

Notas relevantes referentes al desarrollo de FFT mediante Yocto Project

Por: German A. Quesada

1. Descripción general:

- La transformada discreta de Fourier representa un muestreo de la transformada de Fourier de la función de la cual se tomaron los puntos utilizados para generar la transformada discreta de Fourier.
- Entre más alto el valor de N , más procesamiento necesita el computador. N determina la complejidad del proceso.
- FFT es un algoritmo eficiente para el calculo de la DFT
 - Comúnmente utiliza el algoritmo Radix-2 o Radix-4 para el calculo de DFT
- Video relevante: <https://www.youtube.com/watch?v=ysjbyYvHZOY>

2. Cualidades de DFT:

- N : numero de muestras

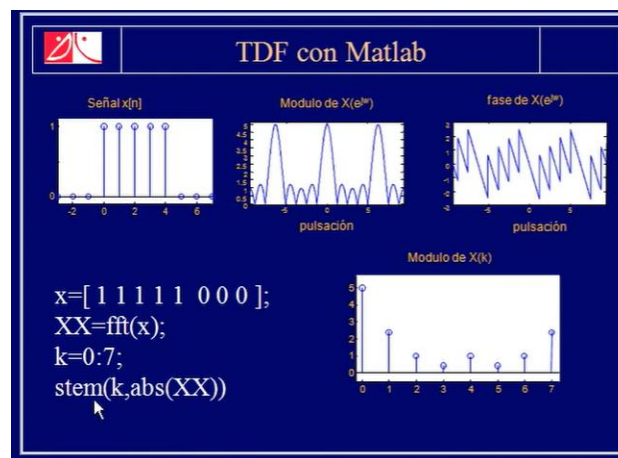
3. Requerimientos de DFT:

- Para números N menores que cero la función debe ser cero
- Para numero N mayores que $N-1$ la función debe ser cero

4. Enventanado de una función

- Multiplicar la función a la que se le desea hacer la TDF por una función ventana para cumplir los requerimientos anteriores. Entre más ancha la ventana más estrecha el resultado en el dominio de frecuencia.

5. Matlab



6. Notas referentes a algoritmo utilizados:

- Libro DSP: <http://www.elcom-hu.com/Electrical/Digital%20Singnal%20Proccessing/4th/4th%20Digital%20Signal%20Procesing%20-%20Proakis%20and%20Manolakis.pdf>
- Mariposa:

- Videos relevantes:
 1. Algoritmo con 2 muestras: <https://www.youtube.com/watch?v=XtypWS8HZco>
 2. Algoritmo con 8 muestras: <https://www.youtube.com/watch?v=fCTfKL3XluA>
 3. Algoritmo DSP: <https://www.youtube.com/watch?v=8cjDKirNIko&list=PLuh62Q4Sv7BUSzx5Jr8Wrxn-U10qG1et&index=13>
- Imágenes relevantes:

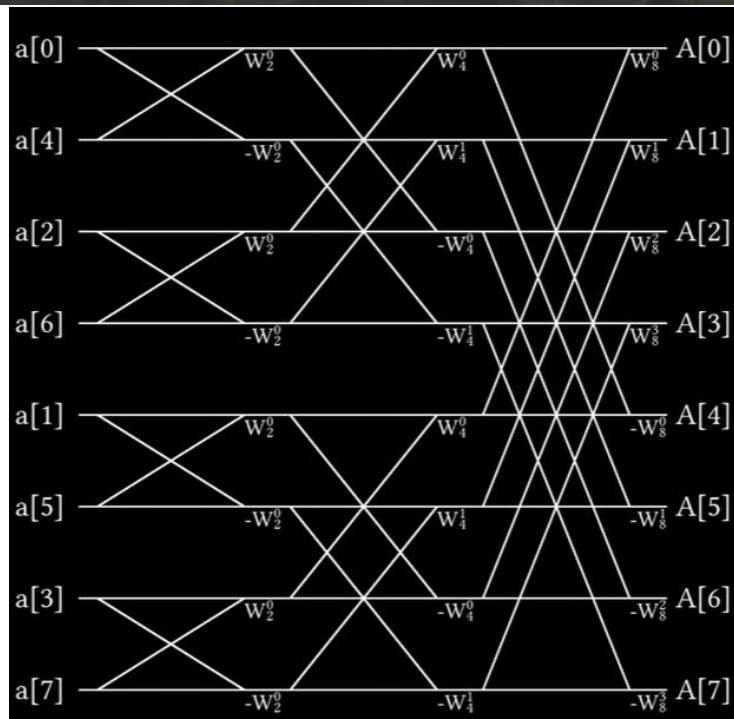
$$F_k = \sum_{n=0}^{N-1} f_n \left[e^{-2\pi i k n / N} \right] \rightarrow \omega_N^k = e^{-2\pi i k / N}$$

