ES 203: DIGITAL SYSTEMS PROJECT
FPGA IMPLEMENTATION OF FAST FOURIER TRANSFORM

Jainish Chauhan(17110038) | Ojas Mithbavkar(17110083) | Mrinal Anand(17110087) | Rohit Patil(17110126)

• Objective:

FPGA Implementation of Fast Fourier Transform(FFT) algorithm

•Theory

- The Discrete Fourier Transform(DFT) is a linear transformation of the vector x_n (the time domain signal samples) to the vector Xm (the set of coefficients of component sinusoids of time domain signal). Suppose our signal is x_n for n=0...N −1, and $x_n = x_{n+jN}$ for all n and j. The discrete Fourier transform of x is given by

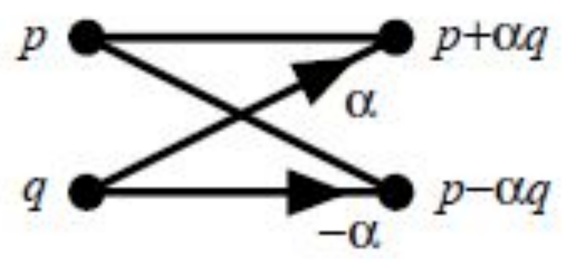
$$X_m = \sum_{n=0}^{N-1} x_n w^{nm}$$

where N is the size of the vector, $\omega = e^{2\pi i/N}$ are the "roots-of-unity" (twiddle factors), and $0 \le m < N$.

