# Department of Electronics & Communication Engg.

**Course : Digital Signal Processing -15EC52.** **Sem.: 5th (2018-19)**

## Course Coordinator:

## Prof. S. S. Ittannavar

# Sampling the Fourier Transform

- Consider an a periodic sequence with a Fourier transform

$$x[n] \xleftrightarrow{\text{DTFT}} X\left(e^{j\omega}\right)$$

- Assume that a sequence is obtained by sampling the DTFT

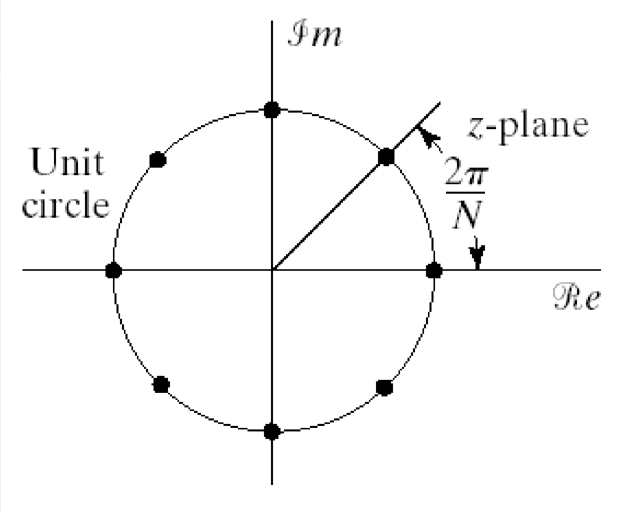
$$\tilde{X}[k] = X\left(e^{j\omega}\right)\Big|_{\omega=(2\pi/N)k} = X\left(e^{j(2\pi/N)k}\right)$$

- Since the DTFT is periodic resulting sequence is also periodic
- We can also write it in terms of the z-transform

$$\tilde{X}[k] = X(z)\Big|_{z=e^{(2\pi/N)k}} = X\left(e^{j(2\pi/N)k}\right)$$

- The sampling points are shown in figure
- $\tilde{X}[k]$ could be the DFS of a sequence
- Write the corresponding sequence

$$\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1}\tilde{X}[k]e^{j(2\pi/N)kn}$$



# Sampling the Fourier Transform Cont'd

- The only assumption made on the sequence is that DTFT exist

$$X(e^{j\omega}) = \sum_{m=-\infty}^{\infty} x[m]e^{-j\omega m} \qquad \tilde{X}[k] = X\left(e^{j(2\pi/N)k}\right) \qquad \tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1}\tilde{X}[k]e^{j(2\pi/N)kn}$$

- Combine equation to get

$$\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1}\left[\sum_{m=-\infty}^{\infty} x[m]e^{-j(2\pi/N)km}\right]e^{j(2\pi/N)kn}$$

$$= \sum_{m=-\infty}^{\infty} x[m]\left[\frac{1}{N}\sum_{k=0}^{N-1} e^{j(2\pi/N)k(n-m)}\right] = \sum_{m=-\infty}^{\infty} x[m]\tilde{p}[n-m]$$

