第九章：离散傅里叶变换的计算

- 9.1 离散傅里叶变换的高效计算
- 9.2 Goertzel算法
- 9.3 按时间抽取的FFT算法
- 9.4 按频率抽取的FFT算法
- 9.5 实现问题考虑
- 9.6 用卷积实现DFT
- 9.7 有限寄存器长度的影响

9.1 傅里叶变换的高效计算

◆ DFT直接计算复杂度分析

长度为N的有限长序列的DFT变换对可表示为

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

$$x[n]=(1/N)\sum_{k=0}^{N-1}X[k]W_N^{-kn}$$

式中 $W_N=e^{-j2\pi/N}$

假设$x[n]$为复数

计算DFT每个输出值需要N次复数乘法和N-1次复数加法

计算全部N个值需要N^2次复数乘法和N (N-1) 次复数加法

由于DFT与IDFT仅差复指符号，计算方法和复杂度相同

9.1 傅里叶变换的高效计算

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

◆ DFT直接计算复杂度分析（续）

若用实数运算来表示有限长复序列的DFT，可得

$$X[k]=\sum_{n=0}^{N-1}\Big[\Big(\mathrm{Re}\{x[n]\}\mathrm{Re}\{W_N^{kn}\}-\mathrm{Im}\{x[n]\}\mathrm{Im}\{W_N^{kn}\}\Big)$$
$$+j\Big(\mathrm{Re}\{x[n]\}\mathrm{Im}\{W_N^{kn}\}-\mathrm{Im}\{x[n]\}\mathrm{Re}\{W_N^{kn}\}\Big)\Big],\quad k=0,1,\cdots,N-1$$

由于计算每个复乘需要4次实数乘法和2次实数加法，每个复加需要2次实数加法

所以计算DFT每一个值需要4N次实数乘法和2N+2(N-1)= 4N-2次实数加法

因此，计算全部N个值需要4N^2次实数乘法和N (4N-2) 次实数加法，即计算次数与时间大致与4N^2成正比

另外，计算每个$X[k]$需要计算或存储N个系数 W_N^{kn}， $n=0,\ldots,N-1$
计算全部N个值需要计算或存储N^2个系数。

9.1 傅里叶变换的高效计算

改善DFT计算效率的大多数方法均利用 W_N^{kn} 的对称性和周期性

1） $W_N^{kn}=W_N^{k(n+N)}=W_N^{(k+N)n}$ （对 n 和 k 的周期性）；

2） $W_N^{k(N-n)}=W_N^{-kn}=\left(W_N^{kn}\right)^*$ （对 n 的复共轭对称性）；

利用复共轭对称性，计算 $X[k]$的实数运算中含 n 和$(N-n)$的项可合并为

$$\mathrm{Re}\{x[n]\}\mathrm{Re}\{W_N^{kn}\}+\mathrm{Re}\{x[N-n]\}\mathrm{Re}\{W_N^{k(N-n)}\}$$

$$=\left(\mathrm{Re}\{x[n]\}+\mathrm{Re}\{x[N-n]\}\right)\mathrm{Re}\{W_N^{kn}\}$$

$$\mathrm{Im}\{x[n]\}\mathrm{Im}\{W_N^{kn}\}+\mathrm{Im}\{x[N-n]\}\mathrm{Im}\{W_N^{k(N-n)}\}$$

$$=\left(\mathrm{Im}\{x[n]\}-\mathrm{Im}\{x[N-n]\}\right)\mathrm{Im}\{W_N^{kn}\}$$

