THE FAST FOURIER TRANSFORM: LECTURE 2

Paolo Favaro

Chapter 10 of Textbook

(original material from John Thompson)

Summary of The FFT

- The FFT is a computationally efficient implementation of the DFT
- The idea is to subdivide an N-point DFT into lots of butterfly operations
- Note that the FFT and DFT yield the SAME transform output

Subdividing the DFT

- Can write the DFT equation as:

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

$$= \sum_{\substack{n=0 \\ n \text{ even}}}^{N-1} x(n) W_N^{nk} + \sum_{\substack{n=0 \\ n \text{ odd}}}^{N-1} x(n) W_N^{nk}$$

- Now we make two substitutions:

$$m = n/2 \,(\text{even } n), \;\; m = (n-1)/2 \,(\text{odd } n)$$

- So the DFT equation becomes:

$$X(k) = \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{(2m+1)k}$$

$$= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + W_N^k \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{2mk}$$

$$= X_1(k) + W_N^k X_2(k)$$

- $X_1(k)$ is the $(N/2)$ point DFT of even $x(n)$
- $X_2(k)$ is the $(N/2)$ point DFT of odd $x(n)$

DFT Subdivision

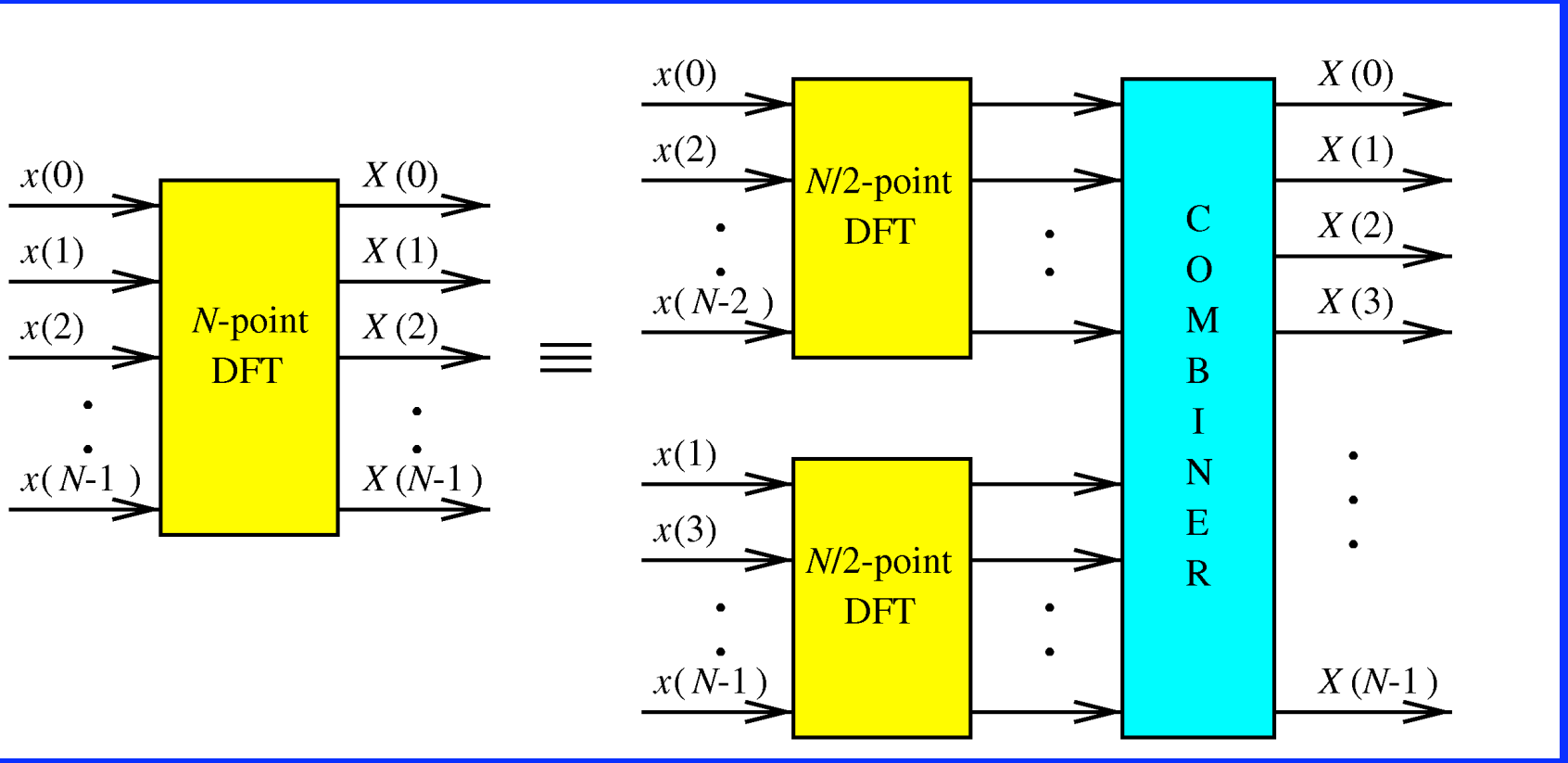
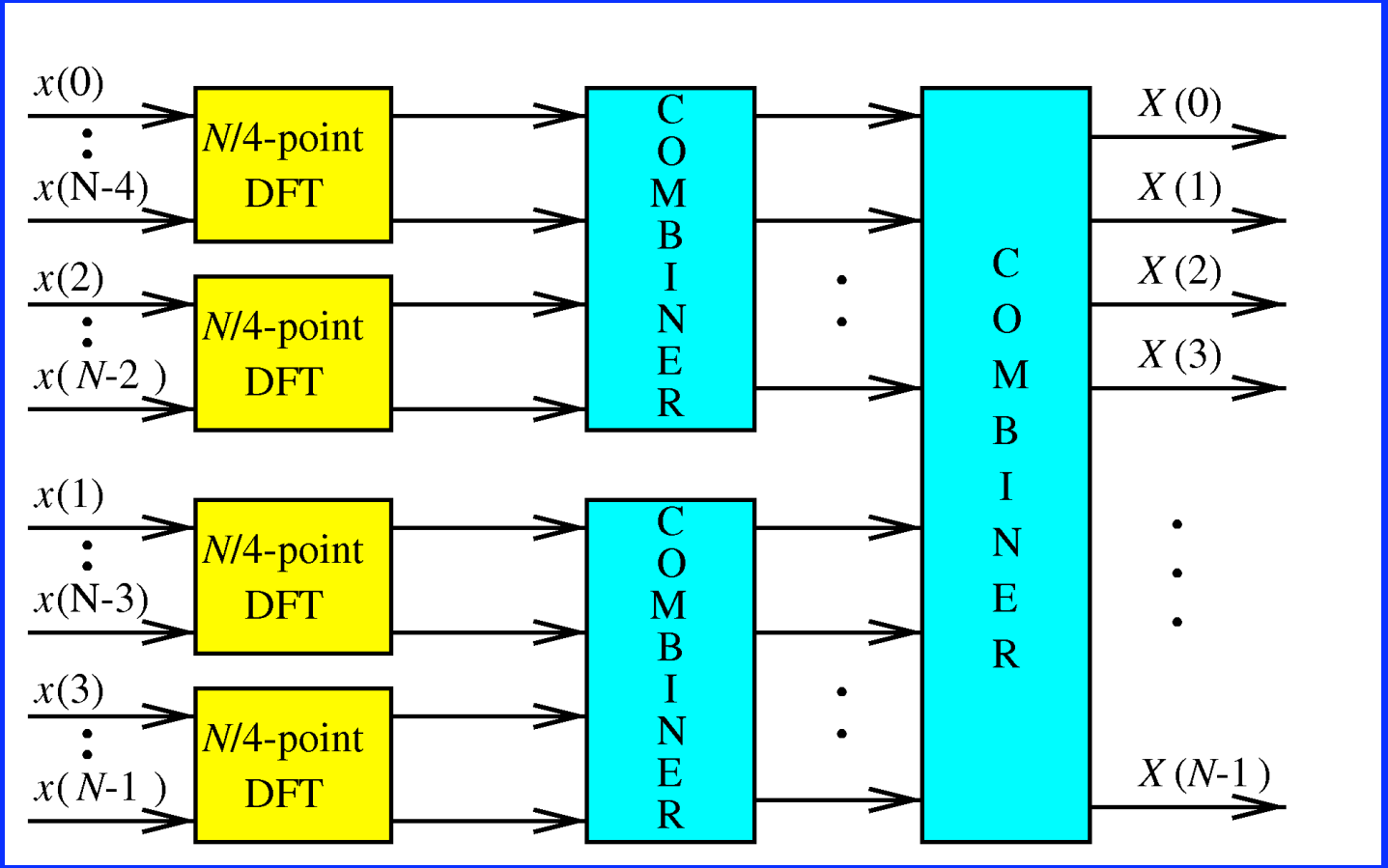


Fig 10.1

Subdividing the DFT Again



DFT Simplification

- If N is power of 2, can simplify DFT to:
  - N/2 separate 2-point DFTs
  - Followed by $[\log_2(N)-1]$ combining stages
- So for the N=8 example, we have four 2-point DFTs and two combining stages

The 2-point DFT

- The 2-point DFT may be written as:

$$\begin{bmatrix} X(0) \\ X(1) \end{bmatrix} = \begin{bmatrix} W_2^0 & W_2^0 \\ W_2^0 & W_2^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \end{bmatrix}$$

- *X(0)* & *X(1)* are sum and difference of *x (0)* and *x(1)*
- No multiplies needed for 2-point DFT!

