# Fast Fourier Transform

# Discrete Fourier Transform

- DFT of discrete-time signal $x[n]$ of length $N$

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}kn} \qquad k = 0,\ 1, \ldots, N-1$$

- Let, k= 0

- Assuming $x[n]$ be complex
  - **Complex multiplications: N per DFT value**
  - **Complex addition: (N-1) per DFT value**

## Discrete Fourier Transform

- DFT of discrete-time signal $x[n]$ of length $N$

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}kn} \qquad k = 0,\ 1,\dots,\ N-1$$

- Let, k= 0

$$X[0] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{4}0n}$$

- Assuming $x[n]$ be complex
  - **Complex multiplications: N per DFT value**
  - **Complex addition: (N-1) per DFT value**

## Discrete Fourier Transform

- DFT of discrete-time signal $x[n]$ of length $N$

$$X[k]=\sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}k\,n} \qquad k=0,\ 1,\ldots,\ N-1$$

- Let, k= 0

$$X[0]=\sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{4}0\,n}$$

$$= x[0]\, e^{-j\frac{2\pi}{4}0\,0} + x[1]\, e^{-j\frac{2\pi}{4}0\,1} + \ldots + x[N-1]\, e^{-j\frac{2\pi}{4}0\,(N-1)}$$

- Assuming $x[n]$ be complex
  - **Complex multiplications: N per DFT value**
  - **Complex addition: (N-1) per DFT value**

## Discrete Fourier Transform

- DFT of discrete-time signal $x[n]$ of length $N$

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}kn} \qquad k = 0,\ 1, \ldots, N-1$$

- Let, k= 0

$$X[0] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{4}0n}$$

$$= x[0]\, e^{-j\frac{2\pi}{4}00} + x[1]\, e^{-j\frac{2\pi}{4}01} + \ldots + x[N-1]\, e^{-j\frac{2\pi}{4}0(N-1)}$$

- Assuming $x[n]$ be complex
  - **Complex multiplications: N per DFT value**
  - **Complex addition: (N-1) per DFT value**
- **For N-point DFT $N^2$ complex multiplications**

  **$N(N-1)$ complex additions**

# Discrete Fourier Transform

## Discrete Fourier Transform

- Separating real and imaginary parts

- Assuming $x[n]$ be complex
  - **Real multiplications: 4N per DFT value**
  - **Real addition: (4N-2) per DFT value**

# Discrete Fourier Transform

- Separating real and imaginary parts

$$X[k] = \sum_{n=0}^{N-1} \mathrm{Re}\{x[n]\}.\cos\left(\frac{2\pi}{4}k\,n\right) - \mathrm{Im}\{x[n]\}.\sin\left(\frac{2\pi}{4}k\,n\right)$$
$$+\,j\left(\mathrm{Re}\{x[n]\}.\sin\left(\frac{2\pi}{4}k\,n\right) - \mathrm{Im}\{x[n]\}.\cos\left(\frac{2\pi}{4}k\,n\right)\right)$$

- Assuming $x[n]$ be complex
    - **Real multiplications: 4N per DFT value**
    - **Real addition: (4N-2) per DFT value**

# Discrete Fourier Transform

- Separating real and imaginary parts

$$X[k] = \sum_{n=0}^{N-1} \mathrm{Re}\{x[n]\}.\cos\left(\frac{2\pi}{4}k\,n\right) - \mathrm{Im}\{x[n]\}.\sin\left(\frac{2\pi}{4}k\,n\right)$$
$$+\, j\left(\mathrm{Re}\{x[n]\}.\sin\left(\frac{2\pi}{4}k\,n\right) - \mathrm{Im}\{x[n]\}.\cos\left(\frac{2\pi}{4}k\,n\right)\right)$$

- Assuming $x[n]$ be complex
    - **Real multiplications: 4N per DFT value**
    - **Real addition: (4N-2) per DFT value**

# Discrete Fourier Transform

- Separating real and imaginary parts

$$X[k] = \sum_{n=0}^{N-1} \mathrm{Re}\{x[n]\}.\cos\left(\frac{2\pi}{4}kn\right) - \mathrm{Im}\{x[n]\}.\sin\left(\frac{2\pi}{4}kn\right)$$
$$+ j\left(\mathrm{Re}\{x[n]\}.\sin\left(\frac{2\pi}{4}kn\right) - \mathrm{Im}\{x[n]\}.\cos\left(\frac{2\pi}{4}kn\right)\right)$$

- Assuming $x[n]$ be complex
  - **Real multiplications: 4N per DFT value**
  - **Real addition: (4N-2) per DFT value**

- **For N-point DFT $4N^2$ real multiplications**
  **$N(4N-2)$ real additions**

# Fast Fourier Transform

# Fast Fourier Transform

- **Fast Fourier Transform (FFT) is an algorithm** that efficiently computes the Discrete Fourier Transform (DFT).
- Computational algorithms that exploit both the **symmetry and the periodicity of the sequence $W_N^{nk}$**.
- Runge (1905) and later Danielson and Lanczos (1942) described algorithms for which computation was roughly proportional to $N\log N$ **rather than** $N^2$.
    - Distinction was **not of great importance for the small values of *N***.
- **Cooley and Tukey (1965)** published an algorithm for the computation of the discrete Fourier transform that is applicable when ***N* is a composite number**, i.e., the product of two or more integers.
    - James w. Cooley, Peter A. W. Lewis, and Peter D. WELCH, “Historical Notes on the Fast Fourier Transform”, *Proceedings IEEE. vol. 55. no. 10., October 1967*.

# Fast Fourier Transform

- **Fast Fourier Transform (FFT) is an algorithm** that efficiently computes the Discrete Fourier Transform (DFT).
- Computational algorithms that exploit both the **symmetry and the periodicity of the sequence $W_N^{nk}$**.
- Runge (1905) and later Danielson and Lanczos (1942) described algorithms for which computation was roughly proportional to $N\log N$ **rather than** $N^2$.
  - Distinction was **not of great importance for the small values of *N*.**
- **Cooley and Tukey (1965)** published an algorithm for the computation of the discrete Fourier transform that is applicable when ***N* is a composite number**, i.e., the product of two or more integers.
  - James w. Cooley, Peter A. W. Lewis, and Peter D. WELCH, "Historical Notes on the Fast Fourier Transform", *Proceedings IEEE. vol. 55. no. 10., October 1967.*

# Fast Fourier Transform

- **Fast Fourier Transform (FFT) is an algorithm** that efficiently computes the Discrete Fourier Transform (DFT).
- Computational algorithms that exploit both the **symmetry and the periodicity of the sequence $W_N^{nk}$**.
- Runge (1905) and later Danielson and Lanczos (1942) described algorithms for which computation was roughly proportional to $N\log N$ **rather than** $N^2$.
  - Distinction was **not of great importance for the small values of *N*.**
- **Cooley and Tukey (1965)** published an algorithm for the computation of the discrete Fourier transform that is applicable when ***N* is a composite number**, i.e., the product of two or more integers.
  - James w. Cooley, Peter A. W. Lewis, and Peter D. WELCH, "Historical Notes on the Fast Fourier Transform", *Proceedings IEEE. vol. 55. no. 10., October 1967.*

# Complexity of FFT algorithms

# Complexity of FFT algorithms

- The number of **arithmetic multiplications and additions as a measure** of computational complexity.

# Complexity of FFT algorithms

- The number of **arithmetic multiplications and additions as a measure** of computational complexity.
- Simple to apply and **directly related to the computational speed** when on general-purpose digital computers or special-purpose microprocessors

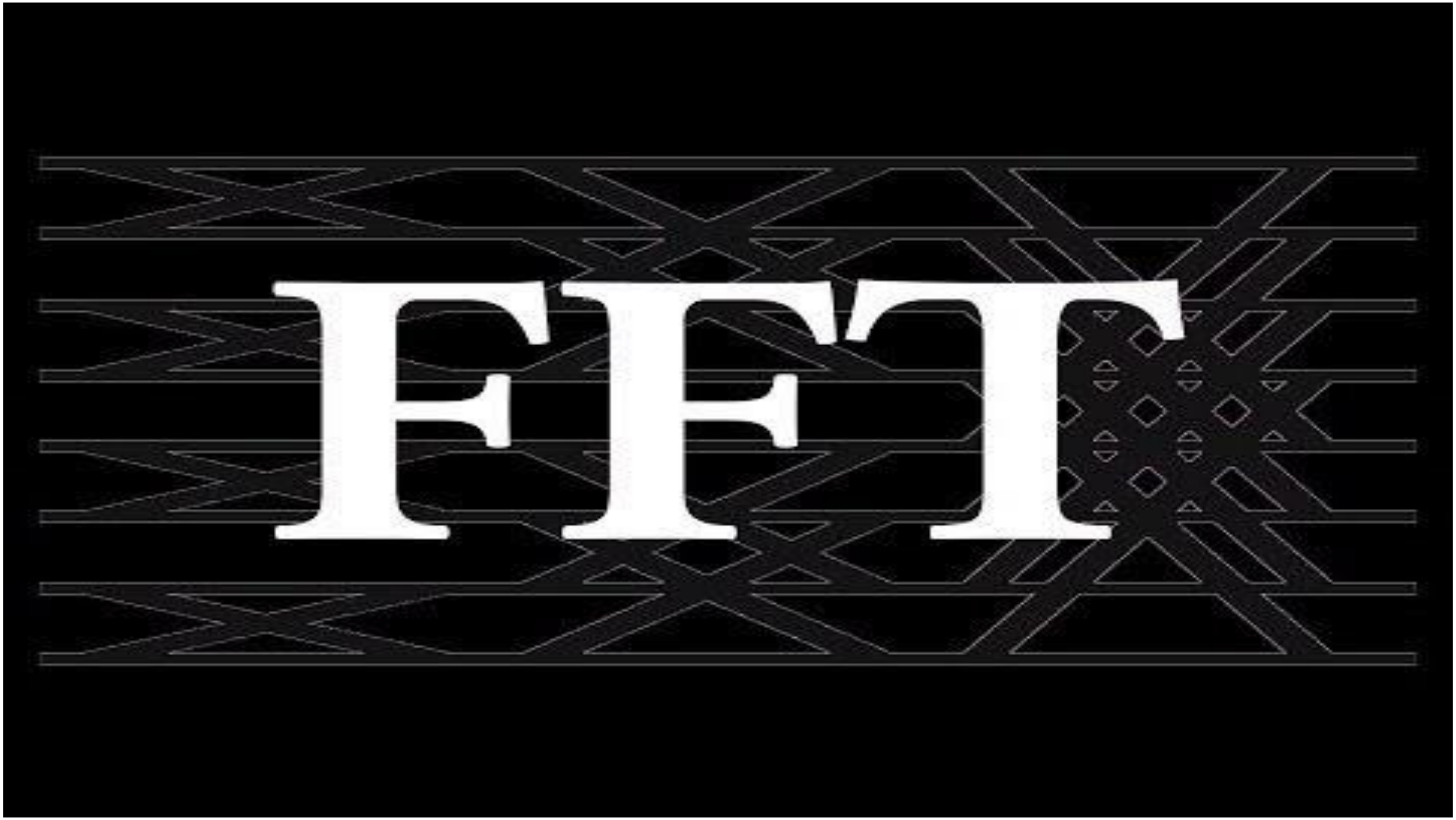
# Complexity of FFT algorithms

- The number of **arithmetic multiplications and additions as a measure** of computational complexity.
- Simple to apply and **directly related to the computational speed** when on general-purpose digital computers or special-purpose microprocessors
- In custom VLSI implementations, the **area of the chip and power requirements** are important considerations.

# Complexity of FFT algorithms

- The number of **arithmetic multiplications and additions as a measure** of computational complexity.
- Simple to apply and **directly related to the computational speed** when on general-purpose digital computers or special-purpose microprocessors
- In custom VLSI implementations, the **area of the chip and power requirements** are important considerations.
- The efficiency of FFT algorithms is so high, that in many cases the **most efficient** procedure for implementing **a convolution is through DFT and IDFT process**.

