

Re-ordering the input data

We realize that the input sequence has to be reordered for FFT.

The secret is to think in binary terms. Table 3.2 shows in the first column the required ordering of the data for input to the butterfly network as given by figure 3.5 of our book.

Each value is assumed to be stored in a binary memory address. These addresses are given in the second column. The third column shows these memory addresses bit reversed. If these bit-reversed addresses are taken to correspond to the binary addresses of the original data sequence, commencing with $x(0)$ at 000, then the corresponding data values are given in the fourth column, which is seen to contain the original data sequence. Thus the addresses of the re-ordered data are seen to be the bit-reversed address of the original data sequence. The program is therefore required to convert the data point numbers (0, to N-1) to binary, to bit-reverse these binary numbers and to convert them back to the denary numbers which are the addresses of the re-ordered data.

Table 3.2 Sequence re-ordering by bit reversal.

Required sequence for Butterfly computation	Binary address of Required sequence data	Bit-Reversed address	Corresponding Sequence=original Data sequence
x_0	000	000	x_0
x_4	100	001	x_1
x_2	010	010	x_2
x_6	110	011	x_3
x_1	001	100	x_4
x_5	101	101	x_5
x_3	011	110	x_6
x_7	111	111	x_7

Computational Advantage of the FFT

The computational advantages of the FFT may be illustrated by considering first the FFT algorithm of figure 3.5. This figure shows that an N -point FFT contains $N/2$ butterflies per stage and $\log_2 N$ stages, that is it contains a total of $(N/2)\log_2 N$ butterflies. We know that each butterfly requires one complex multiplication hence the FFT requires $(N/2)\log_2 N$ **complex multiplications**.

Compared to N^2 in the case of DFT there is a computational saving of $N^2 - (N/2)\log_2 N$.

Also ;

Each butterfly requires two complex additions so the FFT requires $N \log_2 N$ **complex additions** compared with $N(N-1)$ for the DFT.

Thus the saving in complex additions is $N(N-1) - N \log_2 N$.

N	DFT		FFT		Ratio of DFT multiplications To FFT multiplications	Ratio of DFT additions To FFT additions
	Number of Complex Multiplications	Number of Complex Additions	Number of Complex Multiplications	Number of Complex Additions		
2	4	2	1	2	4	1
8	64	56	12	24	5.3	2.3
32	1024	992	80	160	12.8	6.2
128	16384	65280	1024	2048	64	31.9
512	262144	261632	2304	4608	113.8	56.8
1024	1048576	1047552	5120	10240	204.8	102.3