Signal Processing System Design Laboratory (EE69205)

LAB REPORT

by

YASH ANAND

(Roll No.- **23EE65R22**)

SIGNAL PROCESSING AND MACHINE LEARNING

Department of ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Kharagpur - 721302, India

September, 2023

|| Experiment 4 ||
|| Implementation of Radix-2 DIT and DIF FFT algorithm and comparison with the DFT algorithm ||

AIM

To compare the computational complexity, space complexity, and execution complexity of two different Fast Fourier Transform (FFT) algorithms (Radix-2 Decimation in Time, and Radix-2 Decimation in Frequency) with the Discrete Fourier Transform (DFT) algorithm.

FUNCTION USED:

- **cmath.exp():** This function calculates the exponential function of a complex number.
- **fft_radix2_dit():** This function implements the radix-2 decimation in time (DIT) fast Fourier transform (FFT) algorithm.
- **fft_radix2_dif():** This function implements the radix-2 decimation in frequency (DIF) fast Fourier transform (FFT) algorithm.

OBSERVATION:

→ The DIT FFT and DIF FFT algorithms exhibit significantly lower computational complexity compared to the DFT algorithm, particularly as the input size increases.

→ The space complexity of each algorithm remains constant with respect to the input size. It is primarily determined by the size of the input sequence and the temporary variables used in the computation.

→ The DIT FFT and DIF FFT algorithms consistently outperform the DFT algorithm in terms of execution time for a wide range of input sizes. This trend becomes more pronounced as the input size increases.

→ Based on the computational complexity and execution time observations, it is evident that the FFT algorithms (DIT and DIF) are more efficient for computing the Discrete Fourier Transform compared to the straightforward DFT approach.

→ Both FFT algorithms exhibit a logarithmic increase in computational complexity and execution time as the input size grows exponentially. This indicates that they are well-suited for handling large datasets efficiently.

OUTPUT:

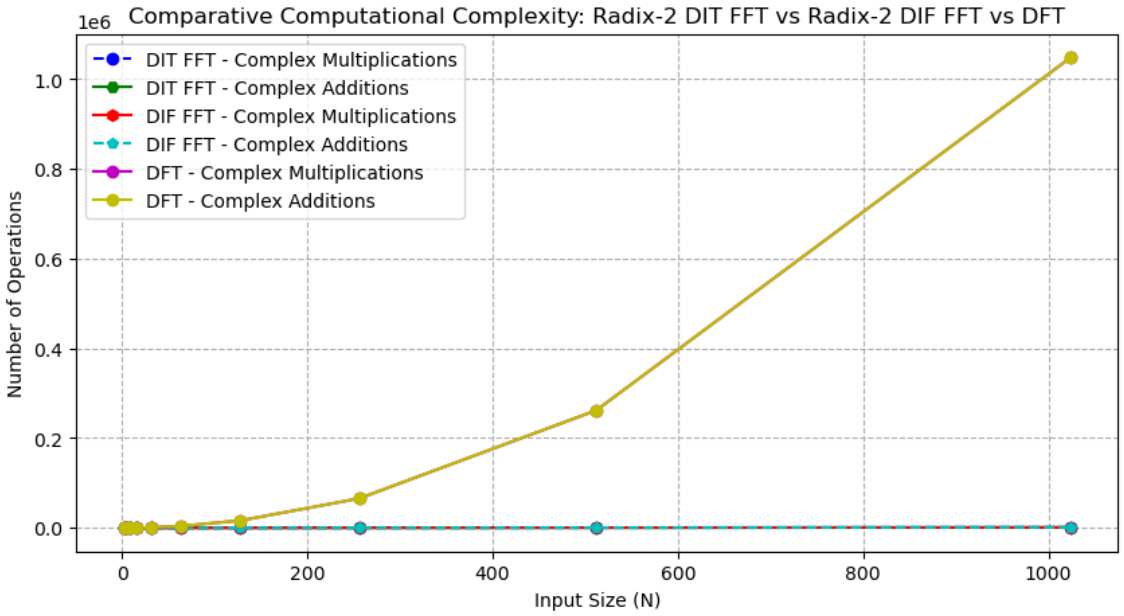

FIGURE 1: **C**omparative Computational Complexity: Radix-2 DIT FFT vs Radix-2 DIF FFT vs DFT