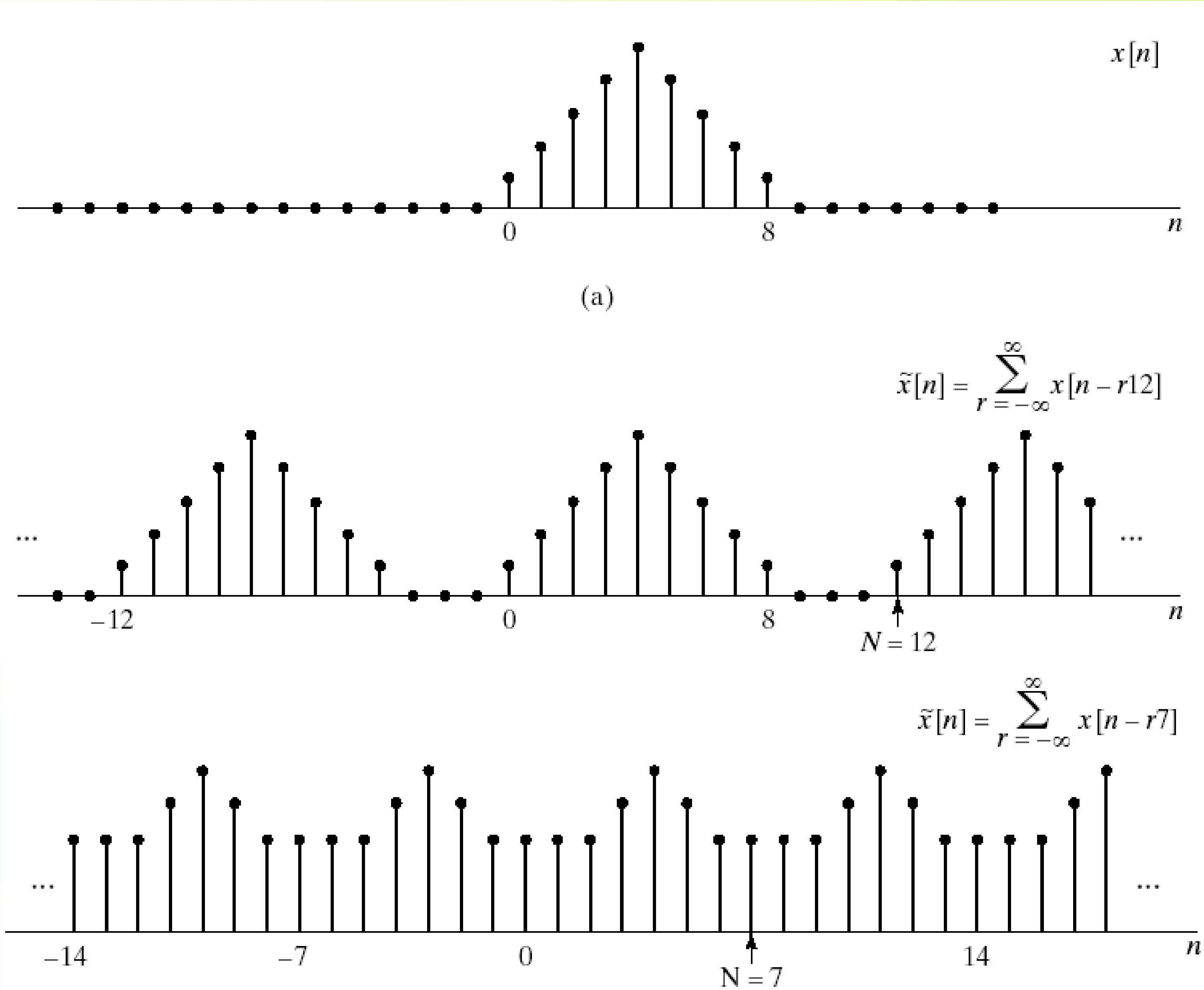
- Term in the parenthesis is

$$\tilde{p}[n-m] = \frac{1}{N}\sum_{k=0}^{N-1} e^{j(2\pi/N)k(n-m)} = \sum_{r=-\infty}^{\infty}\delta[n-m-rN]$$

- So we get

$$\tilde{x}[n] = x[n] * \sum_{r=-\infty}^{\infty}\delta[n-rN] = \sum_{r=-\infty}^{\infty} x[n-rN]$$

# Sampling the Fourier Transform Cont'd



## Sampling the Fourier Transform Cont'd

- Samples of the DTFT of an aperiodic sequence
  - can be thought of as DFS coefficients
  - of a periodic sequence
  - obtained through summing periodic replicas of original sequence
- If the original sequence
  - is of finite length
  - and we take sufficient number of samples of its DTFT
  - the original sequence can be recovered by

$$x[n] = \begin{cases} \tilde{x}[n] & 0 \le n \le N-1 \\ 0 & \text{else} \end{cases}$$

- It is not necessary to know the DTFT at all frequencies
  - To recover the discrete-time sequence in time domain
- Discrete Fourier Transform
  - Representing a finite length sequence by samples of DTFT

## The Discrete Fourier Transform

- Consider a finite length sequence x[n] of length N

$$x[n] = 0 \quad \text{outside of} \quad 0 \le n \le N-1$$

- For given length-N sequence associate a periodic sequence

$$\tilde{x}[n] = \sum_{r=-\infty}^{\infty} x[n - rN]$$

- The DFS coefficients of the periodic sequence are samples of the DTFT of x[n]
- Since x[n] is of length N there is no overlap between terms of x[n-rN] and we can write the periodic sequence as

$$\tilde{x}[n] = x[(n \bmod N)] = x[((n))_N]$$

- To maintain duality between time and frequency
  - We choose one period of $\tilde{X}[k]$ as the Fourier transform of x[n]

$$X[k] = \begin{cases} \tilde{X}[k] & 0 \le k \le N-1 \\ 0 & \text{else} \end{cases} \qquad \tilde{X}[k] = X[(k \bmod N)] = X[((k))_N]$$

## The Discrete Fourier Transform Cont'd

- The DFS pair

$$\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j(2\pi/N)kn} \qquad \tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k] e^{j(2\pi/N)kn}$$

- The equations involve only on period so we can write

$$X[k] = \begin{cases} \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j(2\pi/N)kn} & 0 \le k \le N-1 \\ 0 & \text{else} \end{cases}$$

$$x[n] = \begin{cases} \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k] e^{j(2\pi/N)kn} & 0 \le k \le N-1 \\ 0 & \text{else} \end{cases}$$