- First stage of simplified DFT requires only additions/subtractions

- The 2-point DFT is also called the **Butterfly Operation**:

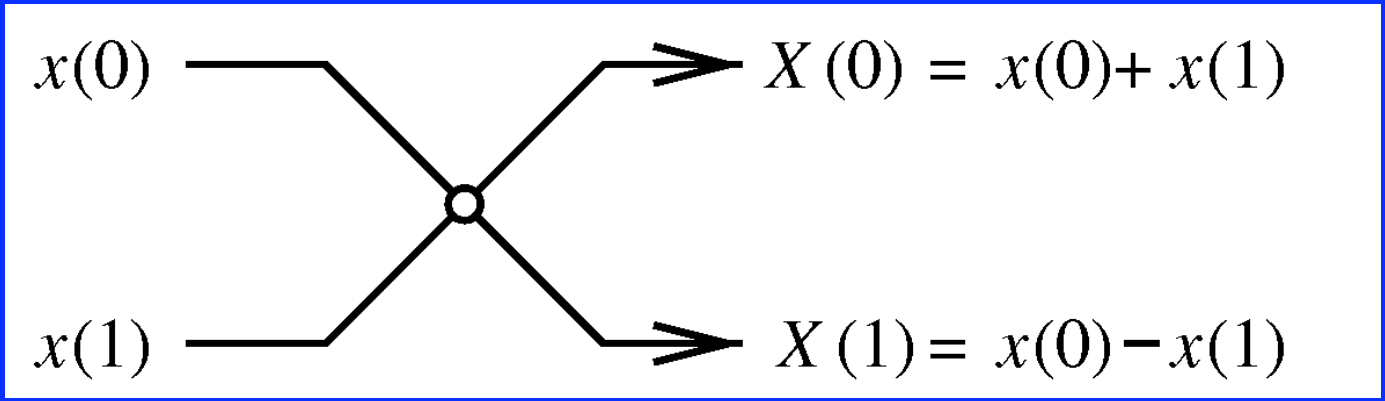


Fig 10.4

Last Combining Stage

- The first simplified DFT was

$$X(k) = X_1(k) + W_N^k X_2(k)$$

- $X_1(k)$ and $X_2(k)$ have period $N/2$, so:

$$X_1(k) = X_1(k + (N/2)) \text{ and } X_2(k) = X_2(k + (N/2))$$

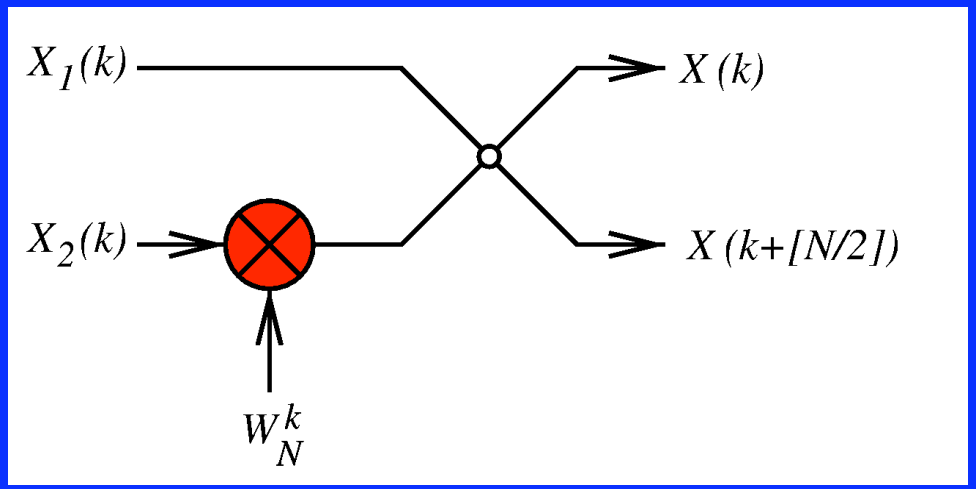
- Also can write $W_N^k = -W_N^{k+(N/2)}$

- So we can now write:

$$X(k) = X_1(k) + W_N^k X_2(k)$$

$$X(k + [N/2]) = X_1(k) - W_N^k X_2(k)$$

- Output is 2-point DFT of $X_1(k)$ and of $W_N^k X_2(k)$:



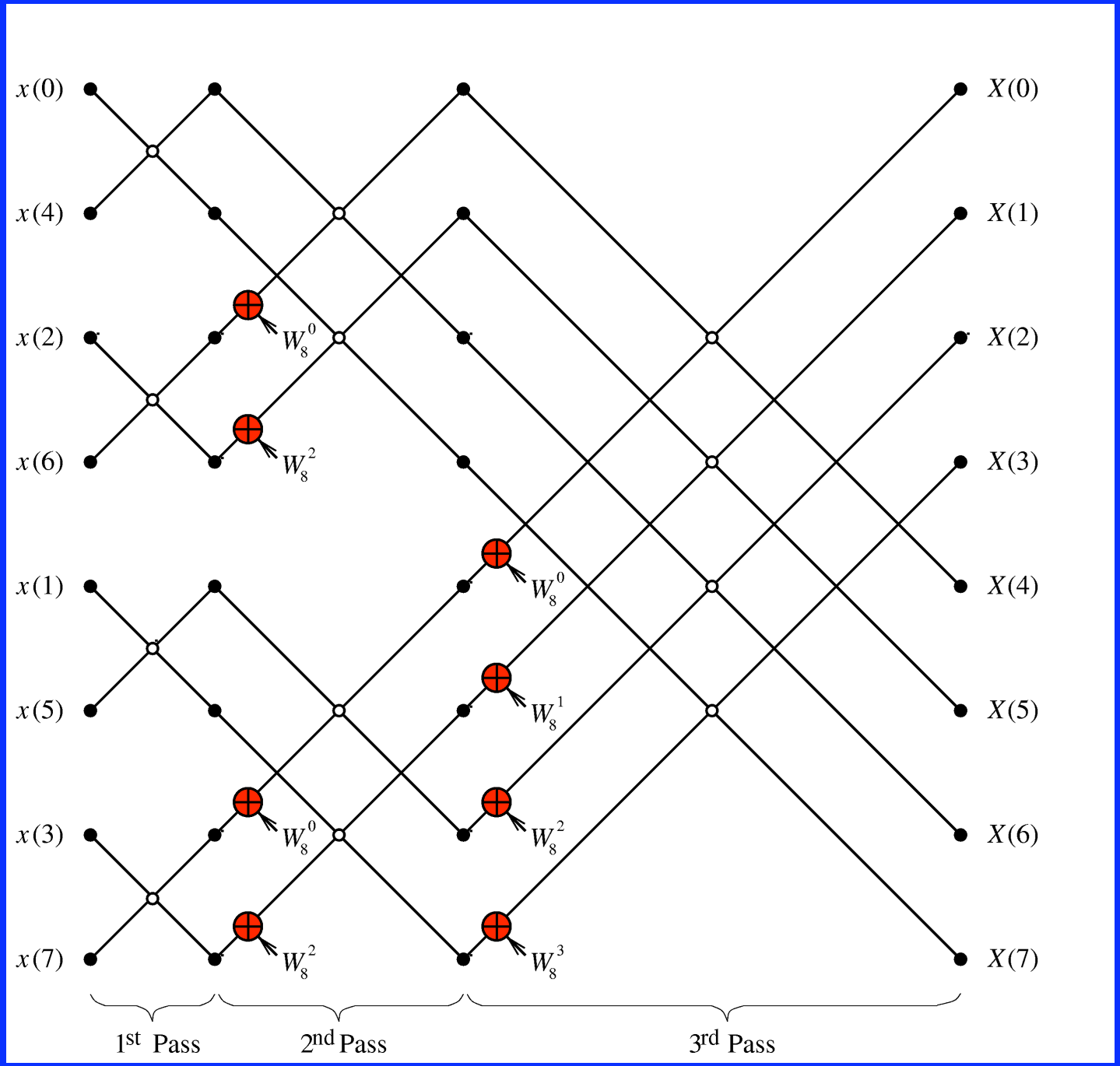