# FFT: Cooley-Tuckey algorithm



# FFT Butterfly

# FFT Butterfly

- Let $x[n] = \{x_0, \ x_1\}$ and corresponding DFT $X(k) = \{X_0, \ X_1\}$

## FFT Butterfly

- Let $x[n] = \{x_0,\ \ x_1\}$ and corresponding DFT $X(k) = \{X_0,\ \ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}k\,n} \qquad \text{...(Taking DFT)}$$

## FFT Butterfly

- Let $x[n] = \{x_0, \ x_1\}$ and corresponding DFT $X(k) = \{X_0, \ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}k\,n} \qquad \text{...(Taking DFT)}$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0\,n}$$

## FFT Butterfly

- Let $x[n] = \{x_0, \ x_1\}$ and corresponding DFT $X(k) = \{X_0, \ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}k\,n} \qquad \text{...(Taking DFT)}$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0\,n} = x[0]\, e^{-j\frac{2\pi}{2}0\,0} + x[1]\, e^{-j\frac{2\pi}{2}0\,1}$$

## FFT Butterfly

- Let $x[n] = \{x_0, \ x_1\}$ and corresponding DFT $X(k) = \{X_0, \ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}k\,n} \qquad \ldots(\text{Taking DFT})$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0\,n} = x[0]\, e^{-j\frac{2\pi}{2}0\,0} + x[1]\, e^{-j\frac{2\pi}{2}0\,1}$$

$$= x_0 + x_1$$

## FFT Butterfly

- Let $x[n] = \{x_0, \ x_1\}$ and corresponding DFT $X(k) = \{X_0, \ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}kn} \qquad \ldots(\text{Taking DFT})$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0n} = x[0]\, e^{-j\frac{2\pi}{2}0\,0} + x[1]\, e^{-j\frac{2\pi}{2}0\,1}$$

$$= x_0 + x_1$$

$$X[1] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}1n}$$

## FFT Butterfly

- Let $x[n] = \{x_0, \; x_1\}$ and corresponding DFT $X(k) = \{X_0, \; X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}kn} \qquad \text{...(Taking DFT)}$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0n} = x[0]\, e^{-j\frac{2\pi}{2}0\,0} + x[1]\, e^{-j\frac{2\pi}{2}0\,1}$$

$$= x_0 + x_1$$

$$X[1] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}1n} = x[0]\, e^{-j\frac{2\pi}{2}1\,0} + x[1]\, e^{-j\frac{2\pi}{2}1\,1}$$

## FFT Butterfly

- Let $x[n] = \{x_0,\ x_1\}$ and corresponding DFT $X(k) = \{X_0,\ X_1\}$

$$X[k] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}kn} \qquad \text{...(Taking DFT)}$$

$$X[0] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}0n} = x[0]\, e^{-j\frac{2\pi}{2}00} + x[1]\, e^{-j\frac{2\pi}{2}01}$$

$$= x_0 + x_1$$

$$X[1] = \sum_{n=0}^{1} x[n]\, e^{-j\frac{2\pi}{2}1n} = x[0]\, e^{-j\frac{2\pi}{2}10} + x[1]\, e^{-j\frac{2\pi}{2}11}$$

