Indian Journal of Science and Technology, Vol 8(20), DOI:10.17485/ijst/2015/v8i20/78462, August 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

FFT Architectures for Real Valued Signals based Different Radices Algorithm

Kodakandla Abhilash* **and P. Reena Monica**

School of Electronics Engineering, VIT University, Chennai - 600048,Tamil Nadu, India;
kodakandla.abhilash2013@vit.ac.in, reenamonica@vit.ac.in

Abstract

Objectives: The objective of this work is to design efficient FFT architectures for real valued signals. For higher value of N, the design of FFT architecture has many butterflies in the same column. A considerable amount of power is needed for such architecture. The hardware complexity also increases. **Methods:** The throughput depends on the number of input data paths available, which also increases hardware complexity for higher values of N. This work aims at overcoming these shortcomings by employing one of the effective techniques such as parallel pipelined architectures. One butterfly structure is introduced in one column. For handling the complex data paths, different butterfly structures have been introduced using parallel processing and Folding methodology. The design is simulated in Xilinxs14.3, and synthesized in Cadence RTL compiler. **Findings:** The hardware complexity is reduced by introducing one butterfly structure in one column. Comparison is made between the different radix algorithms and pipelined architectures and Radix-2 multi path delay commutator (R2MDC) in terms of the required number of adders, delay elements, multiplier units, throughput and control complexity. Although the parallel architecture consumes more power, it occupies less area when compared to the R2MDC architecture. Parallel architecture is more area efficient than R2MDC, whereas the radix-2 multi-path delay commutator is simple to implement, since feedback is not needed in the design. **Conclusion:** This work can be extended to higher order N-values of DIF and DIT-FFT flow graphs.

Keywords: DFT, Fast Fourier Transform (FFT), Folding, Parallel Processing, Pipelining, Radix-2,4,8, Reordering Structures, R2MDC

1. Introduction

Fast Fourier Transform is a sophisticated algorithm to compute the Discrete Fourier Transform. It is the core of many digital signal processing systems. These systems can perform time domain signal processing and frequency domain signal processing. The Fourier transform is achieved by means of performing the Fourier analysis and in Fourier analysis the time domain signals are converted into its equivalent frequency domain signal. The Fourier Transform allows the frequency domain signal to be calculated. Conversely Inverse Fourier Transform can be performed in which the time domain signal are computed from the frequency domain signals. Frequency analysis of these transforms is very important as they are used in different types of TAP Filter designing. There are different types of transformation methods which are followed rigorously in modern day communication processes. They are namely: Fourier series[1], Fourier Transform[2], Discrete Time Fourier Series[3] and Discrete Fourier Transform[4]. These definitions are based on the behavior of the signals, as the signals can be continuous-periodic [1], continuous-a periodic[2], discrete-periodic[4] and discrete-a periodic[3]. So it changes accordingly.

DFT is an important operation in signal processing applications. It is used in both spectrum analysis as well as linear wave shaping filters for signals. The Spectrum analysis is closely related to Fourier Transform of a signal and DFT of the sample of that very same signal[5]. Linear Filtering of a signal is done by the DFT which depicts the property of cyclic convolution[1]. It Transforms Frequency signals into number of point wise products. One of the main drawbacks of DFT in linear filtering is that it is defined only by complex numbers, while most of the signals are real in nature. Also DFT of a real valued sequence contains some redundancies. In

* Author for correspondence

DFT computation half of the co-efficient is sufficient to compute the DFT.

FFT plays a critical role in latest technology applications and a lot of research work is being carried out in the FFT designs like digital video broadcasting and Orthogonal Frequency Division Multiplexing (OFDM) systems frequency domain[9], source tracking[9]. It finds its application in various fields such as communication systems, biomedical applications, sensors, source tracking radar signal processing, beam forming and many more. Beamforming is a digital technique by which one can focus the radar transmitter and receiver in selected direction[11]. The FFT with beam forcing strategy is used to receive the time domain signal and convert it into different frequency components.

DFT of the x[n] is defined as equation 1,

$$X[K]=\sum_{n=0}^{N-1}x(n)W_N^{nk} \tag{1}$$

Where W_N is twiddle factorand is defined by

$$W_N=e^{-j\frac{2\pi nk}{N}}=\cos\left(\frac{2\pi nk}{N}\right)-j\sin\left(\frac{2\pi nk}{N}\right) \tag{2}$$

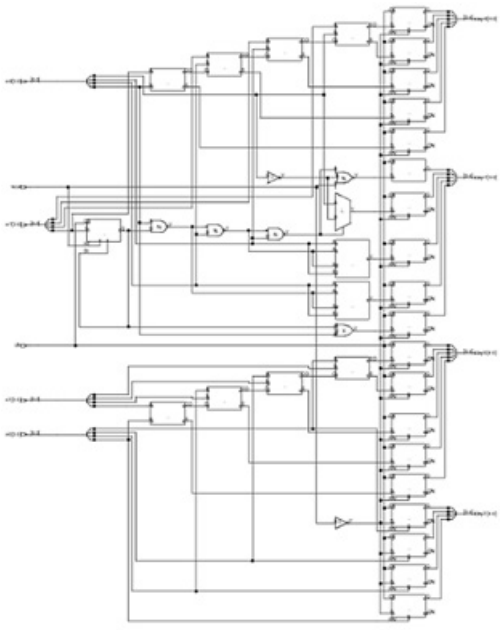


Figure 1. RTL Schematic of two point DFT.