- The Discrete Fourier Transform

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

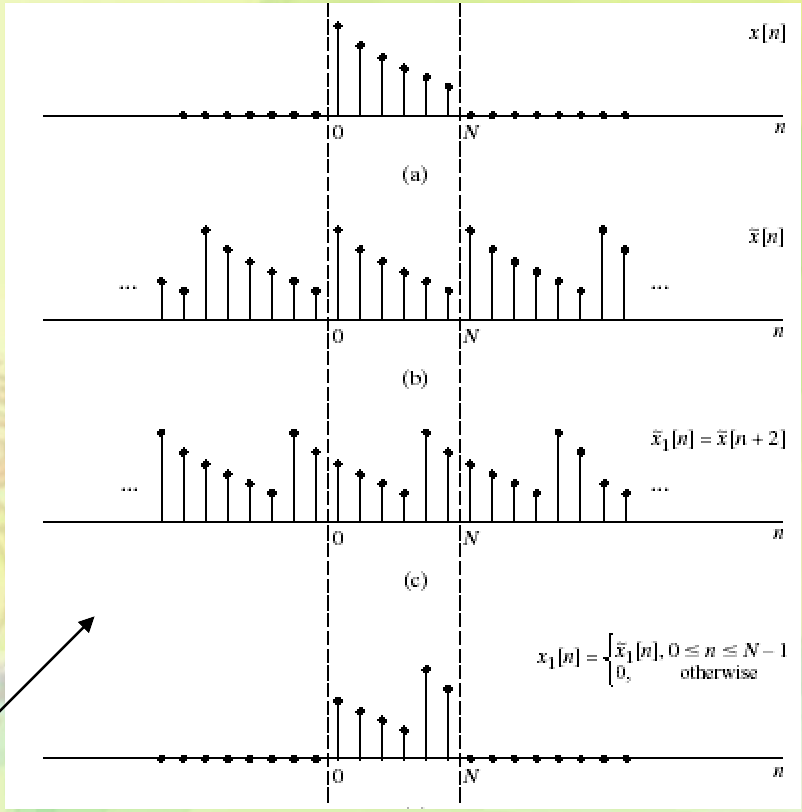
- The DFT pair can also be written as

$$X[k] \xleftrightarrow{\text{DFT}} x[n]$$

# Properties of DFT

- Linearity

$$x_1[n] \xleftrightarrow{DFT} X_1[k]$$
$$x_2[n] \xleftrightarrow{DFT} X_2[k]$$
$$ax_1[n] + bx_2[n] \xleftrightarrow{DFT} aX_1[k] + bX_2[k]$$




- Duality

$$x[n] \xleftrightarrow{DFT} X[k]$$
$$X[n] \xleftrightarrow{DFT} Nx[((-k))_N]$$

- Circular Shift of a Sequence

$$x[n] \xleftrightarrow{DFT} X[k]$$
$$x[((n-m))_N]\ 0 \le n \le N-1 \xleftrightarrow{DFT} X[k]e^{-j(2\pi k/N)m}$$

# Symmetry Properties

| | | |
|---|---|---|
| 5. | $x[((n-m))_N]$ | $W_N^{km}X[k]$ |
| 6. | $W_N^{-\ell n}x[n]$ | $X[((k-\ell))_N]$ |
| 7. | $\sum_{m=0}^{N-1} x_1(m)x_2[((n-m))_N]$ | $X_1[k]X_2[k]$ |
| 8. | $x_1[n]x_2[n]$ | $\frac{1}{N}\sum_{\ell=0}^{N-1} X_1(\ell)X_2[((k-\ell))_N]$ |
| 9. | $x^*[n]$ | $X^*[((-k))_N]$ |
| 10. | $x^*[((-n))_N]$ | $X^*[k]$ |
| 11. | $\mathcal{R}e\{x[n]\}$ | $X_{\mathrm{ep}}[k] = \frac{1}{2}\{X[((k))_N] + X^*[((-k))_N]\}$ |
| 12. | $j\mathcal{I}m\{x[n]\}$ | $X_{\mathrm{op}}[k] = \frac{1}{2}\{X[((k))_N] - X^*[((-k))_N]\}$ |
| 13. | $x_{\mathrm{ep}}[n] = \frac{1}{2}\{x[n] + x^*[((-n))_N]\}$ | $\mathcal{R}e\{X[k]\}$ |
| 14. | $x_{\mathrm{op}}[n] = \frac{1}{2}\{x[n] - x^*[((-n))_N]\}$ | $j\mathcal{I}m\{X[k]\}$ |

Properties 15–17 apply only when $x[n]$ is real.

