第五章

快速傅立叶变换

（FFT）

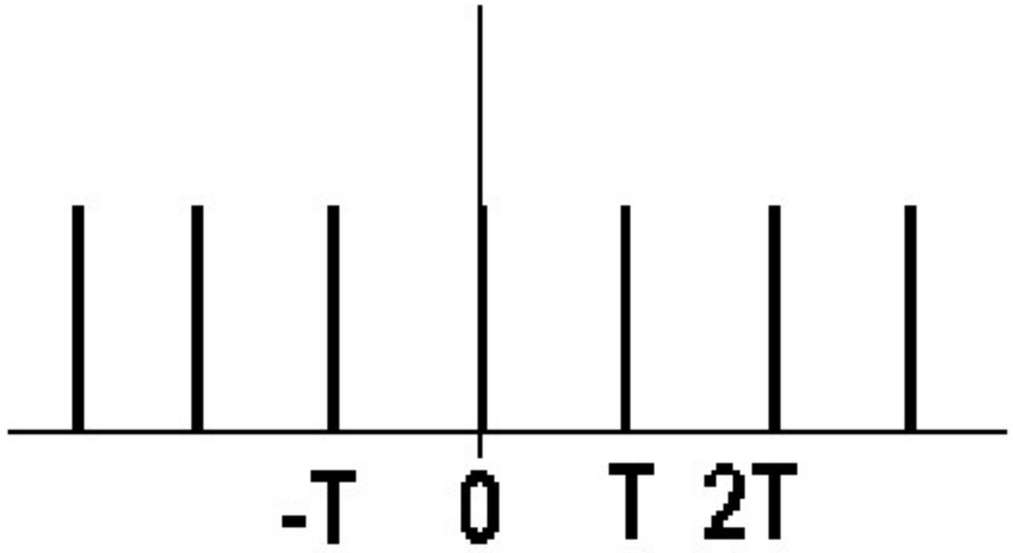
傅立叶变换

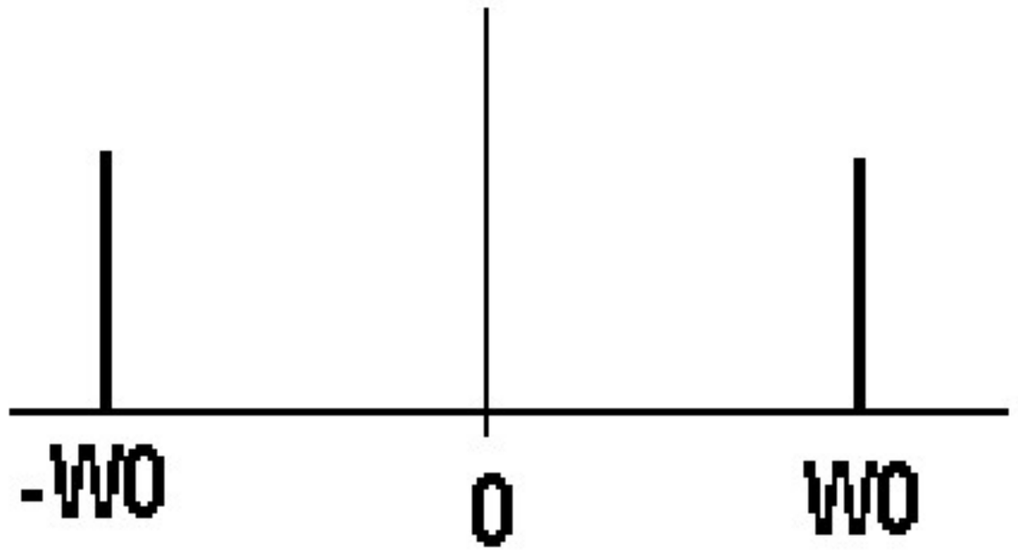
$$X(f) = \int_{-\infty}^{+\infty} \chi(t) e^{-j2\pi ft} dt$$