即两次实乘运算通过合并变为一次实乘运算，这样DFT的总乘法次数减少1/2，但计算量仍正比于$2N^2$

利用 W_N^{kn} 的周期性可进一步降低DFT的计算量。

9.2 Goertzel算法

Goertzel算法是一种利用系数序列（ W_N^{kn} ） 的周期性来降低DFT计算量（输入计算量、系数计算和存储量）的算法。

由 W_N^{kn} 的周期性可得 $W_N^{-kN}=e^{jkN(2\pi/N)}=e^{j2\pi k}=1,\quad k=0,\cdots,N-1$

因此序列 $x[r]$ 的DFT可表示为

$$X[k]=W_N^{-kN}\sum_{r=0}^{N-1}x[r]W_N^{kr}=\sum_{r=0}^{N-1}x[r]W_N^{-k(N-r)}$$

定义序列

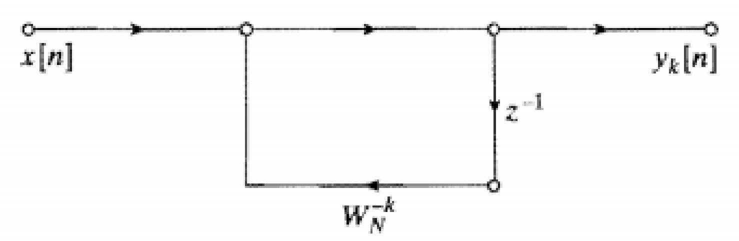
$$y_k[n]=\sum_{r=-\infty}^{\infty}x[r]W_N^{-k(n-r)}u[n-r]=x[n]*W_N^{-kn}u[n]$$

由于 $x[n]=0,\quad n\notin[0\quad N-1]$，可得卷积输出第$N$个样值为

$$y_k[n]\big|_{n=N}=\sum_{r=0}^{N-1}x[r]W_N^{-k(N-r)}u[N-r]=\sum_{r=0}^{N-1}x[r]W_N^{-k(N-r)}=X[k]$$

即有限长序列 $x[n]$ 的DFT的第 k 个值 $X[k]$，可表示为脉冲响应为 $W_N^{-kn}u[n]$ 的系统对输入序列 $x[n]$ 在 $n=N$ 时的响应。

◆ DFT的一阶复系数递推实现



右图为脉冲响应为 $W_N^{-kn}u[n]$ 的系统流图，相应的差分方程为

$$y_k[n]=W_N^{-k}y_k[n-1]+x[n]$$

对于复数输入序列，计算每个输出值 $y_k[n]$ 需4次实乘和4次实加而获得一个 $X[k]$ 值需计算全部N个输出值，共需$4N$次实乘和$4N$次实加

尽管比直接法所需的实加数($4N$-2)稍多，但对每个$X[k]$的递推实现过程仅需计算和存储1个系数 W_N^{kn}， $k=0,\ldots,N-1$ ，即系数的计算和存储量为直接法的1/N。

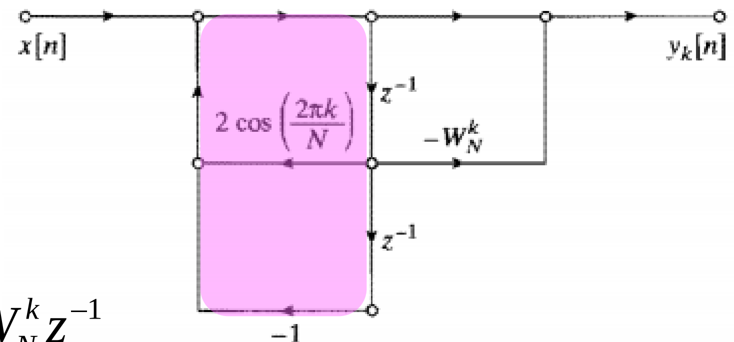
采用一阶复系数递推实现，可以降低系数计算和存储量；
但是，该实现结构无法降低对输入序列的计算量。

9.2 Goertzel算法

◆ DFT的二阶实系数递推实现（Goertzel算法）

脉冲响应 $W_N^{-kn}u[n]$ 对应的系统函数为

$$H_k(z)=\frac{1}{1-W_N^{-k}z^{-1}}$$



系统函数分子分母乘相同的因式可得

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

对于复输入序列，该系统极点（反向链路）只需对每个样值做2次实乘和4次实加，零点的系数 $-W_N^k$ 复乘不必在每次迭代中都计算，即只需计算最后一次n=N时值的1次复乘和1次复加。