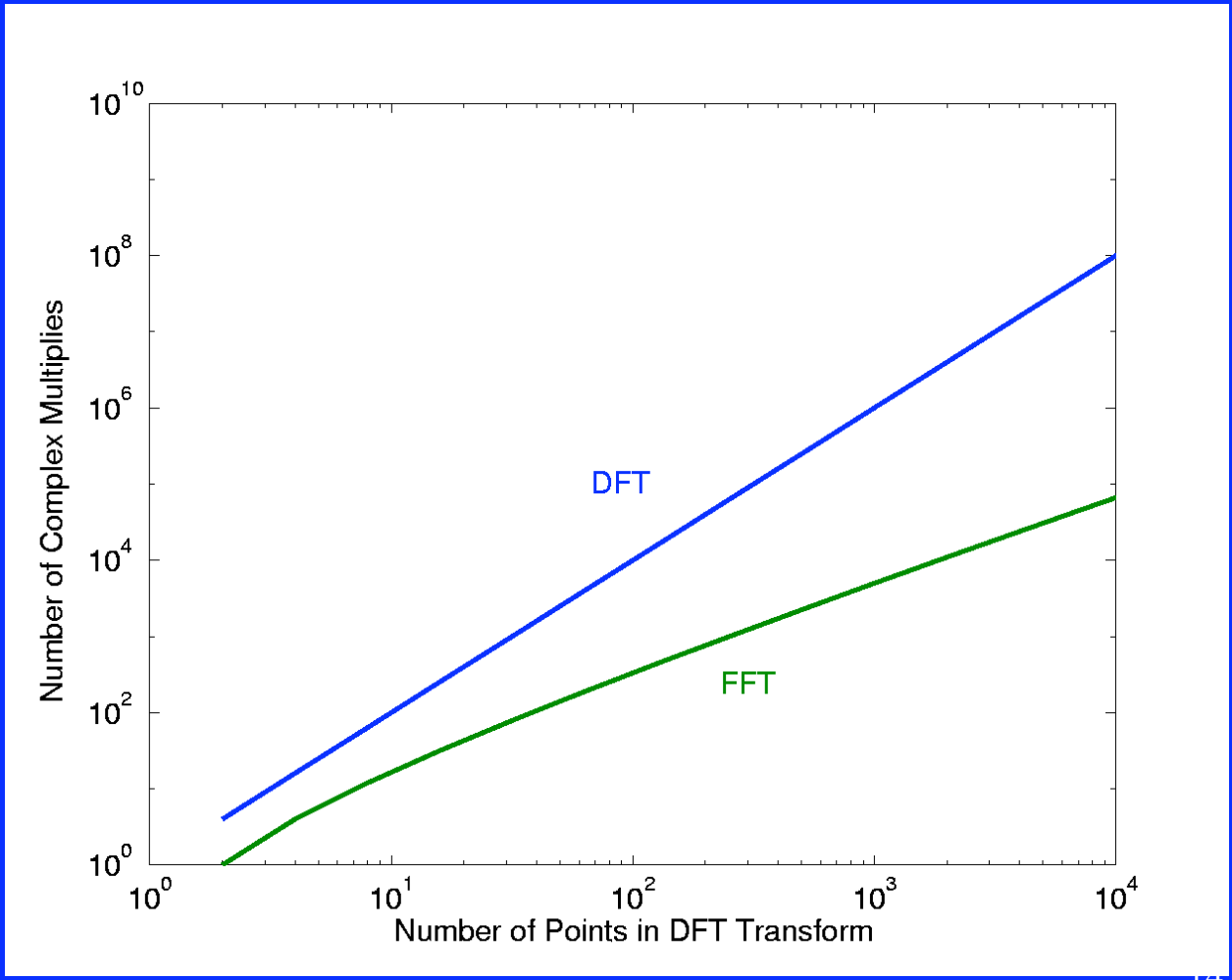


