Introduction to Fast Fourier Transform (FFT) Algorithms

R.C. Maher

EE477 DSP

Spring 2006

Discrete Fourier Transform (DFT)

- The DFT provides uniformly spaced samples of the Discrete-Time Fourier Transform (DTFT)
- DFT definition:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi nk}{N}} \qquad x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] e^{j\frac{2\pi nk}{N}}$$

- Requires N^2 complex multiplies and N(N-1) complex additions

Faster DFT computation?

- Take advantage of the symmetry and periodicity of the complex exponential (with $W_N=e^{-j2\pi/N}$):
  - symmetry: $W_N^{k[N-n]} = W_N^{-kn} = (W_N^{kn})^*$
  - periodicity: $W_N^{kn} = W_N^{k[n+N]} = W_N^{[k+N]n}$

- Note that two length N/2 DFTs take less computation than one length N DFT: $2(N/2)^2<N^2$

- Algorithms that exploit computational savings are collectively called *Fast Fourier Transforms*

Decimation-in-Time Algorithm

- Consider expressing DFT with even and odd input samples:

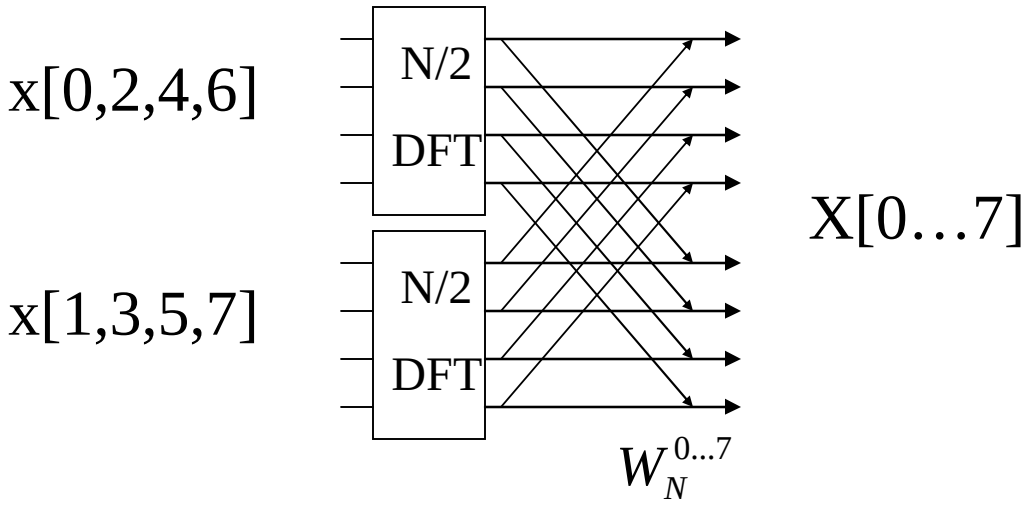
$$
\begin{aligned}
X[k] &= \sum_{n=0}^{N-1} x[n] W_N^{nk} \\
&= \sum_{n\ even} x[n] W_N^{nk} + \sum_{n\ odd} x[n] W_N^{nk} \\
&= \sum_{r=0}^{\frac{N}{2}-1} x[2r] (W_N^2)^{rk} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] (W_N^2)^{rk} \\
&= \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_{N/2}^{rk} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_{N/2}^{rk}
\end{aligned}
$$

DIT Algorithm (cont.)

- Result is the sum of two N/2 length DFTs

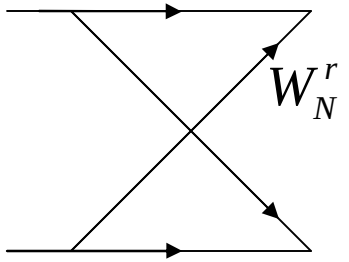
$$X[k] = \underbrace{G[k]}_{\text{N/2 DFT of even samples}} + W_N^k \cdot \underbrace{H[k]}_{\text{N/2 DFT of odd samples}}$$

- Then repeat decomposition of N/2 to N/4 DFTs, etc.

