated.

$$X[0] = (x[0] + x[1] + x[2] + x[3] + x[4] + x[5] + x[6] + x[7]) - j(0)$$

$$X[1] = \big(x[0] - x[4] + K(x[1] - x[3] - x[5] + x[7])\big) - j\big(x[2] - x[6] + K(x[1] - x[7] + x[3] - x[5])\big)$$

$$X[2] = (x[0] + x[4] - x[2] - x[6]) - j(x[1] - x[7] - x[3] + x[5])$$

$$X[3] = \left(x[0] - x[4] - K(x[1] - x[3] - x[5] + x[7])\right) - j(K(x[1] - x[7] + x[3] - x[5]) - x[2] + x[6])$$

$$X[4] = (x[0] - x[1] + x[2] - x[3] + x[4] - x[5] + x[6] - x[7]) - j(0) \tag{13}$$

In equation (13), $K = Angle = \cos\frac{\pi}{4}$ is calculated as mentioned in equation (9).

The sub modules are designed by exploring symmetry property of the DFT thus reducing number of computational blocks. Figure 1 shows the schematic of the top module of the proposed NEDA based DFT architecture.

From the figure 1, it is evident that the core is divided into two sub modules which compute coefficients for real and imaginary parts separately. From the lower level of RTL schematic, the number of adder/subtractors required for computing the entire transform is 31. In addition to the above, the number of shifters used in the proposed design is 10. As there are no multipliers, though there is an increase in number of adders, it greatly reduces the hardware complexity.

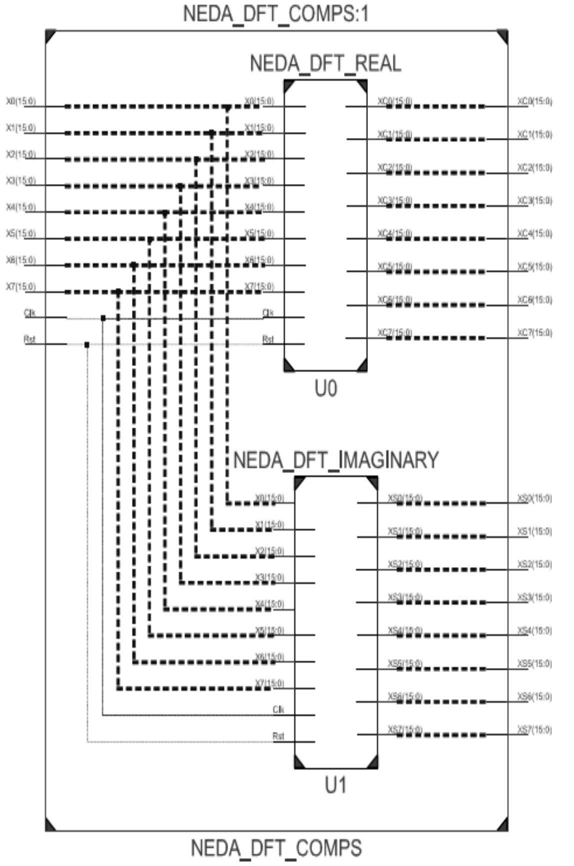


Fig. 1. RTL schematic of top module of the proposed design

The inputs and outputs of the design are registered thus making the design sequential and also synchronous.

Compared to the traditional implementation of FFT that consumes 24 adder/subtractors and 12 multipliers, the proposed design, though has 31 adder/subtractors in total, the hardware is greatly reduced as there are no multipliers.

The width of the inputs is taken to be 16 bits and the same is maintained throughout the design, which makes the design have a constant data path of 16 bits. Thus the width of the outputs is also 16 bits. The average of the inputs is limited to 4095 as the width of the data path is fixed.

IV. RESULTS AND DISCUSSION

The design is coded in VHDL using the software Xilinx ISE 10.1. The design is mapped onto the Xilinx made XC2VP30-7FF896 FPGA. All inputs and outputs are declared in signed two's complement number system. As mentioned earlier, the data path width considered in the design is 16 bits and thus, the total number of bonded IOBs required for the proposed design is 386. The inputs are considered both in integer and fixed point. As the width of the data path is fixed, no overflow should occur in carrying out the arithmetic operations.

Table I compares the results of the proposed design, taken using Xilinx ISim, to the results obtained by MATLAB. The inputs and outputs are declared using fixed point representation.