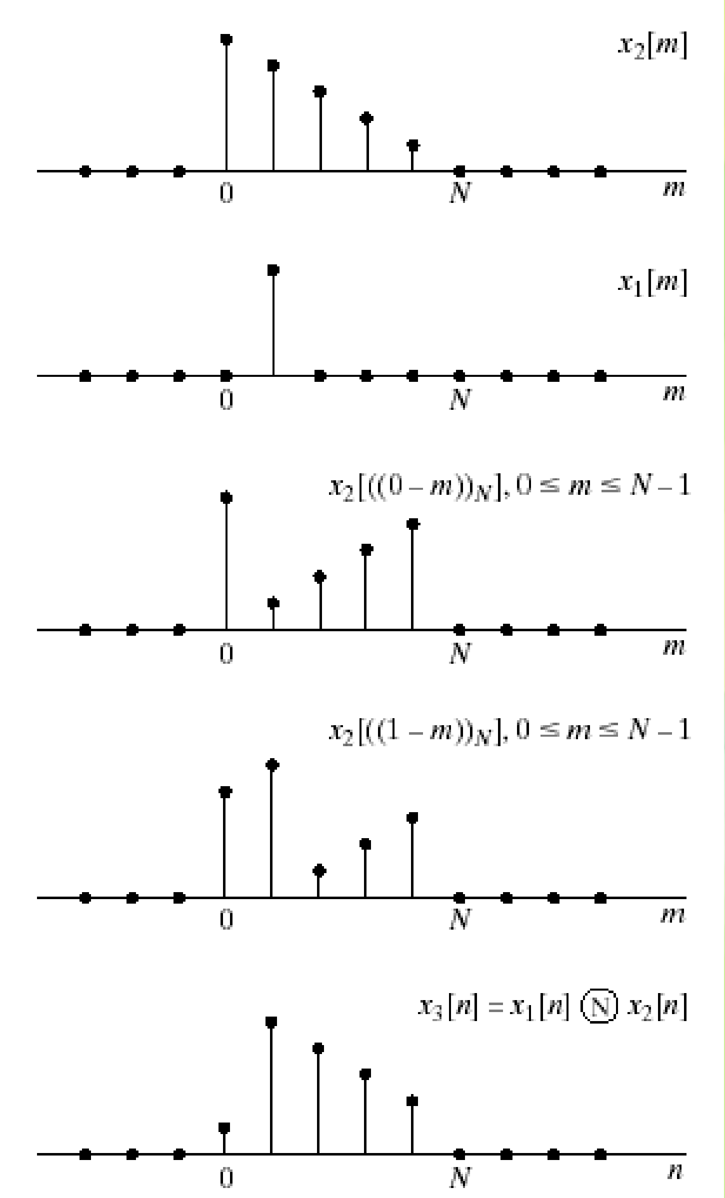
| | | |
|---|---|---|
| 15. | Symmetry properties | $\begin{cases} X[k] = X^*[((-k))_N] \\ \mathcal{R}e\{X[k]\} = \mathcal{R}e\{X[((-k))_N]\} \\ \mathcal{I}m\{X[k]\} = -\mathcal{I}m\{X[((-k))_N]\} \\ \lvert X[k]\rvert = \lvert X[((-k))_N]\rvert \\ \sphericalangle\{X[k]\} = -\sphericalangle\{X[((-k))_N]\} \end{cases}$ |

# Circular Convolution

- Circular convolution of of two finite length sequences

$$x_3[n] = \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

$$x_3[n] = \sum_{m=0}^{N-1} x_2[m] x_1[((n-m))_N]$$



# Fast Fourier Transforms

# Discrete Fourier Transform

- The DFT pair was given as

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

- Baseline for computational complexity:
  - Each DFT coefficient requires
    - N complex multiplications
    - N-1 complex additions
  - All N DFT coefficients require
    - $N^2$ complex multiplications
    - N(N-1) complex additions
- Complexity in terms of real operations
    - $4N^2$ real multiplications
    - 2N(N-1) real additions
- Most fast methods are based on symmetry properties
  - Conjugate symmetry $e^{-j(2\pi/N)k(N-n)} = e^{-j(2\pi/N)kN} e^{-j(2\pi/N)k(-n)} = e^{j(2\pi/N)kn}$
  - Periodicity in n and k $e^{-j(2\pi/N)kn} = e^{-j(2\pi/N)k(n+N)} = e^{j(2\pi/N)(k+N)n}$

# The Goertzel Algorithm

- Makes use of the periodicity

$$e^{j(2\pi/N)Nk} = e^{j2\pi k} = 1$$

- Multiply DFT equation with this factor

$$X[k] = e^{j(2\pi/N)kN} \sum_{r=0}^{N-1} x[r] e^{-j(2\pi/N)rn} = \sum_{r=0}^{N-1} x[r] e^{j(2\pi/N)r(N-n)}$$

- Define

$$y_k[n] = \sum_{r=-\infty}^{\infty} x[r] e^{j(2\pi/N)k(n-r)} u[n-r]$$

- With this definition and using x[n]=0 for n<0 and n>N-1

$$X[k] = y_k[n]\Big|_{n=N}$$

- X[k] can be viewed as the output of a filter to the input x[n]
  - Impulse response of filter:

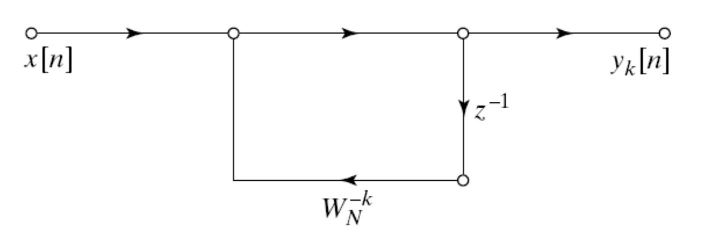
$$e^{j(2\pi/N)kn} u[n]$$

  - X[k] is the output of the filter at time n=N

# The Goertzel Filter

- Goertzel Filter

$$H_k(z) = \frac{1}{1 - e^{j\frac{2\pi}{N}k} z^{-1}}$$



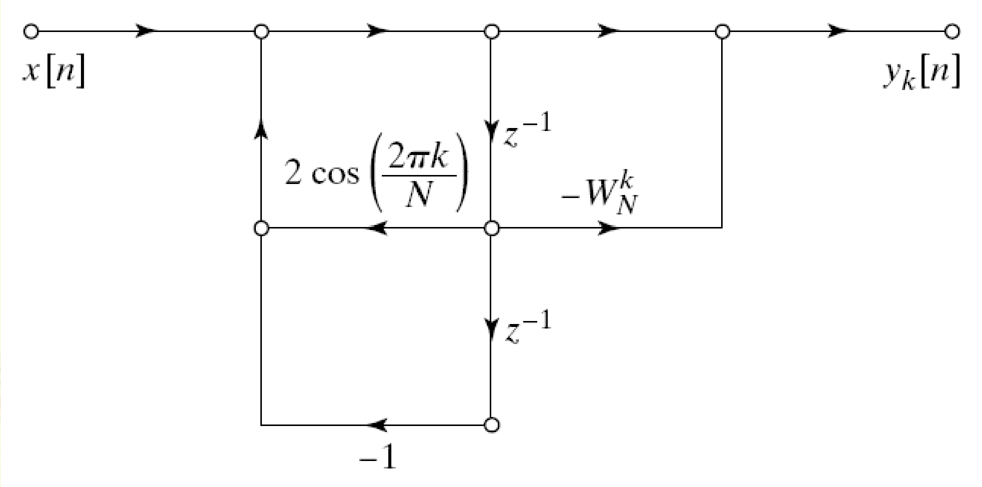

- Computational complexity
  - 4N real multiplications
  - 2N real additions
  - Slightly less efficient than the direct method

- Multiply both numerator and denominator

$$H_k(z) = \frac{1 - e^{-j\frac{2\pi}{N}k} z^{-1}}{\left(1 - e^{j\frac{2\pi}{N}k} z^{-1}\right)\left(1 - e^{-j\frac{2\pi}{N}k} z^{-1}\right)} = \frac{1 - e^{-j\frac{2\pi}{N}k} z^{-1}}{1 - 2\cos\frac{2\pi k}{N} z^{-1} + z^{-2}}$$

# Second Order Goertzel Filter

- Second order Goertzel Filter




$$H_k(z) = \frac{1 - e^{-j\frac{2\pi}{N}k} z^{-1}}{1 - 2\cos\frac{2\pi k}{N} z^{-1} + z^{-2}}$$

- Complexity for one DFT coefficient
  - Poles: 2N real multiplications and 4N real additions
  - Zeros: Need to be implement only once
    - 4 real multiplications and 4 real additions
- Complexity for all DFT coefficients
  - Each pole is used for two DFT coefficients
    - Approximately $N^2$ real multiplications and $2N^2$ real additions
- Do not need to evaluate all N DFT coefficients
  - Goertzel Algorithm is more efficient than FFT if
    - less than M DFT coefficients are needed
    - $M < \log_2 N$

# Decimation-In-Time FFT Algorithms

- Makes use of both symmetry and periodicity
- Consider special case of N an integer power of 2
- Separate x[n] into two sequence of length N/2
  - Even indexed samples in the first sequence
  - Odd indexed samples in the other sequence

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j(2\pi/N)kn} = \sum_{n\ \text{even}}^{N-1} x[n]e^{-j(2\pi/N)kn} + \sum_{n\ \text{odd}}^{N-1} x[n]e^{-j(2\pi/N)kn}$$