Fig 10.5: 8-pt FFT

FFT Complexity

- An *N*-point DFT can be simplified into $log_2(N)$ stages or ‘passes’
- Each stage requires *N/2* butterfly operations
- Thus require $(N/2)\ log_2(N)$ butterfly operations in total
- Each butterfly uses two complex adds plus one complex multiply

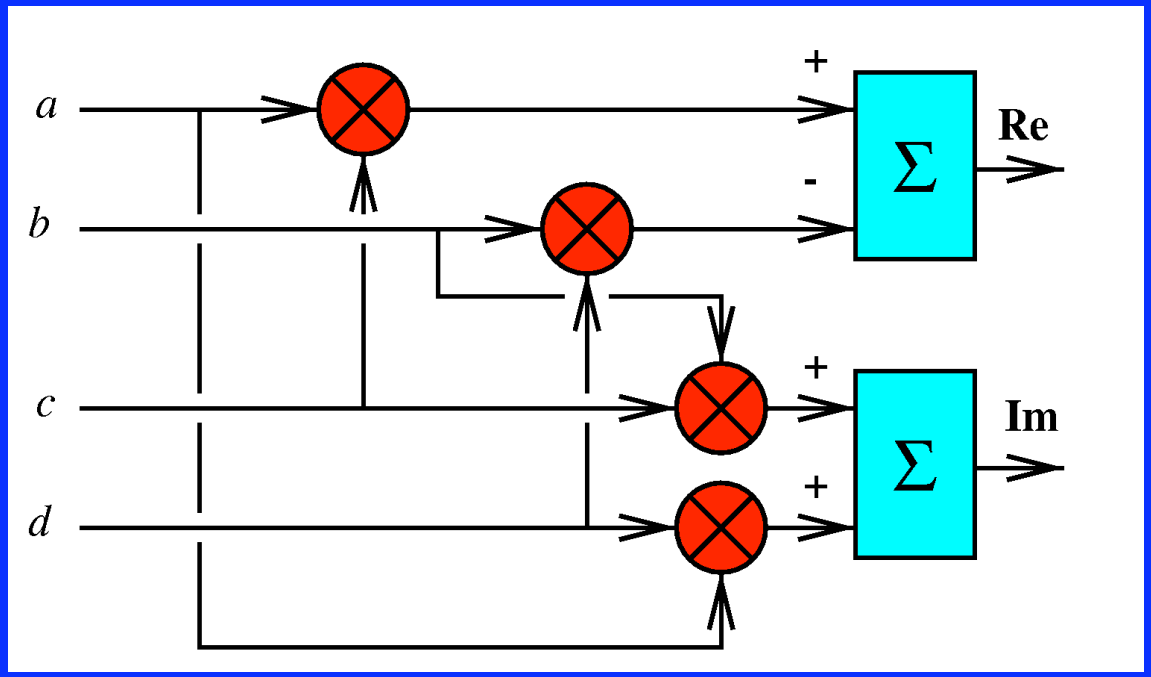
Complex Multiplies in DFT/FFT



Complex Multiplies

- Complex multiplication can be written:

$$(a + jb)(c + jd) = (ac - bd) + j(ad + bc)$$



- One complex multiply requires:
  - 4 real multiplications
  - 2 real additions
- One complex addition requires 2 real additions
- Hence the main FFT computation is $(2N)\ log_2(N)$ real multiplications
- The other key implementation issue is retrieval/calculation of twiddle factors

Radix 4 FFTs

- The four point DFT/Butterfly is:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

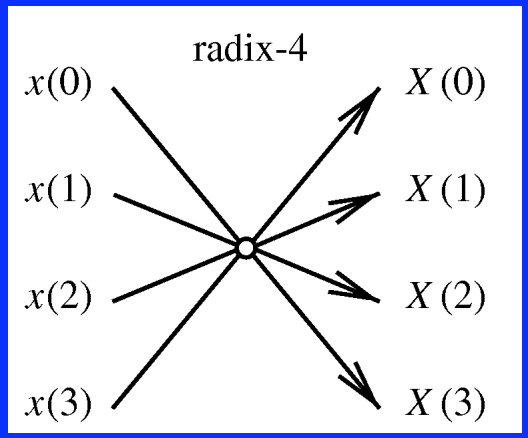


Fig 10. 7

- Multiplying by $\pm j$ swaps Real/Imag parts, with one sign change
- Radix 4 halves No of passes needed

Decimation in Frequency(DIF) FFT

- Can also split DFT outputs *X(k)* into ‘odd’ and ‘even’ groups
- The resulting FFT has multipliers on one Butterfly output (not input)
- Note that DIT or DIF algorithms may have bit reversed inputs or outputs