傅立叶变换

时域	<——>	频域
离散信号		周期谱
周期信号		离散谱
离散周期信号		离散周期谱

$$\delta_T(t) \leftrightarrow \omega_0 \delta_{\omega_0}(\omega)$$





DFT的定义

$$X(k)=\sum_{n=0}^{N-1}x[n]e^{-j2\pi nk/N}$$

$$x[n]=\frac{1}{N}\sum_{k=0}^{N-1}X(k)e^{j2\pi nk/N}$$

$$(k=0,1,\ldots,N-1)$$

DFT的定义

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

$$W_N=e^{-j\frac{2\pi}{N}}$$

DFT的定义

$$W_N^{(n+mN)(k+lN)} = W_N^{nk}$$

$$m,l = 0,\pm 1,\pm 2,\ldots$$

FFT

Number of Points	Direct Computation of the DFT		Radix-2 FFT	
	Complex Multiplies	Complex Additions	Complex Multiplies	Complex Additions
N	N^2	N^2-N	$(N/2)\log_2 N$	$N\log_2 N$
4	16	12	4	8
16	256	240	32	64
64	4096	4032	192	384
256	65536	65280	1024	2048
1024	1048576	1047552	5120	10240

将N点的序列分为两个N/2点的序列

$$x1[n] = x[2n]$$

$$x2[n] = x[2n+1]$$

将N点DFT分为两个N/2点DFT

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

n为偶数　　n为奇数

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

$$=\sum_{n=0}^{N/2-1}x1[n]W_{N/2}^{nk}+W_N^k\sum_{n=0}^{N/2-1}x2[n]W_{N/2}^{nk}$$

$$=X1(k)+W_N^kX2(k)$$

将N点DFT分为两个N/2点DFT

上式的最后一步是因为

$$W_N^2 = [e^{-j(\frac{2\pi}{N})}]^2$$

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

$$= W_{N/2}$$

将N点DFT分为两个N/2点DFT

- 我们已经将一个N点的DFT分解成为两个N/2点的DFT。
- 但是，X（k）有N点，但X1（k）和X2（k）都只有N/2点，因此，前面计算的只是X（k）的前一半项的结果。