$$= x_0 - x_1$$

# FFT Butterfly

## FFT Butterfly

$$X[k] = \left\{ x_0 + x_1, \quad x_0 - x_1 \right\}$$

## FFT Butterfly

$$X[k]=\{x_0+x_1, \quad x_0-x_1\}$$

$x_0$

$x_1$

## FFT Butterfly

$$X[k]=\left\{x_0+x_1,\quad x_0-x_1\right\}$$

$x_0$ $X_0$

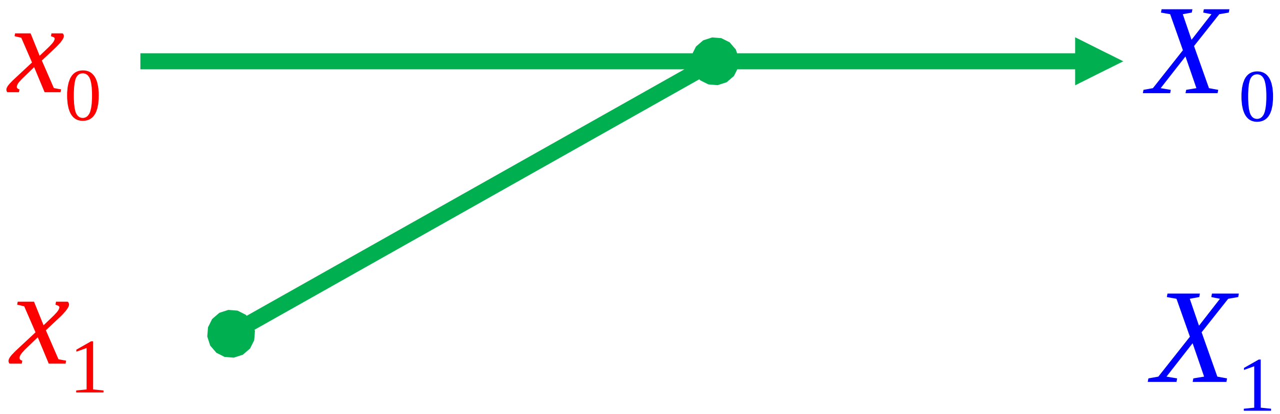
$x_1$ $X_1$

## FFT Butterfly

$$X[k] = \{ x_0 + x_1, \quad x_0 - x_1 \}$$

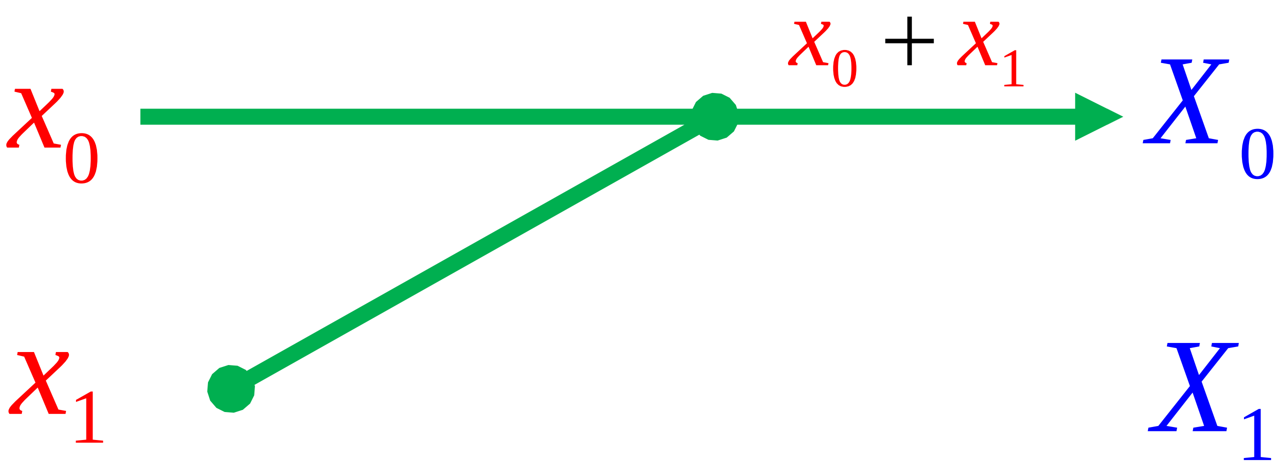
## FFT Butterfly

$$X[k]=\{x_0+x_1, \quad x_0-x_1\}$$



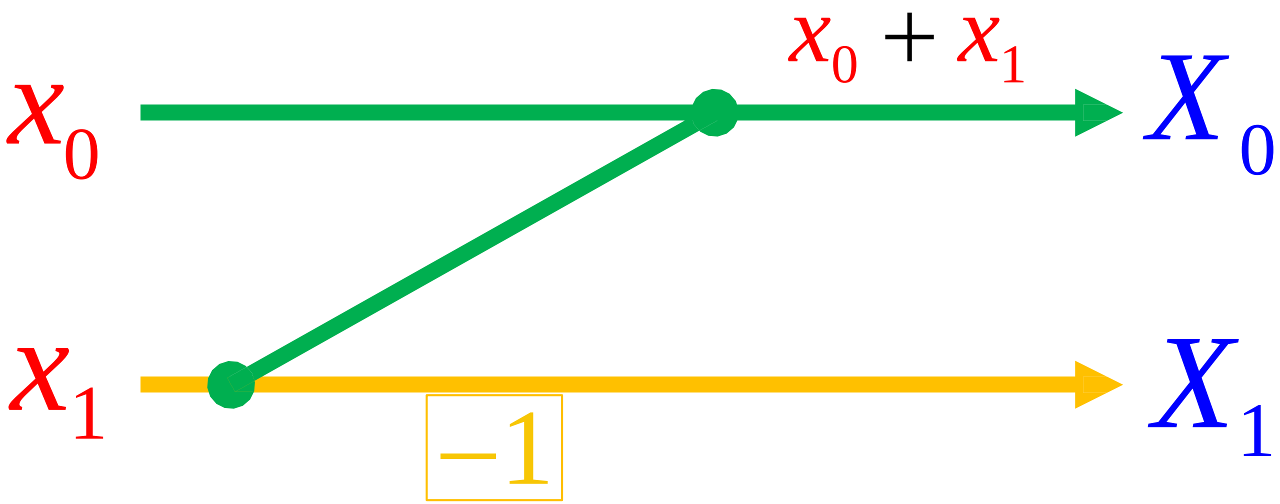
## FFT Butterfly

$$X[k] = \{ x_0 + x_1, \quad x_0 - x_1 \}$$



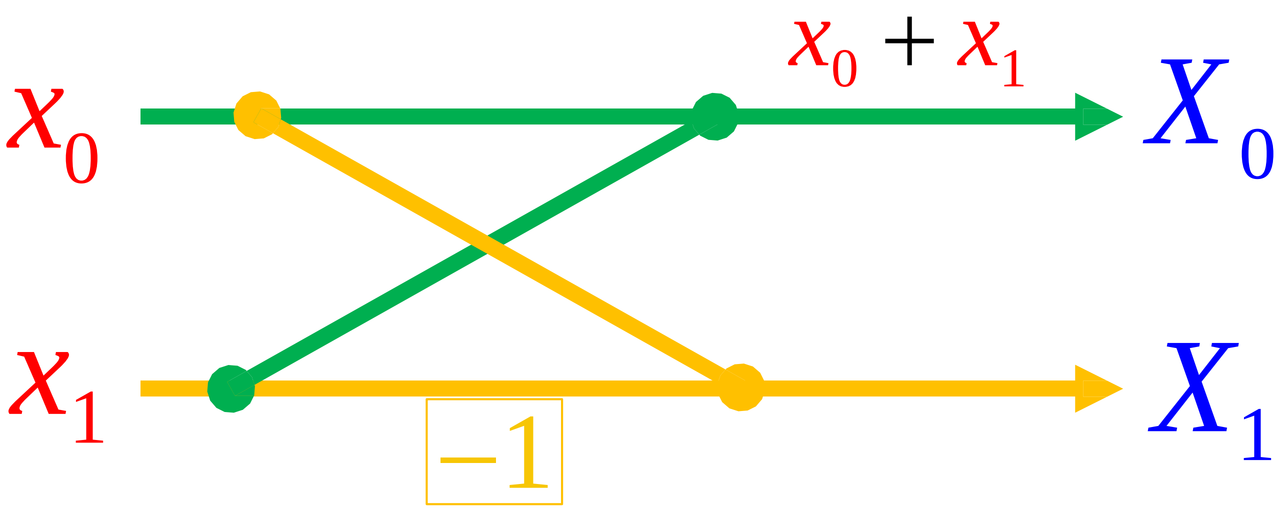
## FFT Butterfly

$$X[k] = \{ x_0 + x_1, \quad x_0 - x_1 \}$$



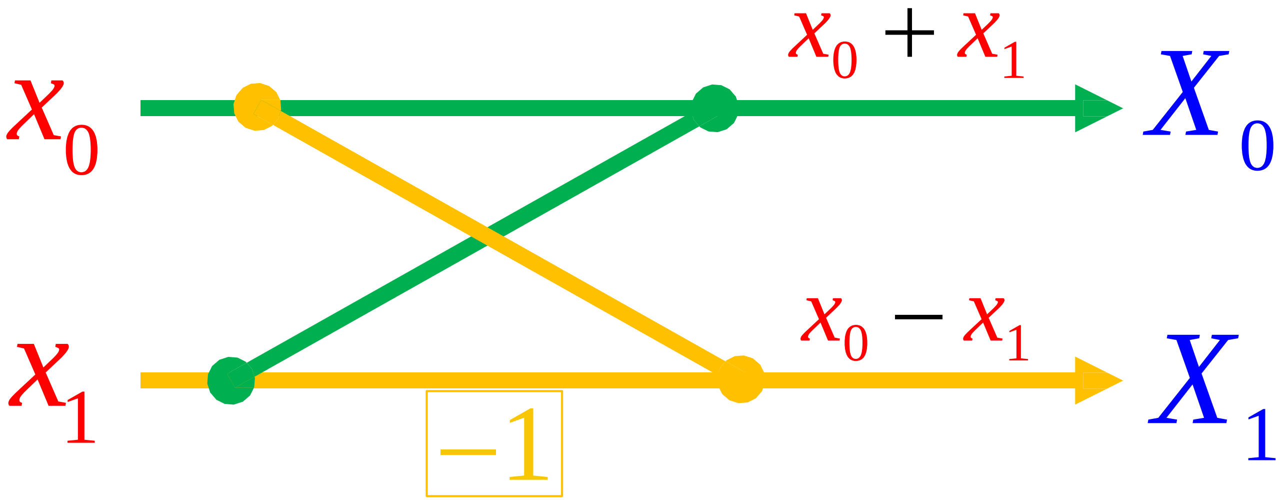
# FFT Butterfly

$$X[k] = \{ x_0 + x_1, \quad x_0 - x_1 \}$$



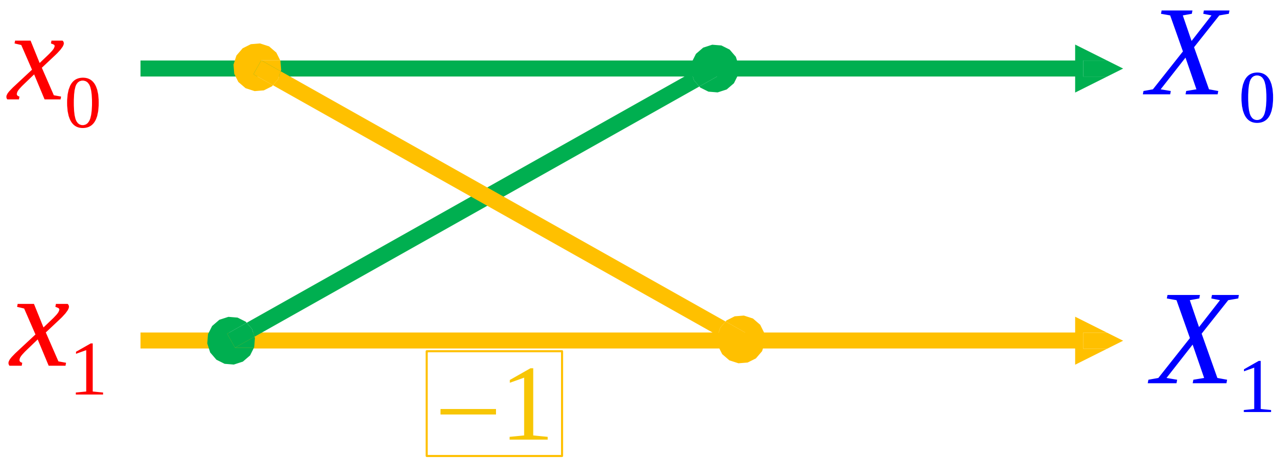
## FFT Butterfly

$$X[k] = \{ x_0 + x_1, \quad x_0 - x_1 \}$$



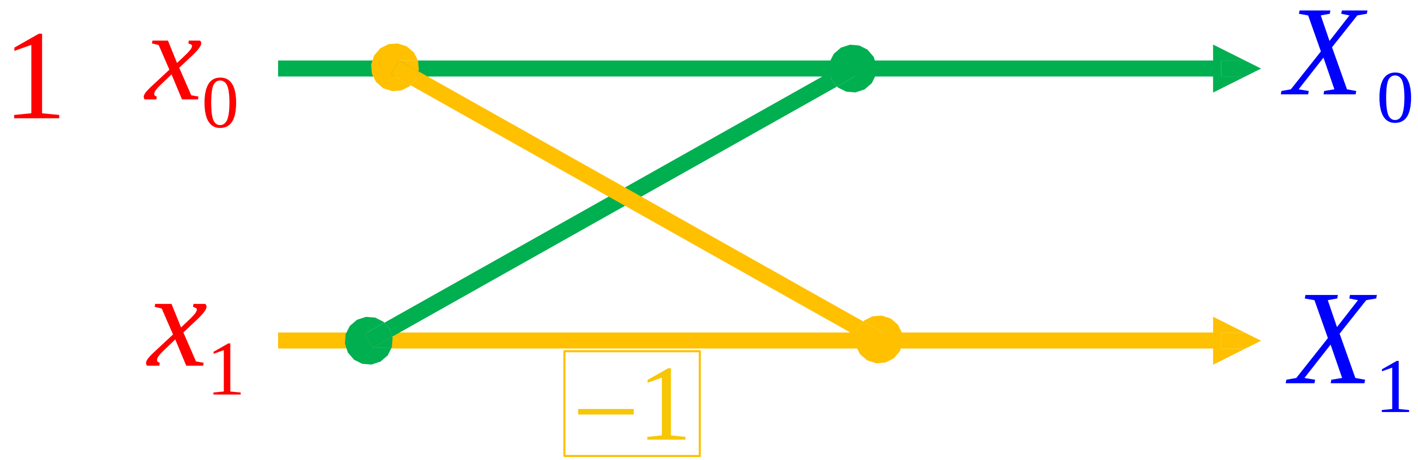
## Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm



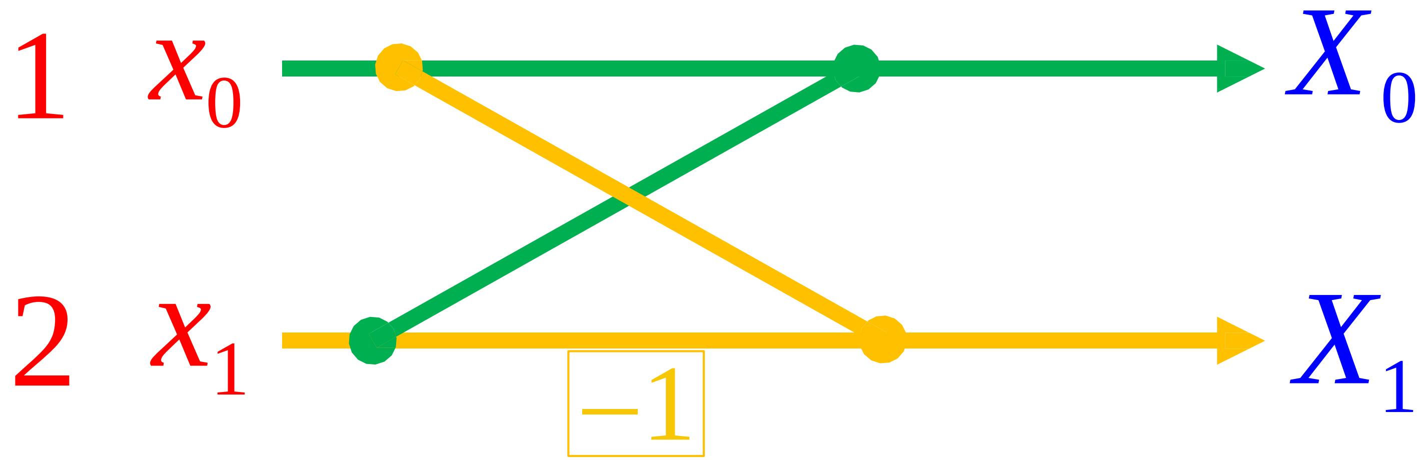
# Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm



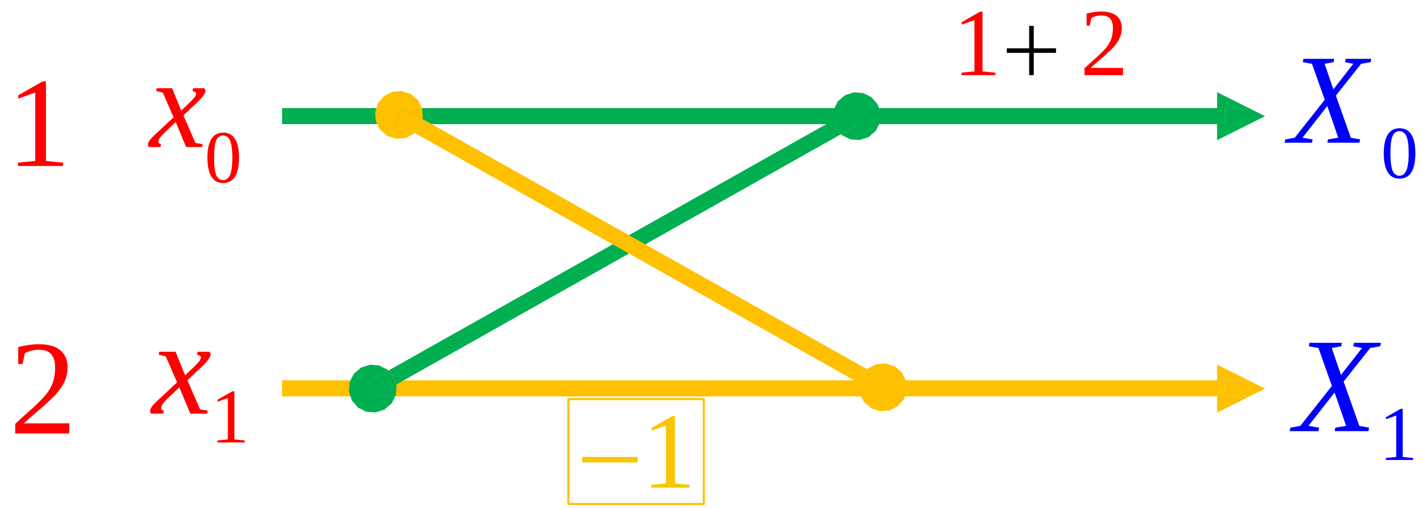
# Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm



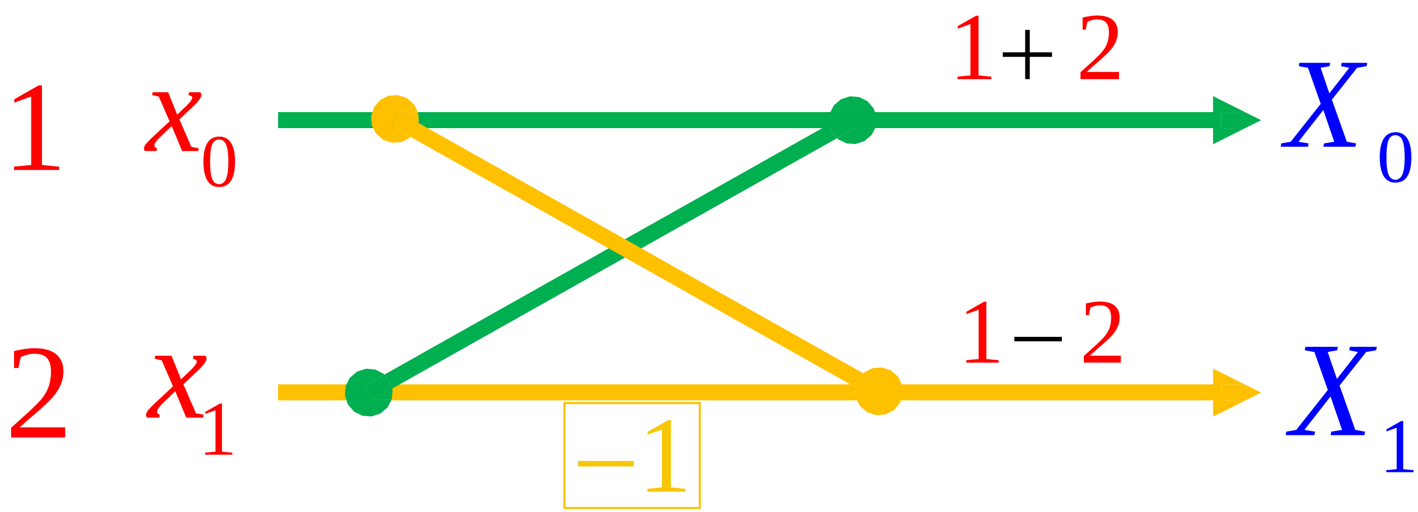
## Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm



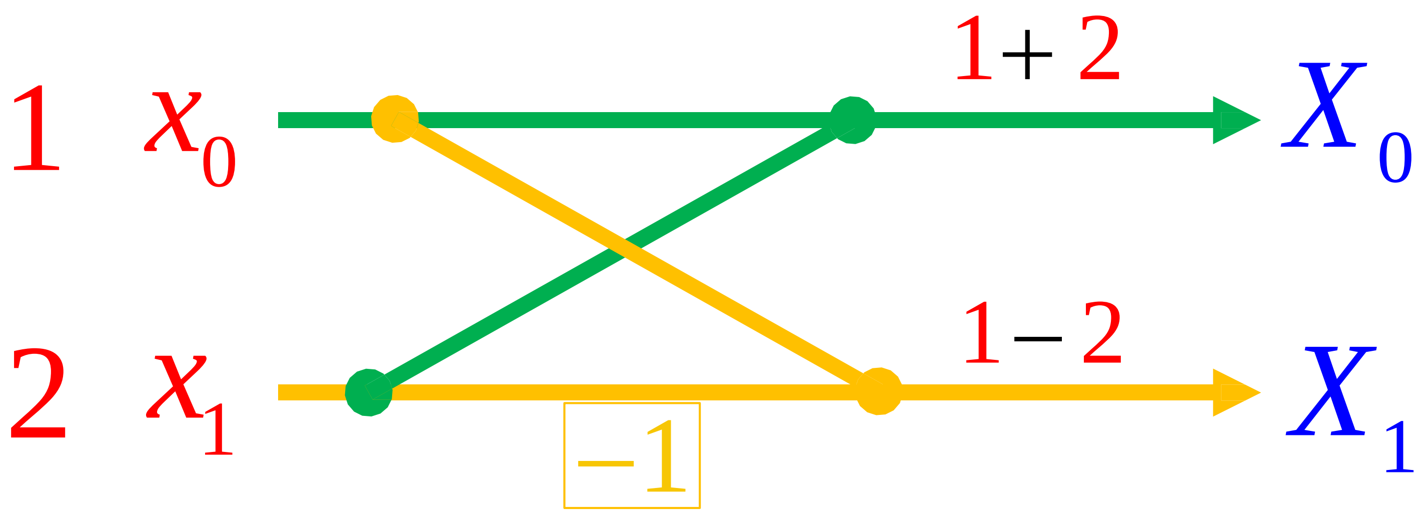
## Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm



# Example

Compute DFT of $x[n] = \{1, 2\}$ using FFT algorithm




$$X[k] = \{3, \quad -1\}$$

# Fast Fourier Transform

# Fast Fourier Transform

- Fundamental principle of **decomposing** the computation of N-point DFT into **successively smaller DFTs**.

