

数字信号处理

周治国

2019.4

第四章 快速傅里叶变换

§ 4-4 按频率抽取(DIF)的FFT算法 (Sande-Tukey算法)

一、算法原理

$$\forall x(n), \quad 0 \leq n \leq N-1, \quad N = 2^v$$



$$\begin{aligned} X(k) &= DFT[x(n)] \\ &= \sum_{n=0}^{N-1} x(n)W_N^{kn}, \quad 0 \leq k \leq N-1 \end{aligned}$$

将 $x(n)$, $0 \leq n \leq N-1$ 按顺序分为前后两半

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)W_N^{kn} + \sum_{n=0}^{\frac{N}{2}-1} x\left(\frac{N}{2} + n\right)W_N^{k\left(n + \frac{N}{2}\right)} \quad (4-23)$$

$k=0, 1, \dots, N-1$

注意: $W_N = e^{-j\frac{2\pi}{N}} \neq W_{N/2}$

\therefore 两个 Σ 和式并不是 $N/2$ -DFT

不过，由于 $W_N^{k\frac{N}{2}} = e^{-j\frac{2\pi}{N}k\frac{N}{2}} = e^{-j\pi k} = (-1)^k$

所以，式(4-23)可改写为

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)W_N^{kn} + \sum_{n=0}^{\frac{N}{2}-1} x\left(\frac{N}{2} + n\right)W_N^{k\left(\frac{N}{2} + n\right)} \quad (4-23)$$

$$= \sum_{n=0}^{\frac{N}{2}-1} [x(n) + (-1)^k x\left(\frac{N}{2} + n\right)]W_N^{kn}, \quad k = 0, 1, \dots, N-1 \quad (4-24)$$

$$\therefore (-1)^k \begin{cases} 1 & k \text{ 为偶数} \\ -1 & k \text{ 为奇数} \end{cases}$$

∴可按 k 的奇偶取值将 $X(k)$ 分为两部分：

$$\begin{aligned}
X(2r) &= \sum_{n=0}^{\frac{N}{2}-1} [x(n) + x(\frac{N}{2} + n)] W_N^{2rn} \\
&= \sum_{n=0}^{\frac{N}{2}-1} [x(n) + x(\frac{N}{2} + n)] W_{N/2}^{rn} \quad r = 0, 1, \dots, \frac{N}{2} - 1 \quad (4-25)
\end{aligned}$$

$$\begin{aligned}
X(2r+1) &= \sum_{n=0}^{\frac{N}{2}-1} [x(n) - x(\frac{N}{2} + n)] W_N^{(2r+1)n} \\
&= \sum_{n=0}^{\frac{N}{2}-1} [x(n) - x(\frac{N}{2} + n)] W_N^n \cdot W_N^{2rn} \\
&= \sum_{n=0}^{\frac{N}{2}-1} [x(n) - x(\frac{N}{2} + n)] W_N^n \cdot W_{N/2}^{rn} \quad r = 0, 1, \dots, \frac{N}{2} - 1 \quad (4-26)
\end{aligned}$$

显然，若令

$$x_1(n) \stackrel{\Delta}{=} [x(n) + x(n + \frac{N}{2})], \quad n = 0, 1, \dots, \frac{N}{2} - 1$$

$$X_1(k) \stackrel{\Delta}{=} X(2k), \quad k = 0, 1, \dots, \frac{N}{2} - 1$$

$$x_2(n) \stackrel{\Delta}{=} [x(n) - x(n + \frac{N}{2})]W_N^n, \quad n = 0, 1, \dots, \frac{N}{2} - 1$$

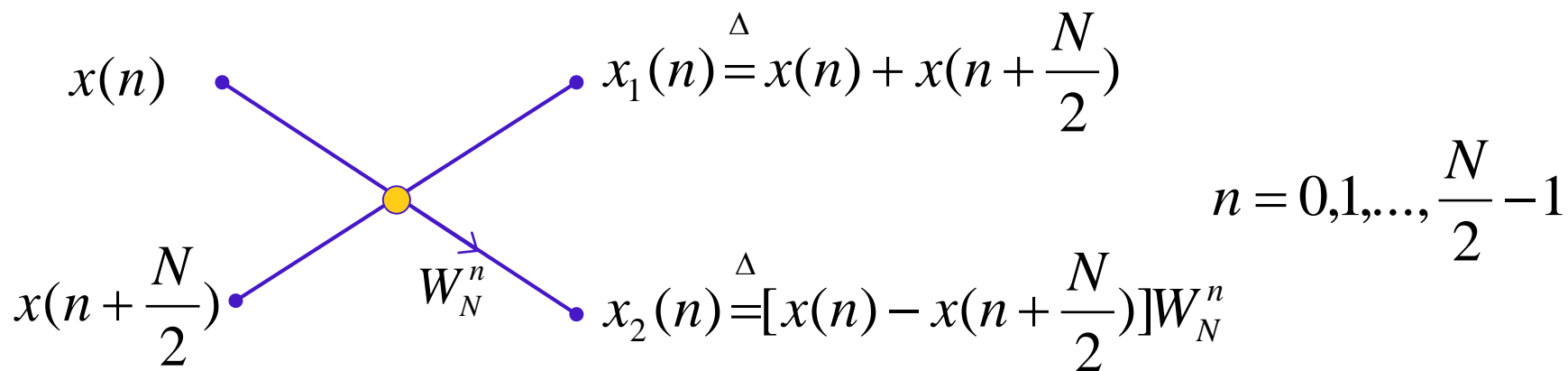
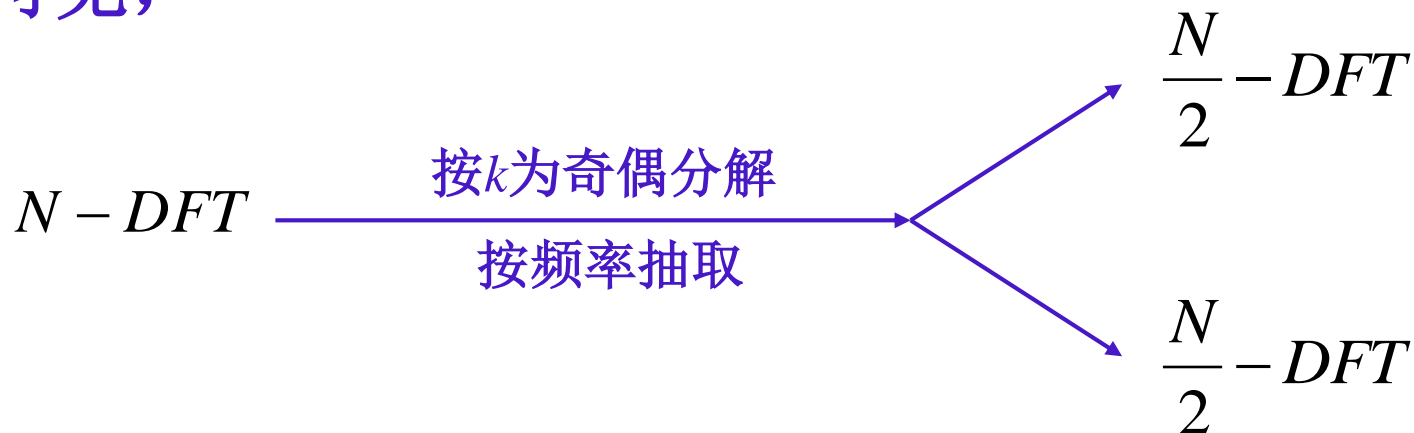
$$X_2(k) \stackrel{\Delta}{=} X(2k + 1), \quad k = 0, 1, \dots, \frac{N}{2} - 1$$