$a[0]$ \searrow $b[0]$ \downarrow ARRAY OF ω 's
 $a[1]$ \swarrow $b[1]$

$$b[0] = a[0] + \omega_2^0 a[1]$$

$$b[1] = a[0] + \omega_2^1 a[1]$$

$\omega_2^0 = -\omega_2^1$



DECIMATION IN TIME (N EVEN)

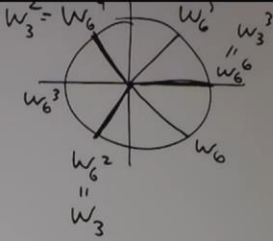
$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk}$$

$$= \sum_{n \text{ EVEN}} x[n] W_N^{nk} + \sum_{n \text{ ODD}} x[n] W_N^{nk}$$

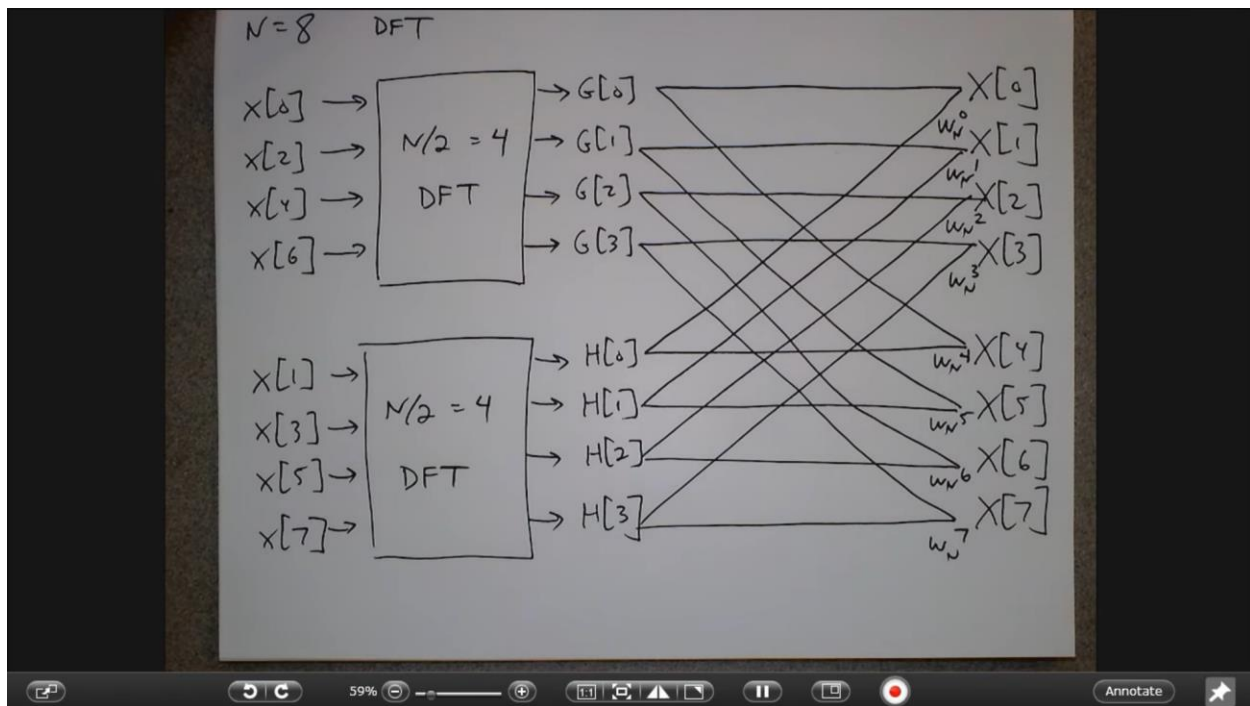
$$\stackrel{n=2r}{=} \sum_{r=0}^{N/2-1} x[2r] W_N^{2rk} + \sum_{r=0}^{N/2-1} x[2r+1] W_N^{(2r+1)k}$$