# Fast Fourier Transform

- Fundamental principle of **decomposing** the computation of N-point DFT into **successively smaller DFTs**.
- The manner in which this principle is implemented leads to a variety of different algorithms.

# Fast Fourier Transform

- Fundamental principle of **decomposing** the computation of N-point DFT into **successively smaller DFTs**.
- The manner in which this principle is implemented leads to a variety of different algorithms.
  - **Decimation in Time (DIT)**

    The sequence *x[n]* (generally thought of as a time sequence) is decomposed into successively smaller subsequences.

# Fast Fourier Transform

- Fundamental principle of **decomposing** the computation of N-point DFT into **successively smaller DFTs**.
- The manner in which this principle is implemented leads to a variety of different algorithms.
  - **Decimation in Time (DIT)**

    The sequence *x[n]* (generally thought of as a time sequence) is decomposed into successively smaller subsequences.
  - **Decimation in Frequency (DIF)**

    The sequence of DFT coefficients *X[k]* is decomposed into smaller subsequences

# Decimation-In-Time FFT Algorithm

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of $\boldsymbol{N}$ **an integer power of 2**, i.e., $N = 2^v$.

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n] \, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

- Separating $x[n]$ into two $(N/2)$-point sequences consisting of the even-indexed points in $x[n]$ and the odd-indexed points in $x[n]$.

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{kn} \qquad k, n = 0,1,\ldots, N-1$$

- Separating $x[n]$ into two $(N/2)$-point sequences consisting of the even-indexed points in $x[n]$ and the odd-indexed points in $x[n]$.

$$X[k] = \sum x[n]\, \mathbf{W}_N^{kn} + \sum x[n]\, \mathbf{W}_N^{kn}$$

# Decimation-In-Time FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

- Separating $x[n]$ into two $(N/2)$-point sequences consisting of the even-indexed points in $x[n]$ and the odd-indexed points in $x[n]$.

$$X[k] = \sum_{n\ \text{even}} x[n]\, \mathbf{W}_N^{kn} + \sum_{n\ \text{odd}} x[n]\, \mathbf{W}_N^{kn}$$

- Even indexed $x[2m]$ and odd indexed $x[2m+1]$ each $\boldsymbol{N/2}$ long

# Decimation-In-Time FFT Algorithm

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} \mathbf{W}_N^{k}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} \mathbf{W}_N^{k}$$

$$\left(\mathbf{W}_N^{2}\right)^{k\,m}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} \mathbf{W}_N^{k}$$

$$\left(\mathbf{W}_N^{2}\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N}2}\right)^{k\,m}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \left(\mathbf{W}_N^{2}\right)^{k\,m} \mathbf{W}_N^{k}$$

$$\left(\mathbf{W}_N^{2}\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N}2}\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N/2}}\right)^{k\,m}$$

# Decimation-In-Time FFT Algorithm

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

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

$$\left(W_N^2\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N}2}\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N/2}}\right)^{k\,m} = W_{N/2}^{k\,m}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_N^{k\,2m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_N^{k\,(2m+1)}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \left(\mathbf{W}_N^2\right)^{k\,m} + \sum_{m=0}^{N/2-1} x[2m+1]\, \left(\mathbf{W}_N^2\right)^{k\,m} \mathbf{W}_N^k$$

$$\left(\mathbf{W}_N^2\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N}2}\right)^{k\,m} = \left(e^{-j\frac{2\pi}{N/2}}\right)^{k\,m} = \mathbf{W}_{N/2}^{k\,m}$$

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_{N/2}^{k\,m} + \mathbf{W}_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_{N/2}^{k\,m}$$

# Decimation-In-Time FFT Algorithm

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, W_{N/2}^{km}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathrm{W}_{N/2}^{km} + \mathrm{W}_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, \mathrm{W}_{N/2}^{km}$$

$$X[k] = G[k] + \mathrm{W}_N^k H[k]$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathrm{W}_{N/2}^{km} + \mathrm{W}_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, \mathrm{W}_{N/2}^{km}$$

$$X[k] = G[k] + \mathrm{W}_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \ldots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \ldots, (N/2) - 1$.

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, W_{N/2}^{km}$$

$$X[k] = G[k] + W_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \dots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \dots, (N/2) - 1$.

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, \mathbf{W}_{N/2}^{km} + \mathbf{W}_N^{k} \sum_{m=0}^{N/2-1} x[2m+1]\, \mathbf{W}_{N/2}^{km}$$

$$X[k] = G[k] + \mathbf{W}_N^{k} H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \ldots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \ldots, (N/2) - 1$.

$$X[k] = G[k] + \mathbf{W}_N^{k} H[k] \qquad k = (N/2), \ldots, N-1$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2 - 1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2 - 1} x[2m+1]\, W_{N/2}^{km}$$

$$X[k] = G[k] + W_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \ldots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \ldots, (N/2) - 1$.

$$X[k] = G[k] + W_N^k H[k] \qquad k = (N/2), \ldots, N-1$$

$$W_N^{(k+N/2)}$$

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, W_{N/2}^{km}$$

$$X[k] = G[k] + W_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \ldots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \ldots, (N/2) - 1$.

$$X[k] = G[k] + W_N^k H[k] \qquad k = (N/2), \ldots, N-1$$

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

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, W_{N/2}^{km}$$

$$X[k] = G[k] + W_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \dots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \dots, (N/2) - 1$.

$$X[k] = G[k] + W_N^k H[k] \qquad k = (N/2), \dots, N-1$$

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

# Decimation-In-Time FFT Algorithm

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]\, W_{N/2}^{km}$$

$$X[k] = G[k] + W_N^k H[k]$$

- $G[k]$ and $H[k]$ are $N/2$-point DFTs.
- Although $k = 0, 1, \ldots, N-1$, since $G[k]$ and $H[k]$ are each periodic in $k$ with period $N/2$ sums must be computed only for $k = 0, 1, \ldots, (N/2) - 1$.

$$X[k] = G[k] + W_N^k H[k] \qquad k = (N/2), \ldots, N-1$$

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

# Decimation-In-Time FFT Algorithm

# Decimation-In-Time FFT Algorithm

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

# Decimation-In-Time FFT Algorithm

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

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

# Decimation-In-Time FFT Algorithm

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

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

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

# Decimation-In-Time FFT Algorithm

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

$$G[k] = \sum_{l=0}^{N/4-1} g[2l]\, \left(W_{N/2}^{2}\right)^{kl} + \sum_{l=0}^{N/4-1} g[2l+1]\, \left(W_{N/2}^{2}\right)^{kl} W_{N/2}^{k}$$

$$G[k] = \sum_{l=0}^{N/4-1} g[2l]\, W_{N/4}^{kl} + W_{N/2}^{k} \sum_{l=0}^{N/4-1} g[2l+1]\, W_{N/4}^{kl}$$

Similarly

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

# 4-point DITFFT

# 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$

## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$

$$x_0$$

$$x_2$$

## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$

$x_2$

$x_1$

$x_3$

## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$ $X_0$

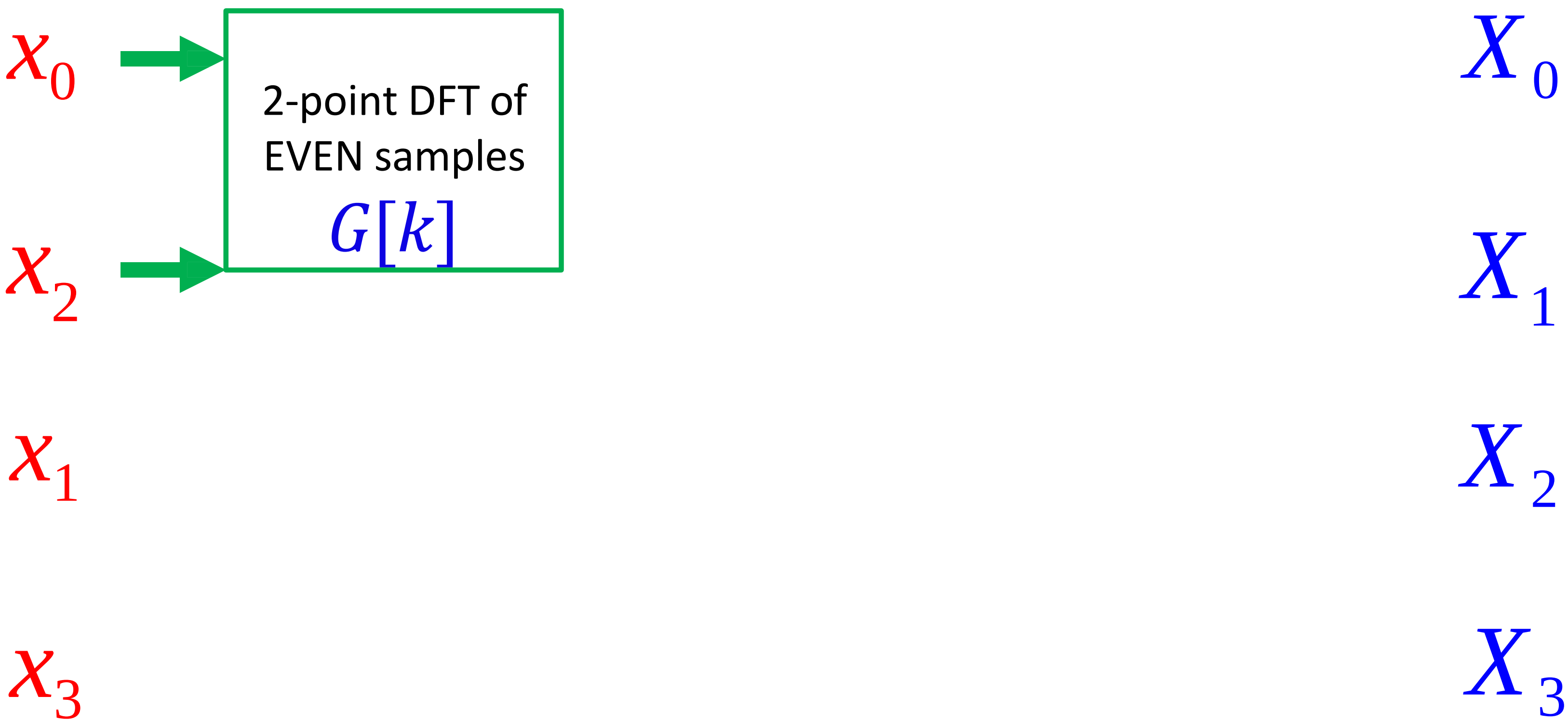
$x_2$ $X_1$

$x_1$ $X_2$

$x_3$ $X_3$

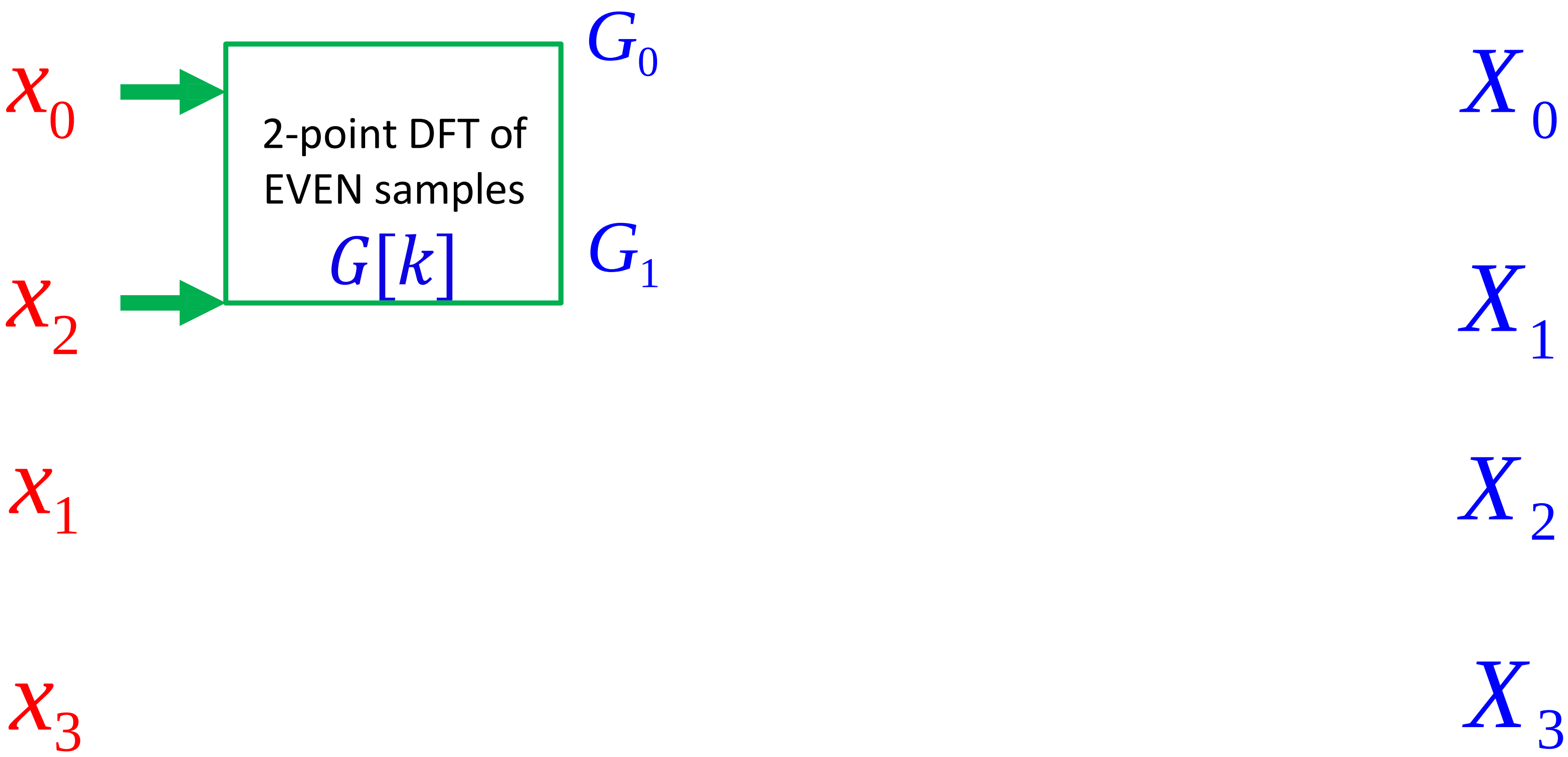
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