将N点DFT分为两个N/2点DFT

对于后一半X（k），有

$$X(k) = X1(k - N/2) + W_N^k X2(k - N/2)$$

$$= X1(k) - W_N^k X2(k)$$

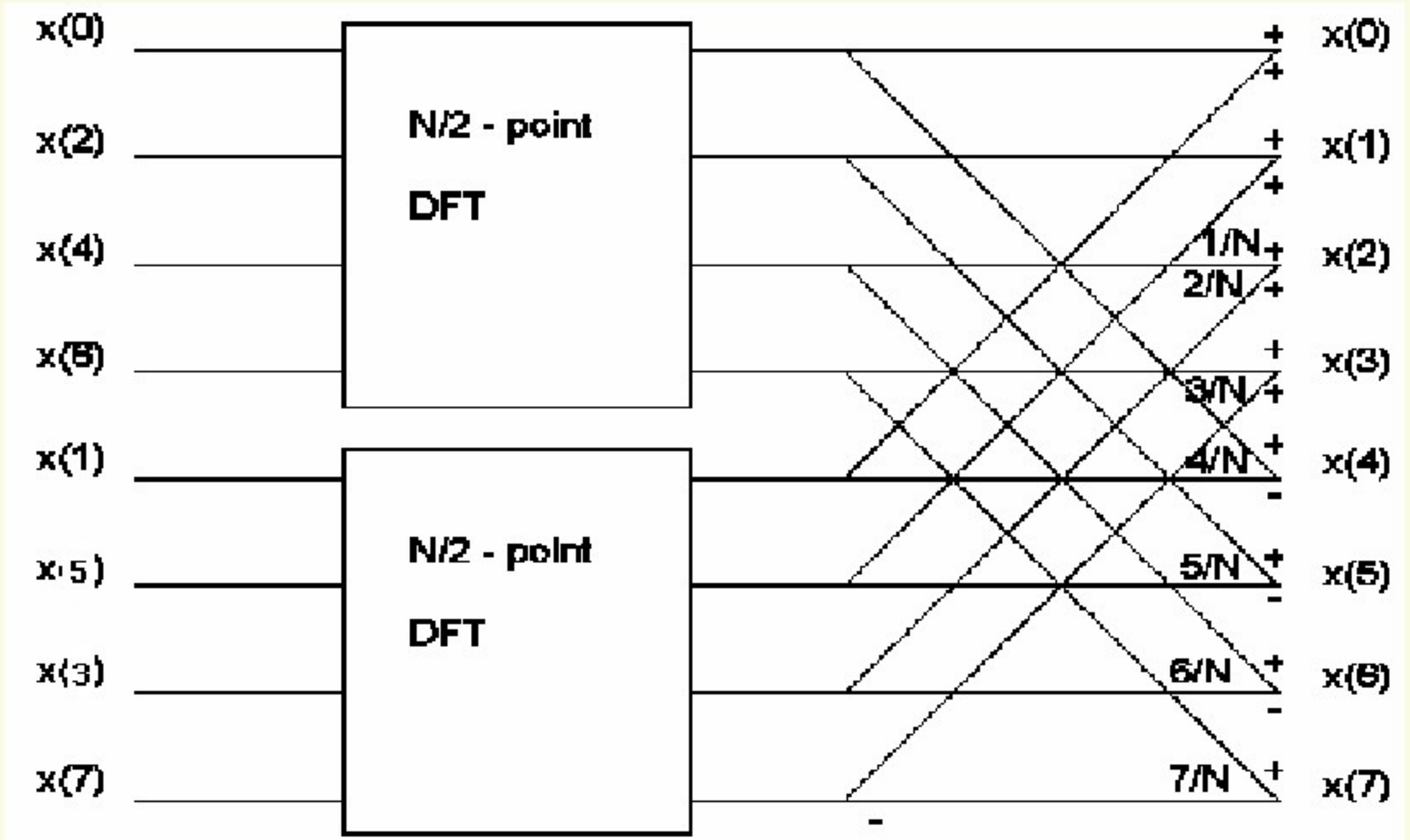
这是因为

