Efficient Computation of DFT
Fast Fourier Transform

Phan Duy Hùng

Fast Fourier Transform ?

Divide-and-Conquer Approach to Computation of the DFT

$n \rightarrow$	0	1		…	$N-1$
	$x(0)$	$x(1)$	$x(2)$	…	$x(N-1)$

(a)

row index l \ m column index	0	1		$M-1$
0	$x(0, 0)$	$x(0, 1)$	…	
1	$x(1, 0)$	$x(1, 1)$	…	
2	$x(2, 0)$	$x(2, 1)$	…	
⋮	⋮	⋮	…	⋮
$L-1$			…	

(b)

Figure 8.1.1 Two dimensional data array for storing the sequence $x(n)$, $0 \leq n \leq N-1$.

Row-wise $n = Ml + m$

$l \diagdown m$	0	1	2		$M-1$
0	$x(0)$	$x(1)$	$x(2)$	$\dots$	$x(M-1)$
1	$x(M)$	$x(M+1)$	$x(M+2)$	$\dots$	$x(2M-1)$
2	$x(2M)$	$x(2M+1)$	$x(2M+2)$	$\dots$	$x(3M-1)$
	$\vdots$	$\vdots$	$\vdots$	$\dots$	$\vdots$
$L-1$	$x((L-1)M)$	$x((L-1)M+1)$	$x((L-1)M+2)$	$\dots$	$x(LM-1)$

(a)

Column-wise $n = l + mL$

$l \diagdown m$	0	1	2		$M-1$
0	$x(0)$	$x(L)$	$x(2L)$	$\dots$	$x((M-1)L)$
1	$x(1)$	$x(L+1)$	$x(2L+1)$	$\dots$	$x((M-1)L+1)$
2	$x(2)$	$x(L+2)$	$x(2L+2)$	$\dots$	$x((M-1)L+2)$
	$\vdots$	$\vdots$	$\vdots$	$\dots$	$\vdots$
$L-1$	$x(L-1)$	$x(2L-1)$	$x(3L-1)$	$\dots$	$x(LM-1)$

(b)

Figure 8.1.2 Two arrangements for the data arrays.