则有（式(4-25)(4-26)分别变为）

$$X_1(k) = X(2k) = DFT[x_1(n)] = \sum_{n=0}^{\frac{N}{2}-1} x_1(n)W_{N/2}^{kn}, \quad k = 0, 1, \dots, \frac{N}{2} - 1$$

$$X_2(k) = X(2k + 1) = DFT[x_2(n)] = \sum_{n=0}^{\frac{N}{2}-1} x_2(n)W_{N/2}^{kn}, \quad k = 0, 1, \dots, \frac{N}{2} - 1$$

可见,



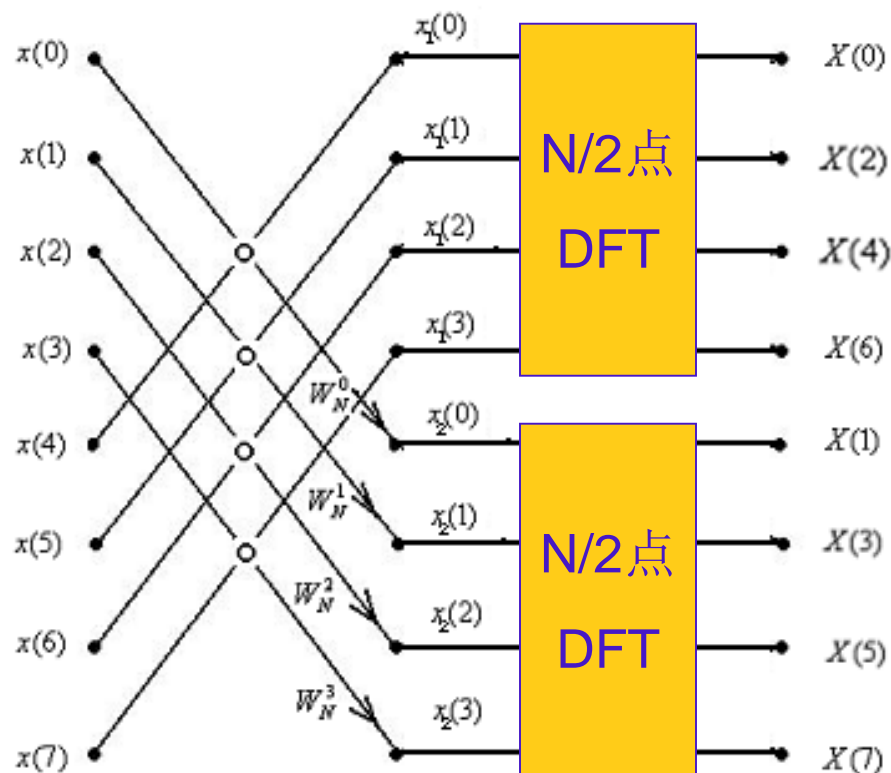
例：N=8，DIF的分解过程（见图4-16） [P.137]

$\because N = 2^v, \frac{N}{2} = 2^{v-1}$ 仍为偶数 ($v \geq 3$)