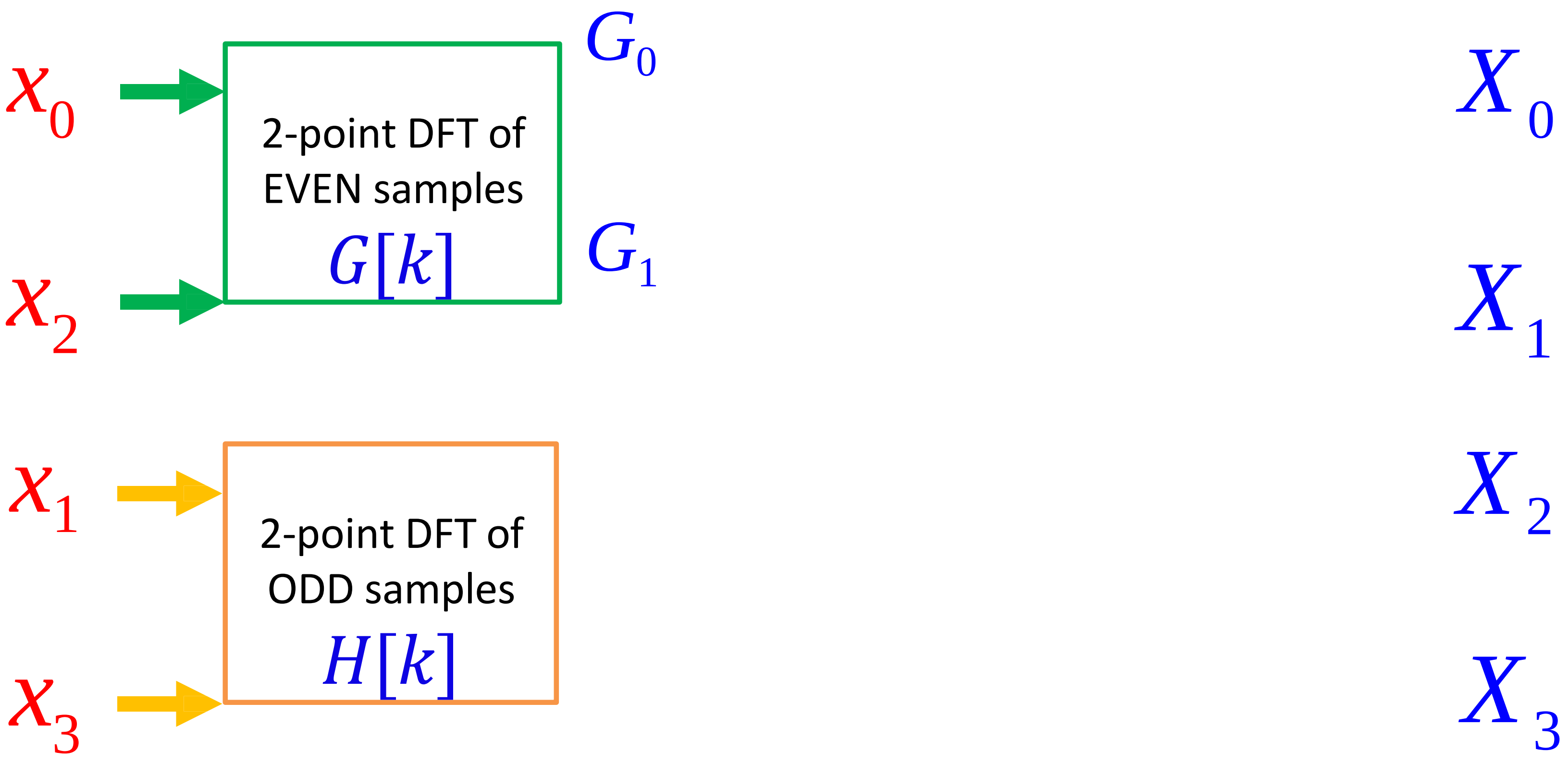
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



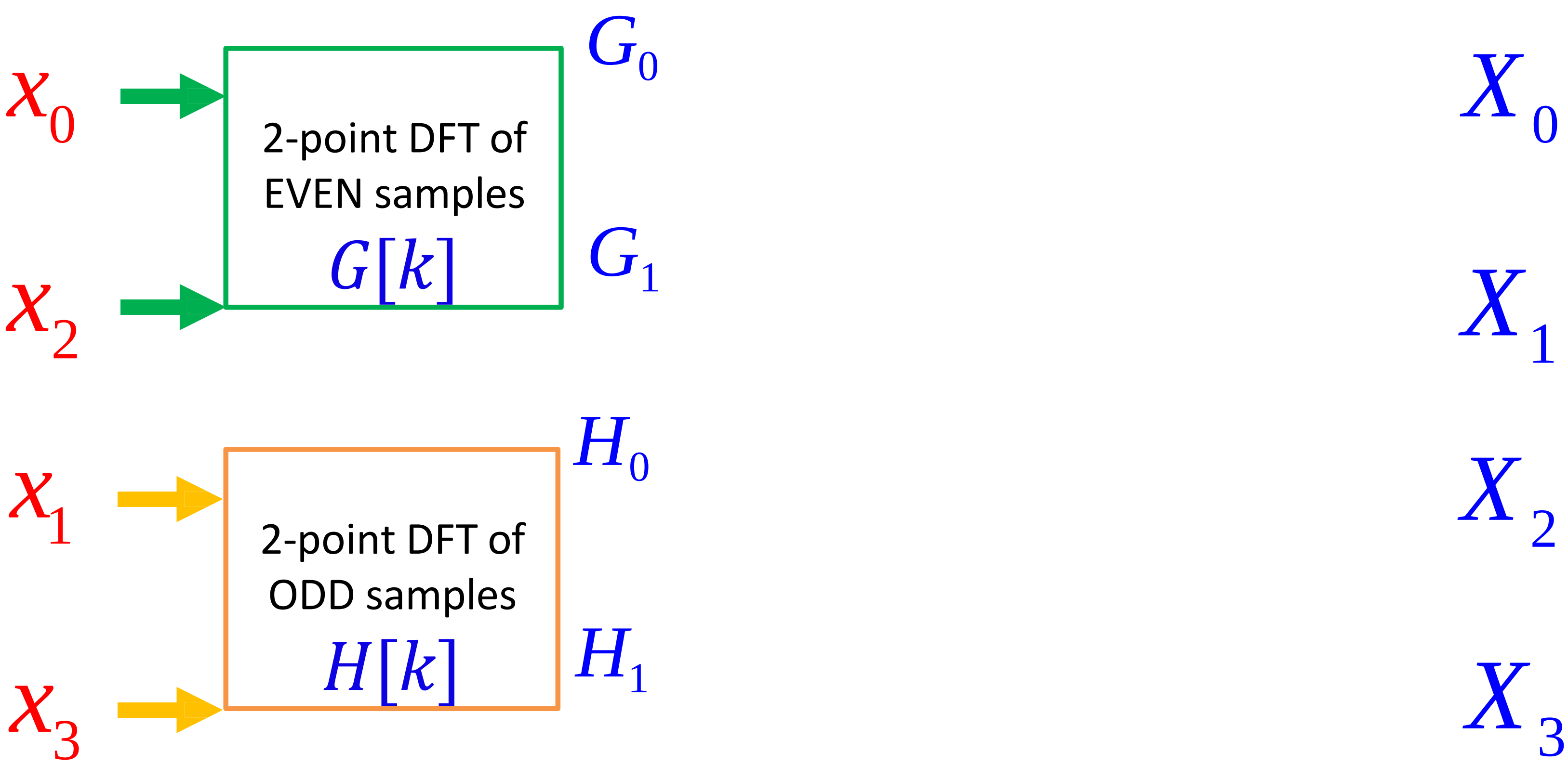
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

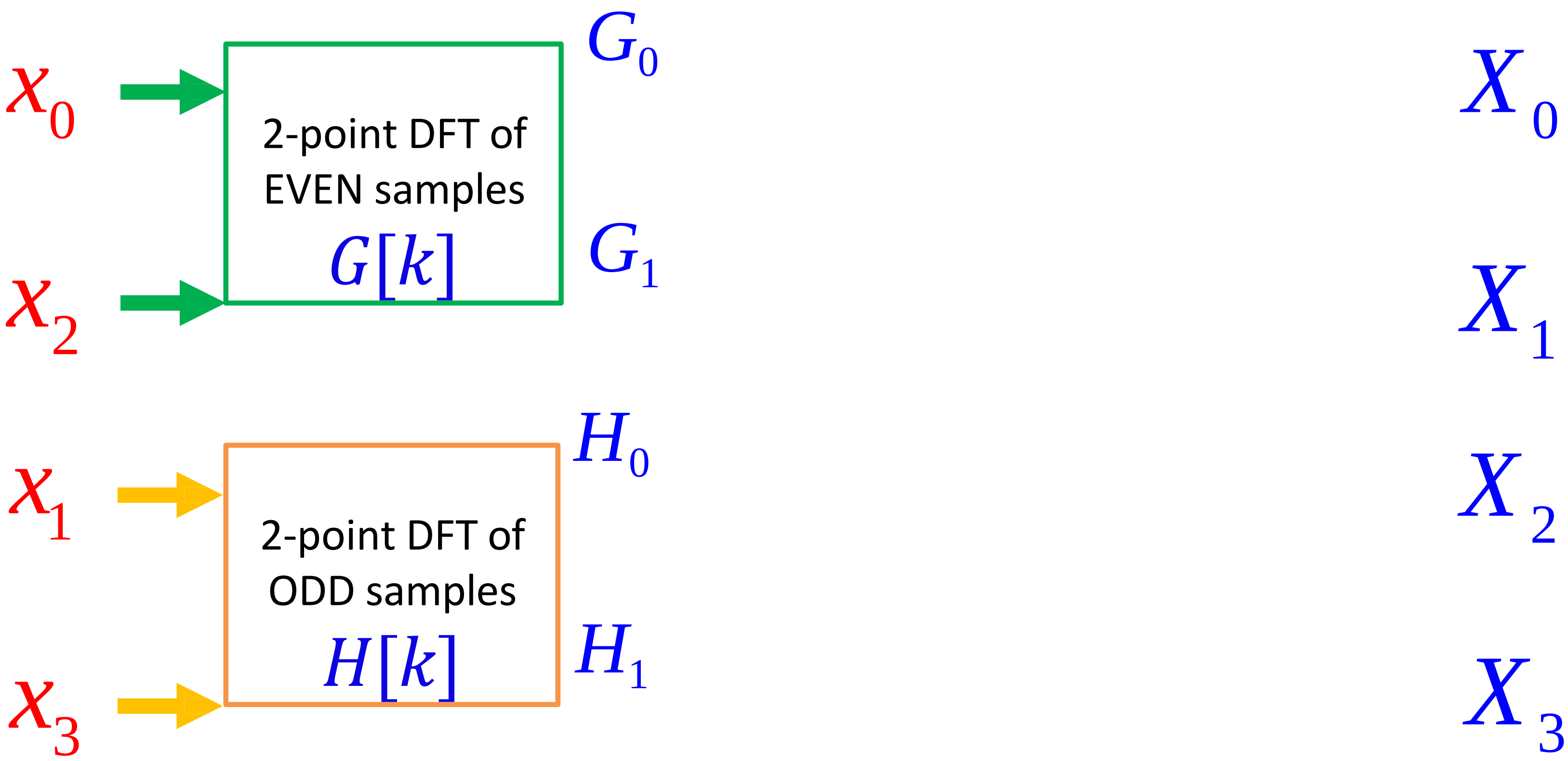
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + W_4^k H[k]$$