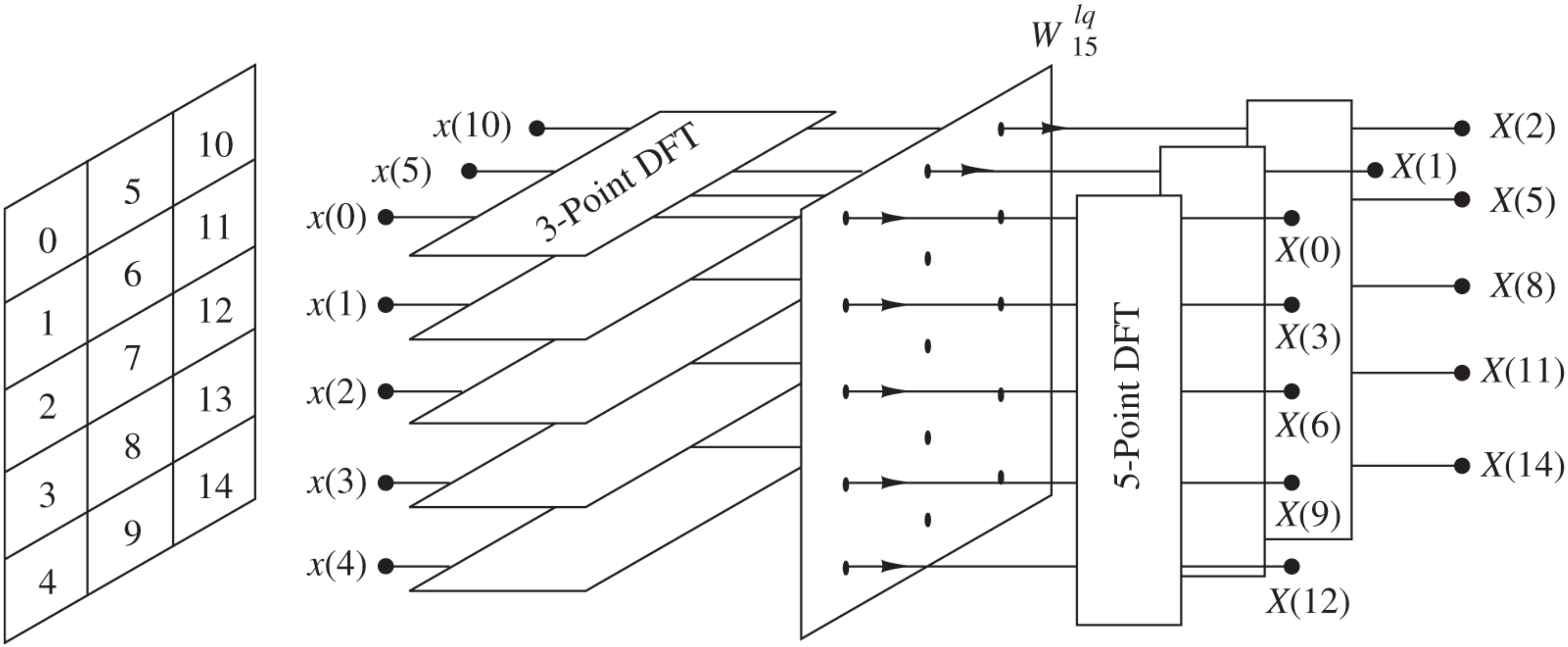


Figure 8.1.3 Computation of $N = 15$-point DFT by means of 3-point and 5-point DFTs.