\therefore 上述分解过程可继续下去，
直至分解v次/步后变成求
N/2 个 2-DFT 为止。

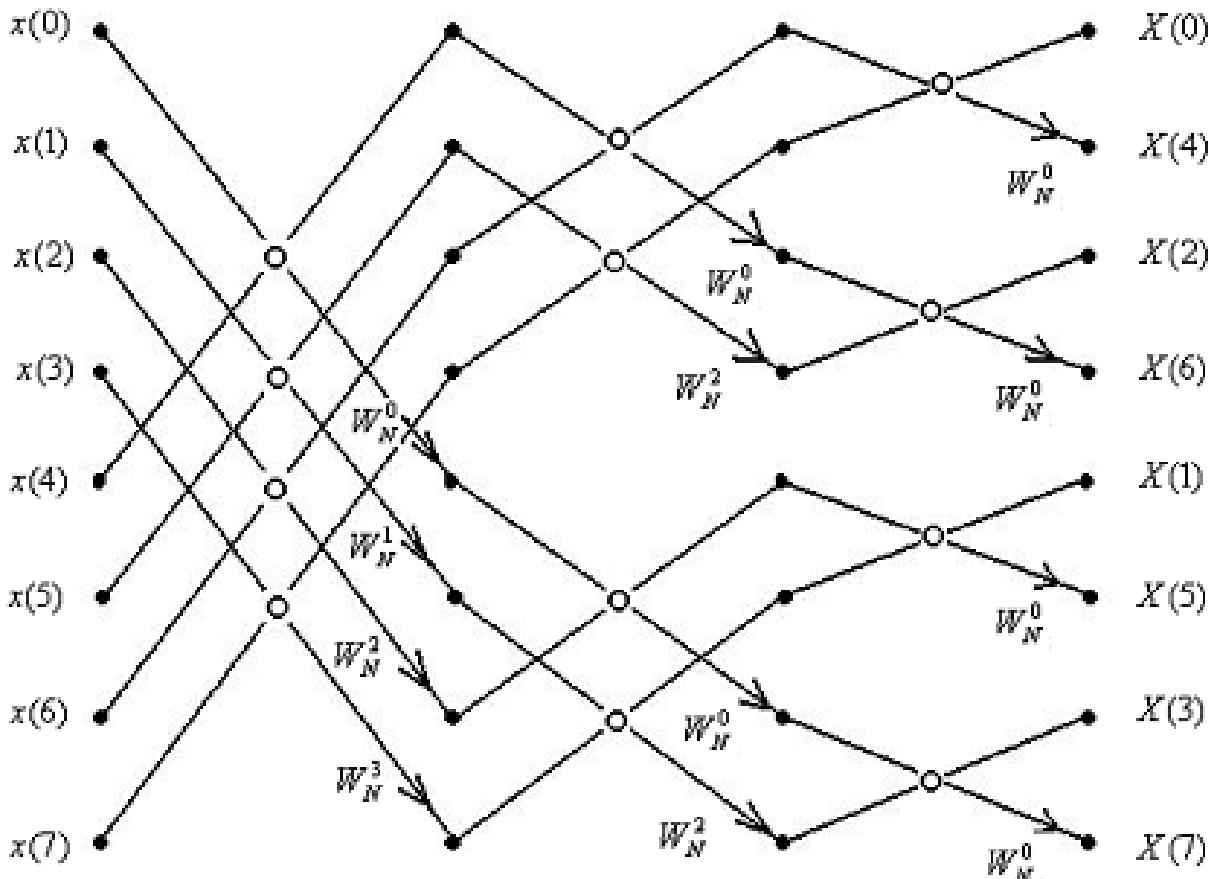
—————> DIF-FFT算法

（显然与DIT-FFT算法的分解类似）



例：N=8，DIF-FFT算法流图 [图4-18, P.138]

注意：① $W_N^n, n=0,1,\dots,N/2-1$ $n=0,1,\dots,N/2-1$
 $W_{N/2}^n, n=0,1,\dots,N/4-1 \longrightarrow W_N^n, n=0,1,\dots,N/2-2$
 $W_{N/4}^n, n=0,1,\dots,N/8-1$ $n=0,1,\dots,N/2-4$



② $W_N^0 (=1)$, 依然保留
 (为了算法结构上的统一)

二、DIF-FFT与DIT-FFT的比较

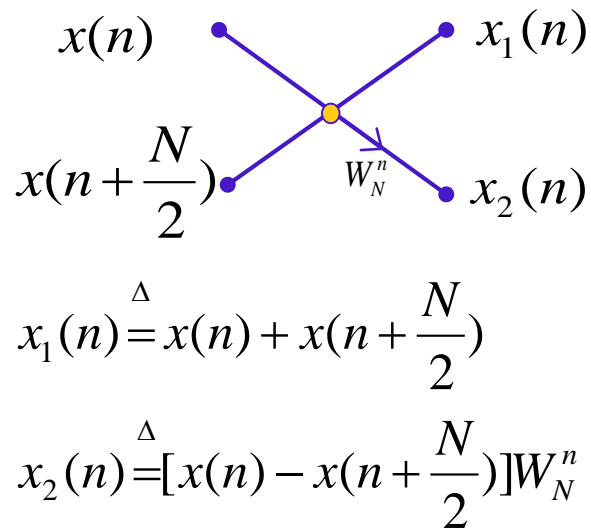
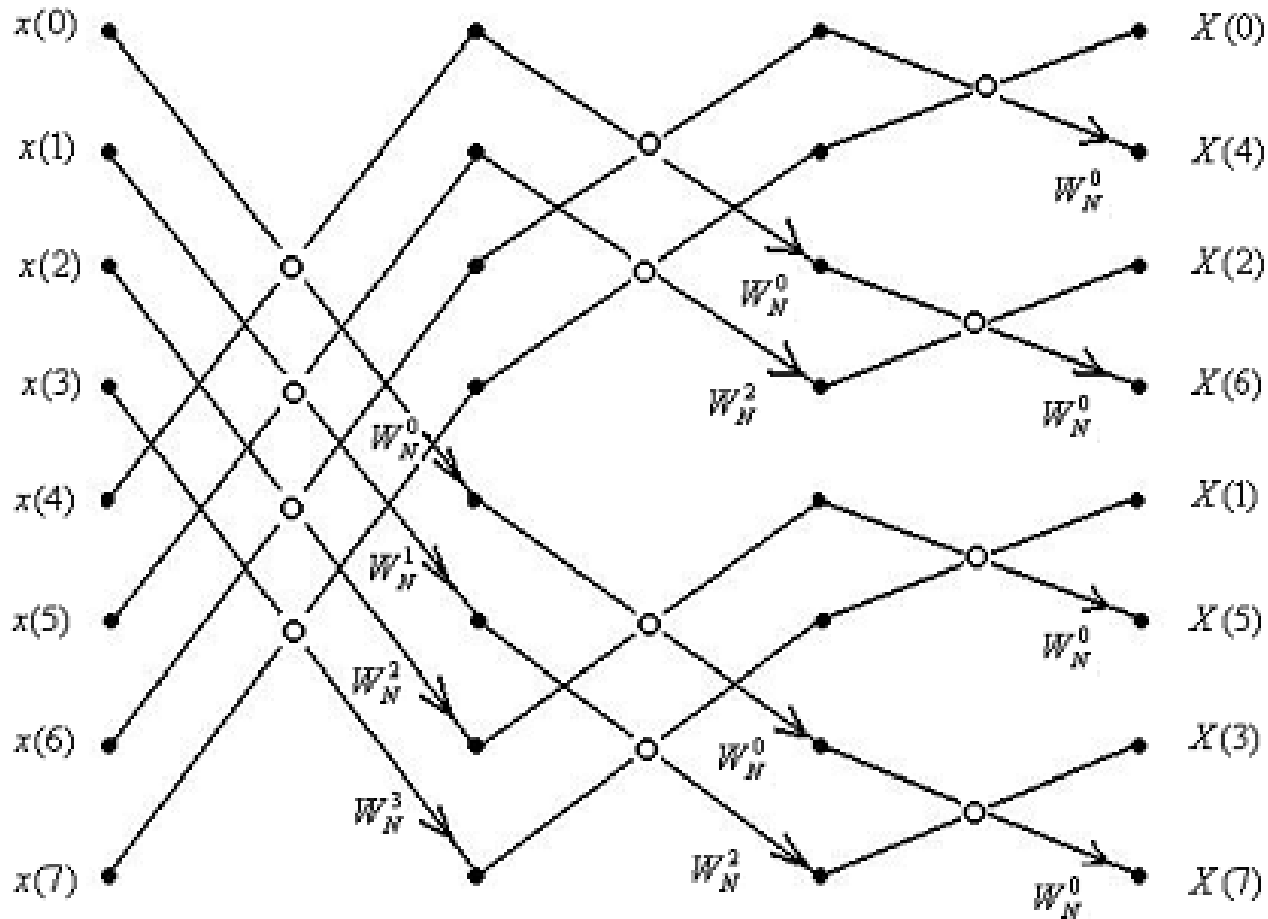