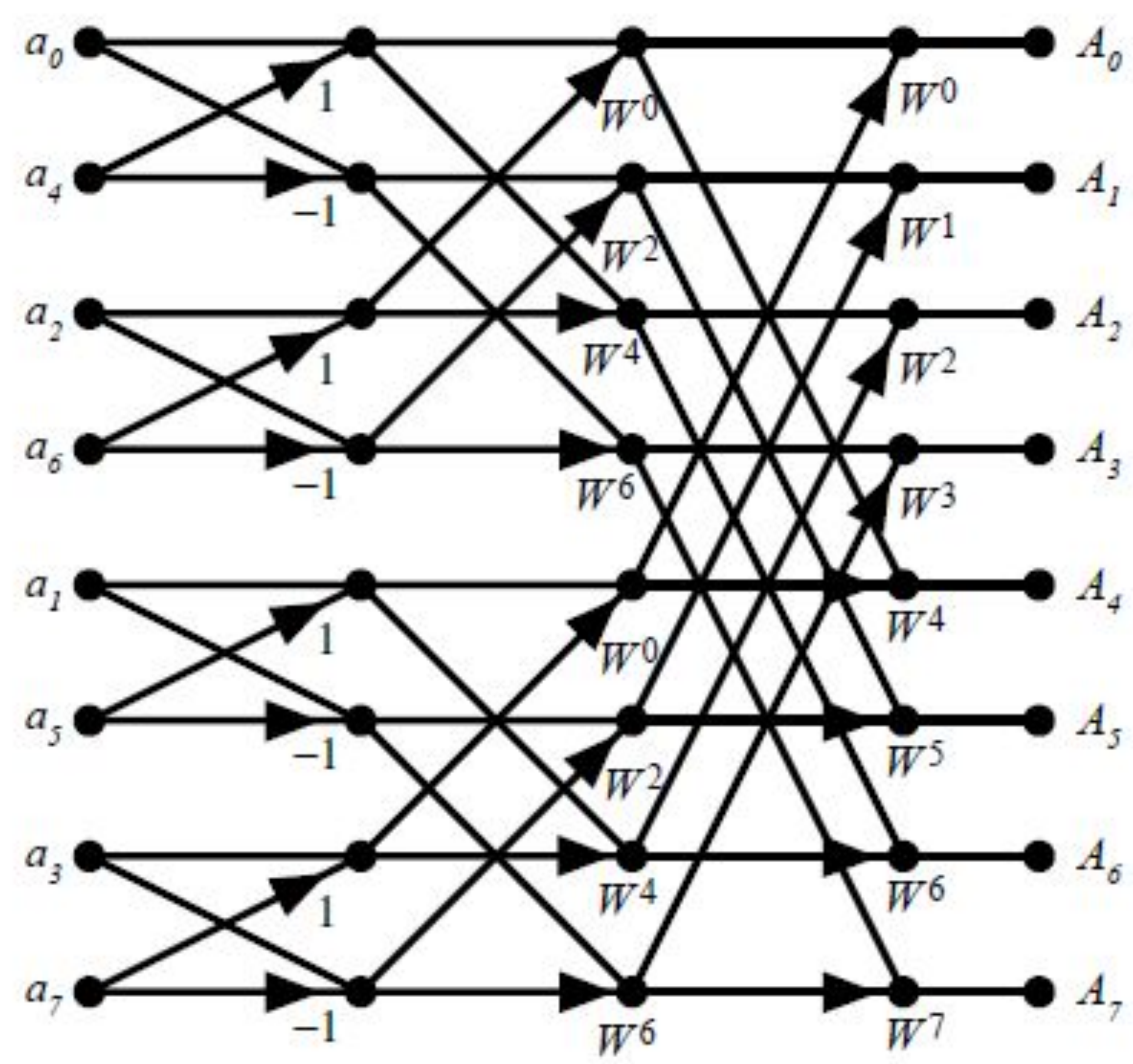
- To compute the DFT of an N-size sequence using above equation would take $O(N^2)$ multiplies and adds.
- Precisely, every '**point**' (x_n)would require 4N multiplies (multiplying 2 complex numbers). These 4N multiplies for N times would give $4N^2$ multiplies.
- The FFT is a fast algorithm for computing the DFT. If we take the 2-point DFT and 4-point DFT and generalize them to 8-point, 16-point, ..., 2^r-point, we get the FFT algorithm.
- The FFT algorithm computes the DFT using $O(N\log_2 N)$ multiplies and adds.There are many variants of the FFT algorithm. We employ the Cooley-Tukey Radix-2 FFT algorithm here.
- The FFT algorithm decomposes the DFT into $\log_2 N$ stages, each of which consists of N=2 butterfly computations.