Radix-2 FFT Algorithms

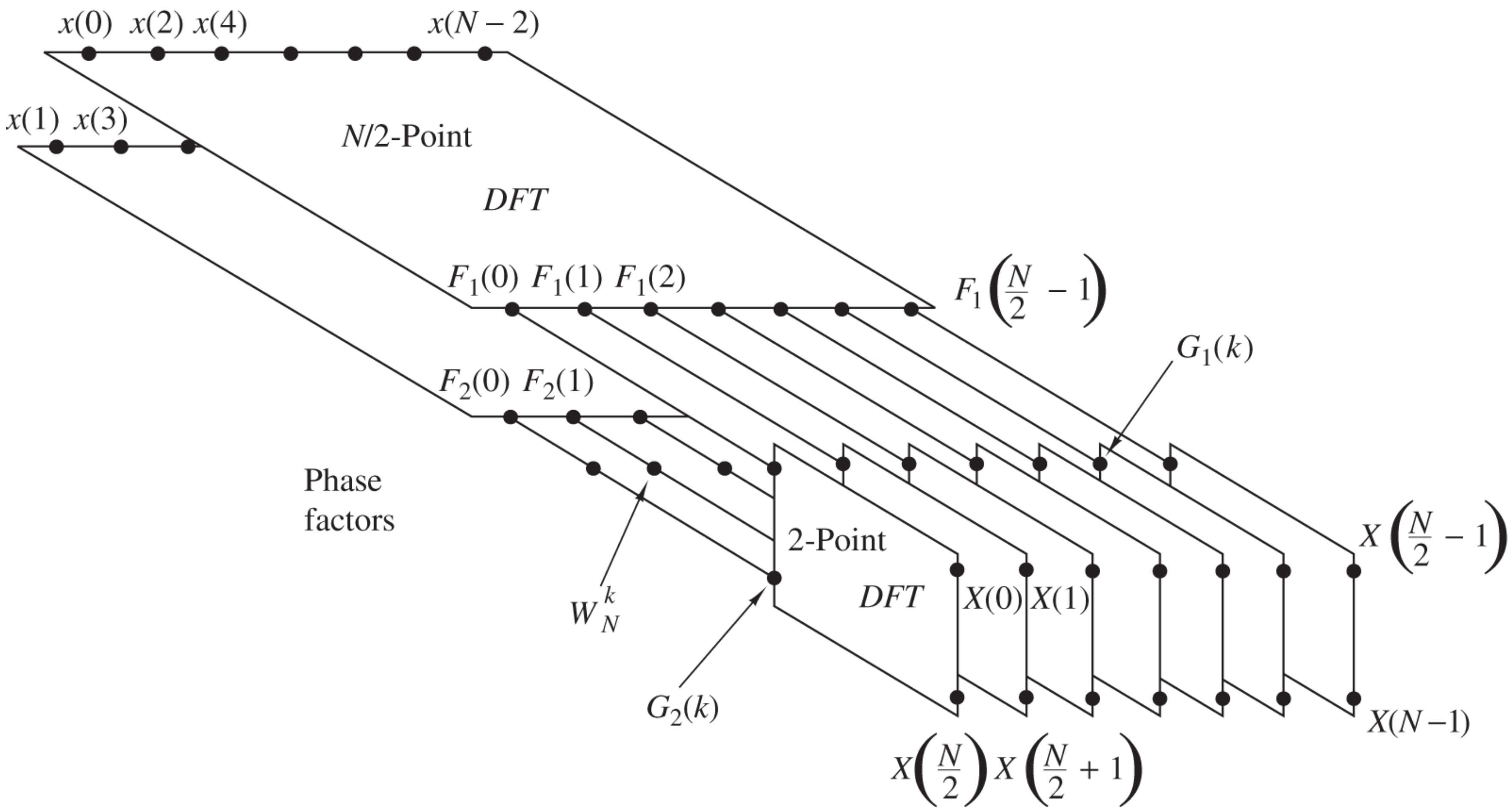

$x(0)$ $x(2)$ $x(4)$ $x(N-2)$
$x(1)$ $x(3)$
$N/2$-Point
DFT
$F_1(0)$ $F_1(1)$ $F_1(2)$
$F_1\left(\frac{N}{2}-1\right)$
$G_1(k)$
$F_2(0)$ $F_2(1)$
Phase
factors
W_N^k
2-Point
DFT
$G_2(k)$
$X(0)$ $X(1)$
$X\left(\frac{N}{2}-1\right)$
$X(N-1)$
$X\left(\frac{N}{2}\right)$ $X\left(\frac{N}{2}+1\right)$

Figure 8.1.4 First step in the decimation-in-time algorithm.