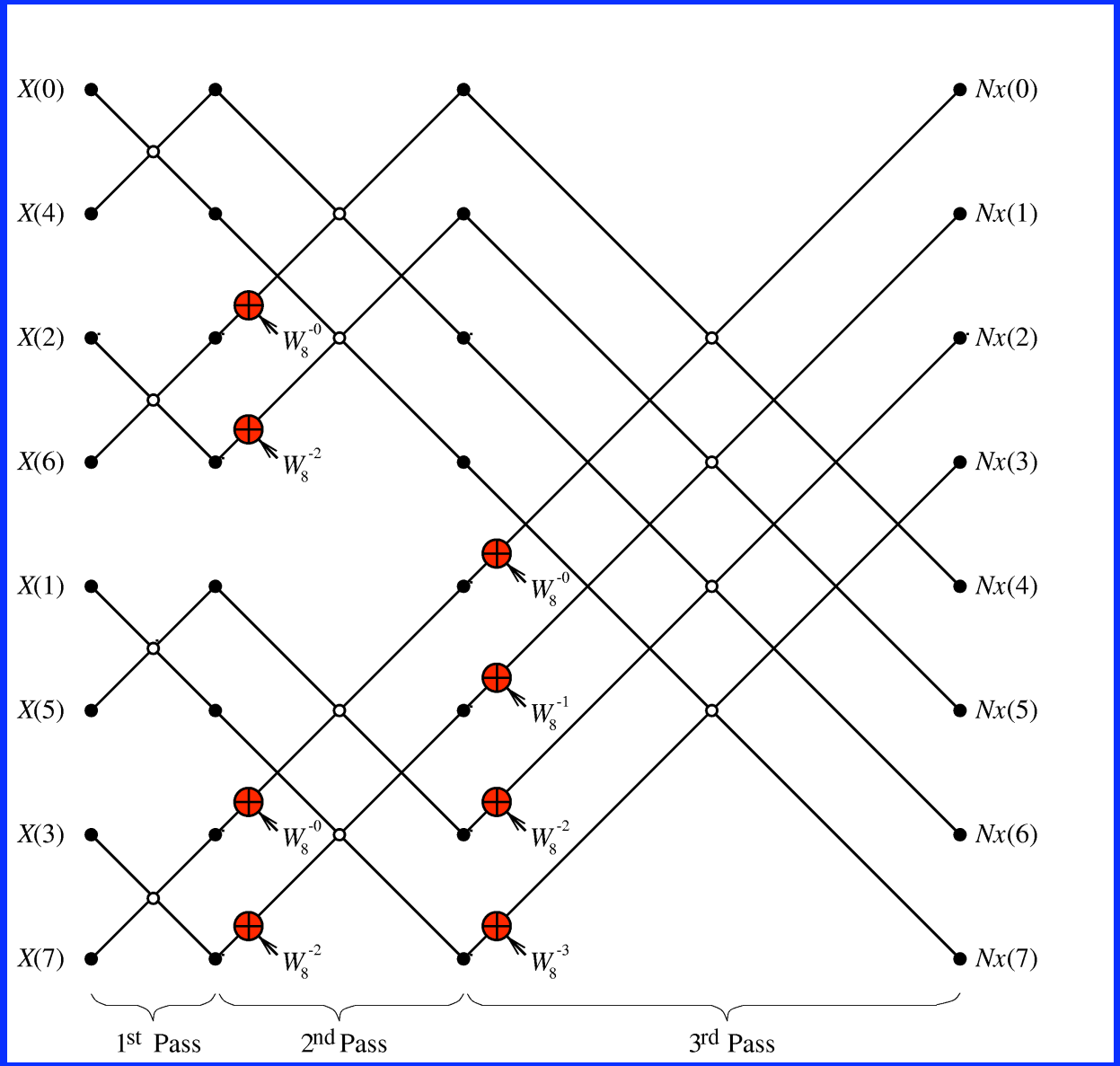
Inverse FFTs

- The inverse DFT is:

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-nk}, \qquad n = 0 \ldots N-1$$

- FFT can also perform inverse DFT:
  - Use complex conjugate twiddle values
  - Scale outputs by *1/N*

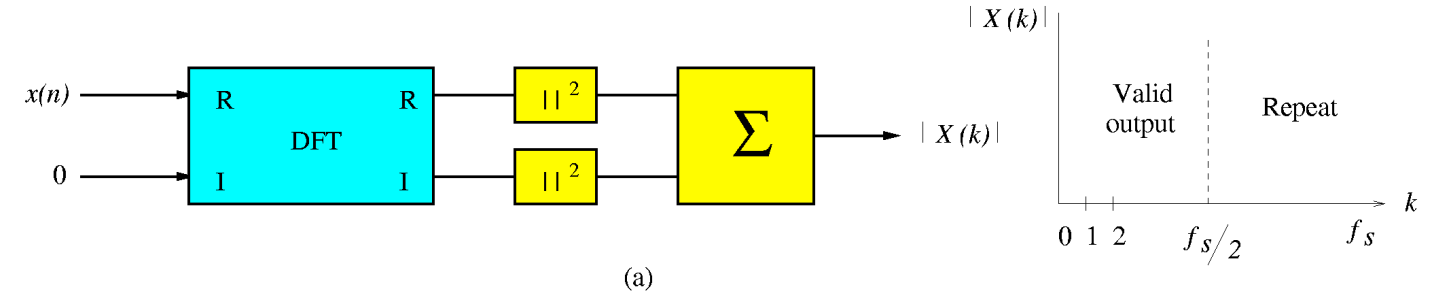
8-point Decimation in frequency Inverse FFT (**Note** similarity to DIT FFT)