计算一个 $X[k]$ 值共需(2N+4)次实乘和(4N+4)次实加，比直接法的4N和4N-2次减少约1/2，且不必计算和存储随n变化的系数。

采用两阶实系数递推实现，对比直接法，系数计算和存储量降低为约1/N的同时，输入计算量降低约一半，即正比于2N^2。

9.2 Goertzel算法

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

- 将DFT输出的对称性用于Goertzel算法

对于$x[n]$的z变换在单位圆上位于共轭位置的值$X[k]$和$X[N-k]$的计算，所需的网络形式和极点相同，零点互为复共轭关系

$$H_{N-k}(z)=\frac{1}{1-W_N^{-(N-k)}z^{-1}}=\frac{1}{1-W_N^k z^{-1}}=\frac{1-W_N^{-k}z^{-1}}{\left(1-W_N^k z^{-1}\right)\left(1-W_N^{-k}z^{-1}\right)}=\frac{1-\left(W_N^k\right)^* z^{-1}}{1-\cos(2\pi k/N)z^{-1}+z^{-2}}$$

因此，可利用同一极点（用$2N$次实乘和$4N$次实加）同时算出两个DFT值；而两者不同的零点仅需在最后一次迭代计算中实现。

这样，利用Goertzel算法计算N点DFT，需约$N/2$x$2N$=N^2次实乘和$N/2$x$4N$= $2N^2$次实加，即该算法计算量仍正比于N^2。

忽略了零点的计算量

Goertzel算法可以计算任意M个$X[k]$值，其计算量正比于MN；当$M<\log_2 N$时，该算法是最有效的。

当要计算$X[k]$的全部N个值时，若需进一步降低计算量，还需要采用能同时利用系数周期性和对称性的更有效算法。

- 9.1 离散傅里叶变换的高效计算
- 9.2 Goertzel算法
- 9.3 按时间抽取的FFT算法
- 9.4 按频率抽取的FFT算法
- 9.5 实现问题考虑
- 9.6 用卷积实现DFT
- 9.7 有限寄存器长度的影响