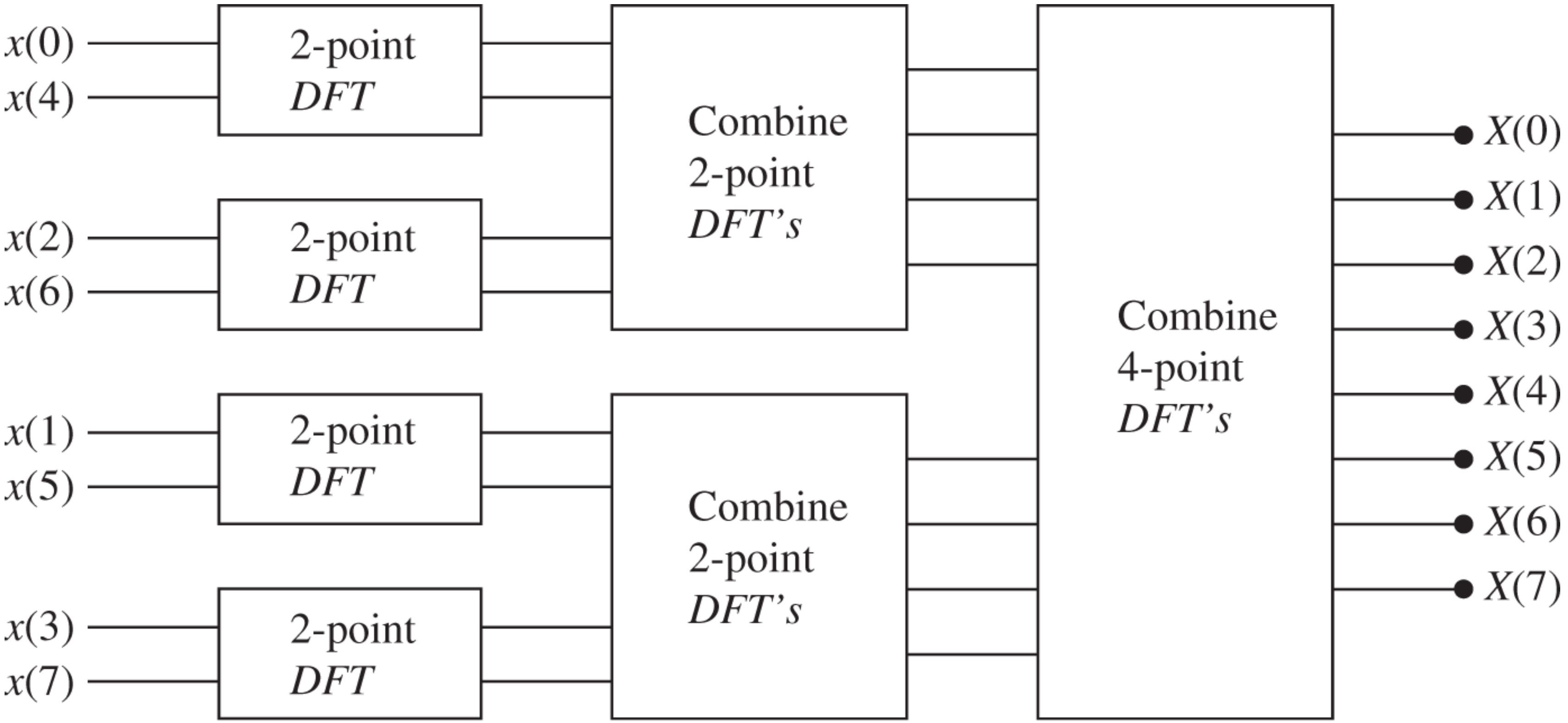


Figure 8.1.5 Three stages in the computation of an $N = 8$-point DFT.