图4-18 N=8,DIF-FFT算法流图

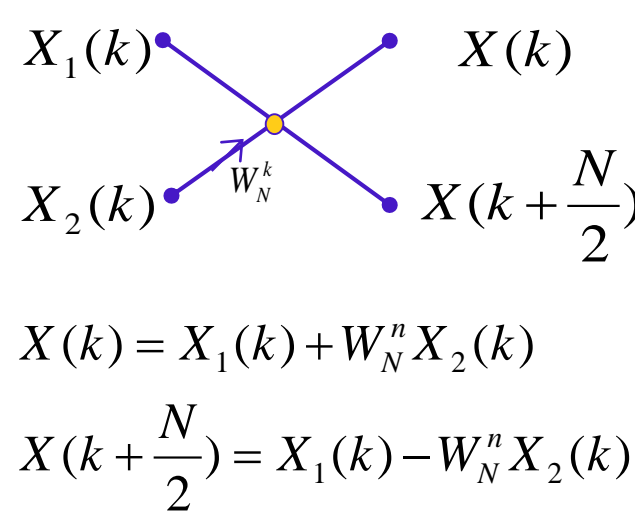
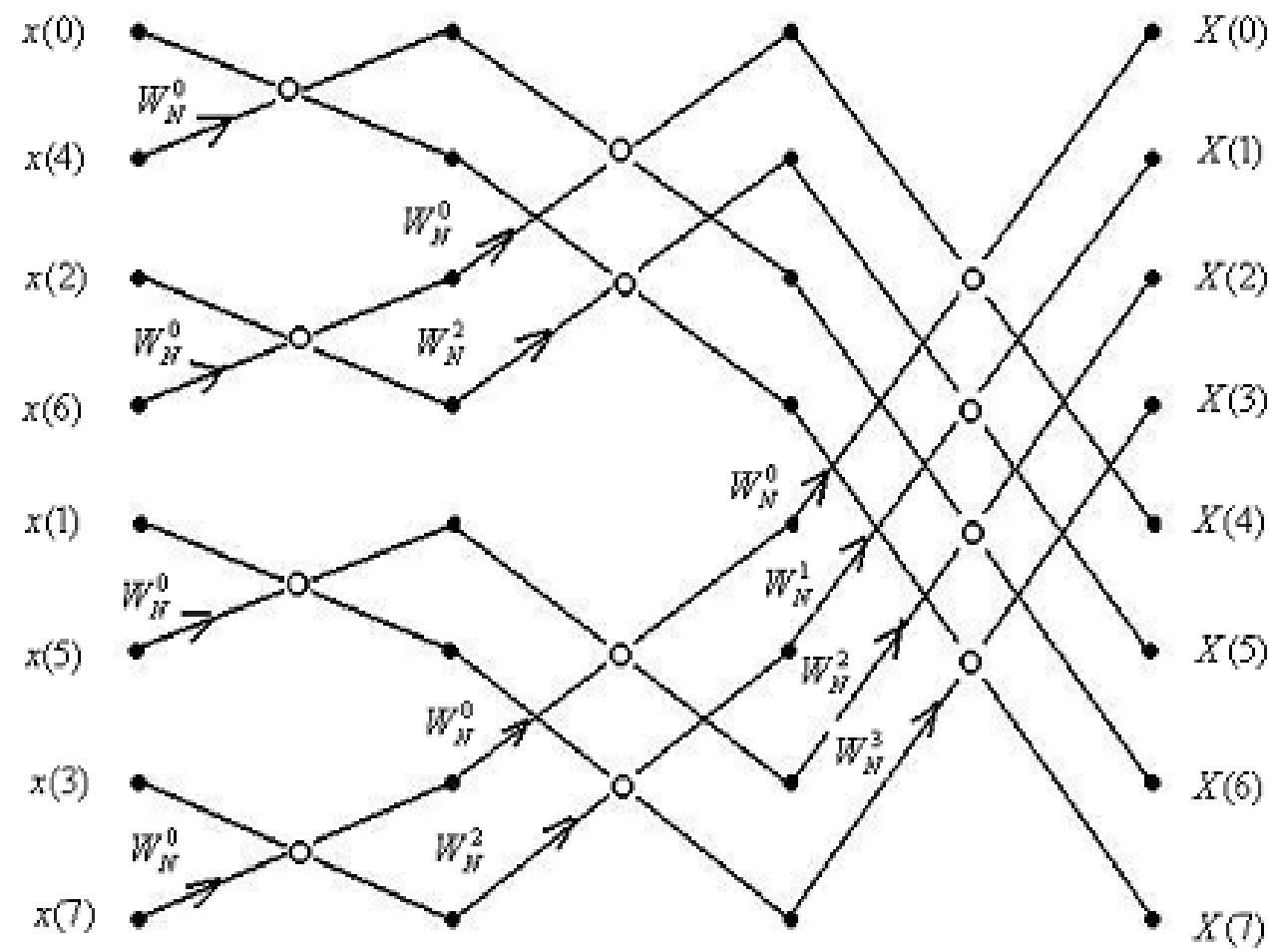
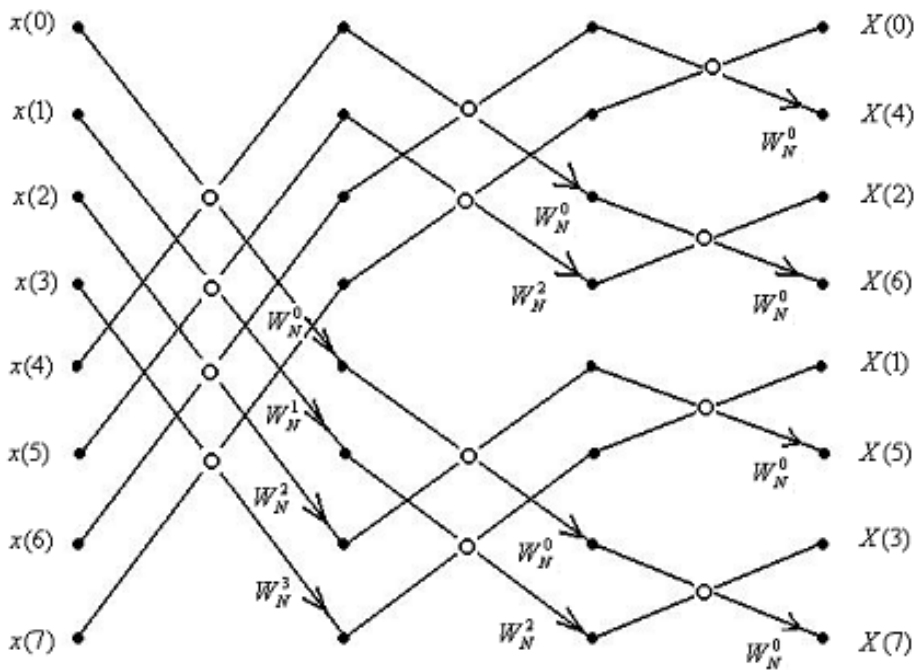
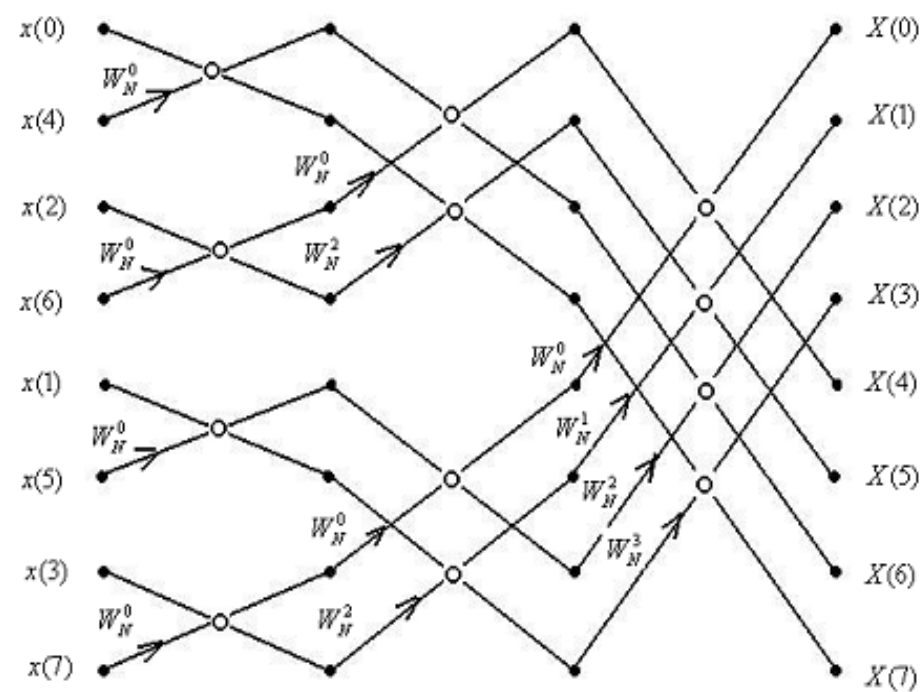