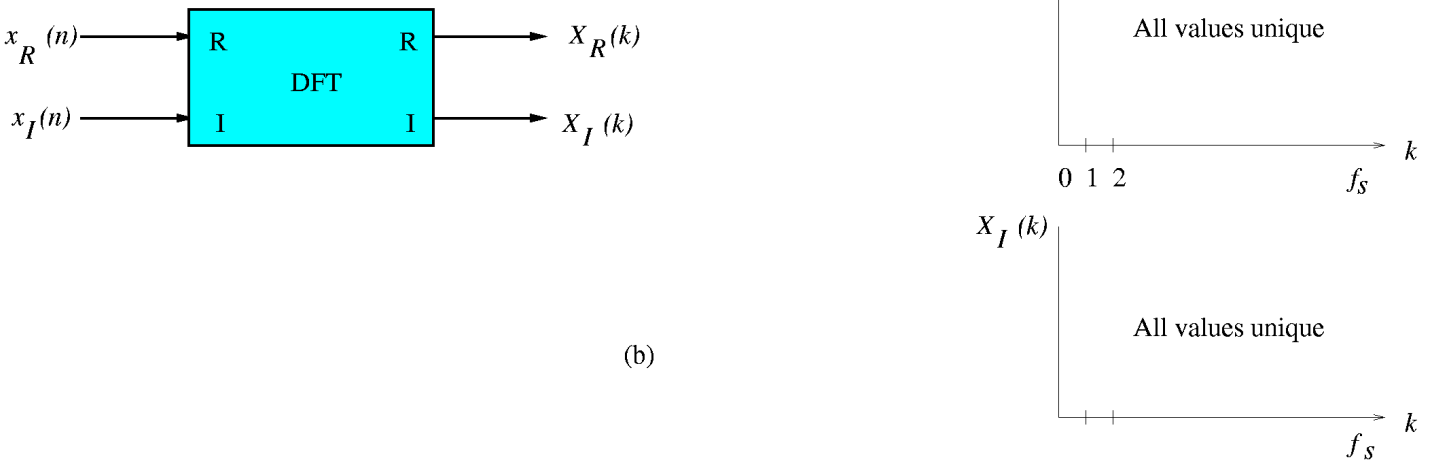
Real Valued Data

- One *N*-point DFT can yield the DFTs for two real signals *a(n)* and *b(n)*:
  - Form composite signal: $x(n)=a(n)+jb(n)$
  - Calculate DFT *X(k)* for *x(n)*
- DFT of *a(n)* is $A(k) = [X(k)+X^*(N-k)]/2$
- DFT of *b(n)* is $B(k) = [X(k)-X^*(N-k)]/2j$

Real/Complex DFTs



(a)



(b)

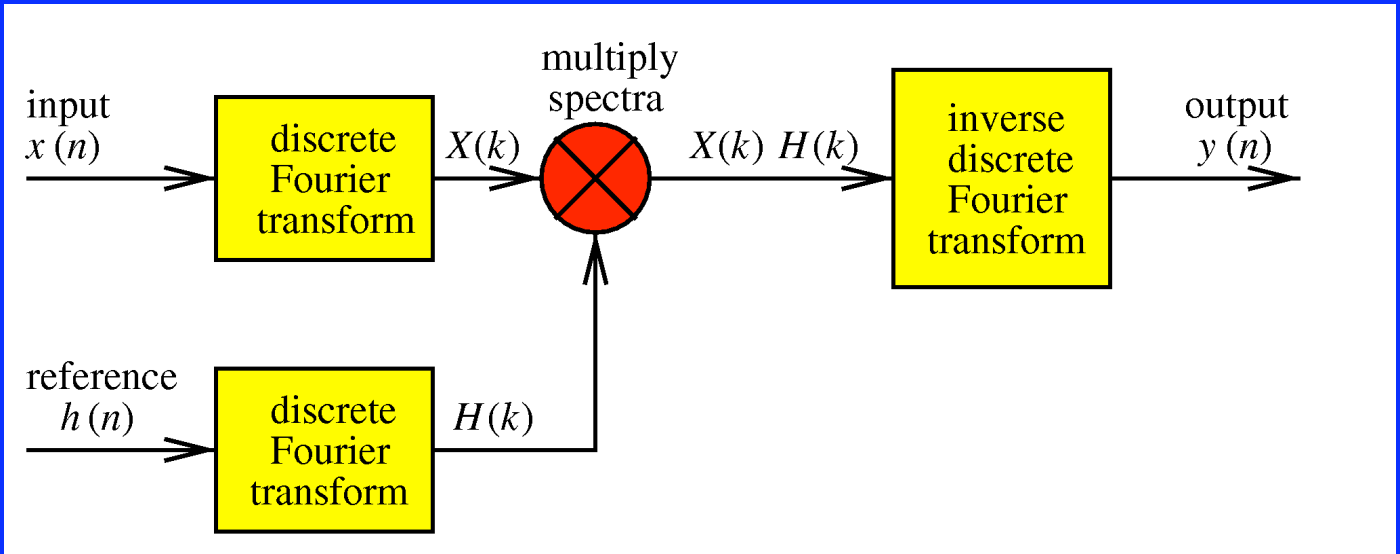
Fig 10. 8

FFT Sizes

- Radix-2 FFTs require an input size *N* which is a power of 2
- If data length *M* is not a power of 2, use zero padding to increase length to *N*
- Remember that with zero padding, resolution depends on *M* **NOT** *N*

FFT Applications

- Fast convolution:



- Order $2N\log_2(2N)$ complexity compared to N^2 for N-tap FIR filter