For an N point DFT, it must be noted that, there are N^2 complex multiplication and N^2-N complex addition necessary. Two point DFT as shown in Figure 1. For reducing these complex addition and multiplications the property of symmetry and periodicity are used respectively as follows in the equation 3 and 3.1.

FFTs play an important role in design of digital systems. FFTs are categorized generally into two types which further depend on the input and output sequence order. Therefore, in order to design the necessary pipeline architecture there is a need to study the following architecture: Decimation In Frequency (DIF-FFT) and Decimation In Time (DIT-FFT). Cooley-Tukey radix-2^1 is one of the most popular FFT design, than Radix-4^2, radix-8^3, R2MDC[4] (Radix-2 multi path delay commutator) and R2SDC[5](Radix-2 single path delay feedback). These are pipelined and have been developed based on the radix-2 FFT.

$$W_N^{k+\frac{N}{2}}=-W_N^k \tag{3}$$

$$W_N^{k+N}=W_N^k \tag{3.1}$$

2. Butterfly Structure

Butterfly structures consist of two adder circuitry and one multiplier. The placing of the multiplier will depend on the type of FFT, i.e., DIF-FFT or DIT-FFT. Their different types of butterflies handle real values and complex values of operation. But the normal butterfly structure can handle only any one of the numbers either real or complex. The basic structure of butterfly is shown in Figure 2.

Butterfly-I type consists of two adders and one inverter as shown in Figure 3, which is used to give inverted output to the input of the second adder. This inverter is used to convert the positive numbers to negative numbers, and vice versa. Figure 3 shows the RTL Schematic of Butterfly-I.

Butterfly-II is designed in such a way that it contains the two floating point adders and inverter, in addition to two 2-input multiplexers as shown in Figure 4. The control signal can be selected in such a way that it can be used for butterfly operation for first half of the sample and second half is transferred directly to the output. The clock signal is to be halved for the control signal to be twice that of the clock signal. When the inputs in0 and in1 change, the outputs respond as in0+in1 and in0-in1 or in0 and in1.

Butterfly-III type is shown in Figure 5. It can handle the real and complex inputs and depends on the control signal. It acts as a normal butterfly operation when inputs are real. But when the inputs are in complex form then it acts as butterfly III.

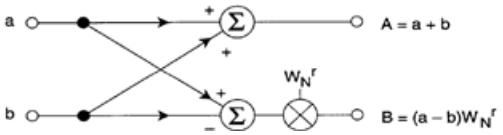


Figure 2. Butterfly structure.

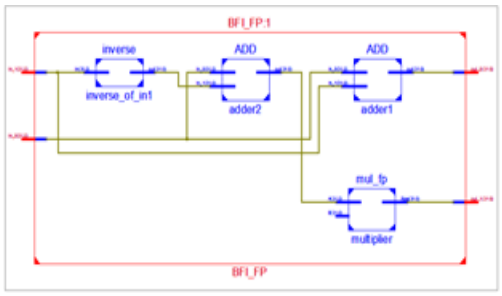


Figure 3. RTL Schematic of Butterfly-I.

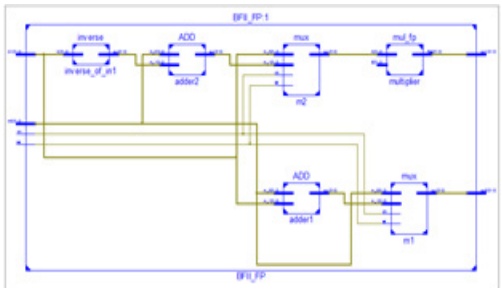


Figure 4. RTL Schematic of Butterfly-II.

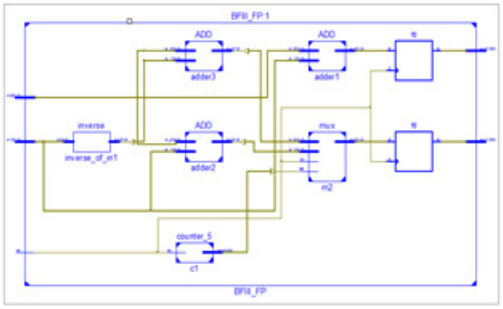
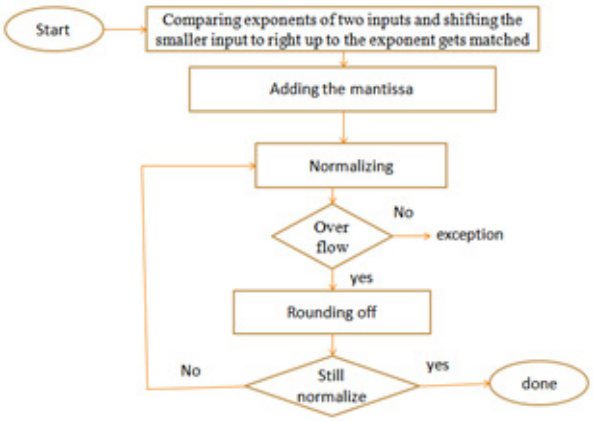