图4-5 N=8,DIT-FFT算法流图



N=8,DIF-FFT



N=8,DIT-FFT

1. 二者的区别

(1) 输入与输出

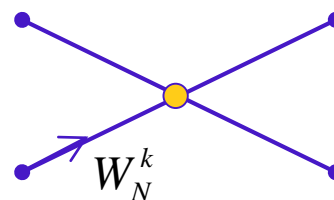
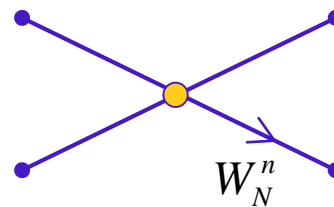
DIF: 顺序 反序

DIT: 反序 顺序

(2) 蝶形运算

DIF: 先加减, 后相乘

DIT: 先相乘, 后加减



2. 二者的相似之处

(1) 分解过程

DIF: v 列 每列 $N/2$ 个蝶形运算

$$m_F = \frac{N}{2} \log_2^N, a_F = \log_2^N$$

DIT: v 列 同上

同上

(2) 原位运算 (∵所有运算均由蝶形运算构成)

3. 二者关系



三、逆DFT的快速算法(IFFT)

1. 算法一: $IDFT: x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k)W_N^{-kn}, \quad n = 0, 1, \dots, N-1$

$DFT: X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn}, \quad k = 0, 1, \dots, N-1$

↓
FFT

比较IDFT与DFT, 可见 $FFT \xrightarrow[\begin{matrix} W_N \rightarrow \frac{1}{2}W_N^{-1} \\ x(n) \rightarrow X(k) \end{matrix}]{\hspace{10em}} IFFT$

(1) $DIT\text{-}FFT \xrightarrow[\begin{matrix} W_N \rightarrow \frac{1}{2}W_N^{-1} \\ x(n) \rightarrow X(k) \end{matrix}]{\hspace{10em}} DIF\text{-}IFFT$

(2) $DIF\text{-}FFT \xrightarrow[\begin{matrix} W_N \rightarrow \frac{1}{2}W_N^{-1} \\ x(n) \rightarrow X(k) \end{matrix}]{\hspace{10em}} DIT\text{-}IFFT$

例：N=8，DIT-IFFT算法流图[图4-19， P.138]