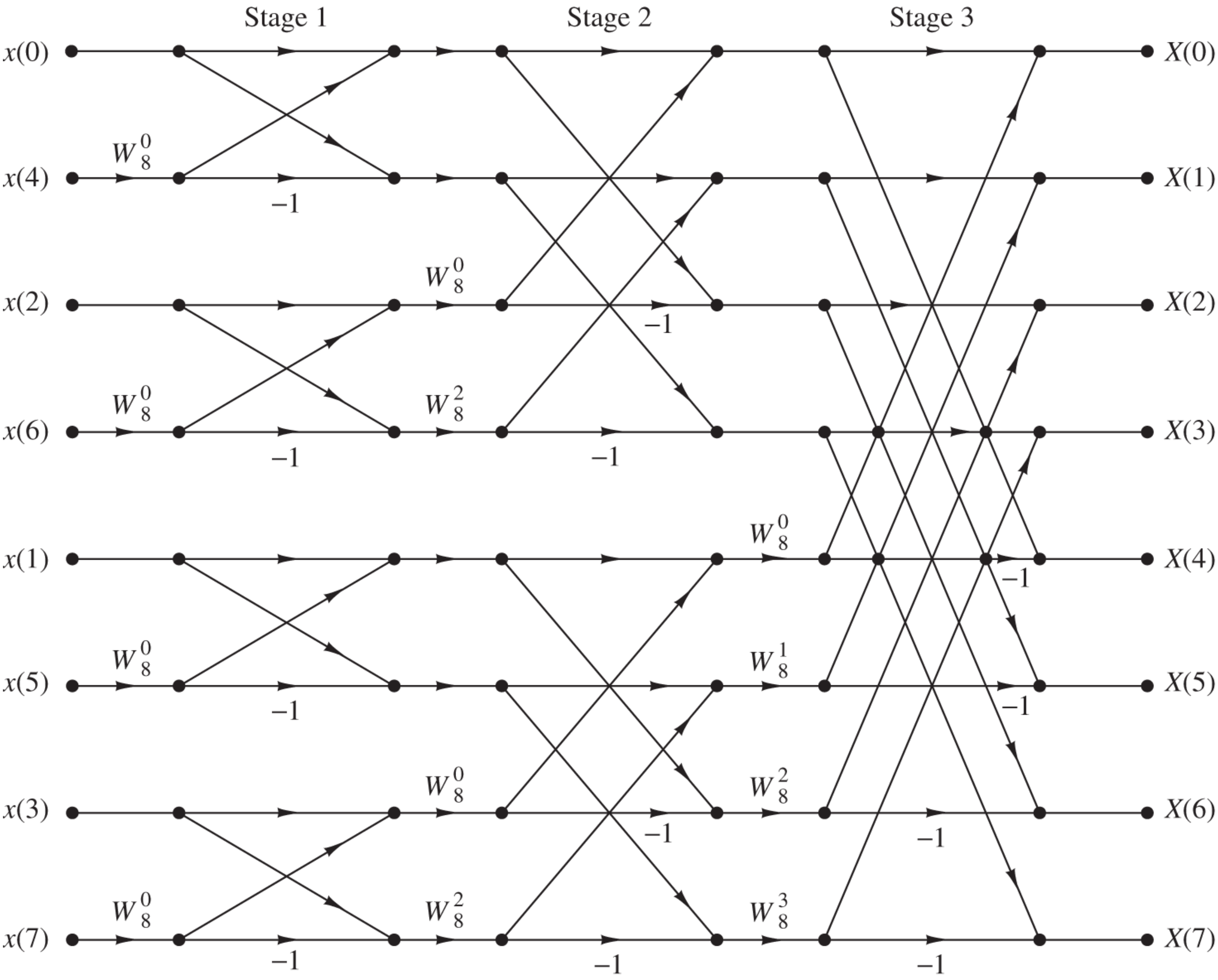


Figure 8.1.6 Eight-point decimation-in-time FFT algorithm.