From table I, it is evident that the proposed design is producing near exact outputs. Thus from the table I, the input sequence considered is of 8 samples.

TABLE I. COMPARISON OF OUTPUTS OBTAINED USING MATLAB AND XILINX ISE, FOR FIXED POINT REPRESENTATION

Inputs	MATLAB outputs	Xilinx ISim outputs
20.1367	112.64	112.6523
21.1289	1.37 – 4.18j	1.3594 – 5.8242j
15.0703	5.95 – 36.07j	5.9492 – 37.9296j
5.0003	6.77 – 4.41j	6.7813 – 5.5977j
16.0664	20.27	20.2617
20.0003	6.77 + 4.41j	6.7813 + 5.5977j
15.1835	5.95 + 36.07j	5.9492 + 37.9296j
0.0585	1.37 + 4.18j	1.3594 + 5.8242j

Table II shows the comparison of the results of the proposed design, but this time, the inputs are taken in integer representation.

TABLE II. COMPARISON OF OUTPUTS OBTAINED USING MATLAB AND XILINX ISE, FOR INTEGER REPRESENTATION

Inputs	MATLAB outputs	Xilinx ISim outputs
35	167	167
33	50.53 + 16.27j	48 + 17j
18	-13 – 18j	-13 – 18j
1	-14.53 – 41.73j	-12– 41j
17	67	67
1	-14.53 + 41.73j	-12 + 41j
47	-13 + 18j	-13 + 18j
15	50.53 – 16.27j	48 – 17j

From table II, it is evident that the difference between the outputs obtained through MATLAB and those obtained through Xilinx ISE is permissible.

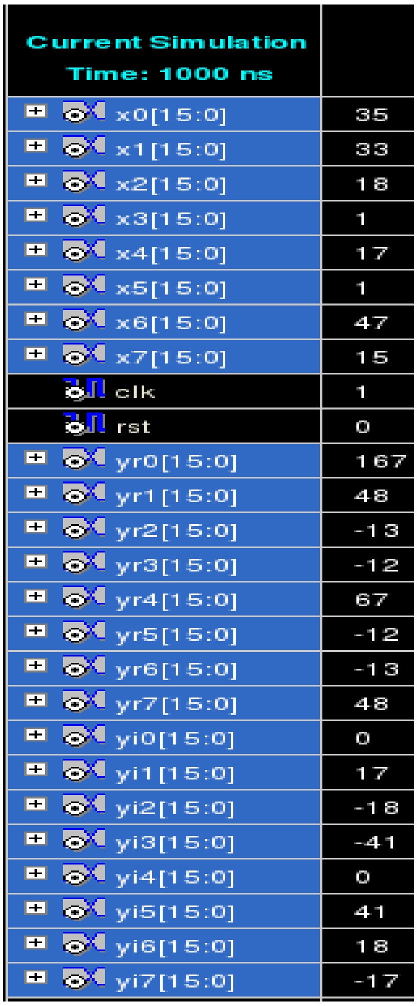
Table III compares the number of arithmetic units required for the proposed design with that of traditional methods. Thus from the table, it is evident that the hardware required for the proposed design is less, though the number of adders is more, since there are no multipliers in the proposed design. The input sequence is assumed to be real in nature.

TABLE III. COMPARISON OF THE NUMBER OF ARITHMETIC UNITS REQUIRED TO PERFORM 8 POINT DFT

Arithmetic unit	**Direct implementation**	**FFT implementation**	**Proposed design (NEDA implementation)**
Adder/Subtractor	56	24	31
Multiplier	64	12	0

The proposed method of implementing DFT can be increased to higher point according to the requirements of the system. If the outputs are expected to be more accurate, then the precision of the W matrix, mentioned in equation (7), has to be increased.

Figure 2 shows the simulation window obtained in Xilinx ISim simulator.