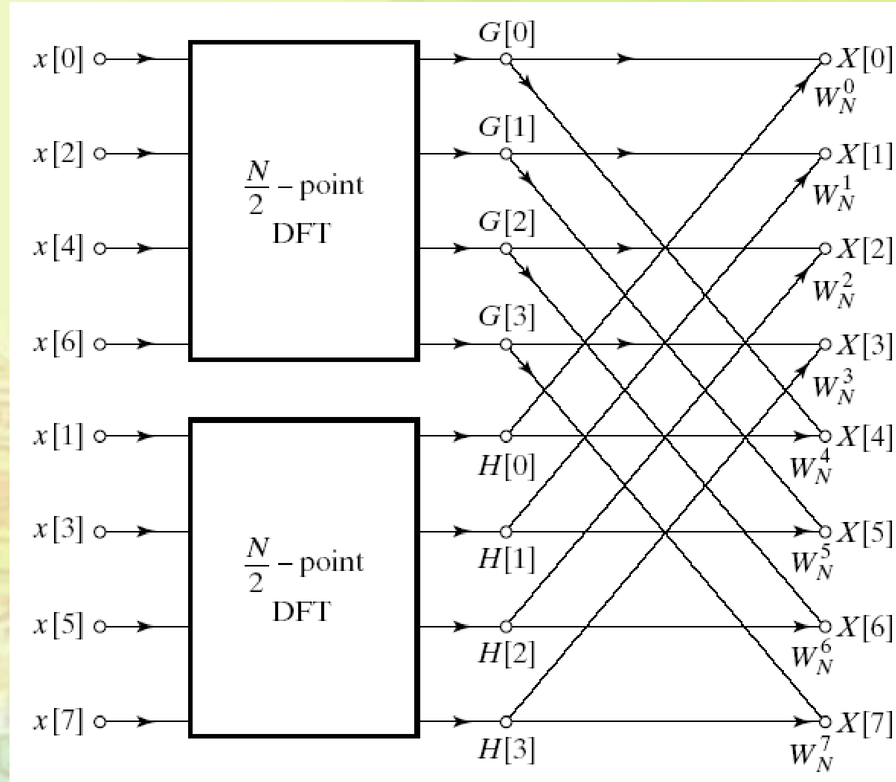
- Substitute variables n=2r for n even and n=2r+1 for odd

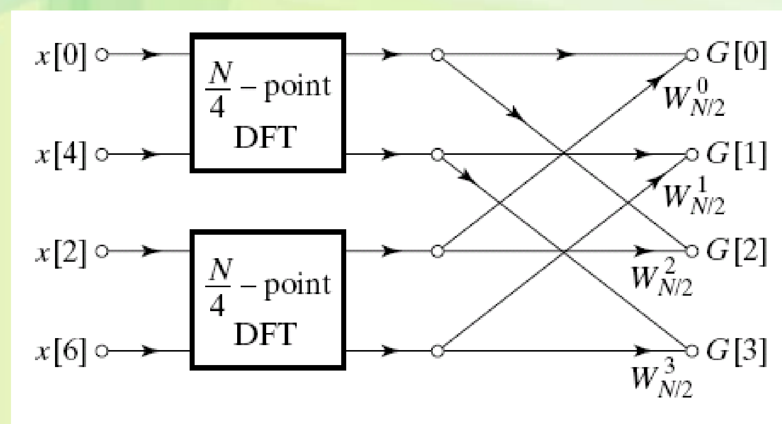
$$\begin{aligned} X[k] &= \sum_{r=0}^{N/2-1} x[2r]W_N^{2rk} + \sum_{r=0}^{N/2-1} x[2r+1]W_N^{(2r+1)k} \\ &= \sum_{r=0}^{N/2-1} x[2r]W_{N/2}^{rk} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1]W_{N/2}^{rk} \\ &= G[k] + W_N^k H[k] \end{aligned}$$