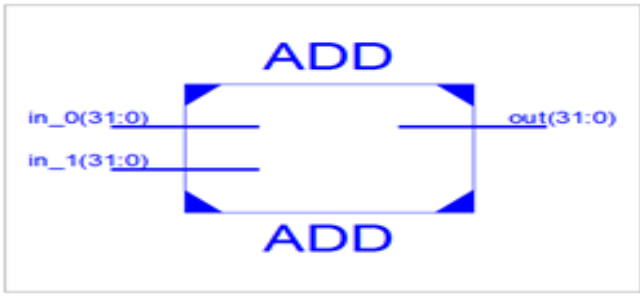
Figure 5. RTL Schematic of Butterfly-III.

3. IEEE754

IEEE-754 standardis a technical standard established by IEEE and the most widely used standard for floating-point computation. Single-precision floating-point format is a computer number format that occupies 32 bits in a computer memory and represents a wide dynamic range of values by using a floating point. Floating point addition design follows the flow chart in Figure 6 (a), and RTL schematic of floating point number is shown in Figure 6 (b).



(a)



(b)

Figure 6. . (a) Flowchart for floating point number addition. (b) RTL schematic of floating point number adder.

4. FFT Architectures

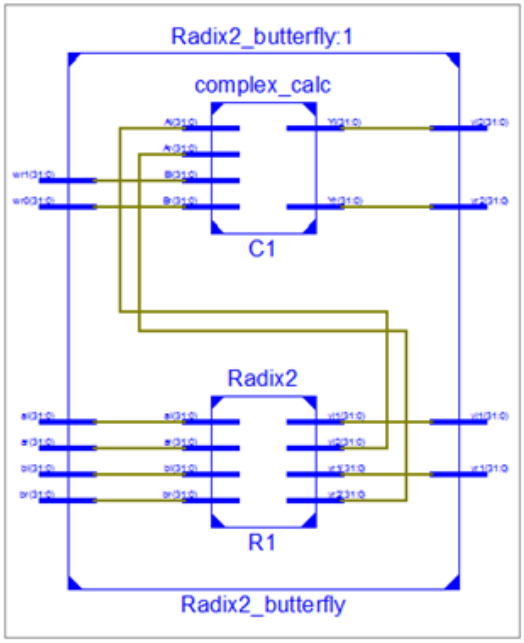


Figure 7. RTL schematic of Radix-2 DIF-FFT.

4.1 Radix – 2 FFT

The basic butterfly structure of radix-2 consists of two inputs and two outputs, which is shown in Figure 7. The inputs are x[n], x[n + N/2] and the digits of output in reversed order X[K], X[K+N/2]. Decimation in Frequency FFT is derived from the equation 1.

The N number of samples is divided into two intervals: One is even samples and the other is odd samples.

$$X[K]=\sum_{n=even}^{N-1} x(n)W_N^{nk}+\sum_{n=odd}^{N-1} x(n)W_N^{nk} \quad (4)$$

$$=\sum_{m=0}^{\frac{N}{2}-1} x(2m)W_N^{2mk}+\sum_{m=0}^{\frac{N}{2}-1} x(2m+1)W_N^{(2m+1)k} \quad (4.1)$$

Equation 4.1 is rewritten in the form of equation 4.2

$$X[K]=F_1[K]+W_N^k F_2[K] \quad (4.2.1)$$

$$X[K+N/2]=F_1[K]-W_N^k F_2[K] \quad (4.2.2)$$

Where as F_1 and F_2 are periodic with period of N/2, from equation 4.2.1 and 4.2.2 radix-2 DIF-FFT designed for 32 bit two input that is N=2 point is as shown in Figure 7. Radix-2 DIF-FFT consist of butterfly-I structure as shown in Figure 3, and complex multiplier that is floating point multiplier unit which considers the real and imaginary inputs of inr0, inr1, ini0, ini1 of the butterfly unit. Wr0, Wr1 are one of the inputs of multiplier and other inputs are output of the butterfly fed to the input multiplier. Outputs are outr0, outi0, outr1 and outi1.

4.2 Radix-4 FFT

For higher radix FFT, that is the N=32 or 64 point FFT the number of the butterfly stages are more and the hardware complexity also increases. For this reason a radix-2^2 or 4 was chosen. The radix-4 butterfly structure is shown in Figure 8. From the Figure 8, it is clear that it contains a four point DIF-FFT. This type of 4 point FFT can be designed by the help of Radix-2 butterfly structure but it contains 4 blocks of radix 2 architectures.

$$X[K]=\sum_{n=\frac{3N}{4}}^{N-1} x(n)W_N^{nk}+\sum_{n=\frac{2N}{4}}^{\frac{3N}{4}-1} x(n)W_N^{nk}+\sum_{n=\frac{N}{4}}^{\frac{2N}{4}-1} x(n)W_N^{nk}+\sum_{n=0}^{\frac{N}{4}-1} x(n)W_N^{nk} \quad (4.3)$$

Whereas radix-4 butterfly in Figure 8 contains only single block to design 4-point DIF-FFT. Radix-4 basic computation is as shown in equation 4.3.