$$W_N^{(N/2)+k} = W_N^{(N/2)} \cdot W_N^k = -W_N^k$$

将N点DFT分为两个N/2点DFT

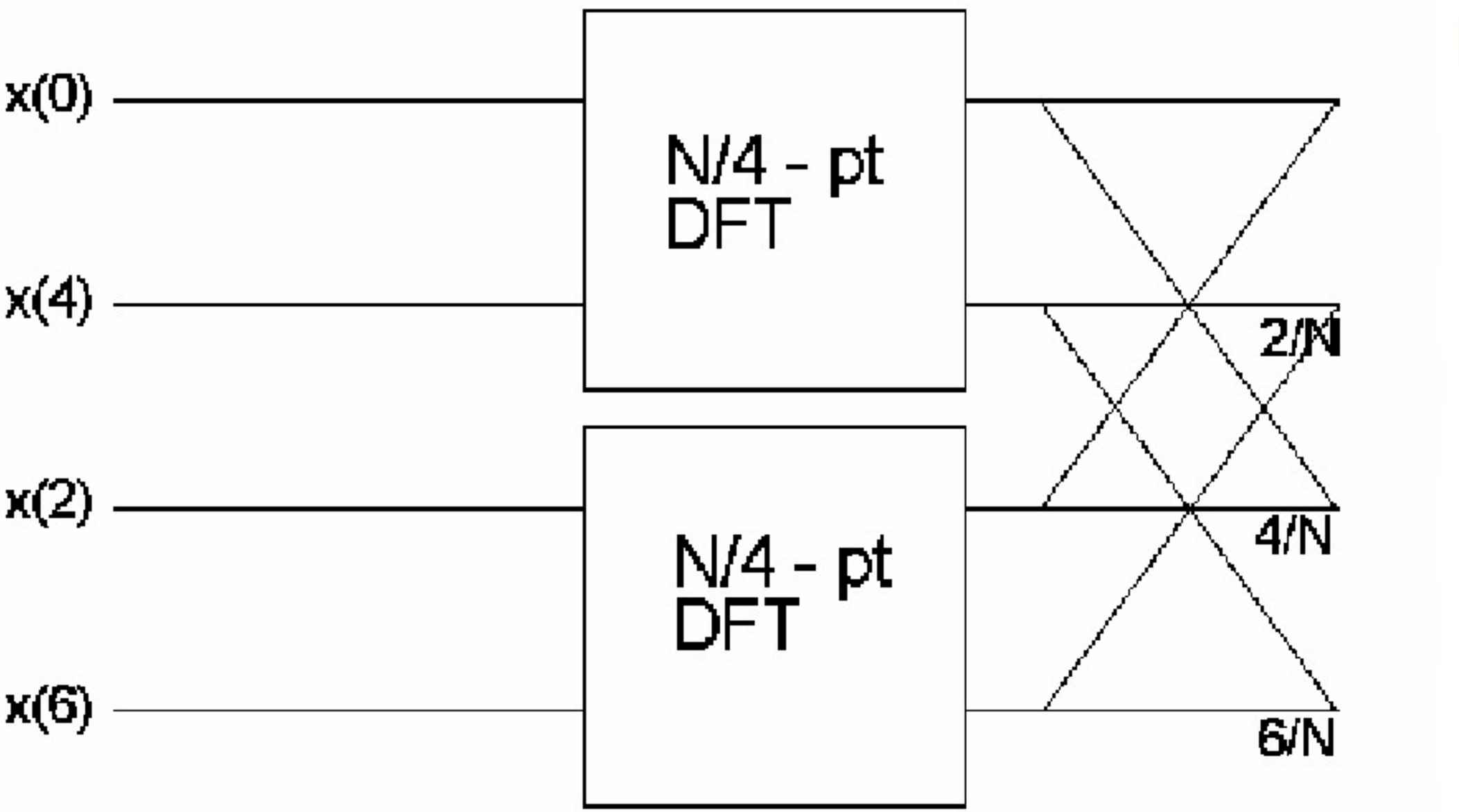
这样，只要计算出（0，N/2-1）区间的X1（k）和X2（k），也就可以很方便地计算整个（0，N-1）区间的全部X（k），从而大大地节省了运算量。