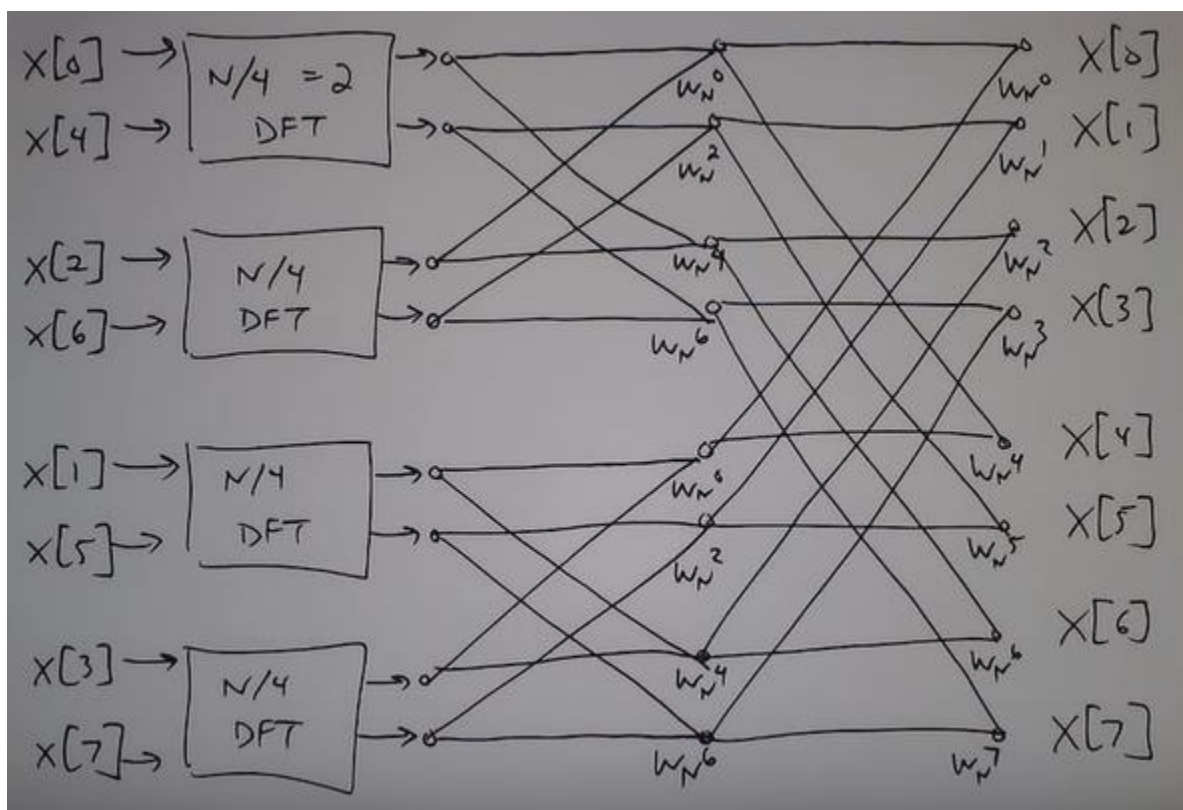
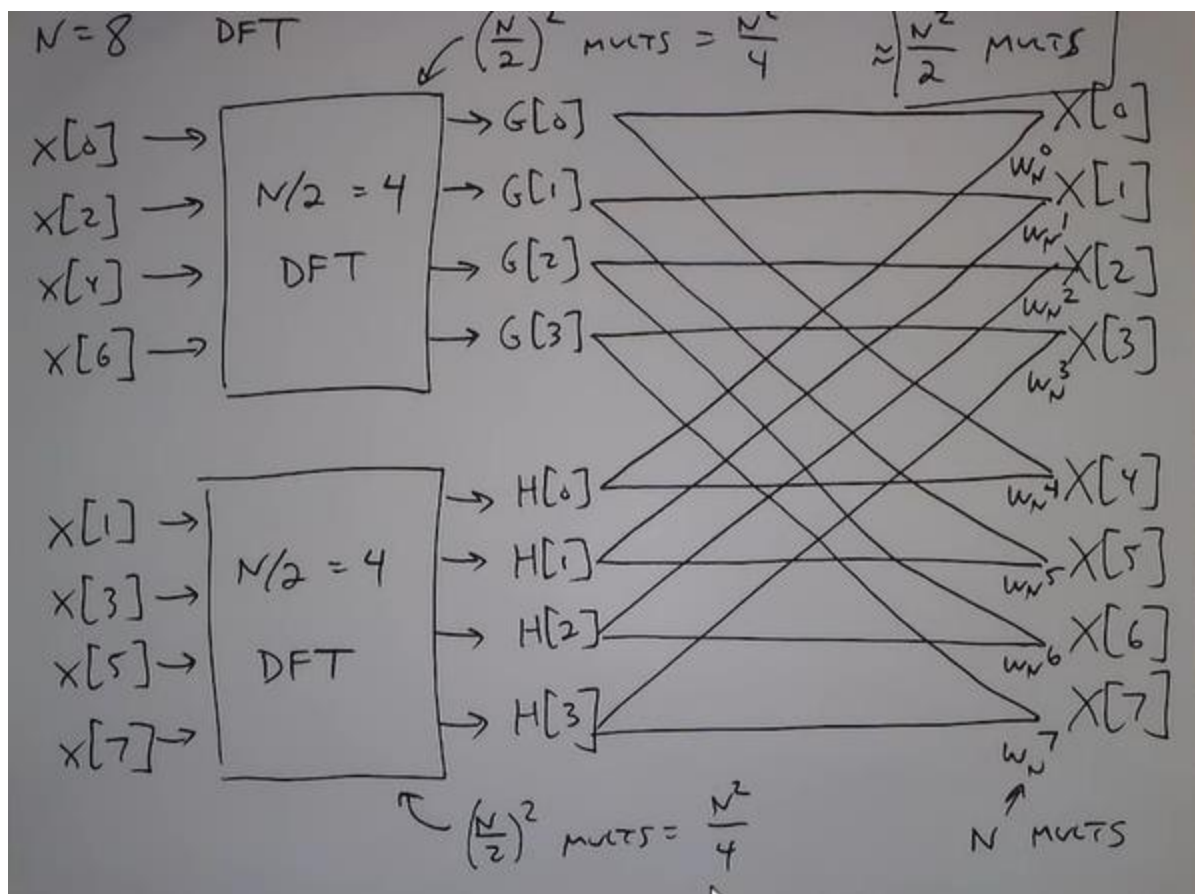
$$= \sum_{r=0}^{N/2-1} x[2r] (W_N^2)^{rk} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] (W_N^2)^{rk}$$