- G[k] and H[k] are the N/2-point DFT's of each subsequence

# Decimation In Time

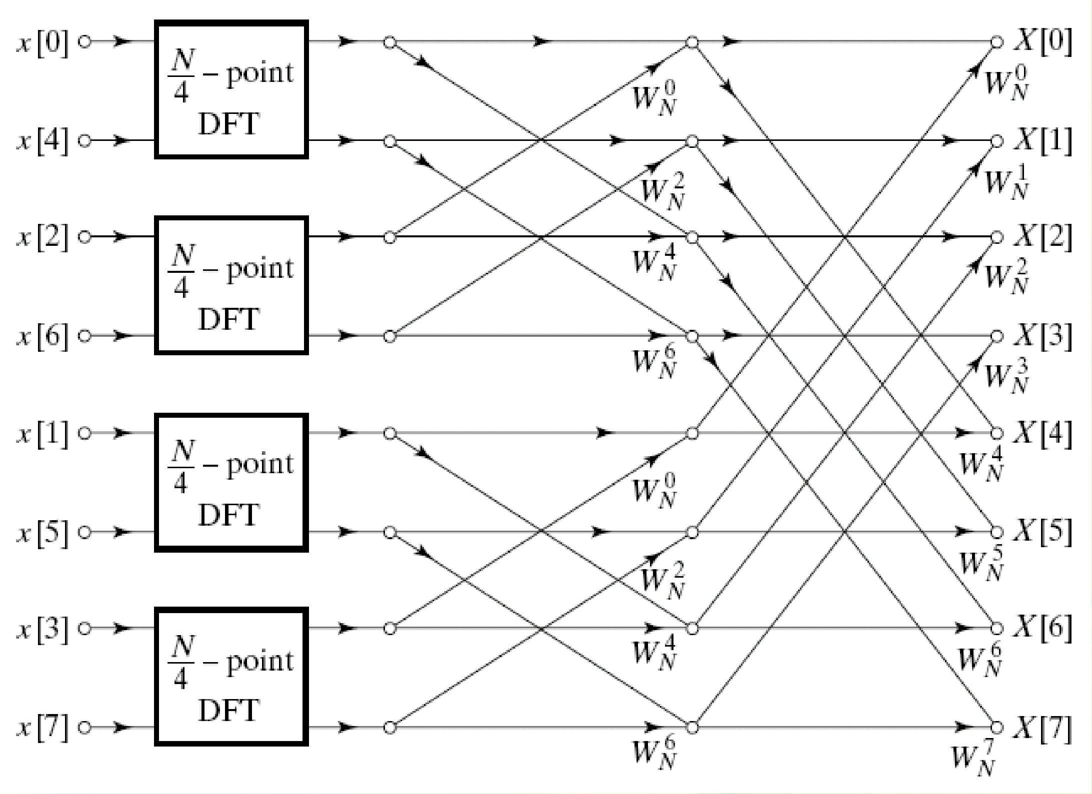
- 8-point DFT example using decimation-in-time
- Two N/2-point DFTs
  - $2(N/2)^2$ complex multiplications
  - $2(N/2)^2$ complex additions
- Combining the DFT outputs
  - N complex multiplications
  - N complex additions
- Total complexity
  - $N^2/2+N$ complex multiplications
  - $N^2/2+N$ complex additions
  - More efficient than direct DFT
- Repeat same process
  - Divide N/2-point DFTs into
  - Two N/4-point DFTs
  - Combine outputs