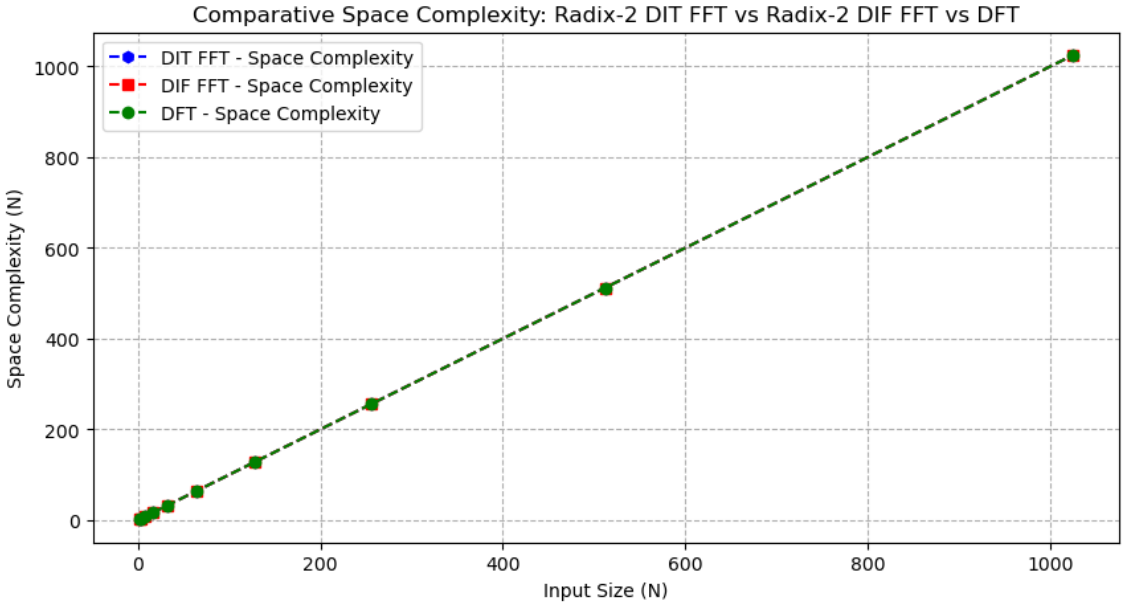

FIGURE 2: **C**omparative Space Complexity: Radix-2 DIT FFT vs Radix-2 DIF FFT vs DFT

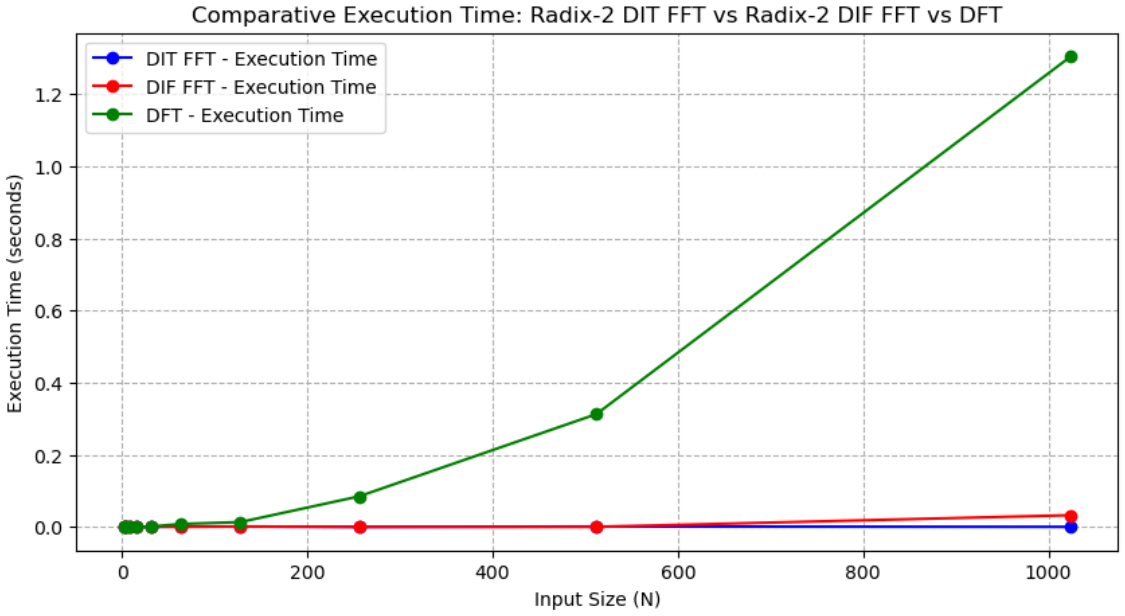

FIGURE 3: **C**omparative Execution Time: Radix-2 DIT FFT vs Radix-2 DIF FFT vs DFT