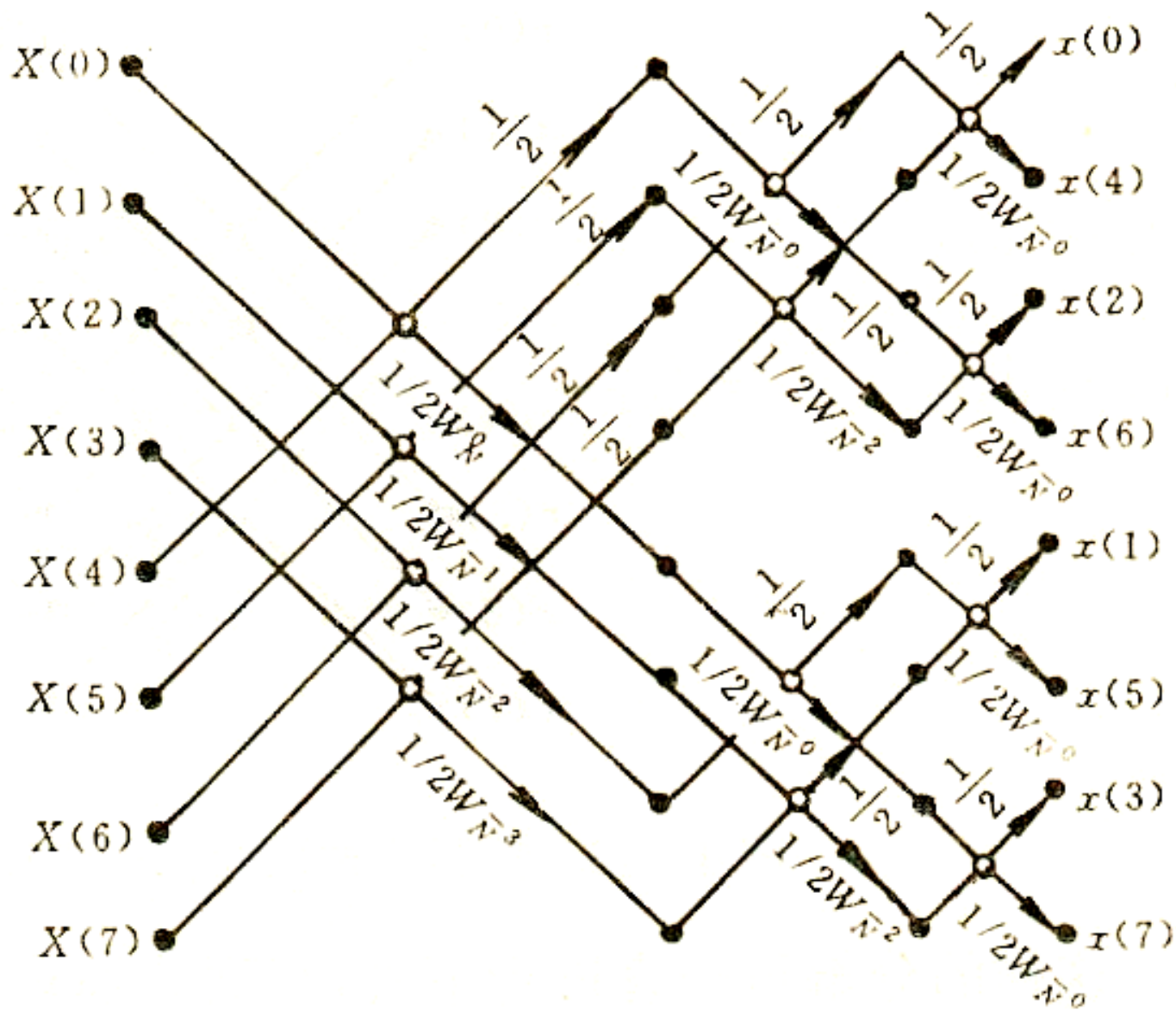


图 4-19 N=8 的时间抽取 IFFT 流图

例：N=8，DIT-IFFT算法流图[图4-19，P.138]

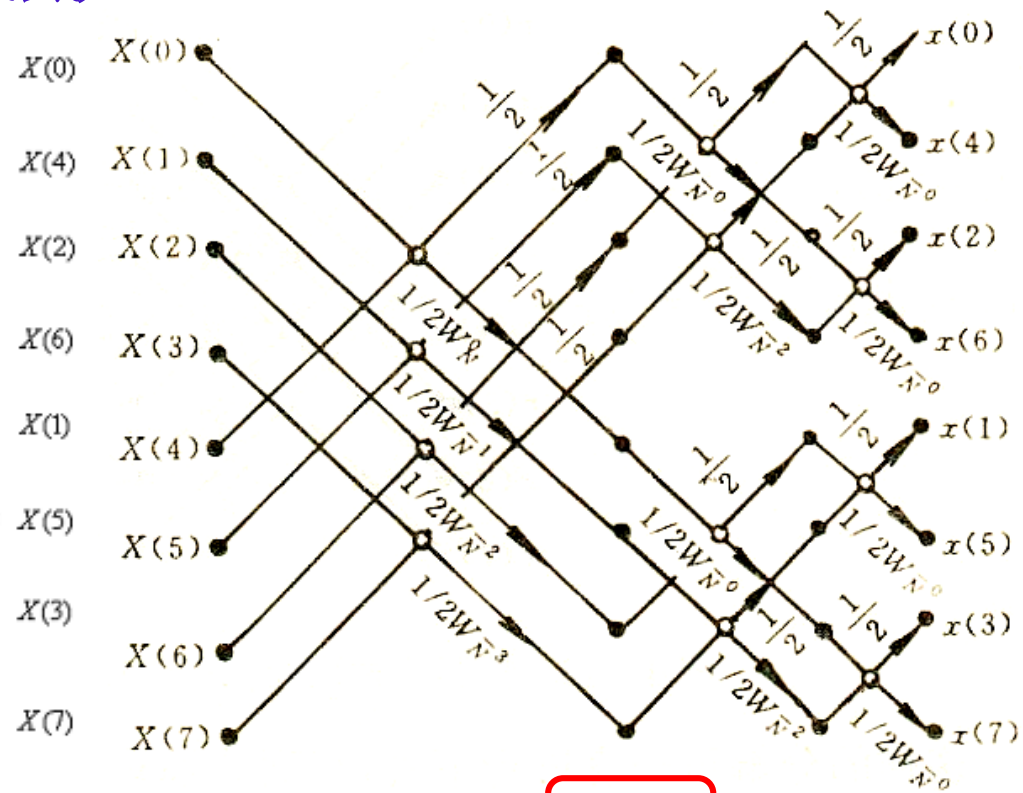
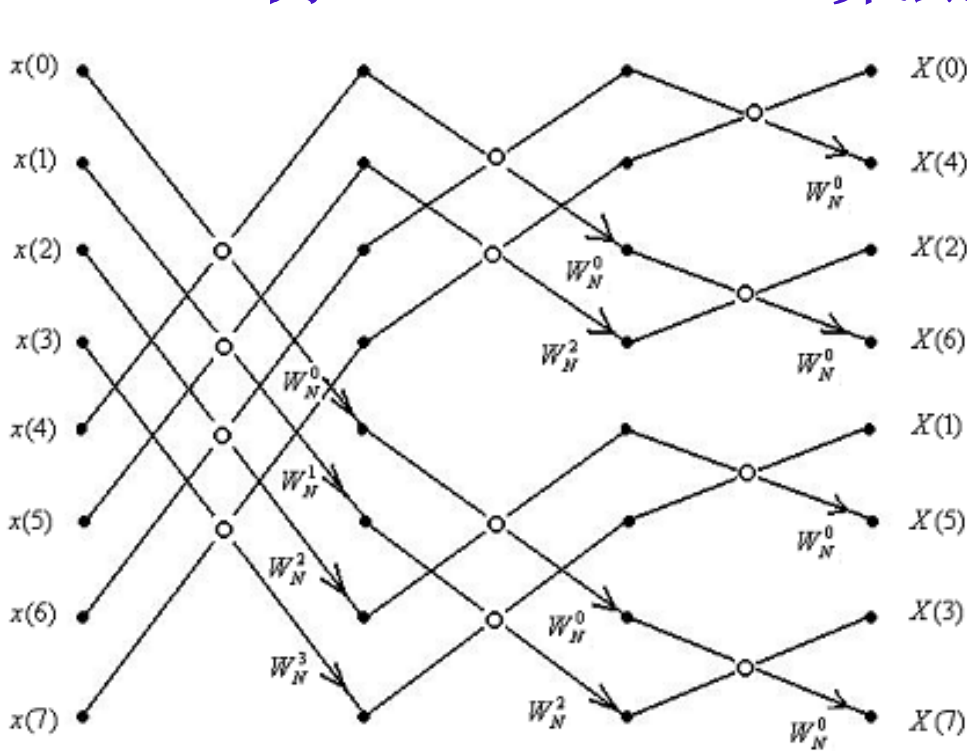