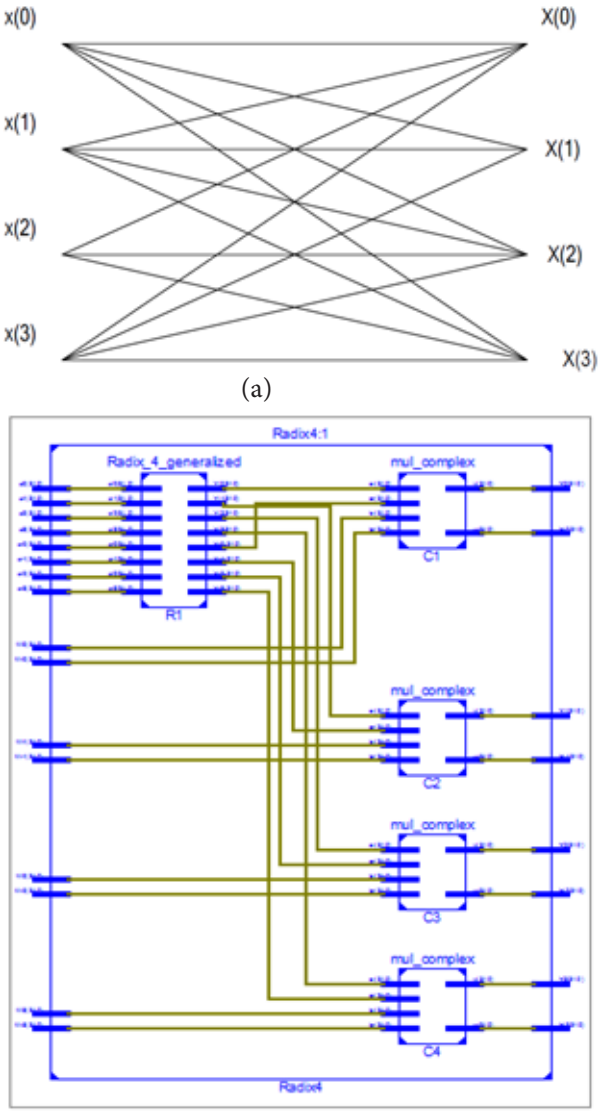


Figure 8. (a) Radix-4 butterfly structure. (b) RTL Schematic radix-4 butterfly structure.

$$X[4K]=\sum_{n=0}^{\left(\frac{N}{4}\right)-1}\left[x(n)+x\left(n+\frac{N}{4}\right)+x\left(n+\frac{2N}{4}\right)+x\left(n+\frac{3N}{4}\right)\right]W_{\frac{N}{4}}^{nk} \quad (4.3.1)$$

$$X[4K+1]=\sum_{n=0}^{\left(\frac{N}{4}\right)-1}\left[x(n)-jx\left(n+\frac{N}{4}\right)-x\left(n+\frac{2N}{4}\right)+jx\left(n+\frac{3N}{4}\right)\right]W_{\frac{N}{4}}^{nk}W_N^n \quad (4.3.2)$$

$$X[4K+2]=\sum_{n=0}^{\left(\frac{N}{4}\right)-1}\left[x(n)-x\left(n+\frac{N}{4}\right)-x\left(n+\frac{2N}{4}\right)-x\left(n+\frac{3N}{4}\right)\right]W_{\frac{N}{4}}^{nk}W_N^{2n} \quad (4.3.3)$$

$$X[4K+3]=\sum_{n=0}^{\left(\frac{N}{4}\right)-1}\left[x(n)+jx\left(n+\frac{N}{4}\right)-x\left(n+\frac{2N}{4}\right)-jx\left(n+\frac{3N}{4}\right)\right]W_{\frac{N}{4}}^{nk}W_N^{3n} \quad (4.3.4)$$

A N=4 point radix-4 DIF-FFT is defined as equation 4.3.1 to 4.3.4.

$X(0) = x(0) + x(4) + x(8) + x(12)$ (4.4.1)

$X(1) = x(0) - jx(4) - x(8) + jx(12)$ (4.4.2)

$X(2) = x(0) - x(4) + x(8) + x(12)$ (4.4.3)

$X(3) = x(0) + jx(4) - x(8) - jx(12)$ (4.4.4)

Equations 4.4.1 to 4.4.4 are simulated and synthesized as shown in Figure 8.1.

4.3 Radix-8 FFT

Radix-8 butterfly unit mainly uses pipelined architectures since it has high throughput. Also Radix-8 contains 8 different inputs which are directly fed to each and every output with corresponding twiddle factor as shown in Figure 9 (a). Radix-8 butterfly can depict the structures which have higher value of N in terms of power of 2 as shown in Figure 9 (b).