9.3 按时间抽取FFT算法

同时利用序列 W_N^{kn} 的对称性和周期性的计算算法，可使DFT的计算量显著降低，即大体上正比于$N\log N$，该类算法称为快速傅里叶变换（FFT）。

FFT算法是基于可以将一个长度为 N 的序列的DFT逐次分解为较短的DFT来计算这一基本原理。

◆ 两类FFT算法

时间抽取FFT算法：将（时间）序列 $x[n]$逐次分解为较短的子序列，并采用较小的DFT来实现原计算。

频率抽取FFT算法：将（频域）DFT系数 $X[k]$逐次分解为较短的子序列，并采用较小的DFT来实现原计算。

9.3 按时间抽取FFT算法

◆ 按时间抽取FFT的DFT的分解

将长度N为2的整数幂的序列 $x[n]$ 的N点DFT分解为2个$N/2$ 点序列计算，其中一个为 $x[n]$ 的偶数点，另一个为 $x[n]$ 的奇数点。

这样 $x[n]$的N点DFT可分解表示为

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

$$=\sum_{n为偶数}x[n]W_N^{kn}+\sum_{n为奇数}x[n]W_N^{kn},\quad k=0,1,\cdots,N-1$$

对于偶数和奇数 n 分别用变量 $n=2r$ 和 $n=2r+1$ 代替，可得

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

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

◆ 按时间抽取FFT的DFT的分解（续1）

由于 $W_N^2 = e^{-2j(2\pi/N)} = e^{-j2\pi(N/2)} = W_{N/2}$

因此 $x[n]$的N点DFT可表示为

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

式中$G[k]$和$H[k]$分别为 $x[n]$的偶数和奇数样值$N/2$点DFT的延拓序列

即， $G[k]$和 $H[k]$ 都是周期为$N/2$的周期序列，则

$$G[k] = G[k+N/2] \quad 和 \quad H[k] = H[k+N/2]$$

式中 W_N^k 是以所求DFT的点数N为周期的复指数序列。

9.3 按时间抽取FFT算法

◆示例：N=8点DFT分解为
2个N/2=4点DFT

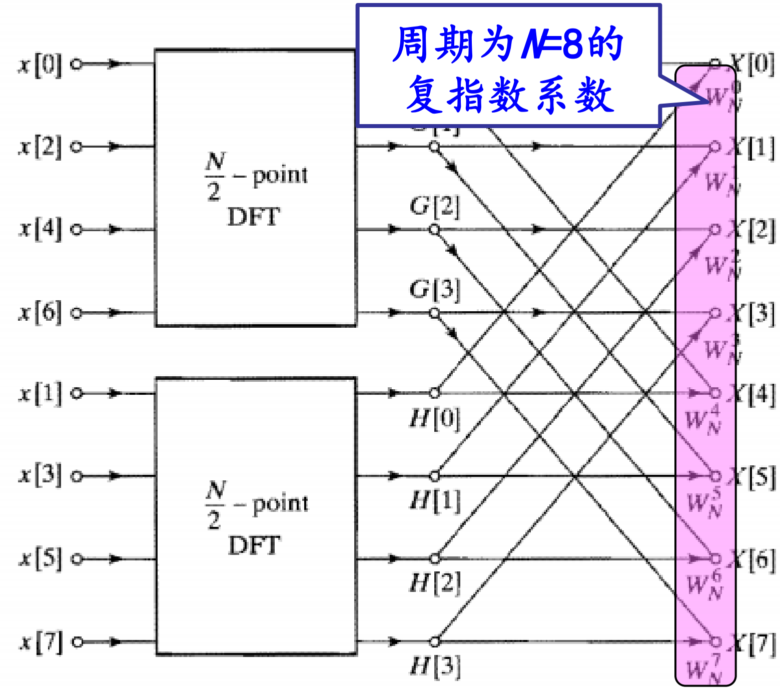
分解的DFT可表示为

$$X[k]=G[k]+W_N^k H[k],\quad k=0,\cdots,7$$

式中 $G[k]$ 和 $H[k]$ 分别为 $x[n]$ 的偶数和奇数的4点DFT拓展序列

并且 $G[k]$ 和 $H[k]$ 周期为4

$$G[0]=G[4] \qquad H[0]=H[4]$$
$$G[1]=G[5] \qquad H[1]=H[5]$$
$$G[2]=G[6] \qquad H[2]=H[6]$$
$$G[3]=G[7] \qquad H[3]=H[7]$$



2个N/2点DFT需2$(N/2)^2$次复乘和2$(N/2)(N/2-1)\approx 2(N/2)^2$次复加

另外，将2个N/2点DFT组合在一起还需N次复乘和N次复加

采用一级分解后，计算全部N点DFT约需$N+N^2/2$次复乘和复加

◆ DFT的分解（续2）

若进一步将2个$N/2$点DFT分别分解为2个$N/4$点DFT，则有

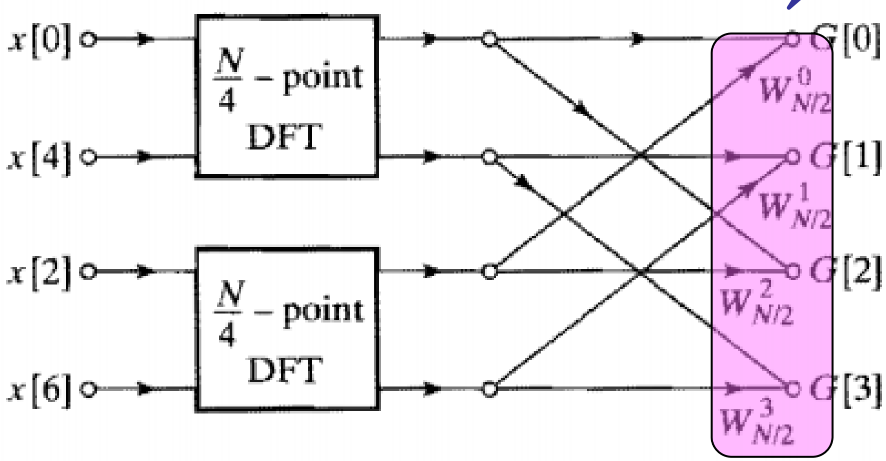
$$G[k]=\sum_{r=0}^{(N/2)-1} g[r]W_{N/2}^{rk}=\sum_{l=0}^{(N/4)-1} g[2l]W_{N/2}^{2lk}+\sum_{l=0}^{(N/4)-1} g[2l+1]W_{N/2}^{(2l+1)k}$$