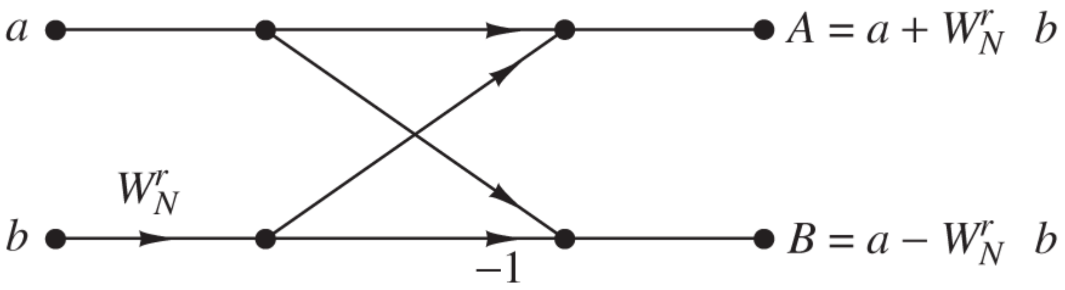
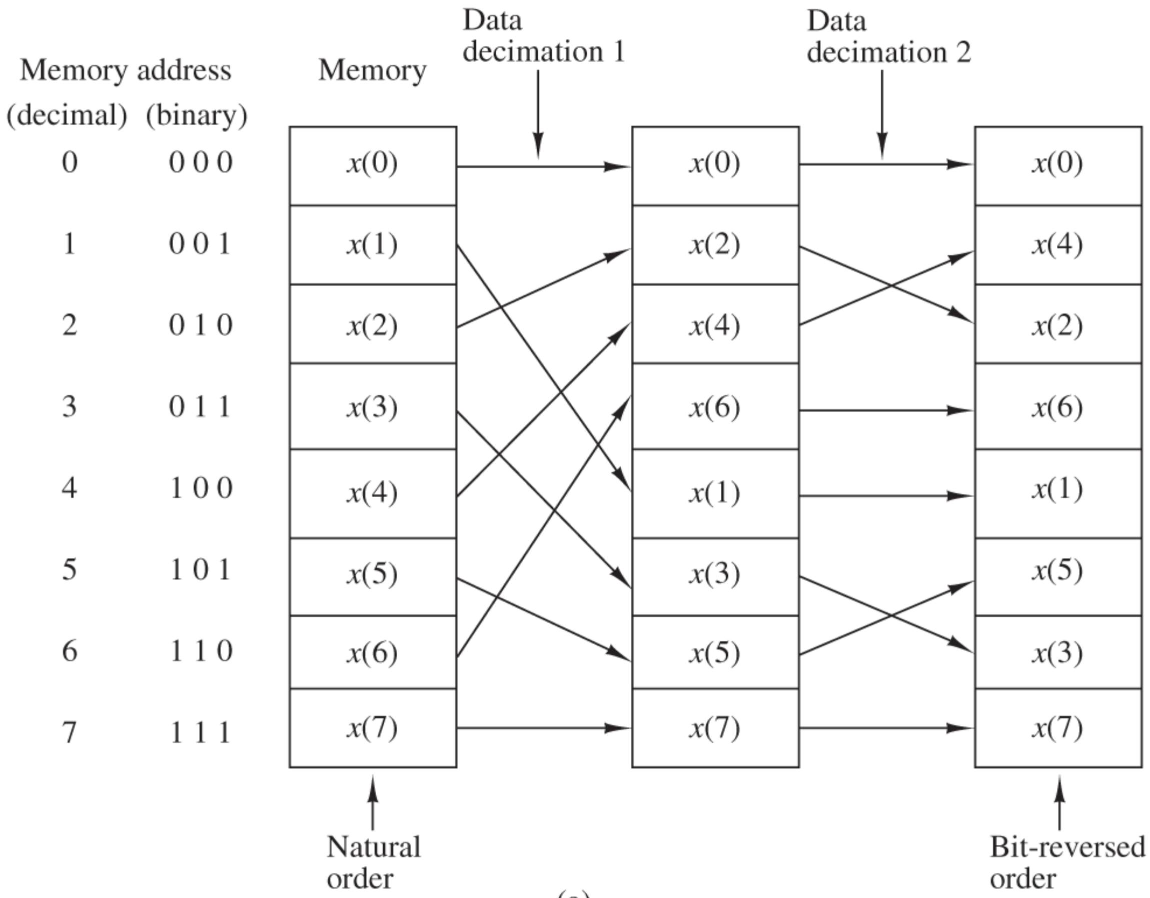


Figure 8.1.7 Basic butterfly computation in the decimation-in-time FFT algorithm.



(a)

$(n_2n_1n_0)$	→	$(n_1n_0n_2)$	→	$(n_0n_1n_2)$
(0 0 0)	→	(0 0 0)	→	(0 0 0)
(0 0 1)	→	(0 1 0)	→	(1 0 0)
(0 1 0)	→	(1 0 0)	→	(0 1 0)
(0 1 1)	→	(1 1 0)	→	(1 1 0)
(1 0 0)	→	(0 0 1)	→	(0 0 1)
(1 0 1)	→	(0 1 1)	→	(1 0 1)
(1 1 0)	→	(1 0 1)	→	(0 1 1)
(1 1 1)	→	(1 1 1)	→	(1 1 1)

(b)

Figure 8.1.8 Shuffling of the data and bit reversal.