将N点DFT分为两个N/2点DFT



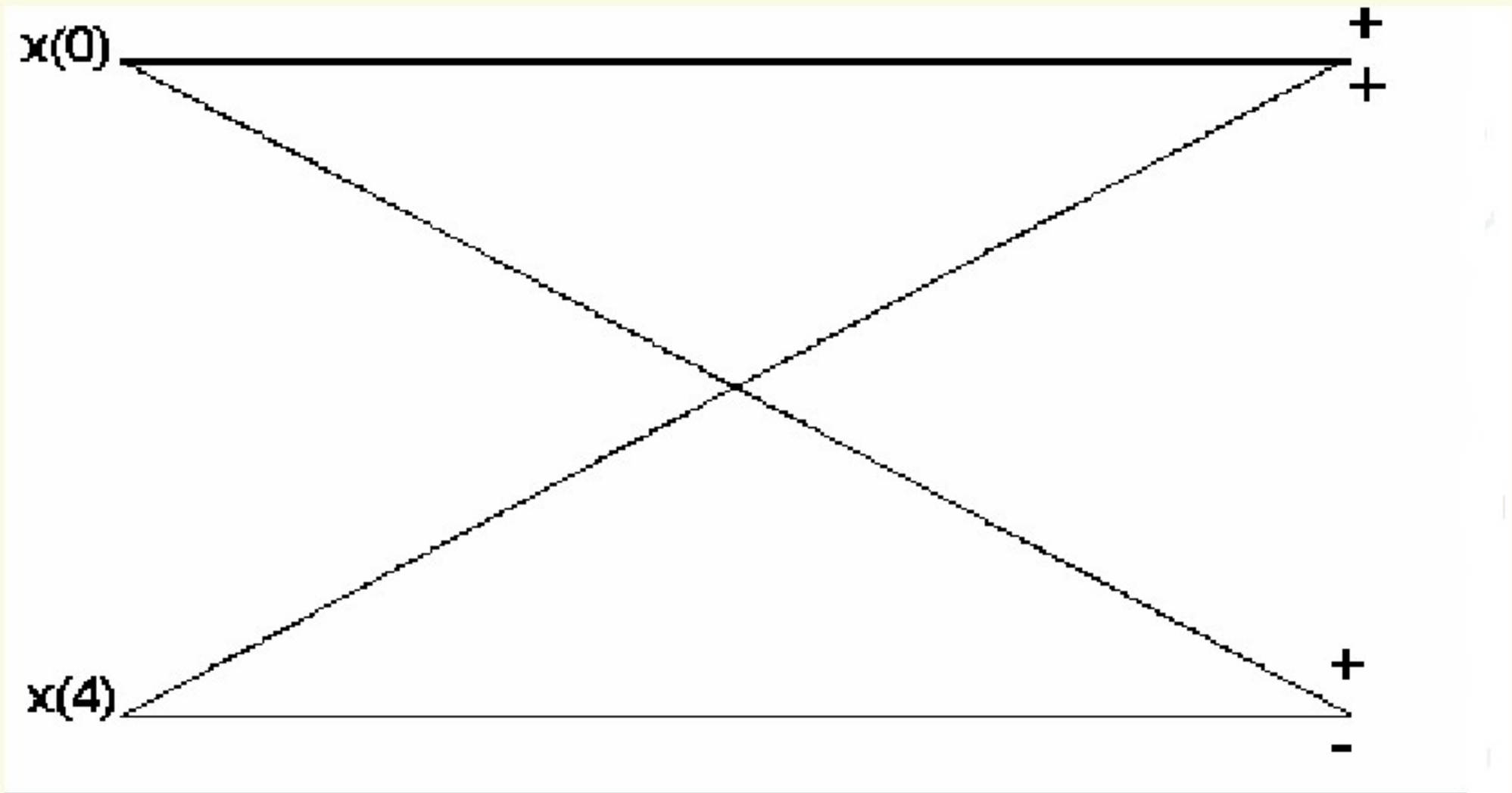
Decimation-in-Time of an N-Point

将N/2点DFT分为两个N/4点DFT