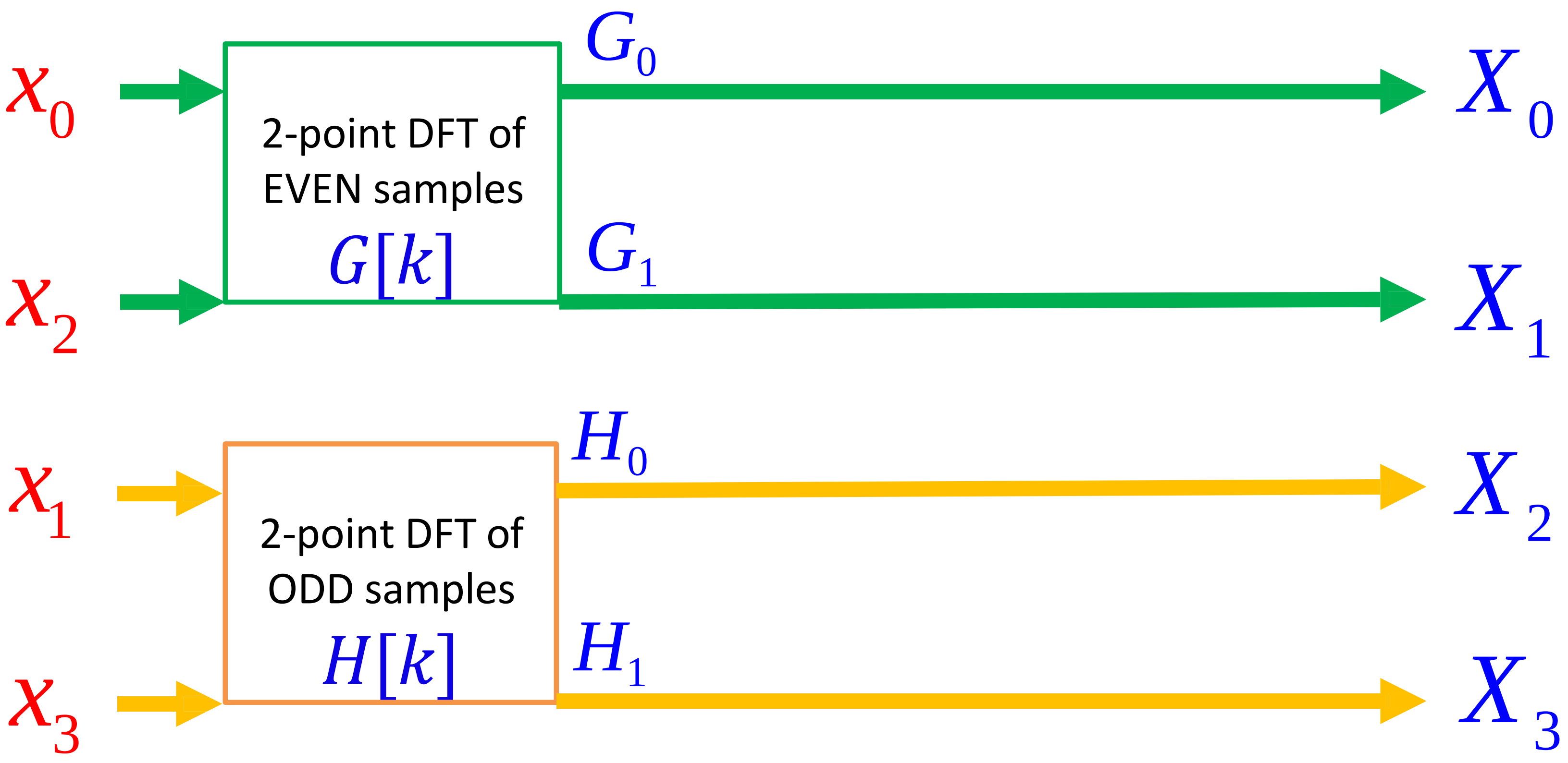
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + \mathbf{W}_4^k H[k]$$

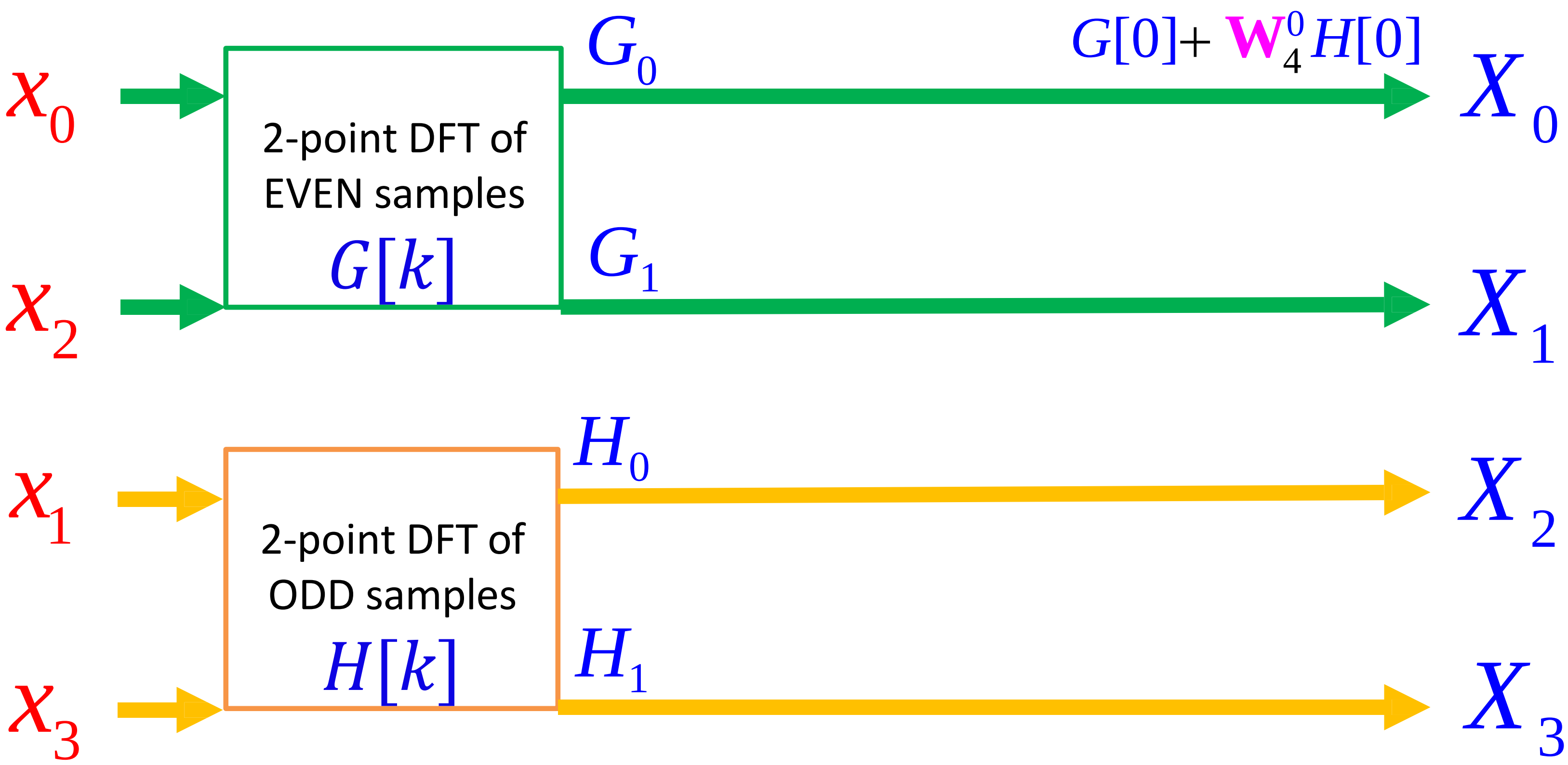
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



# 4-point DITFFT

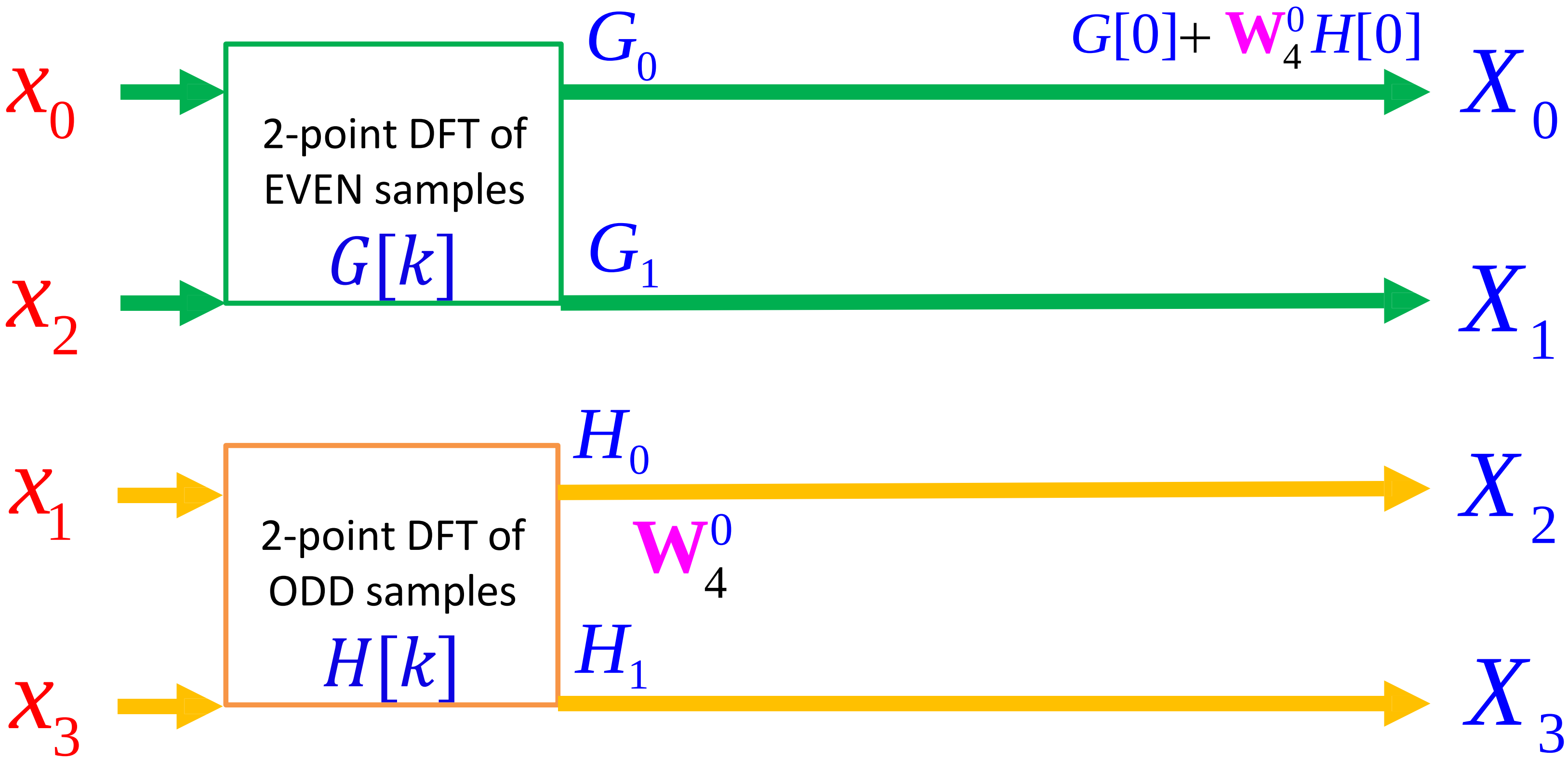
$$X[k] = G[k] + W_4^k H[k]$$

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT $X[k] = G[k] + W_4^k H[k]$