$$=\sum_{l=0}^{(N/4)-1} g[2l]W_{N/4}^{lk}+W_{N/2}^{k}\sum_{l=0}^{(N/4)-1} g[2l+1]W_{N/4}^{lk}$$

$$H[k]=\sum_{l=0}^{(N/4)-1} h[2l]W_{N/4}^{lk}+W_{N/2}^{k}\sum_{l=0}^{(N/4)-1} h[2l+1]W_{N/4}^{lk}$$

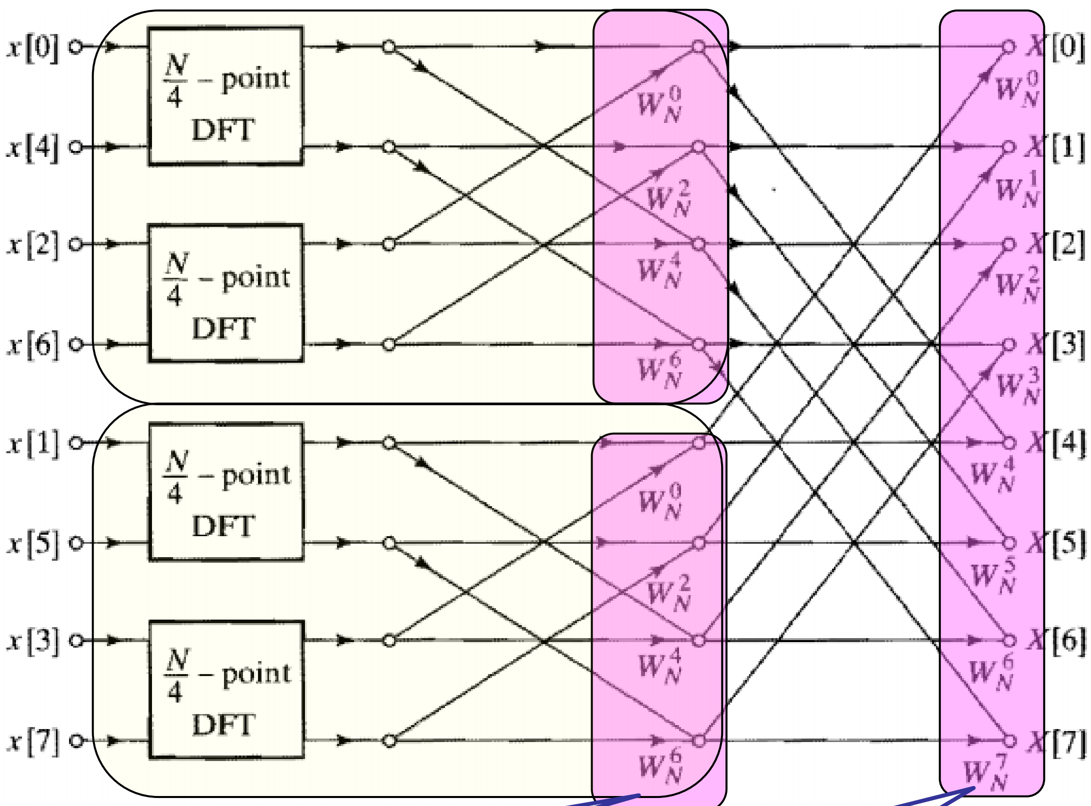
周期为$N/2$的复指数系数

按时间抽取将$(N/2)$点DFT的计算分解为2个$(N/4)$点DFT来计算的流图（N=8，偶数点）



◆按时间抽取FFT的DFT的分解（续3）

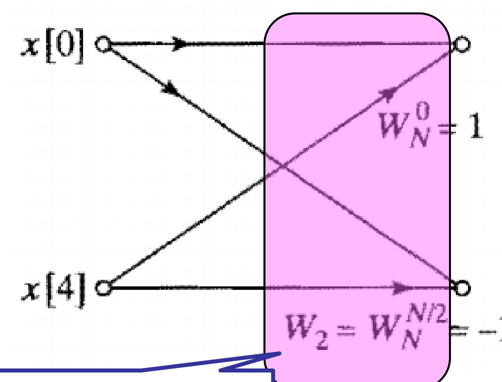
按时间抽取将N点DFT的计算分解为4个($N/4$)点DFT来计算的流图（N=8）