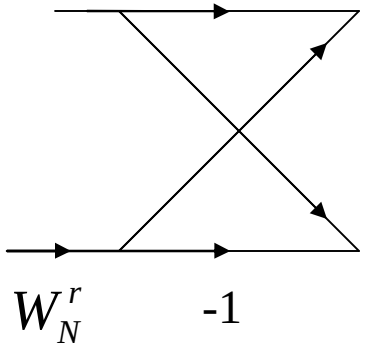


Detail of “Butterfly”

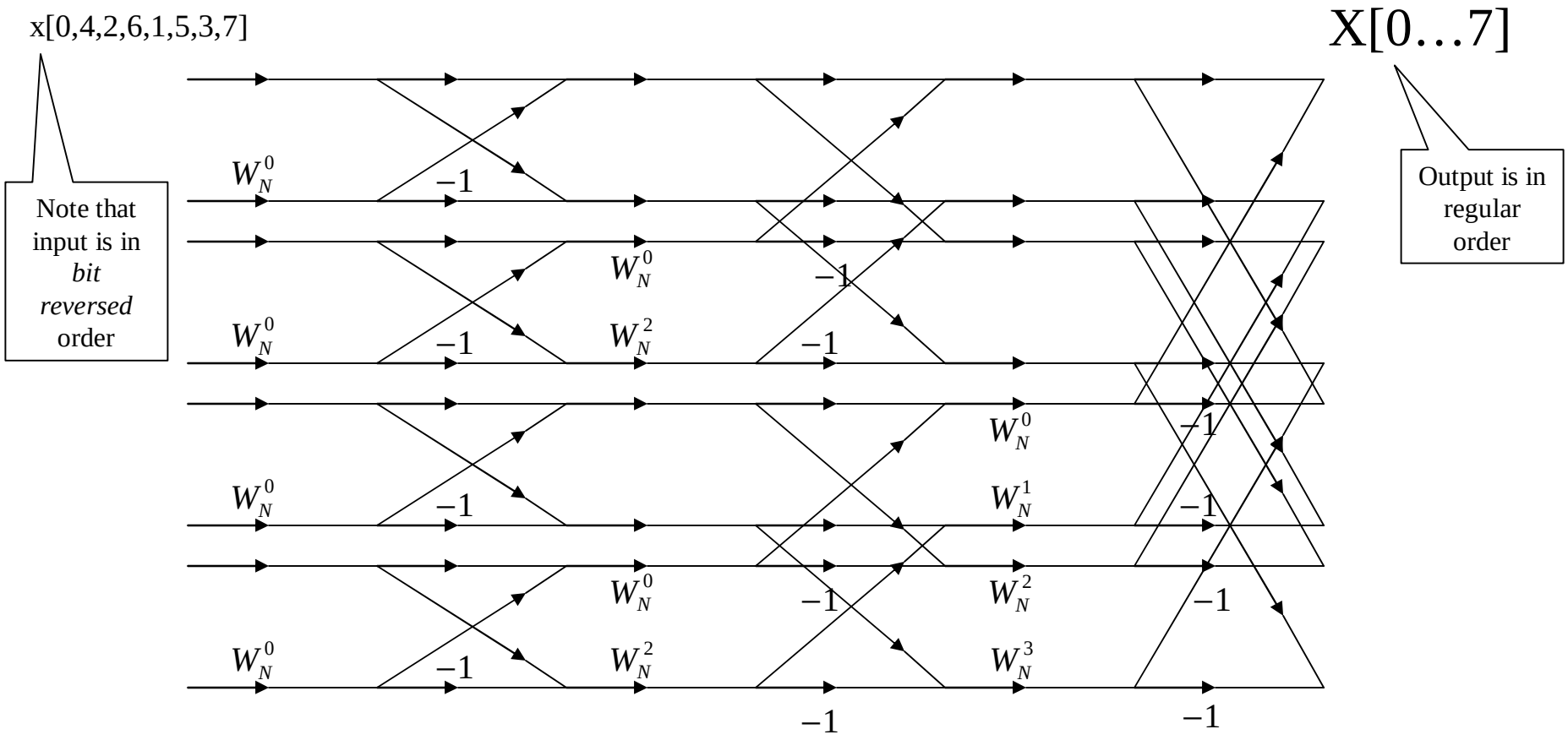
- Cross feed of G[k] and H[k] in flow diagram is called a “butterfly”, due to shape



or simplify:



8-point DFT Diagram



$$W_8^0 = 1; \quad W_8^2 = e^{-j\pi/2} = -j; \quad W_8^4 = e^{-j\pi} = -1$$

Computation on DSP

- Input and Output data
  - Real data in X memory
  - Imaginary data in Y memory
- Coefficients (“twiddle” factors)
  - `cos(real)` values in X memory
  - `sin(imag)` values in Y memory
- Inverse computed with exponent sign change and 1/N scaling