Computation of the radix-8 butterfly unit is shown in equation 4.5.

$$X[K]=\sum_{n=0}^{\frac{N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{N}{8}}^{\frac{2N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{2N}{8}}^{\frac{3N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{3N}{8}}^{\frac{4N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{4N}{8}}^{\frac{5N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{5N}{8}}^{\frac{6N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{6N}{8}}^{\frac{7N}{8}-1}x(n)W_N^{nk}+\sum_{n=\frac{7N}{8}}^{N-1}x(n)W_N^{nk} \quad (4.5)$$

The equation 4.5.1 shows the outputs of 8-point DIF-FFT

$Y[0] = x[0] + x[1] + x[2] + x[3] + x[4] + x[5] + x[6] + x[7]$

$Y[1] = x[0] + w_8^1 x[1] + w_8^2 x[2] + w_8^3 x[3] + w_8^4 x[4] + w_8^5 x[5] + w_8^6 x[6] + w_8^7 x[7]$

$Y[2] = x[0] + w_8^2 x[1] + w_8^4 x[2] + w_8^6 x[3] + x[4] + w_8^2 x[5] + w_8^4 x[6] + w_8^4 x[7]$

$Y[3] = x[0] + w_8^3 x[1] + w_8^6 x[2] + w_8^1 x[3] + w_8^4 x[4] + w_8^7 x[5] + w_8^2 x[6] + w_8^5 x[7]$

$Y[4] = x[0] + w_8^4 x[1] + x[2] + w_8^4 x[3] + x[4] + w_8^4 x[5] + x[6] + w_8^4 x[7]$

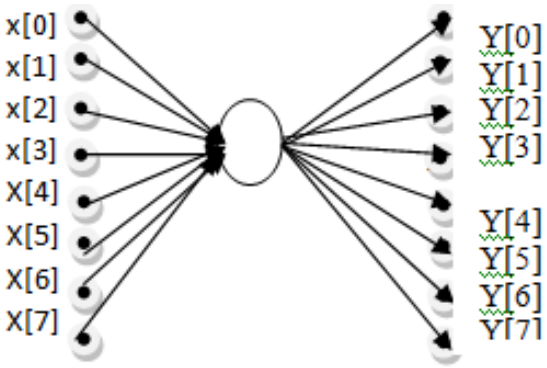
$Y[5] = x[0] + w_8^5 x[1] + w_8^7 x[2] + w_8^4 x[3] + w_8^1 x[4] + w_8^1 x[5] + w_8^6 x[6] + w_8^3 x[7]$

$Y[6] = x[0] + w_8^6 x[1] + w_8^4 x[2] + w_8^2 x[3] + x[4] + w_8^6 x[5] + w_8^4 x[6] + w_8^2 x[7]$

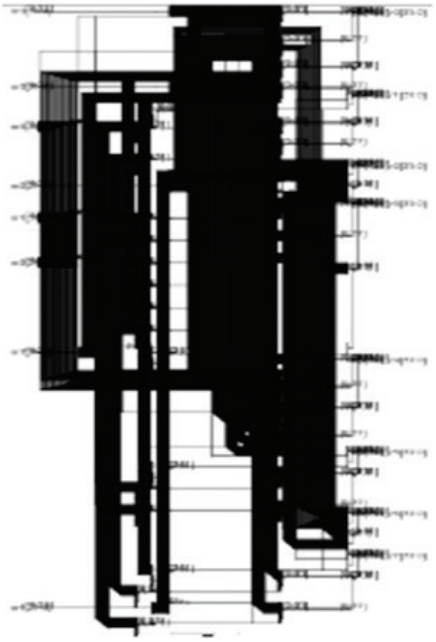
$Y[7] = x[0] + w_8^7 x[1] + w_8^6 x[2] + w_8^5 x[3] + w_8^4 x[4] + w_8^3 x[5] + w_8^2 x[6] + w_8^1 x[7]$ (4.5.1)

Table 1. No. stages and butterfly in different radix

Type N=64	No. Stages log_2^N	No. Butterflies $\frac{N}{2}log_2^N$	Multipliers per stage	Addition per stage
Radix-2	6	32	N/2	N
Radix-4	3	16	3N/4	3N
Radix-8	2	8	$log_8^N - 1$	$4log_2^N - 4$



(a)



(b)

Figure 9. (a) Radix-8 butterfly structure. (b) Radix-8 RTL schematic.

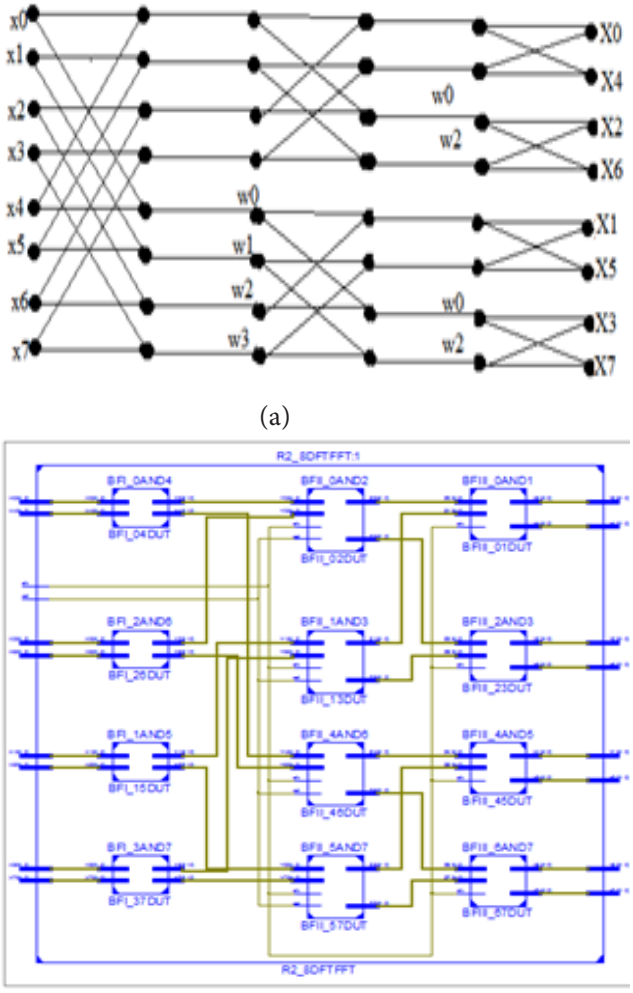
Table 1 shows that N=64 point number of stages and butterfly are required for radix-2, 4, 8. In general radix-r FFT needs N/r radix-r butterflies for each stage and has $\log_r N$ stages.