对于$N/2$点DFT，其系数可由N点DFT的系数表示（N=8）

$$W_{N/2}^{k} = W_N^{2k}, k = 0, \cdots, 3$$

2点DFT流图



周期为$N/2$的复指数系数

周期为N的复指数系数

周期为$N/4$=2的复指数系数

9.3 按时间抽取FFT算法

◆ 按时间抽取FFT的分解实现的DFT计算量分析

对于N为2的整数幂，N点DFT的分解到只剩2点DFT为止。

当一个N点DFT分解成2个$(N/2)$点变换时，N点DFT需要：

$N+2(N/2)^2$次复乘和复加

其中，$(N/2)^2$为直接计算$N/2$点DFT所需的复乘和复加数

当$(N/2)$点DFT再分解成2个$(N/4)$点变换时，N点DFT需要：

$N+2[N/2+2(N/4)^2] = N+N+4(N/4)^2$次复乘和复加

例： 对N=8时，采用2点DFT实现，复乘和复加数为

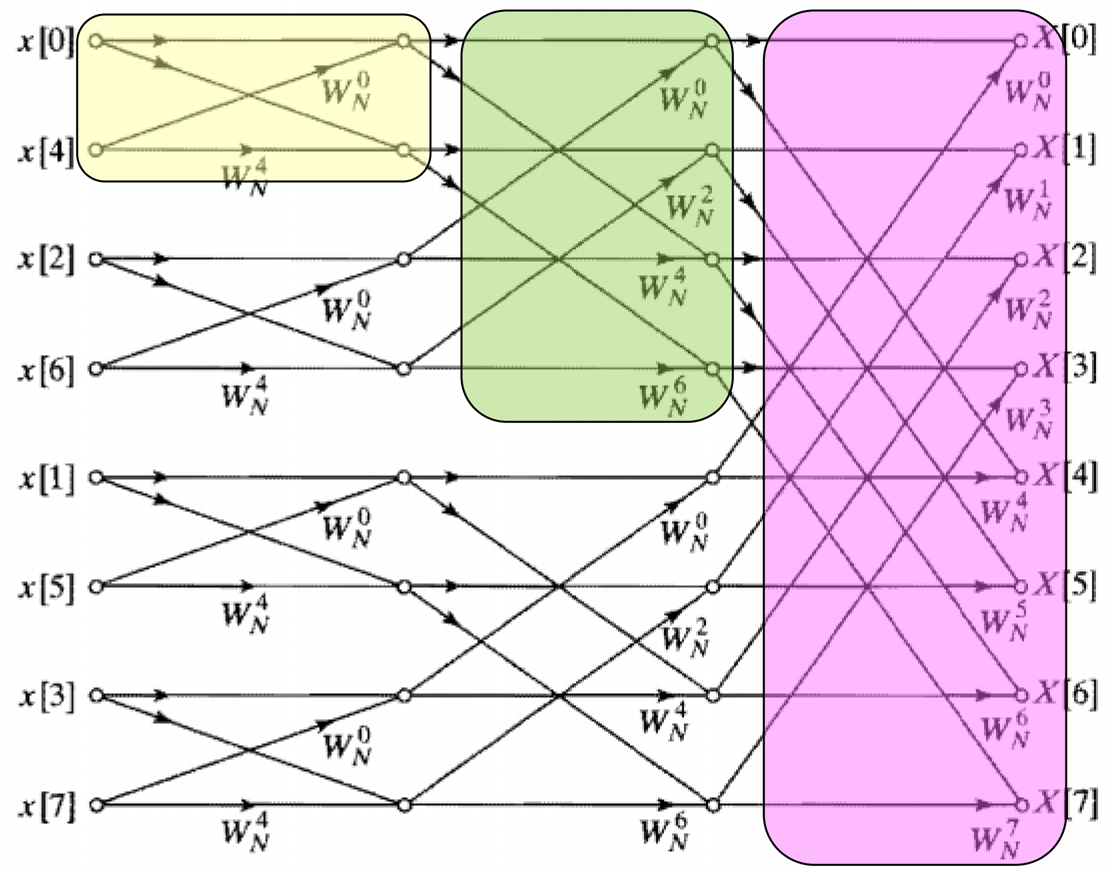
$8+8+4\mathrm{x}(8/4)^2 = 32$

9.3 按时间抽取FFT算法

若 $N=2^v$，最多可分解 $v=\log_2 N$ 级，分解全部完成之后，N点DFT所需的复数乘法和加法的次数为 $Nv=N\log_2 N$。

- 一共有$\log_2 N$级
- 每一级有N次复乘和N次复加

总共有$N\log_2 N$次复乘和复加

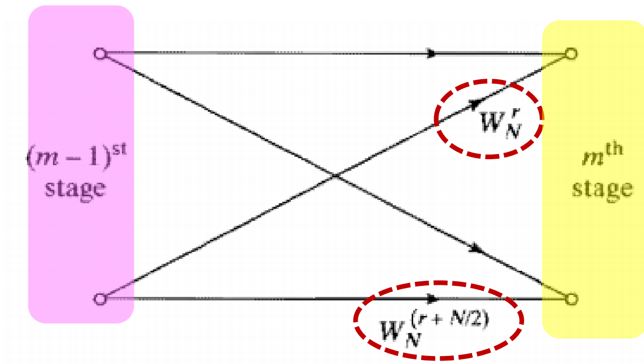


例： 对$N=2^{10}$时，
直接法：N^2=1048576
时间抽取FFT：
$N\log_2 N$=10240

例： 对N=8时，复乘/复加数为 $N\log_2 N$=24

根据前例：N=8时，复乘和复加数为$N+N+4(N/4)^2=32$

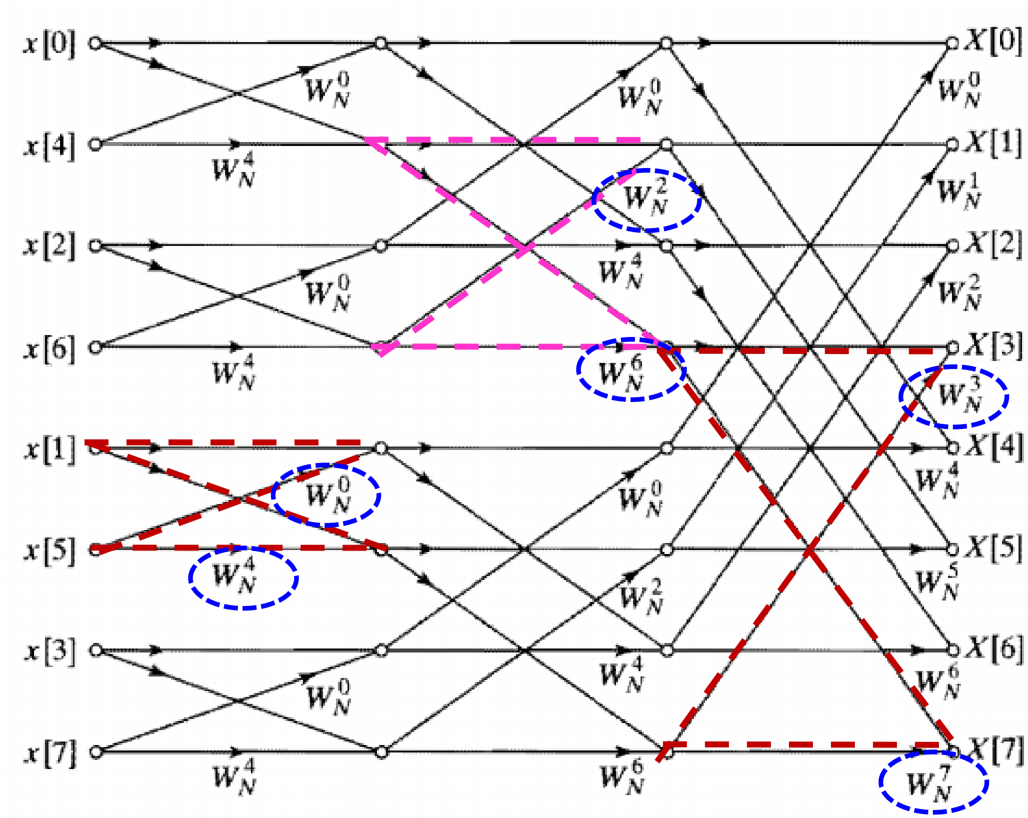