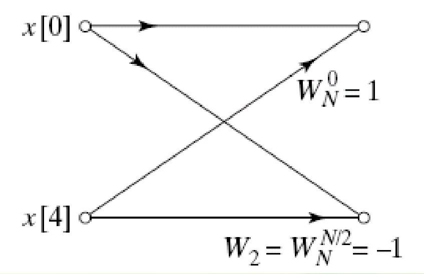



# Decimation In Time Cont'd

- After two steps of decimation in time

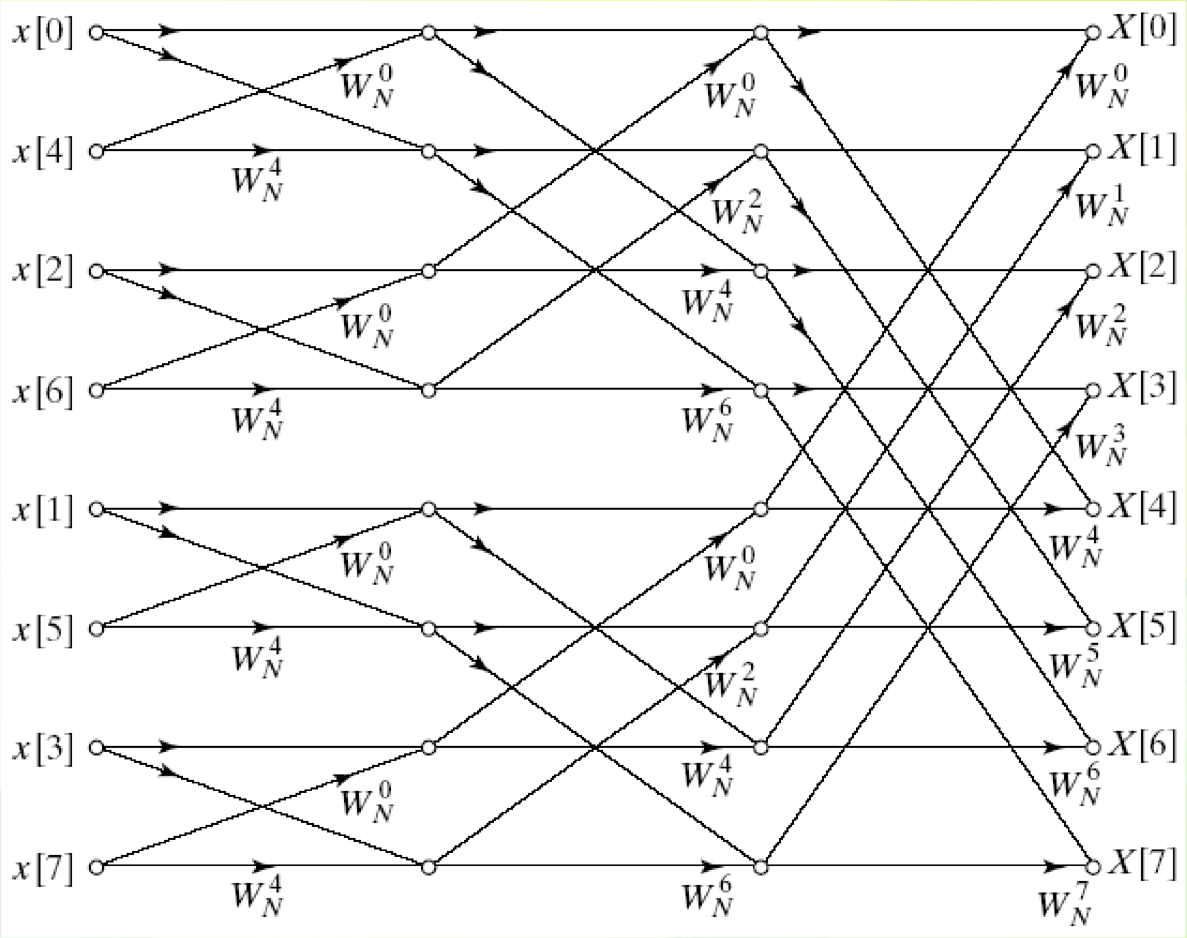



- Repeat until we're left with two-point DFT's



# Decimation-In-Time FFT Algorithm

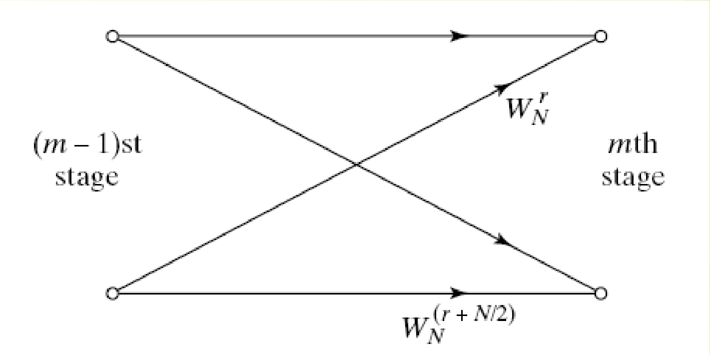
- Final flow graph for 8-point decimation in time



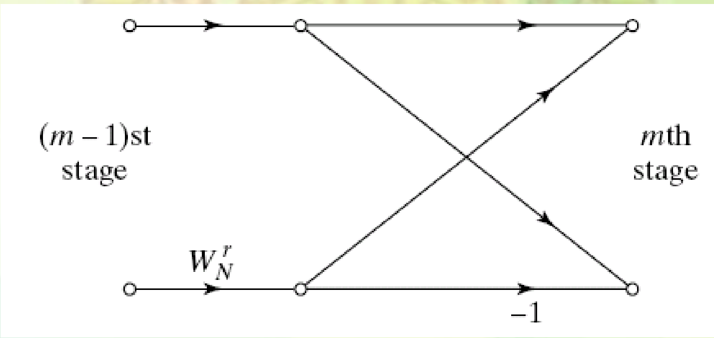

- Complexity:
  - $N\log_2 N$ complex multiplications and additions

# Butterfly Computation

- Flow graph constitutes of butterflies




- We can implement each butterfly with one multiplication




- Final complexity for decimation-in-time FFT
  - $(N/2)\log_2 N$ complex multiplications and additions

# In-Place Computation

- Decimation-in-time flow graphs require two sets of registers
  - Input and output for each stage
- Note the arrangement of the input indices
  - Bit reversed indexing

$$X_0[0] = x[0] \leftrightarrow X_0[000] = x[000]$$
$$X_0[1] = x[4] \leftrightarrow X_0[001] = x[100]$$
$$X_0[2] = x[2] \leftrightarrow X_0[010] = x[010]$$
$$X_0[3] = x[6] \leftrightarrow X_0[011] = x[110]$$
$$X_0[4] = x[1] \leftrightarrow X_0[100] = x[001]$$
$$X_0[5] = x[5] \leftrightarrow X_0[101] = x[101]$$
$$X_0[6] = x[3] \leftrightarrow X_0[110] = x[011]$$
$$X_0[7] = x[7] \leftrightarrow X_0[111] = x[111]$$

# Decimation-In-Frequency FFT Algorithm

- The DFT equation

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

- Split the DFT equation into even and odd frequency indexes

$$X[2r] = \sum_{n=0}^{N-1} x[n] W_N^{n2r} = \sum_{n=0}^{N/2-1} x[n] W_N^{n2r} + \sum_{n=N/2}^{N-1} x[n] W_N^{n2r}$$

- Substitute variables to get

$$X[2r] = \sum_{n=0}^{N/2-1} x[n] W_N^{n2r} + \sum_{n=0}^{N/2-1} x[n+N/2] W_N^{(n+N/2)2r} = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+N/2]\right) W_{N/2}^{nr}$$

- Similarly for odd-numbered frequencies

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left(x[n] - x[n+N/2]\right) W_{N/2}^{n(2r+1)}$$

Decimation-In-Frequency FFT Algorithm

- Final flow graph for 8-point decimation in frequency