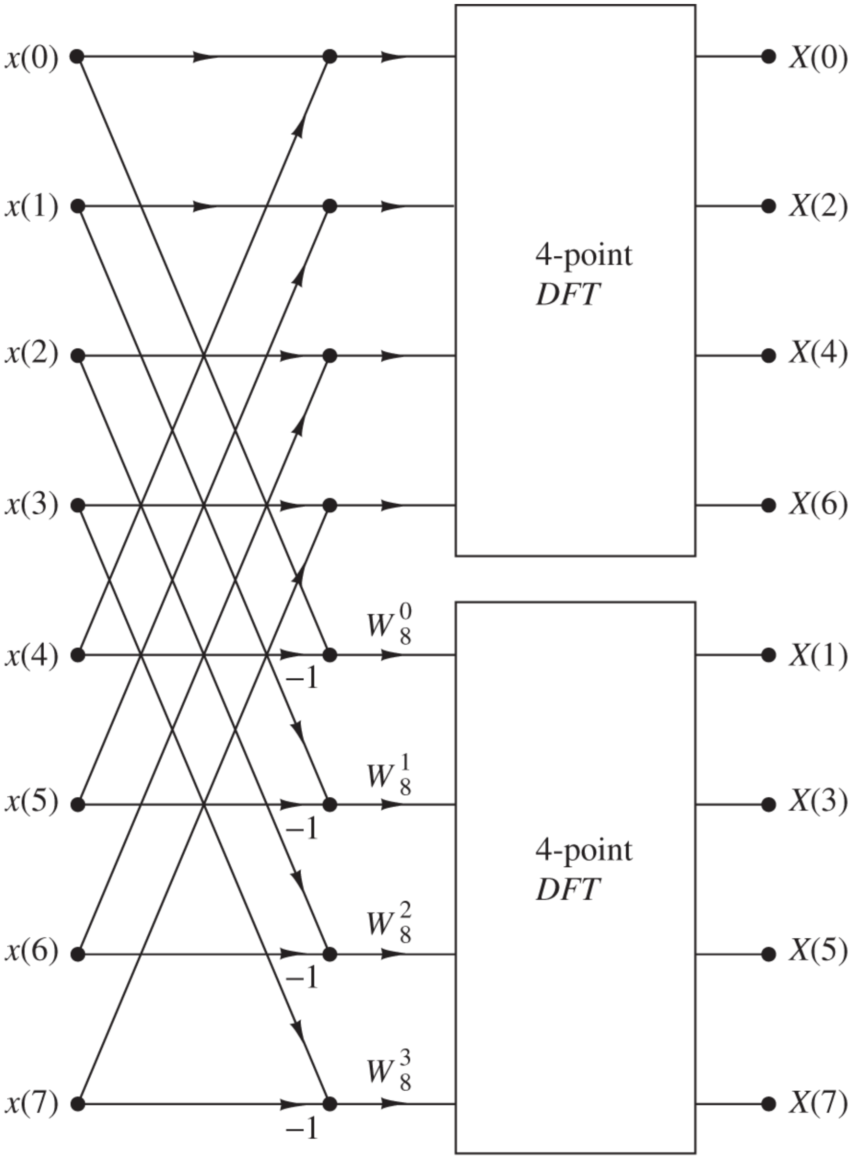


Figure 8.1.9 First stage of the decimation-in-frequency FFT algorithm.

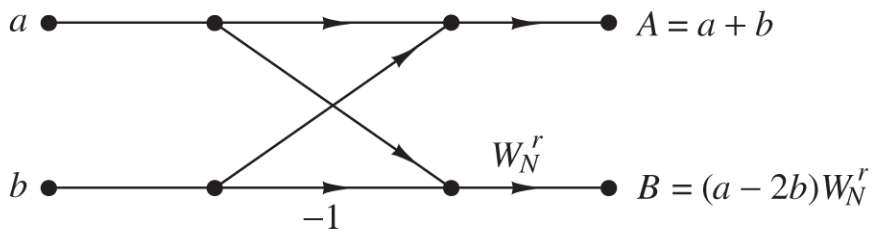


Figure 8.1.10 Basic butterfly computation in the decimation-in-frequency FFT algorithm.