图4-18 N=8, DIF-FFT算法流图

图 4-19 N=8的时间抽取 IFFT 流图

- 解释：
- 1.相同的流图可以采用同一个计算机算法
 - 2.按时间抽取是“抽 $x(n)$ ”和按频率抽取是“抽 $X(k)$ ”
 - 3.DIF-FFT和DIT-IFFT流图结构一样
 - 4.DIT-FFT和DIF-IFFT流图结构一样

例：N=8，DIT-IFFT算法流图[图4-19， P.138]

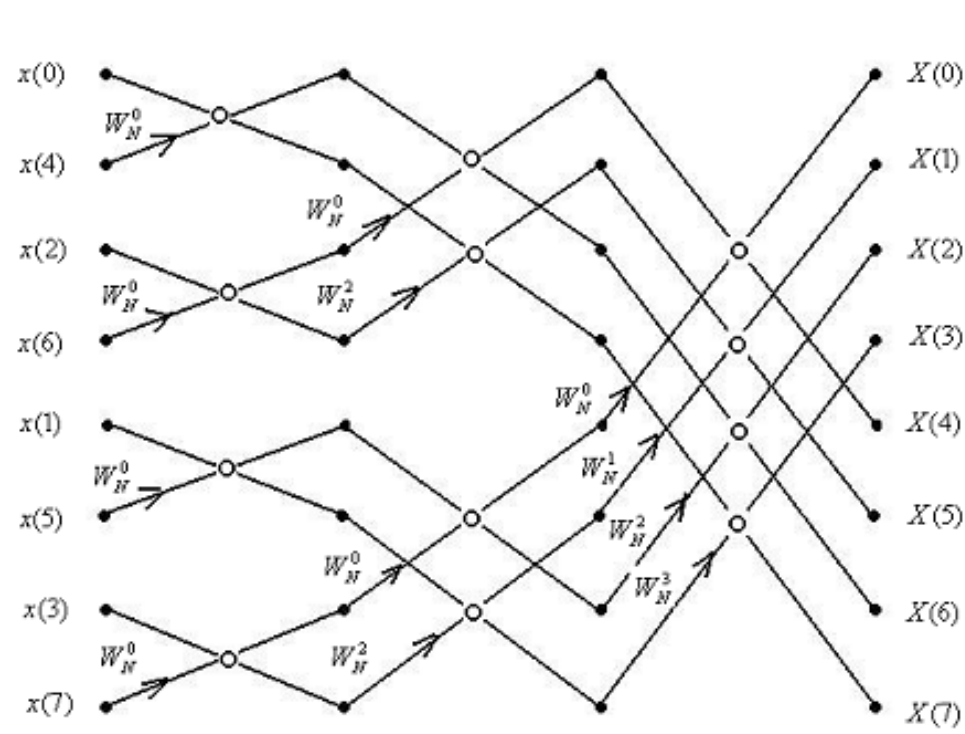


图4-5 N=8时的按时间抽取FFT运算流图

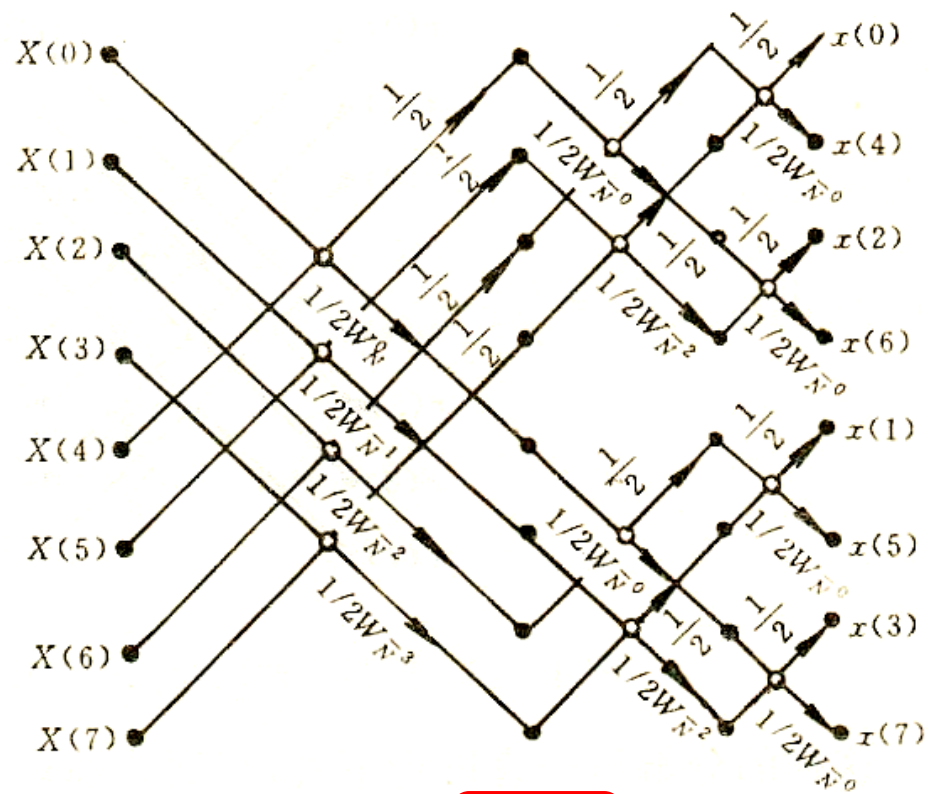


图 4-19 N=8 的时间抽取 IFFT 流图

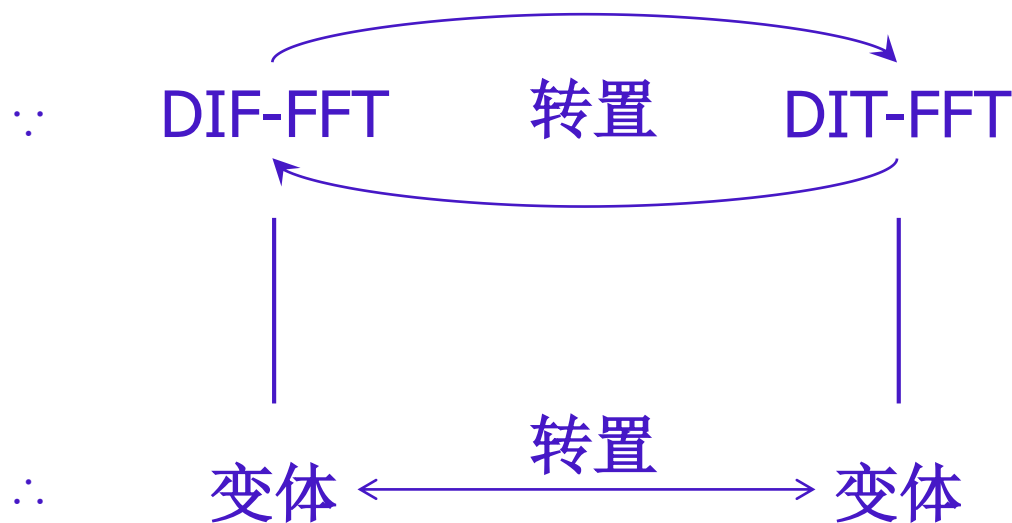
2. 算法二