5. Pipelined Design DIF-FFT

Radix-2 8 point DIF-FFT consists of inputs that are in binary format i.e. points from zero to seven where as outputs are mirror image of input bits. When the input samples are real, then half of the samples are redundant.

The Figure 10 (b) contains groups of butterflies and twiddle factors at corresponding stages. These twiddle

factors are multiplied between each of the intermediate stages. The data flow graph schematic is shown in Figure 10. It consists of three different butterflies for each and every stage.



(a)

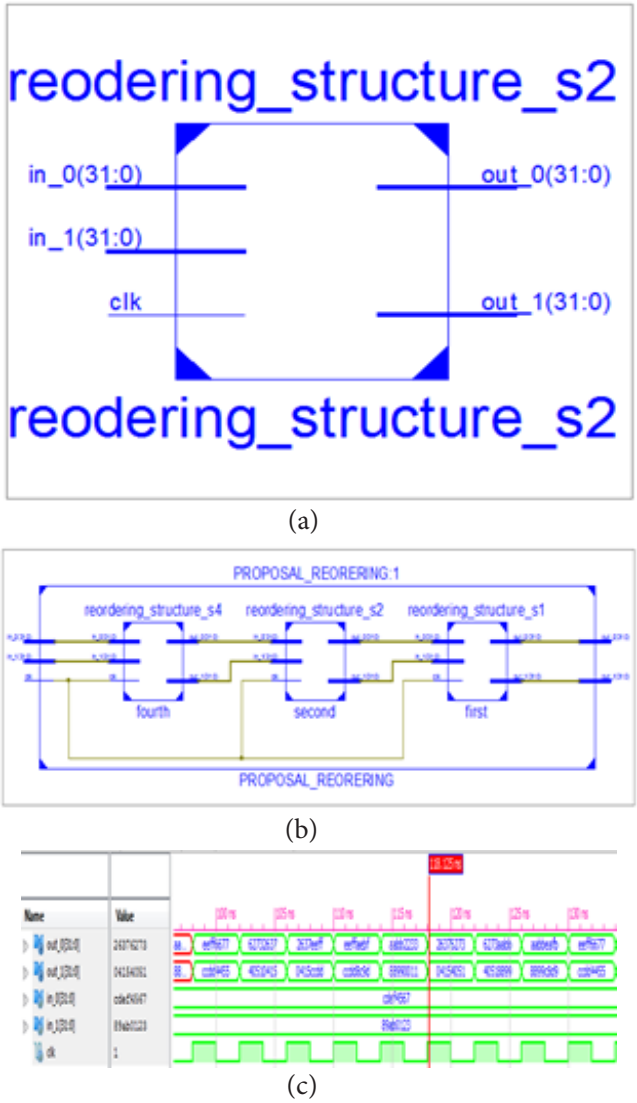
(b)

Figure 10. (a) Radix-2 8-point DIF-FFT flow graph. (b) RTL of eight point data flow graph of radix-2.

5.1 Reordering Structure

The operation of Reordering blocks shown in Figure 11 contains switch and delay elements. Switch consists of control signals generated for different stages of architecture and it can be generated by simple counter block. It is inserted in front of butterflies and it is triggered by a clock pulse.

Figure 11.1 shows the reordering structure which consists of the 4, 2, 1 data sampled reordering stages and hence they are cascaded together. This type of reordering stage can be eliminated since it is an inherent problem of FFT computation. The output samples are in a scrambled order. In order to get the correct order, the reordering is done in such a way that it is used after the butterfly structure of Figure 3. The reordering structure is to be designed as shown in Figure 11.