Current Simulation Time: 1000 ns	
x0[15:0]	35
x1[15:0]	33
x2[15:0]	18
x3[15:0]	1
x4[15:0]	17
x5[15:0]	1
x6[15:0]	47
x7[15:0]	15
clk	1
rst	0
yr0[15:0]	167
yr1[15:0]	48
yr2[15:0]	-13
yr3[15:0]	-12
yr4[15:0]	67
yr5[15:0]	-12
yr6[15:0]	-13
yr7[15:0]	48
yi0[15:0]	0
yi1[15:0]	17
yi2[15:0]	-18
yi3[15:0]	-41
yi4[15:0]	0
yi5[15:0]	41
yi6[15:0]	18
yi7[15:0]	-17

Fig. 2. Simulation output of Xilinx Isim for integer representation of inputs

Table IV compares the proposed design with the designs given in [4]. The designs mentioned in [4] are implemented using three different approaches. The first approach is that the designs are implemented completely using the logical cells of the FPGA. The second approach uses the DSP hardware resources available on board. This approach reduces the total number of logic cells required. The third approach is based on the conventional distributed arithmetic (DA). This approach uses ROM to store the adder combinations to calculate the output coefficients. The DA based approach can be either bit serial or bit parallel in nature. Among the three different approaches carried out in calculating the coefficients in [4], the logic cell approach consumes least area and DA based approach has got maximum frequency of operation.

TABLE IV. COMPARISON OF THE PROPOSED DESIGN WITH THE ONES MENTIONED IN [4]

	Resource usage		**Clock frequency [MHz]**	**Throughput [Mbit/s]**
	[#LC]	[#DSP]		
FFT_LC	4723 (14%)	-	43.51	522.12
FFT_DSP	1554 (5%)	70 (100%)	48.93	587.16
FFT_DA	7222 (22%)	-	74.36	892.32
DFT_NEDA (Proposed)	428 (1%)	-	82.03	984.36

From table IV, it is clear that the proposed design has occupied far less hardware than other designs mentioned. In addition to that, the proposed design also has higher clock frequency and better throughput. The device used to map for generating the comparison is EP2C35F672C6 from Altera's Cyclone II family. The width of the inputs, outputs and the data path considered is 12 bits. The maximum frequency obtained is through the slow model using Altera's TimeQuest Timing Analyzer.

Table V shows the device utilization summary of the proposed design in XC2VP30-7FF896 FPGA.

TABLE V. DEVICE UTILIZATION SUMMARY OF THE PROPOSED DESIGN IN XILINX XC2VP30-7FF896 FPGA

Logic Utilization	**Used**		**Utilization**	
	Behavioral Synth.	**Post PnR Synth.**	**Behavioral Synth.**	**Post PnR Synth.**
Number of slices	295	307	2%	2%
Number of slice Flip Flops	304	304	1%	1%
Number of 4-input LUTs	478	480	1%	1%

Power analysis of the proposed design is done using Xilinx Xpower Analyzer tool. Table VI shows the power distribution of the proposed design for different clock frequencies.

TABLE VI. POWER DISTRIBUTION OF PROPOSED DESIGN ON XILINX XC2VP30-7FF896

Frequency (MHz)	Clock (mW)	Logic (mW)	Signal (mW)
10	2.03	16.19	47.39
20	4.05	274.88	595.25
30	6.08	412.41	887.56
40	8.11	549.9	1179.28
50	10.13	687.5	1472.98

Figure 3 shows the graph of the power distribution of proposed design.

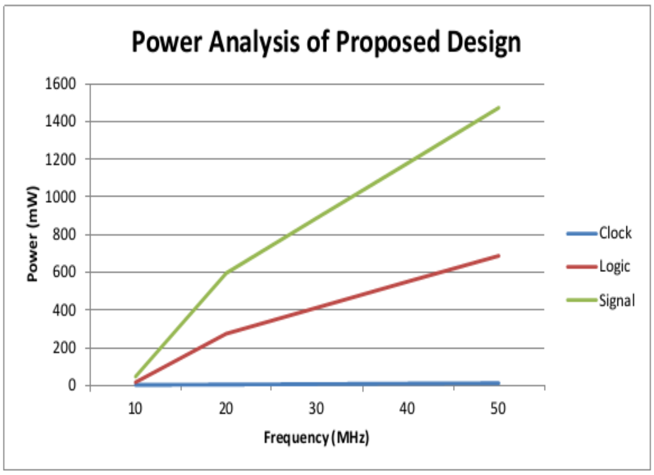


Fig. 3. Power distribution of proposed design on XC2VP30-7FF896 FPGA

Table VII shows the total power consumed by the design proposed for different clock frequencies, when mapped onto the FPGA. From the Xpower analysis, the optimum frequency of the proposed design is 15.732 MHz.