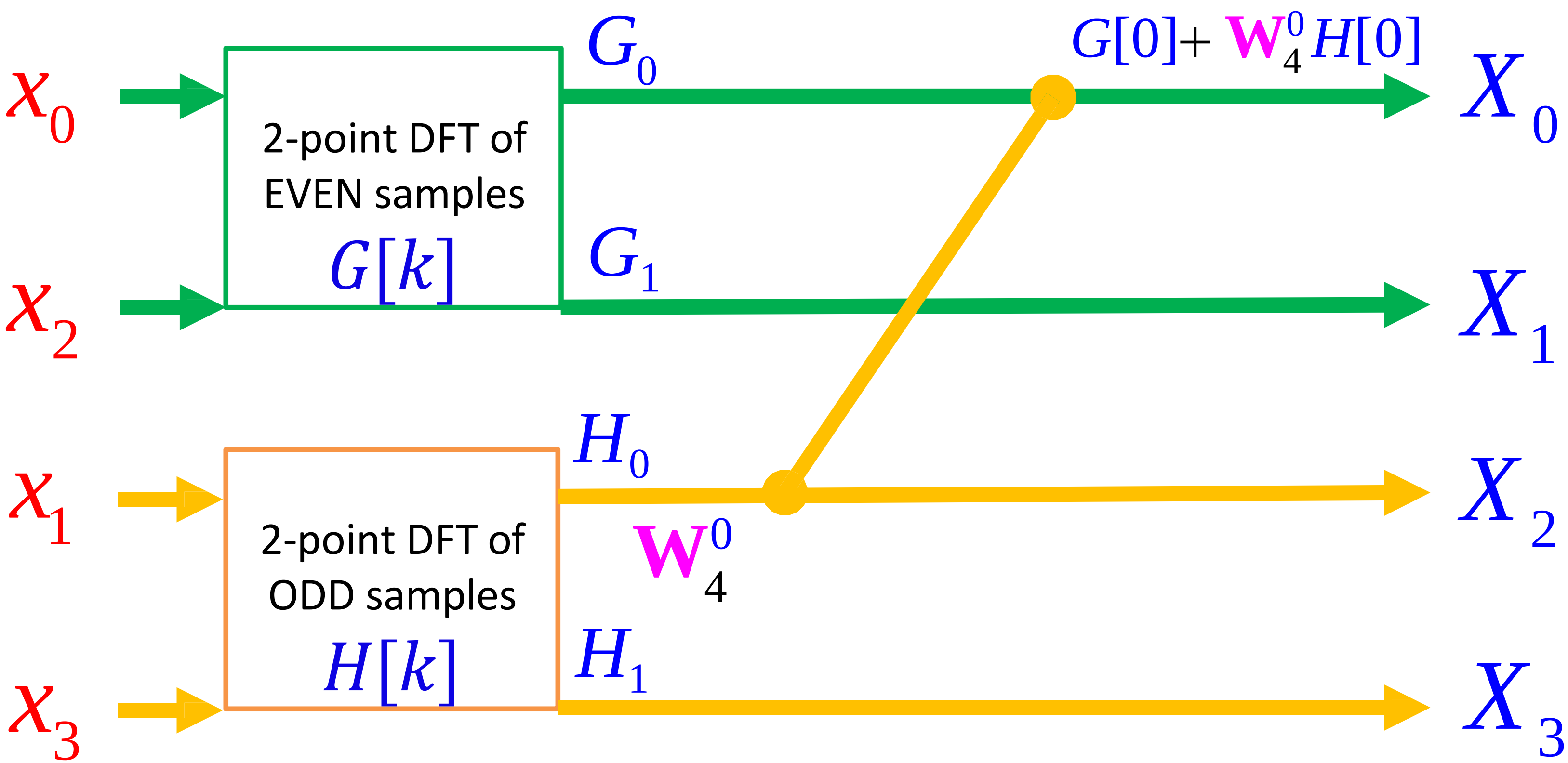
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + W_4^k H[k]$$

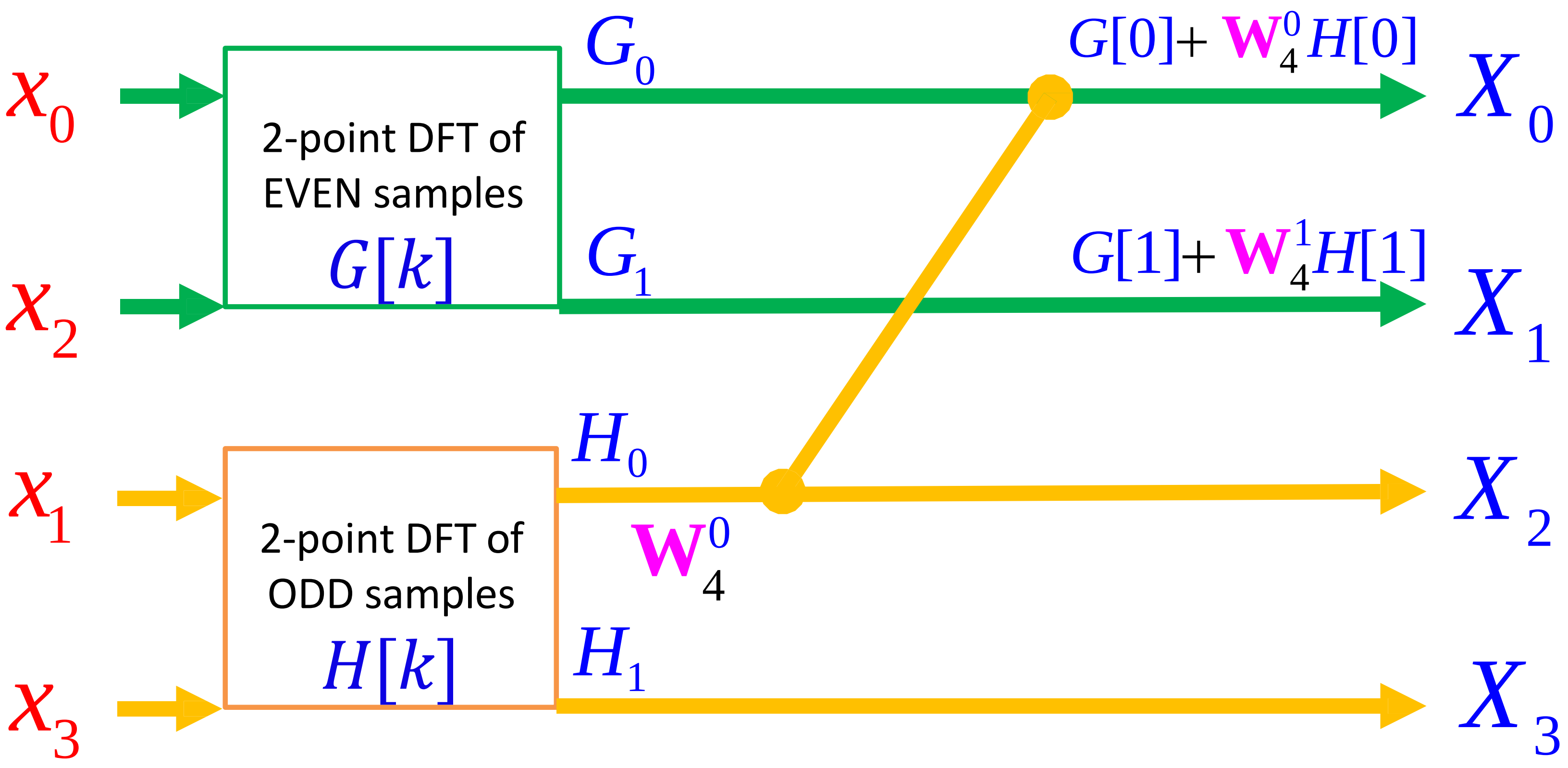
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + W_4^k H[k]$$

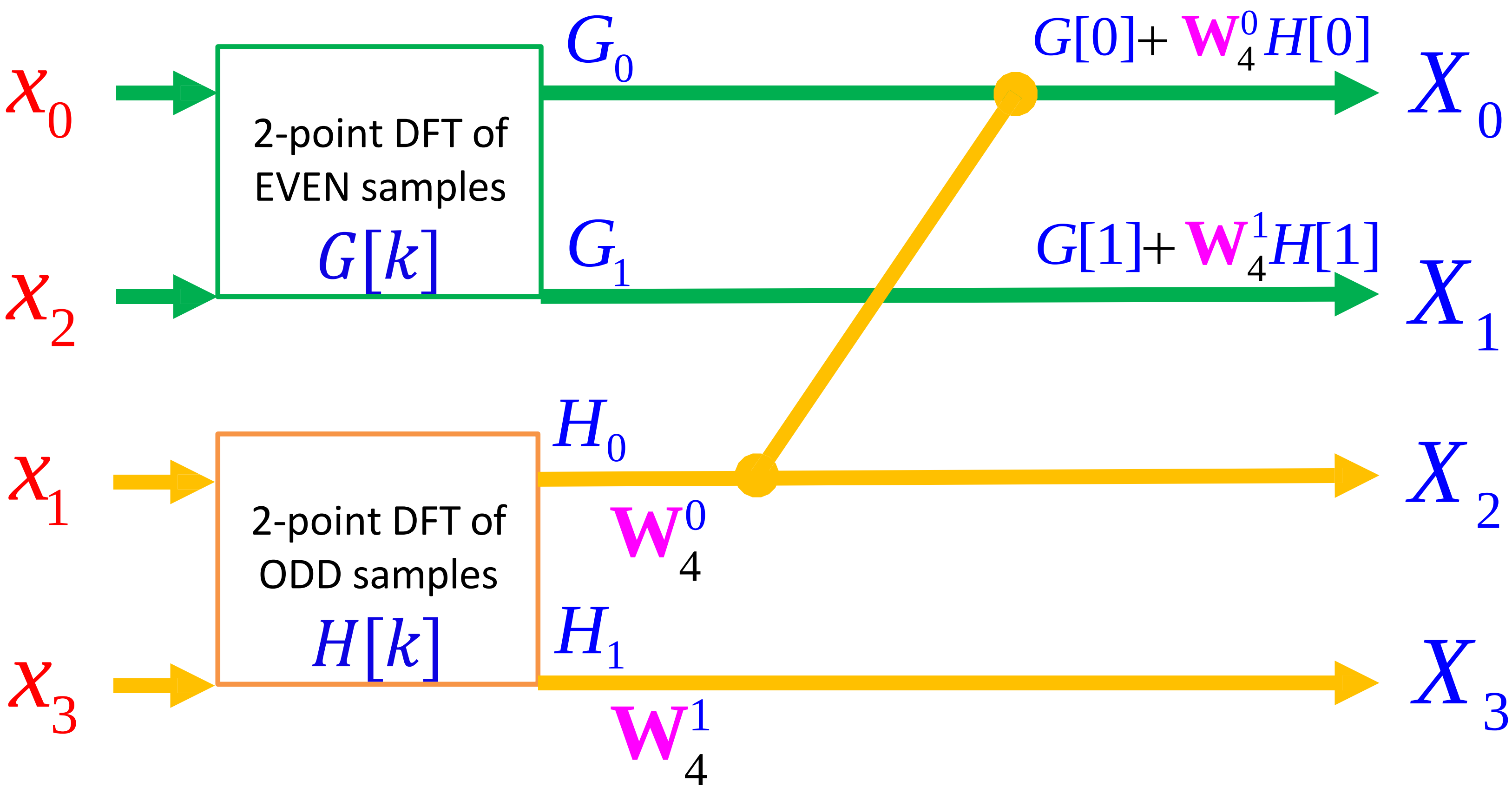
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + W_4^k H[k]$$

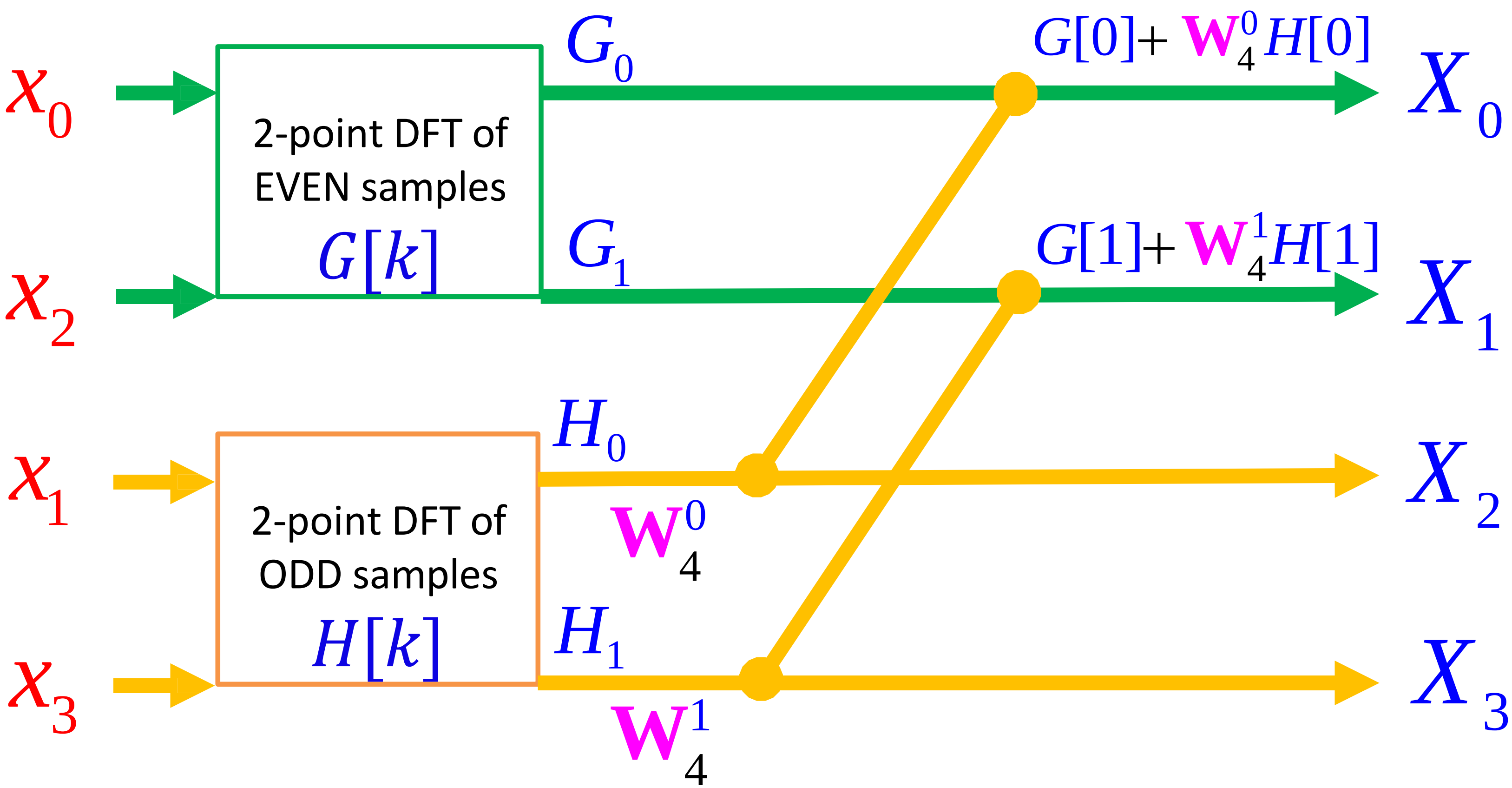
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] + W_4^k H[k]$$