(a)

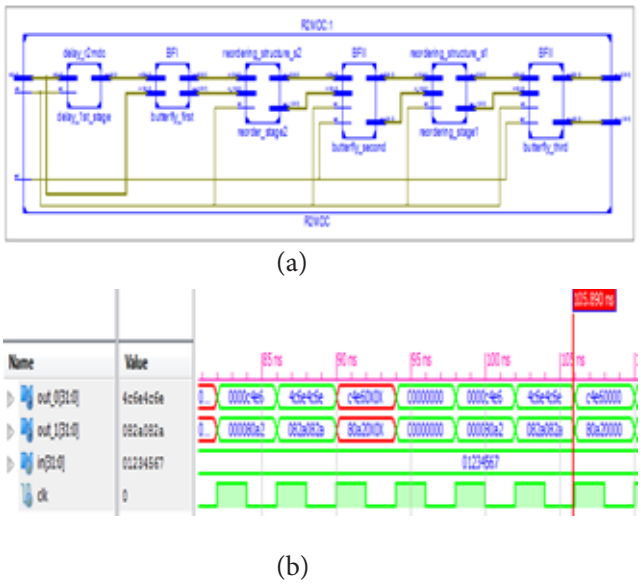
(b)

(c)

Figure 11. (a) Reordering structure for 2 samples. (b) Cascaded reordering stages. (c) Output of reordering structure.

5.2 R2MDC

Radix-2 multi path delay commutator is one of the most general approaches for designing the pipelined implementation of radix-2 FFT designs as shown in Figure 12. In this approach two inputs commutate as fast as possible. Input data samples are broken into two equal data samples and half of the data samples are transferred directly to the first input of the butterfly I, the remaining half sample is delayed and again fed to the other input of the same butterfly.



(a)

(b)

Figure 12. (a) R2MDC architecture. (b) Output response of the R2MDC..

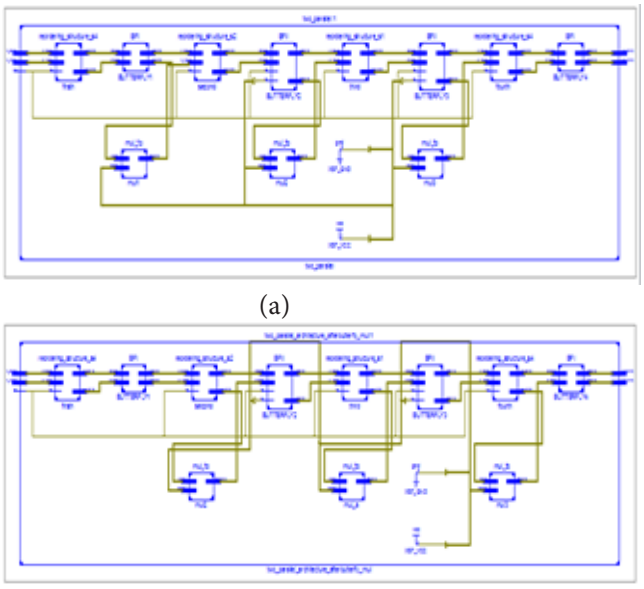
When multiple input multiple output systems data is used, throughput can be very high and it is used in communication systems. These architectures don't support the feedback paths. But it will use feed forward paths as switch boxes as mentioned in Figure 12 (a). Response of R2MDC is shown in Figure 12 (b).

5.3 Two Level Parallel FFT Architecture

The hardware components required in feed-forward architecture is only 50%. This can be observed from the data flow graph of R2MDC in Figure 12, where half of the time period is spent for null operations. It can be extended by some modification in the folding sets and reordering the structure place for the samples. In this way, it can be an efficient architecture in terms of hardware utilization and power consumption.

The DFG of radix-2 16 point DIF-FFT is shown in Figure 13. (a) All the nodes represent the butterfly operations and multiplier unit. It can be observed that the hardware utilization is 100% in this architecture.

Figure 13 (a) all shows the multiplier after the butterfly structure and Figure 13 (b) shows the multiplier's position after the reordering stages. In a general case of N-point DIF-FFT with N power of 2, the architecture needs log_2^N complex butterflies, $3\frac{N}{2}-2$ delay elements or buffers, and log_2^N-1 complex multipliers. The main difference in the two-parallel architectures is the position of the multiplier unit in between the butterflies.



(a)

(b)

Figure 13. (a) Two-Level Parallel Architecture-1. (b) Two-level parallel architecture-2.

6. Comparison and Analysis

Table 2. Comparison of R2MDC and parallel architectures

Architecture	No. of multiplier	No. of adders	No. Of delays	Throughput
R2MDC	$2(log_4^N-1)$	$4log_4^N$	$3\frac{N}{2}-2$	1
Parallel architecture	log_4^N-1	$4log_4^N-2$	$9\frac{N}{8}-2$	2

Comparison is made between the different radix algorithms as shown in Figure 14 and pipelined architectures and R2MDC in terms of the required number of adders, delay elements, multiplier units, throughput and control complexity as shown in Table 2. The Figure 13 shows feed forward architectures which process two samples in parallel with only one clock cycle unit of time.