Figure 4 shows the graph of total power consumption of the proposed design.

TABLE VII. TOTAL POWER CONSUMPTION OF PROPOSED DESIGN ON XILINX XC2VP30-7FF896 FPGA

Frequency (MHz)	Total Quiscent Power (W)	Total Dynamic Power (W)	Total Power (W)
10	0.10312	0.09449	0.19762
20	0.10312	0.89637	0.99949
30	0.10312	1.35024	1.45337
40	0.10312	1.79625	1.89937
50	0.10312	2.2691	2.37223

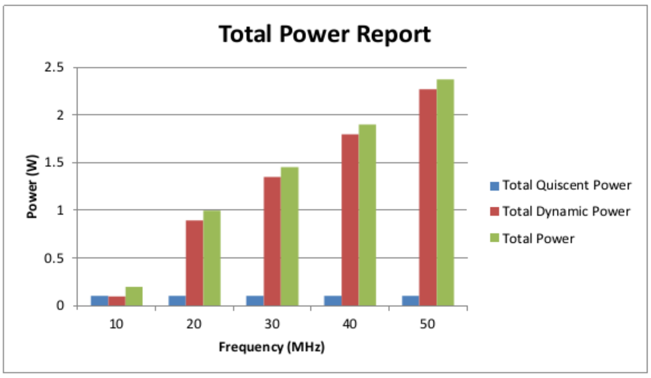


Fig. 4. Total power consumption of proposed design when mapped on Xilinx XC2VP30-7FF896 FPGA

V. CONCLUSIONS

In the present paper, we reported architecture of a DFT core, which is employed in many communication systems, using NEDA, a ROM less and multiplier less method. The proposed design is hardware efficient as compared to other traditional methods as well as architectures that are built using only DA. The proposed design is implemented on Xilinx XC2VP30-7FF896 FPGA, which occupies an area ratio of 2% of the total configurable area. The proposed design in total has 31 adders with zero multipliers and no ROM, thus making possible in implementing higher point DFT. The maximum frequency of operation of the proposed design on the mapped FPGA is 79.339 MHz. The proposed design is also compared to other designs mentioned in [4], which shows great reduction in hardware (72.27%) and improvement in maximum clock frequency (10.31%) and throughput (10.31%).

REFERENCES

[1] Stanley A. White, “Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review,” IEEE ASSP Magazine, vol. 6, no. 3, pp. 4 – 19, Jul. 1989.

[2] Wendi Pan, Ahmed Shams, and Magdy A. Bayoumi, “NEDA: A NEw Distributed Arithmetic Architecture and its Application to One Dimensional Discrete Cosine Transform,” Proc. IEEE Workshop on Signal Processing Syst., pp. 159 – 168, Oct. 1999.

[3] Richard M. Jiang, “An Area-Efficient FFT Architecture for OFDM Digital Video Broadcasting,” IEEE Trans. Consumer Elect., vol. 53, no. 4, pp. 1322 – 1326, Nov. 2007.

[4] M. Rawski, M. Vojtynski, T. Wojciechowski, and P. Majkowski, “Distributed Arithmetic Based Implementation of Fourier Transform Targeted at FPGA Architectures,” Proc. Intl. Conf. Mixed Design, pp. 152 – 156, Jun. 2007.

[5] S. Chandrasekaran, and A. Amira, “Novel Sparse OBC based Distributed Arithmetic Architecture for Matrix Transforms,” Proc. IEEE Intl. Sym. Circuits and Syst., pp. 3207 – 3210, May 2007.

[6] Pramod K. Meher, Jagdish C. Patra, and M. N. S. Swamy, “High-Throughput Memory-Based Architecture for DHT Using a New Convolutional Formulation,” IEEE Trans. Circuits and Syst. – II, vol. 54, no. 7, pp. 606 – 610, Jul. 2007.

[7] Pooja Choudhary, and Dr. Abhijit Karmakar, “CORDIC Based Implementation of Fast Fourier Transform,” Proc. Intl. Conf. Computer and Comm. Tech., pp. 550 – 555, Sept. 2011.

[8] Jayshankar, “Efficient Computation of the DFT of a 2N – Point Real Sequence using FFT with CORDIC based Butterflies,” Proc. IEEE TENCON 2008, pp. 1 – 5, Nov. 2008.