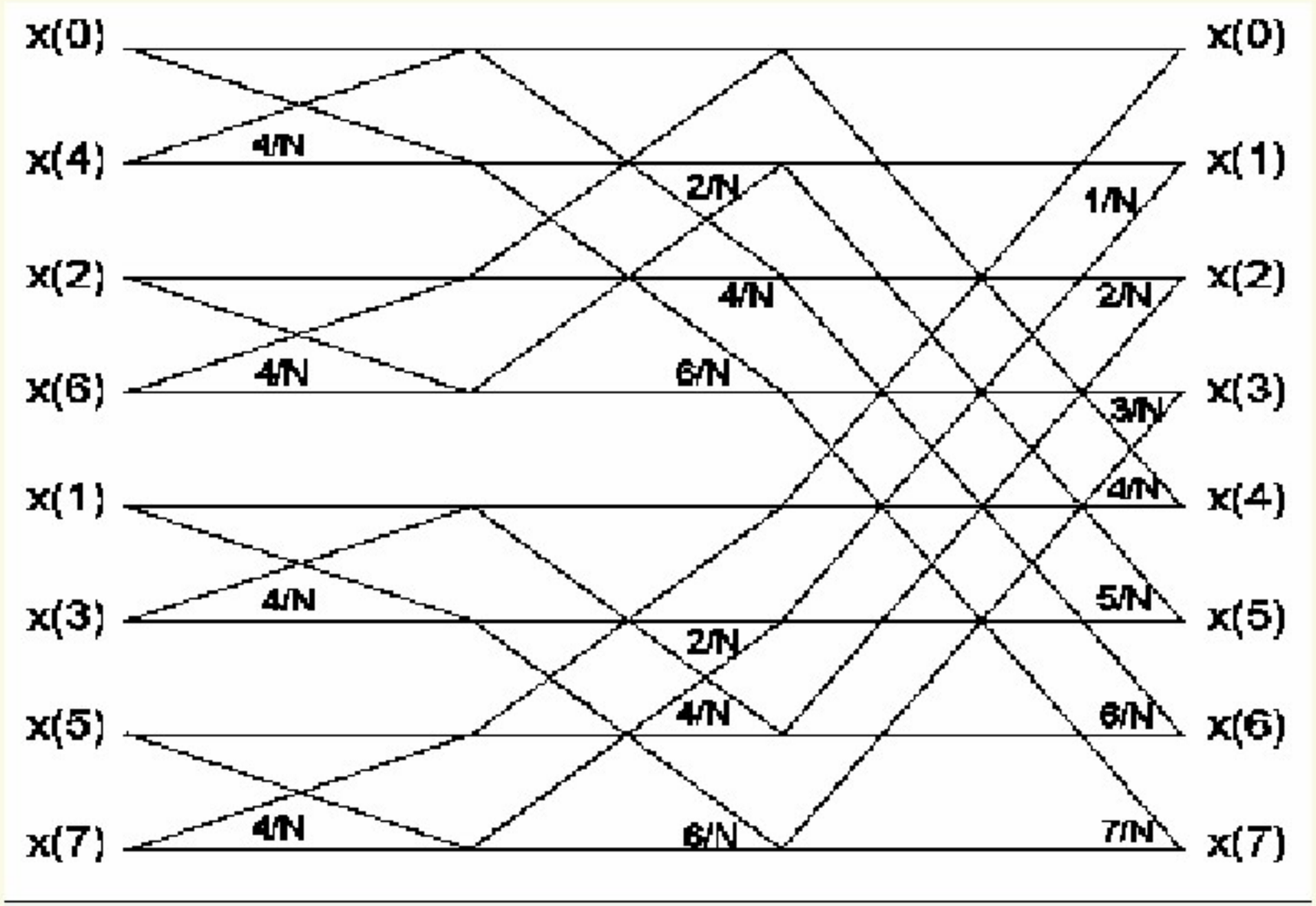
Decimation-in-Time FFT: Step Two

2点DFT



Decimation-in-Time FFT: Final Step
(2-Point DFT)