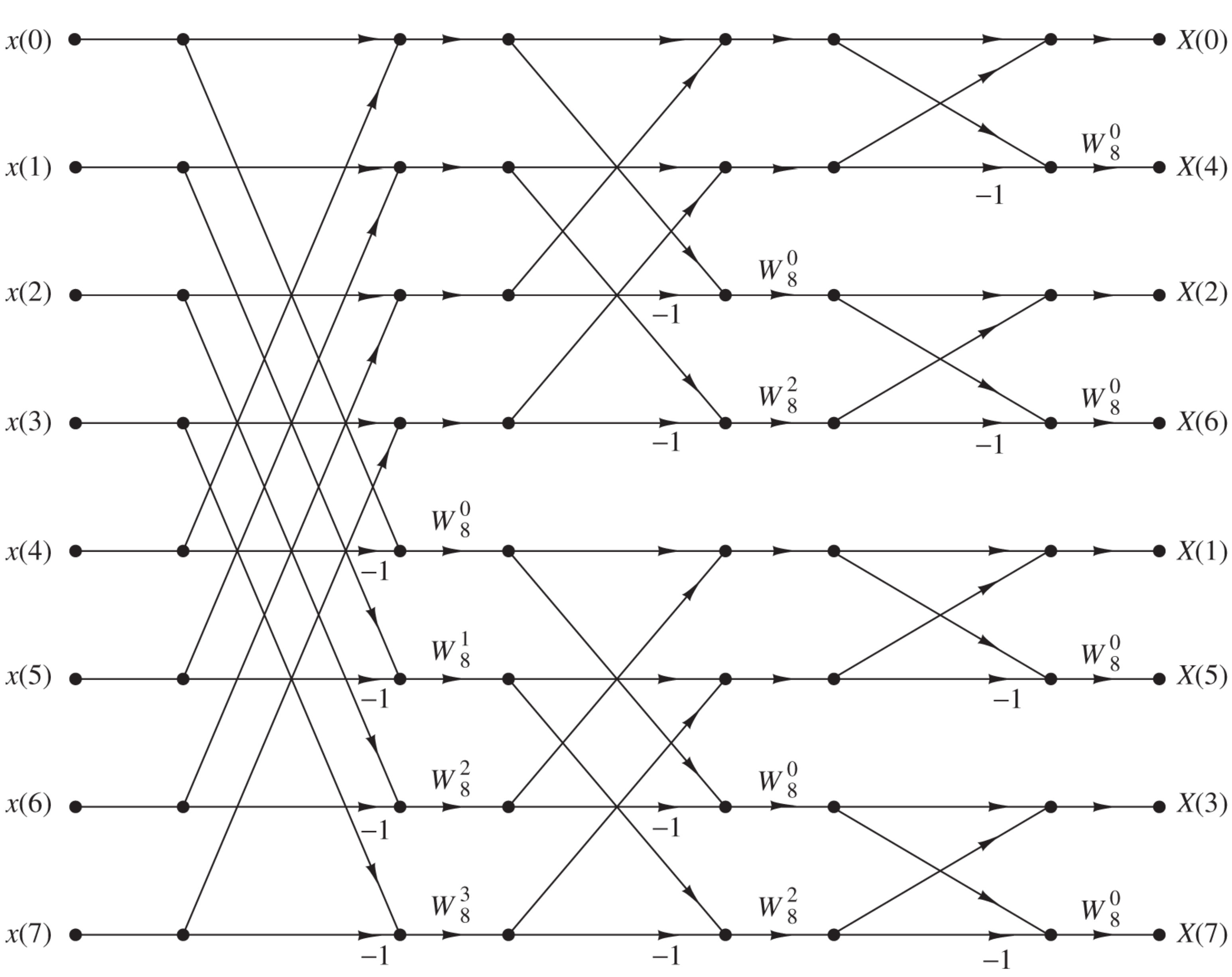


Figure 8.1.11 $N = 8$-point decimation-in-frequency FFT algorithm.

Implementation of FFT Algorithms ?

Applications of FFT Algorithms

- Efficient computation of the DFT of two real sequences
- Efficient computation of the DFT of a 2N point real sequence
- Use of FFT Algorithm in Linear Filtering and Correlation