$N=6$
 $G[0] = G[3]$
 $G[1] = G[4]$
 $G[2] = G[5]$



$$= \underbrace{\sum_{r=0}^{N/2-1} x[2r] W_{N/2}^{rk}}_{\text{LENGTH } N/2 \text{ DFT OF EVEN ENTRIES}} + W_N^k \underbrace{\sum_{r=0}^{N/2-1} x[2r+1] W_{N/2}^{rk}}_{\text{LENGTH } N/2 \text{ DFT OF ODD ENTRIES}}$$

$$X[k] = G[k] + \underline{W_N^k} H[k] \quad k=0,1,\dots,N-1$$





LENGTH 2 DFT:

$$x[0], x[1] \rightarrow X[0], X[1]$$

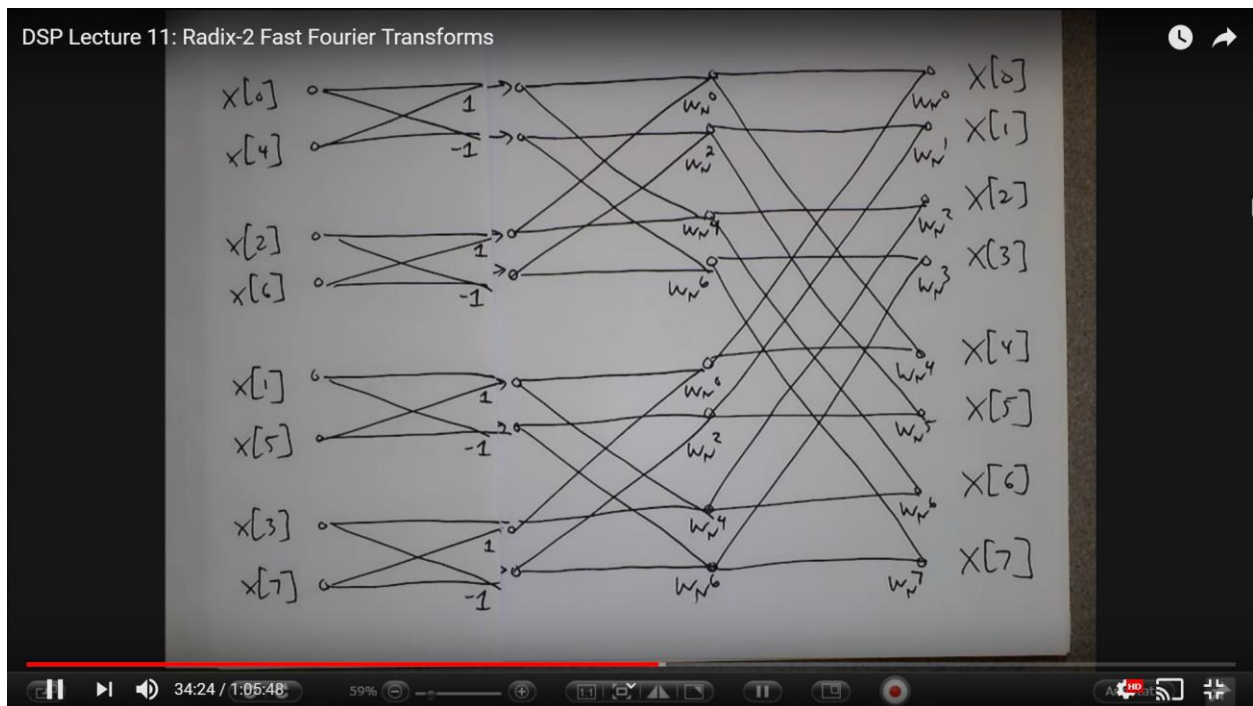
$$X[k] = \sum_{n=0}^1 x[n] W_2^{nk}$$

$$= \sum_{n=0}^1 x[n] (-1)^{nk}$$

$$X[0] = x[0] + x[1]$$

$$X[1] = x[0] - x[1]$$


Video player controls: 59%, 34:24 / 1:05:48, Annotate



Los Estados se refiere a las columnas. El ejemplo del video presenta 3 columnas.

IF N IS A POWER OF 2, WE
CAN RECURSIVELY BREAK THE DFT INTO
 $\log_2 N$ STAGES.