$$X_m = \sum_{n=0}^{N/2-1} x_n w^{nm} + w^{mN/2} \sum_{n=0}^{N/2-1} x_{n+N/2} w^{nm}$$

- The basic computational step of the FFT algorithm is a butterfly.
- Each butterfly computes two complex numbers of the form p +αq and p −αq, so it requires one complex multiply and two complex adds. This works out to 4 real multiplies and 6 real adds per butterfly.



The complete butterfly diagram is :



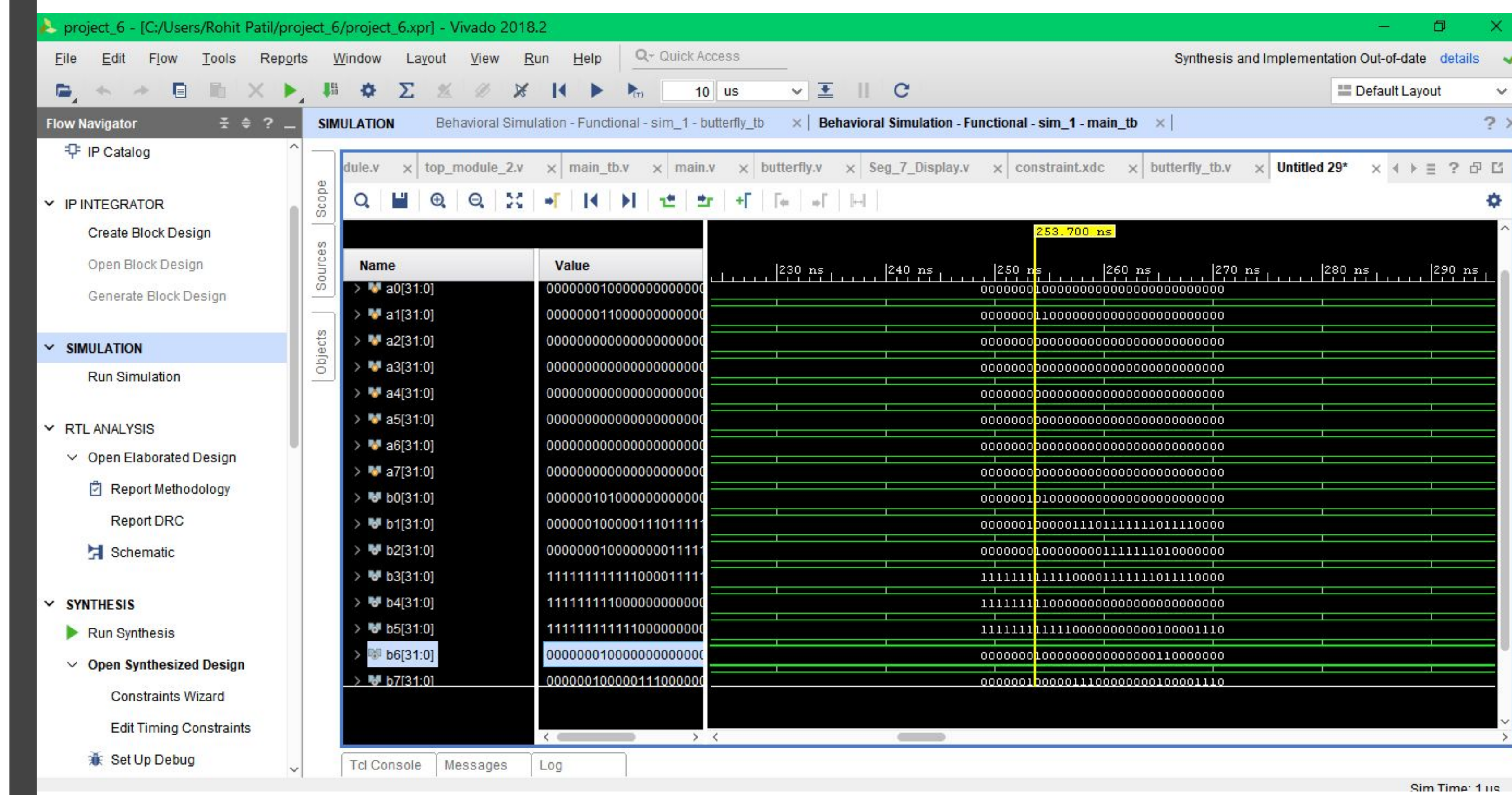
It is observed that the outputs are not in conventional order. Below is a table of j and the index of the jth input sample, n_j:

ILLUSTRATION OF THE BIT-REVERSED INDICES.

Index	binary	Bit reversed index	binary
0	000	0	000
1	001	4	100
2	010	2	010
3	011	6	110
4	100	1	001
5	101	5	101
6	110	3	011
7	111	7	111

The pattern is obvious if input and output indexes are written in binary. It is observed that each output index is the bit-reversal of corresponding input index.

The output for a sample of inputs:



• Applications:

- Frequency Domain Downsampling
- Fractal Image Compression
- Phase Only Correlation
- FFT Processors for OFDM
- Three-Dimensional Face Recognition

• References:

1. Slade, G. W. (2013). The Fast Fourier Transform in Hardware: A Tutorial Based on an FPGA Implementation. I: ResearchGate.
2. Heckbert, P. (1995). Fourier Transforms and the Fast Fourier Transform (FFT) Algorithm. Computer Graphics, 2, 15-463.
3. Brigham, E. O., & Brigham, E. O. (1988). The fast Fourier transform and its applications (Vol. 448). Englewood Cliffs, NJ: prentice Hall.
4. Rao, K. R., Kim, D. N., & Hwang, J. J. (2011). Fast Fourier transform-algorithms and applications. Springer Science & Business Media.
5. https://github.com/ameyk1/Fast-Fourier-Transform