◆ 时间抽取DFT基本计算形式——蝶形计算



蝶形计算流图

1）用前一级的一对数值求得下一级的一对数值

2）系数总是 W_N 的幂，分支上幂指数总是相差$N/2$



9.3 按时间抽取FFT算法

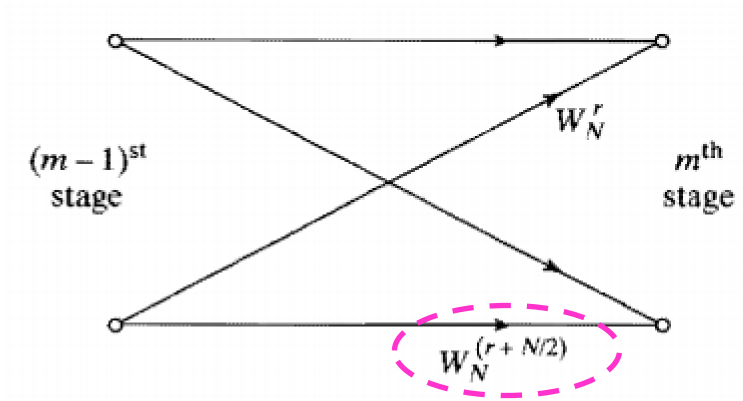
◆ 蝶形计算降低DFT计算量

由于

$$W_N^{N/2} = e^{-j(2\pi/N)N/2} = e^{-j\pi} = -1$$

所以

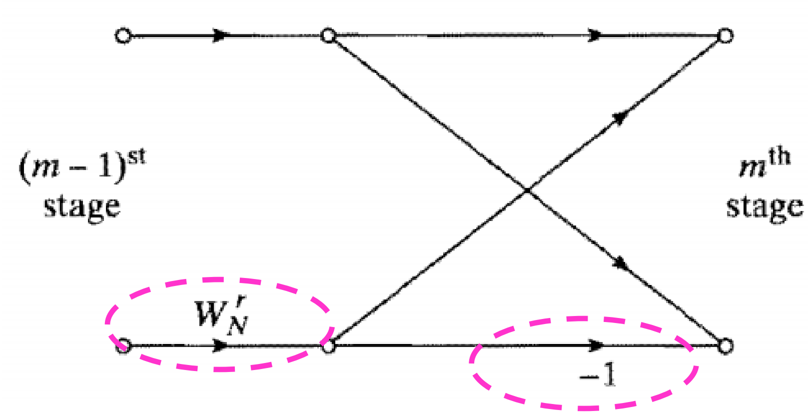
$$W_N^{r+N/2} = W_N^{N/2} W_N^r = -W_N^r$$



蝶形计算流图

蝶形计算流图可简化为右图形式

简化后的蝶形计算所需复乘次数减少一半，即N点DFT所需的复乘数为（$N/2$）$\log_2 N$



仅需一个复乘的蝶形计算流图