$\Rightarrow \boxed{N \log_2 N}$ MULTIPLIES

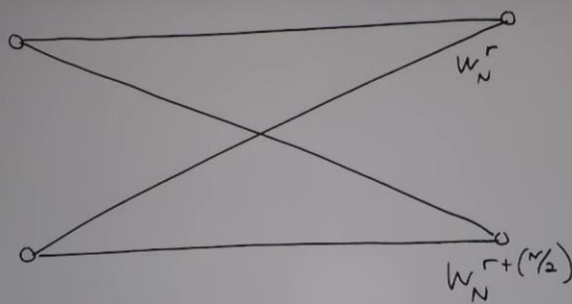
$$N = 2^{10} = 1024$$

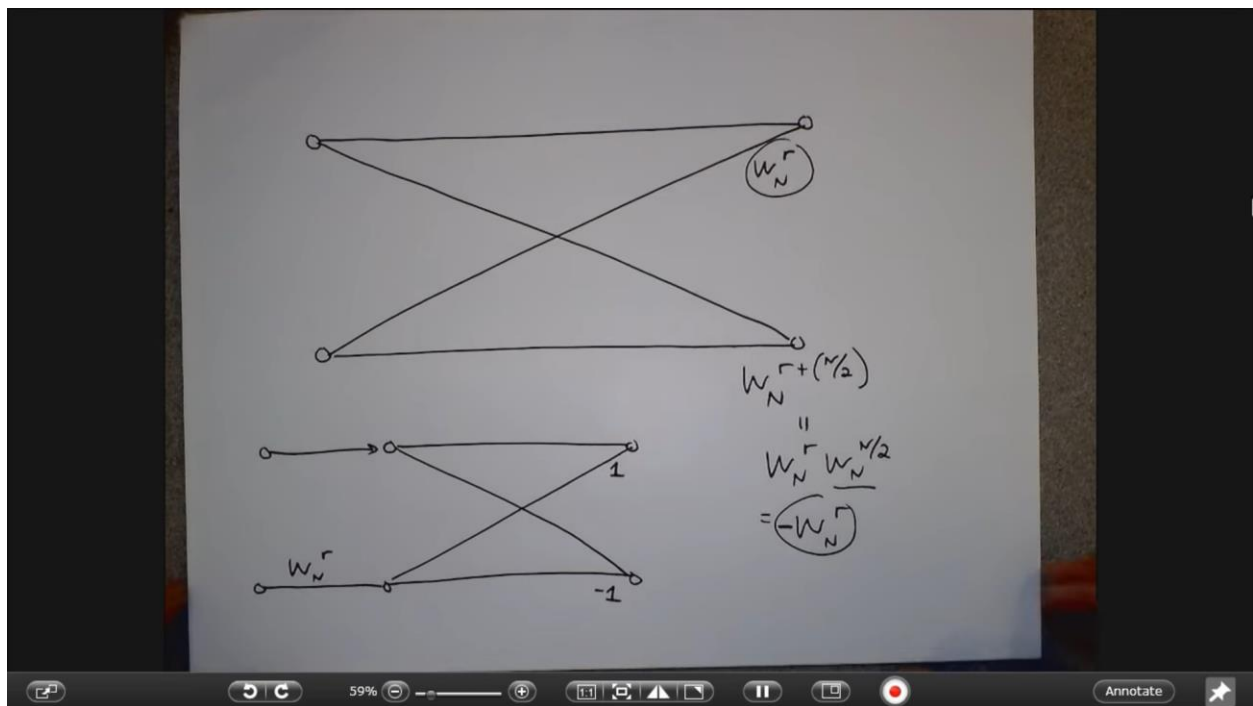
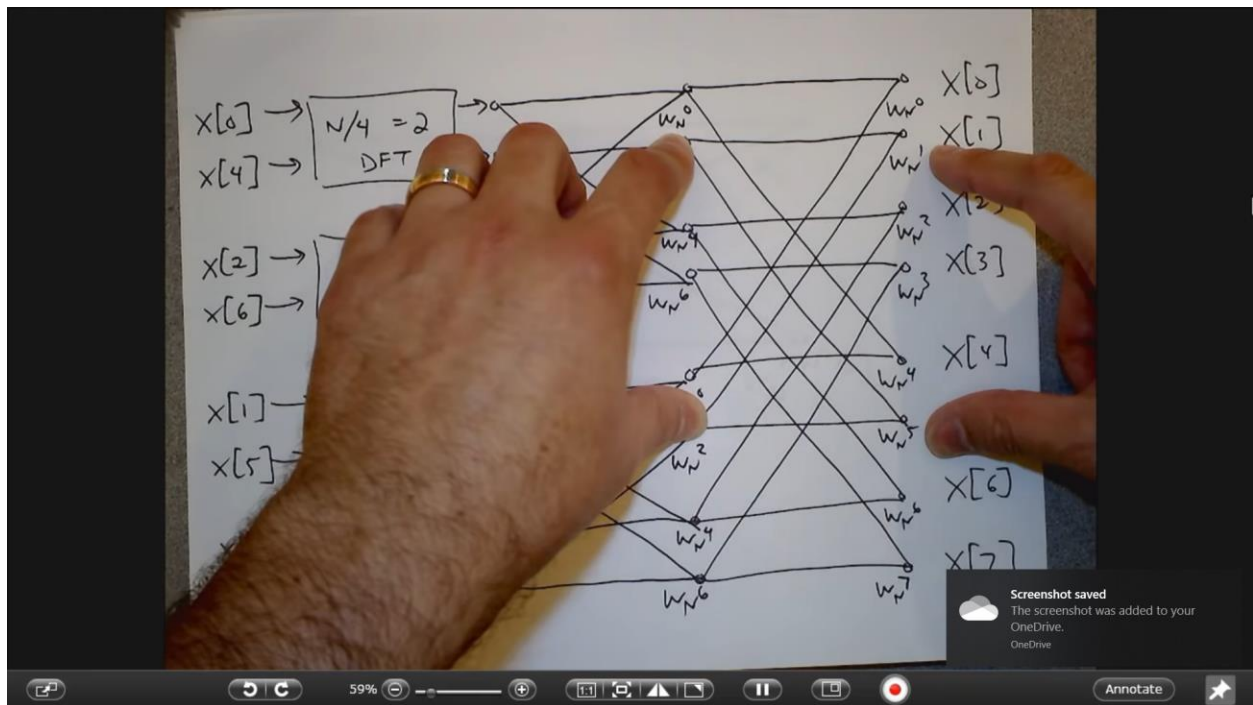
$$N^2 \approx 1000000 \text{ vs.}$$

$$N \log N = 10240. \quad \times 100$$

Note el siguiente patrón el cual nos permite multiplicar de forma previa por la cte W_N y posteriormente agregarla y restarla:

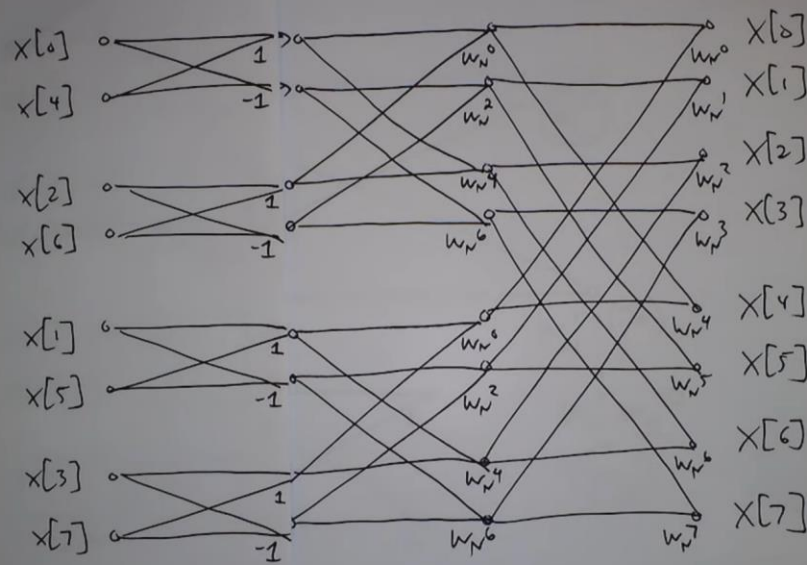
DSP Lecture 11: Radix-2 Fast Fourier Transforms





Ahora bien, note que el proceso tal como se dijo presenta 3 columnas, al final de cada columna siempre tenemos N datos y el proceso se inició con N datos de igual modo lo cual nos permite sobre escribir sobre la matriz que tenía los N datos originales, en otras palabras no es necesario otra matriz para recoger los datos nuevos

SINCE THE p^{th} AND q^{th} VALUES IN THE $(m-1)^{th}$ STAGE ARE USED TO GET THE p^{th} AND q^{th} VALUES IN THE m^{th} STAGE, THE COMPUTATION CAN BE DONE "IN PLACE" - NO EXTRA STORAGE.



Note que los datos pueden acomodarse reescribiendo su valor de forma binaria: el que se encontraba en la muestra 4 en el algoritmo se colocara en la posición 2 por ejemplo.

REVERSE

$x[0]$	000	000	$= x[0]$	
$x[4]$	001	100	$= x[4]$	
x	010	010	$= x[2]$	
	011	110	$= x[6]$	
"BIT- REVERSED ORDER"	100	001	$= x[1]$	
	101	101	$= x[5]$	
	110	011	$= x[3]$	
	111	111	$= x[7]$	

Montaje de la matriz:

THE DFT

$$\begin{bmatrix} X[0] \\ X[1] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \end{bmatrix}$$

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-jk \frac{2\pi}{N} n} \quad \begin{matrix} n=0, 1, \dots, N-1 \\ k=0, 1, \dots, N-1 \end{matrix}$$