The power consumption of the feed forward architectures is calculated using the equation 4.6

$$P_{serial} = C_{serial} V^2 f_{serial} \quad (4.6)$$

Where C_{serial} denotes the total capacitance of the serial circuit. V is the supply voltage. f_{serial} is the clock frequency of the circuit. P_{serial} denotes the power consumption of the

serial architecture. This calculation can be applied to the parallel architecture by introducing new modified power that is shown in equation 4.7

$$Pparallel = Cparallel V^2 \frac{fserial}{L} \quad (4.7)$$

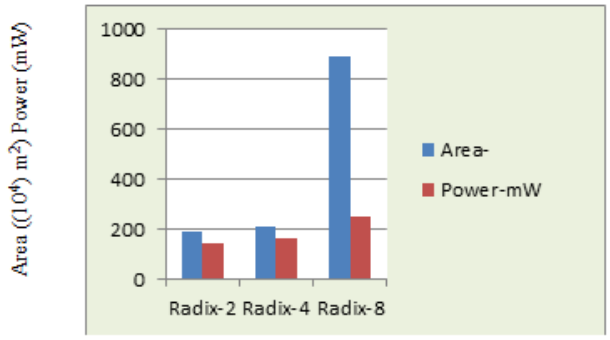


Figure 14. Different Radix-2, 4 and 8 power and area.

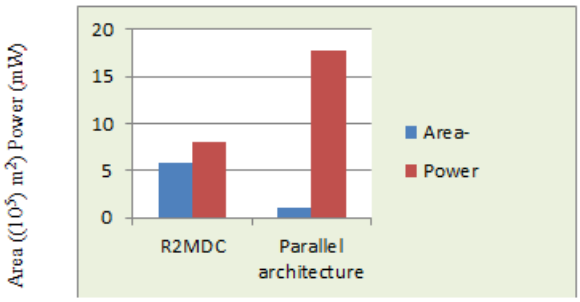


Figure 15. Area and Power of R2MDC and Parallel Architecture.

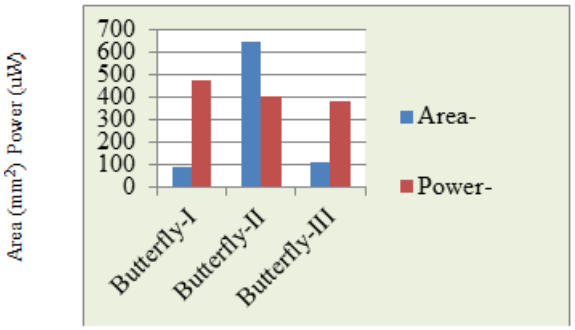


Figure 16. Different Butterfly structures and their area and power.

Figure 16 shows the summary of area and power of different architectures discussed in the paper. Although the parallel architecture consumes more power, it occupies less area when compared to the R2MDC architecture.

7. Conclusions

A novel approach to derive the different radix algorithms of FFT and parallel architectures are developed for better performance for real valued signals. Several architectures have been modeled with Verilog HDL with IEEE-754 floating point number representation. Arithmetic operations like addition and a complex multiplication have also been computed. It is used in floating point unit and central processing unit for arithmetic operations. Parallel architecture is more area efficient than R2MDC, whereas the radix-2 multi-path delay commutator is simple to implement, since feedback is not needed in the design. Also it is a more power efficient architecture as it is evident in the graph shown in Figure 15. However, it requires more area. In R2MDC, area is compromised for low power. The simulation work has been done in Xilinx 14.3. The synthesis has been done in Cadence 180nm technology.

8. References

1. Oppenheim AV, Schafer RW, Buck JR. Discrete-time sig¬nal processing. 2nd ed. Englewood Cliffs, NJ, USA: Prentice Hall; 1998.
2. Ayinala M, Brown M, Parhi K. Pipelined parallel FFT archi¬tectures via folding transformation. IEEE Trans Very Large Scale Integer (VLSI) Syst. 2012 June; 20(6):1068–81.
3. Wang A, Chandrakasan AP. Energy-aware architectures for a real-valued FFT implementation. Proc Int Symp Low Power Electron Design; 2003 Aug. p. 360–5.
4. Ayinala M, Parhi KK. Parallel-pipelined radix-FFT archi¬-tecture for real valued signals. Proc Asilomar Conf Signals Syst Comput; 2010 Nov. p. 1274–8.
5. Parhi KK, Wang CY, Brown AP. Synthesis of control circuits in folded pipelined DSP architectures. IEEE J Solid State Cir¬cuits. 1992; 27(1):29–43.
6. Parhi KK. VLSI digital signal processing systems: Design and implementation. Hoboken, NJ, USA: Wiley; 1999.
7. Smith WW, Smith JM. Handbook of real-time fast fourier transforms. New York, NY, USA: Wiley-IEEE Press; 1995.
8. Yang K-J, Shang-Ho Tsai S-H. IEEE, MDC FFT/IFFT Pro¬cessor with Variable Length for MIMO-OFDM Sys-tems. IEEE transactions on very large scale integration (vlsi) sys¬tems. 2013 Apr; 21(4):720–31.
9. Lee J, Lee H, Cho SI, Choi S. A high-speed two parallel ra-dix-FFT/IFFT processor for MB-OFDM UWB systems. Proc IEEE Int Symp Circuits Systems; 2006 May. p. 4719–22.
10. Sorensen H, Jones D, Heideman M, Burrus. Real-valued fast flourier transform algorithms. IEEE Trans Acoust Speech Signal Process. 1987 Jun; 35(6):849–63.
11. Baas BM. A low-power, high-performance, 1024-point FFT processor. IEEE J Solid-State Circuits. 1999 Mar; 34(3):380–7.