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} = \frac{1}{N} \left[\sum_{k=0}^{N-1} X^*(k) W_N^{kn} \right]^* = \frac{1}{N} \left\{ DFT[X^*(k)] \right\}^*$$

优点:

$$\frac{1}{N} \left\{ FFT[X^*(k)] \right\}^*$$

四、DIF-FFT算法的若干变体



往年真题:

试导出按频率抽取基-2 FFT 算法的蝶形运算公式，并画出相应的 $N=16$ 时的算法流图。

(要求输入正序，输出反序，原位运算)

N=16 基-2 按频率抽取FFT流图

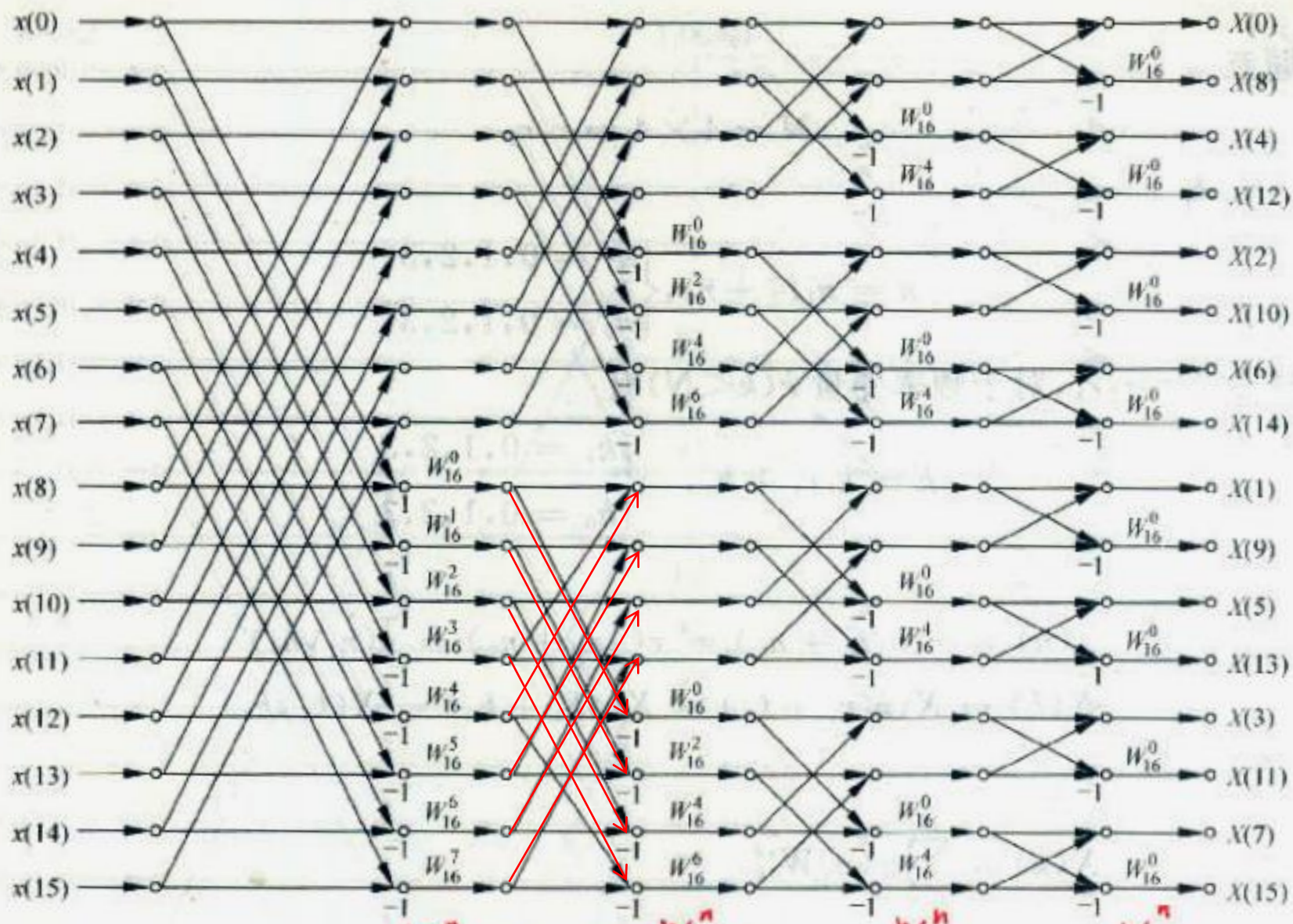


图 P4-3(b)