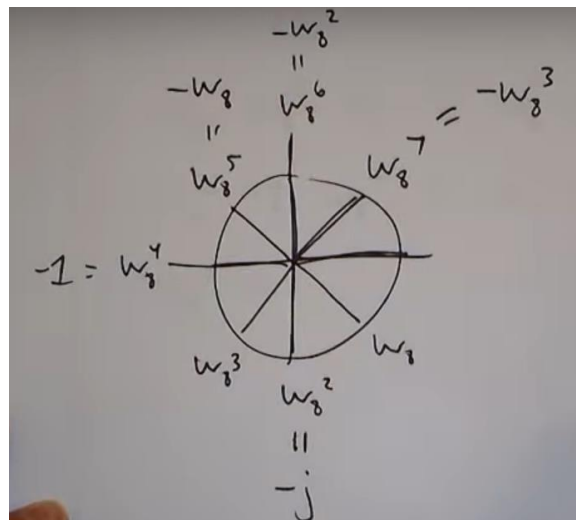
$(a+bj)(c+dj) = (ac-d) + (ad+bc)j$

$$W_N = e^{-j \frac{2\pi}{N}}$$

AN N^{th} ROOT OF 1

$kn \rightarrow N^2$ (COMPLEX) MULTIPLIES
 N (COMPLEX) ADDS

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \\ X[4] \\ X[5] \\ X[6] \\ X[7] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w_8 & w_8^2 & w_8^3 & -1 & -w_8 & -w_8^2 & -w_8^3 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & w_8^3 & j & w_8 & -1 & -w_8^3 & -j & -w_8 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -w_8 & w_8^2 & -w_8^3 & -1 & w_8 & -w_8^2 & w_8^3 \\ 1 & -w_8^2 & -1 & w_8^2 & 1 & -w_8^2 & -1 & w_8^2 \\ 1 & -w_8^3 & -w_8^2 & -w_8 & 1 & w_8^3 & w_8^2 & w_8 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$



Ahora bien, separando la siguiente matriz en columnas, se aprecia un patrón entre las columnas impares y las pares donde las impares se repiten entre la sección superior e inferior y en el caso de las pares es su opuesto o negativo.

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \\ X[4] \\ X[5] \\ X[6] \\ X[7] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W_8 & W_8^2 & W_8^3 & -1 & -W_8 & -W_8^2 & -W_8^3 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & W_8^3 & j & W_8 & -1 & -W_8^3 & -j & -W_8 \\ \hline 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -W_8 & W_8^2 & -W_8^3 & -1 & W_8 & -W_8^2 & W_8^3 \\ 1 & -W_8^2 & -1 & W_8^2 & 1 & -W_8^2 & -1 & W_8^2 \\ 1 & -W_8^3 & -W_8^2 & -W_8 & 1 & W_8^3 & W_8^2 & W_8 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

EVEN COLUMNS =

$$\begin{bmatrix} I_{4 \times 4} \\ I_{4 \times 4} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_8^2 & -1 & -W_8^2 \\ 1 & -1 & 1 & -1 \\ 1 & -W_8^2 & -1 & W_8^2 \end{bmatrix}$$

8x4

$$\begin{bmatrix} I \\ I \\ I \\ I \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4 & W_4^2 & W_4^3 \\ 1 & -1 & 1 & -1 \\ 1 & -W_4 & -1 & W_4 \end{bmatrix}$$

DFT For $N=4$

ODD COLUMNS

$$\begin{bmatrix} I_{4 \times 4} \\ -I_{4 \times 4} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ w_8 & w_8^3 & -w_8 & -w_8^3 \\ w_8^2 & -w_8^2 & w_8^2 & -w_8^2 \\ w_8^3 & w_8 & -w_8^3 & -w_8 \end{bmatrix}$$

$$= \begin{bmatrix} I \\ -I \end{bmatrix} \begin{bmatrix} 1 & w_8 & w_8^2 & w_8^3 \\ & w_8 & w_8^2 & w_8^3 \\ & & w_8 & w_8^2 \\ & & & w_8 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ w_8^2 & -1 & -w_8^2 & \\ -1 & 1 & -1 & \\ -w_8^2 & -1 & w_8^2 & \end{bmatrix}$$

F_4

$$F_8 = \begin{bmatrix} I & I \\ I & -I \end{bmatrix} \begin{bmatrix} I \\ \begin{bmatrix} 1 & w_8 & w_8^2 & w_8^3 \\ & w_8 & w_8^2 & w_8^3 \\ & & w_8 & w_8^2 \\ & & & w_8 \end{bmatrix} \end{bmatrix} \begin{bmatrix} F_4 \\ F_4 \end{bmatrix} \begin{bmatrix} x_{\text{even}} \\ x_{\text{odd}} \end{bmatrix}$$

SUMS AND DIFFERENCES IN "BUTTERFLY"
 "TWIDDLE FACTORS"
 LENGTH 4-DFTS