8点蝶2FFT



An 8-point, radix-2, Decimation-in-Time FFT

按频率抽取（DIF）

$$x1[n] = x[n]$$

$$x2[n] = x[n+N/2]$$

$$n=0，\ 1，\ \ \ldots\ldots，\ N/2-1$$

按频率抽取（DIF）

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

$$=\sum_{n=0}^{N/2-1}(x1[n]+e^{-j\pi k}x2[n])W_N^{nk}$$

按频率抽取（DIF）

$$X(2k)=\sum_{n=0}^{N/2-1}(x1[n]+x2[n])(W_N^2)^{nk}$$

$$=\sum_{n=0}^{N/2-1}(x1[n]+x2[n])W_{N/2}^{nk}$$

$$X(2k+1)=\sum_{n=0}^{N/2-1}(x1[n]-x2[n])W_N^{n(2k+1)}$$

$$=\sum_{n=0}^{N/2-1}W_N^n(x1[n]-x2[n])W_{N/2}^{nk}$$

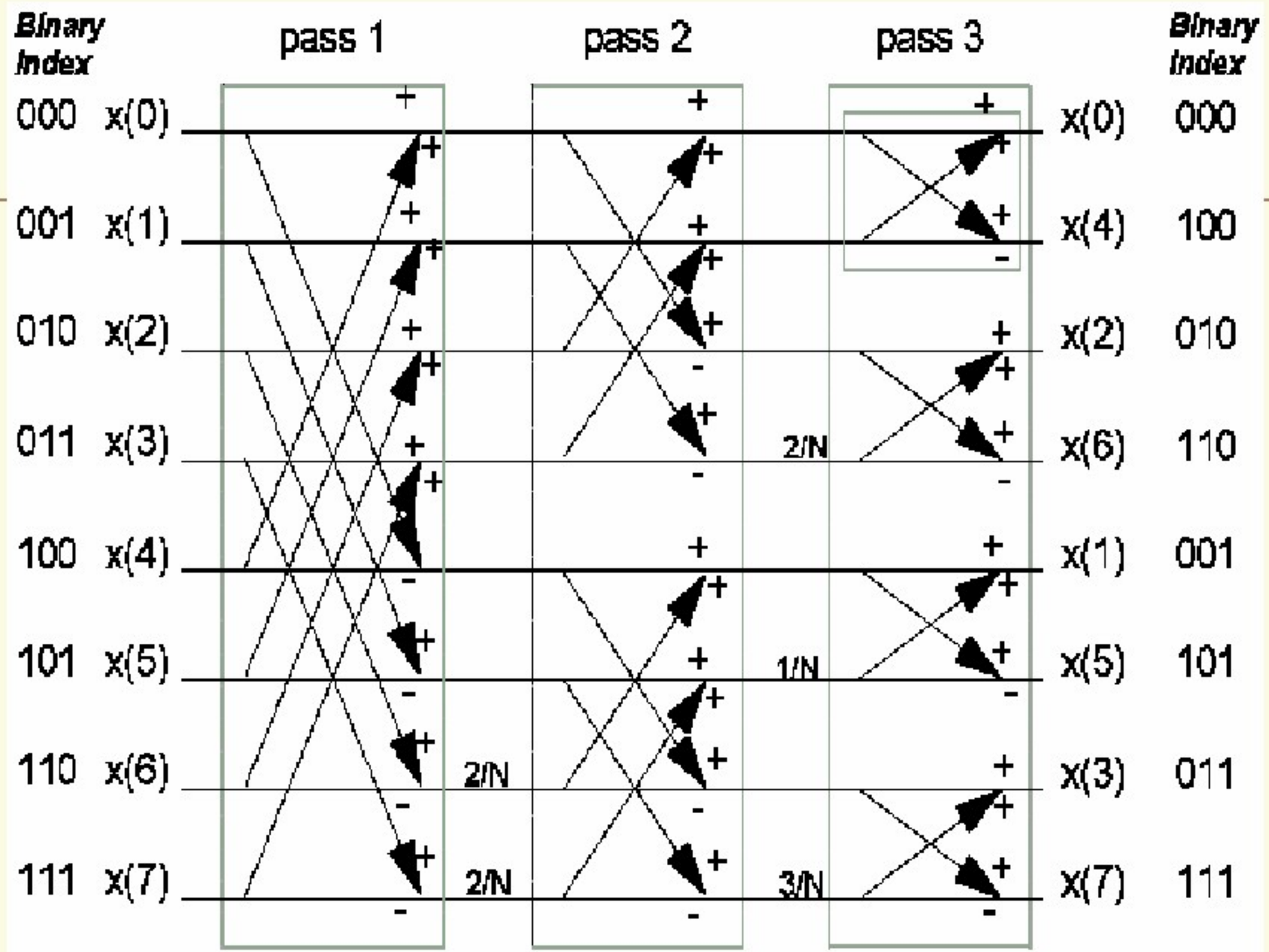
DIT与DIF

DIT输入是混序的，频域的输出是顺序的

DIF输入是顺序的，频域的输出是混序的

DIT的复数乘法出现在加减之前

DIF的复数乘法出现在加减之后



Grouping of Butterflies in the FFT Calculation

定点DSP计算1024点FFT的时间

DSP	时间（ms）
TMS320C25	10.9
TMS320C6201	0.067
DSP56001	1.65

浮点DSP计算1024点FFT的时间

DSP	时间（ms）
TMS320C30	3.87
TMS320C40	1.02
ADSP21060	0.46
DSP96001	0.6
ADSP21160	0.45μs

顺序、混序与位倒序

- DIT与DIF总有一边是混序的
- 绝大多数DSP都提供了位倒序（bit reverse）寻址指令

溢出问题

在多级运算中，要充分注意溢出的问题，尤其是用定点DSP时

无论是C语言，还是DSP汇编语言的FFT程序都有现成的程序可用