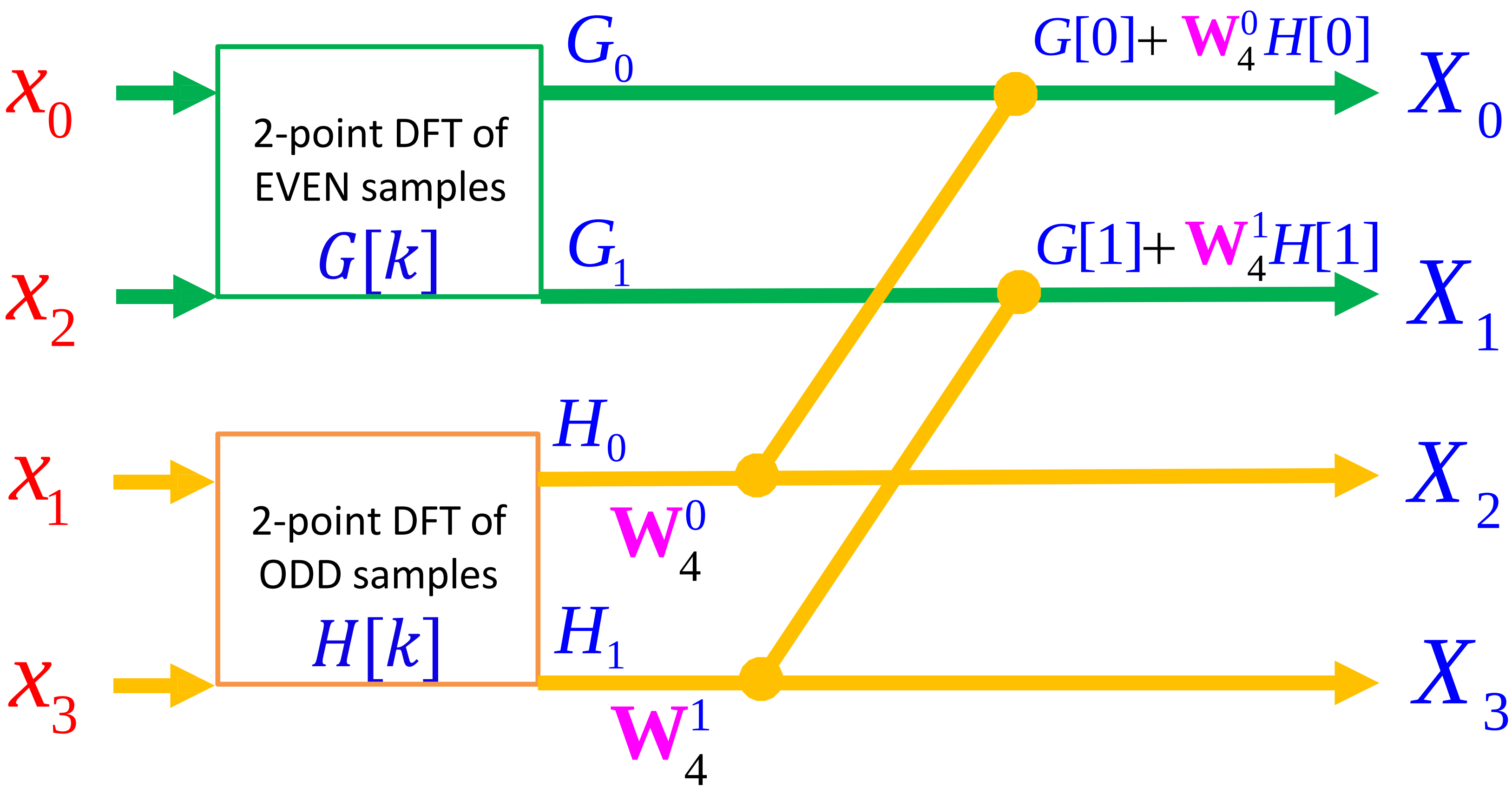
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - \mathbf{W}_4^k H[k]$$

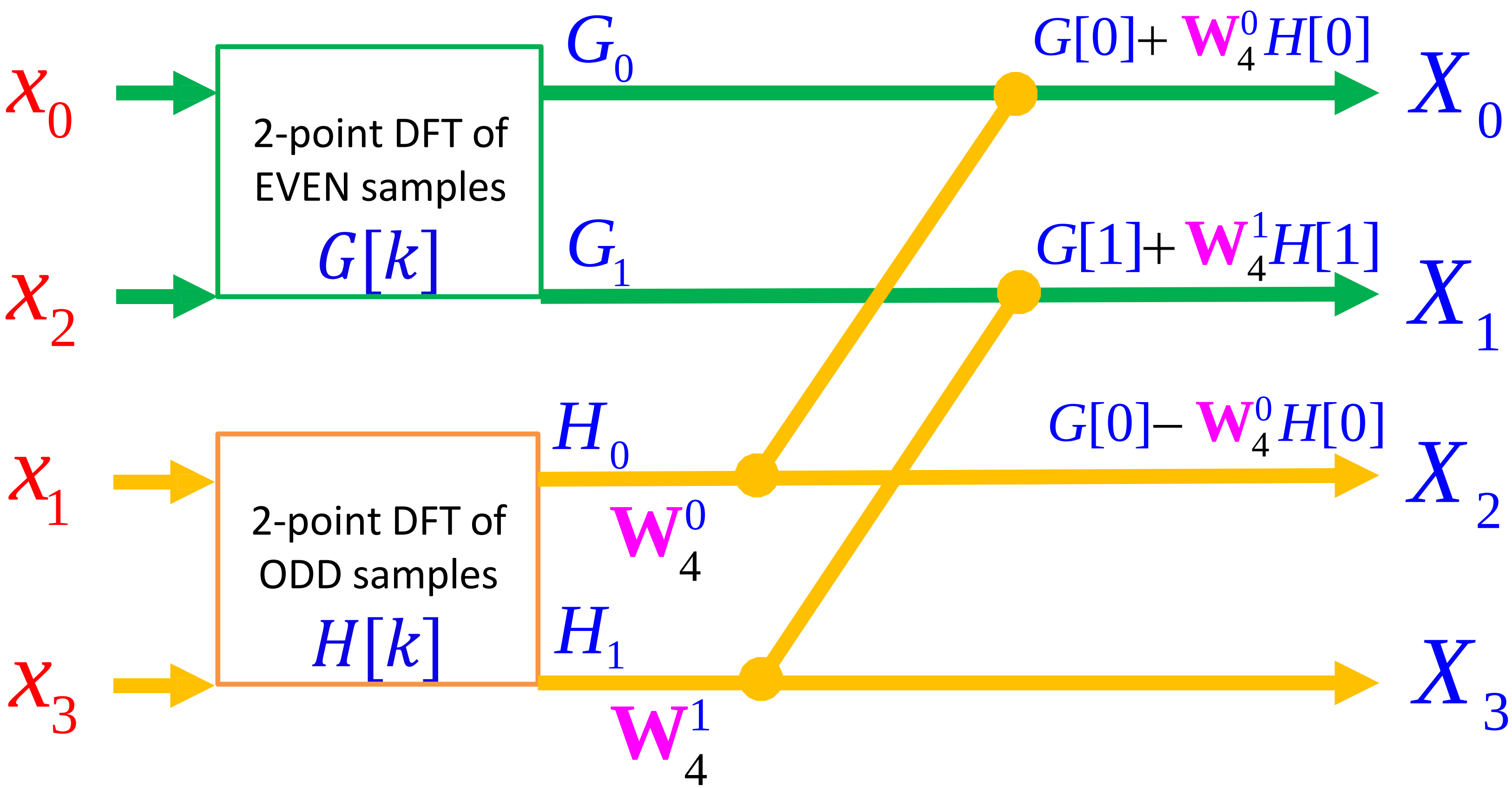
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - \mathbf{W}_4^k H[k]$$

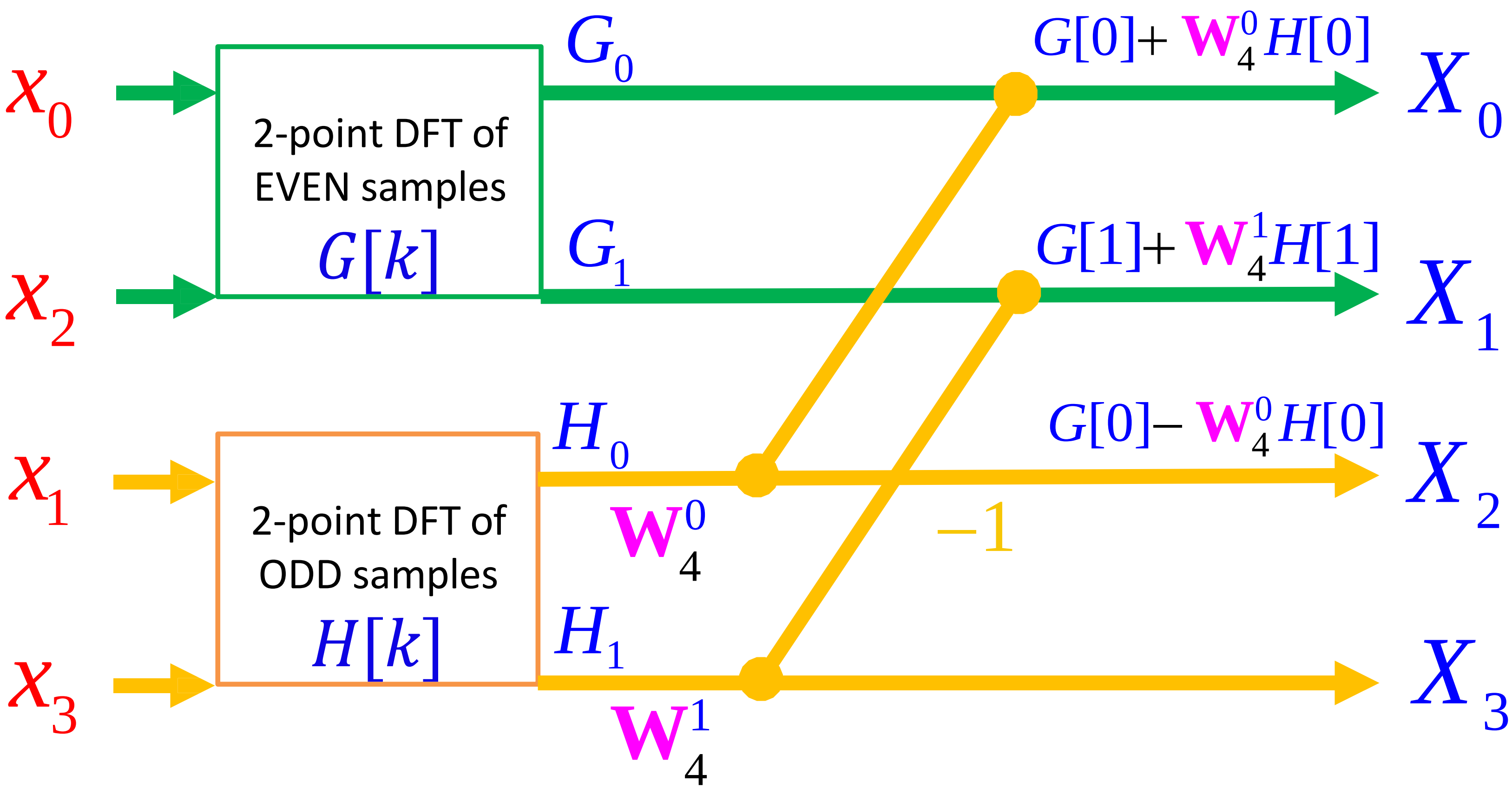
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - \mathbf{W}_4^k H[k]$$

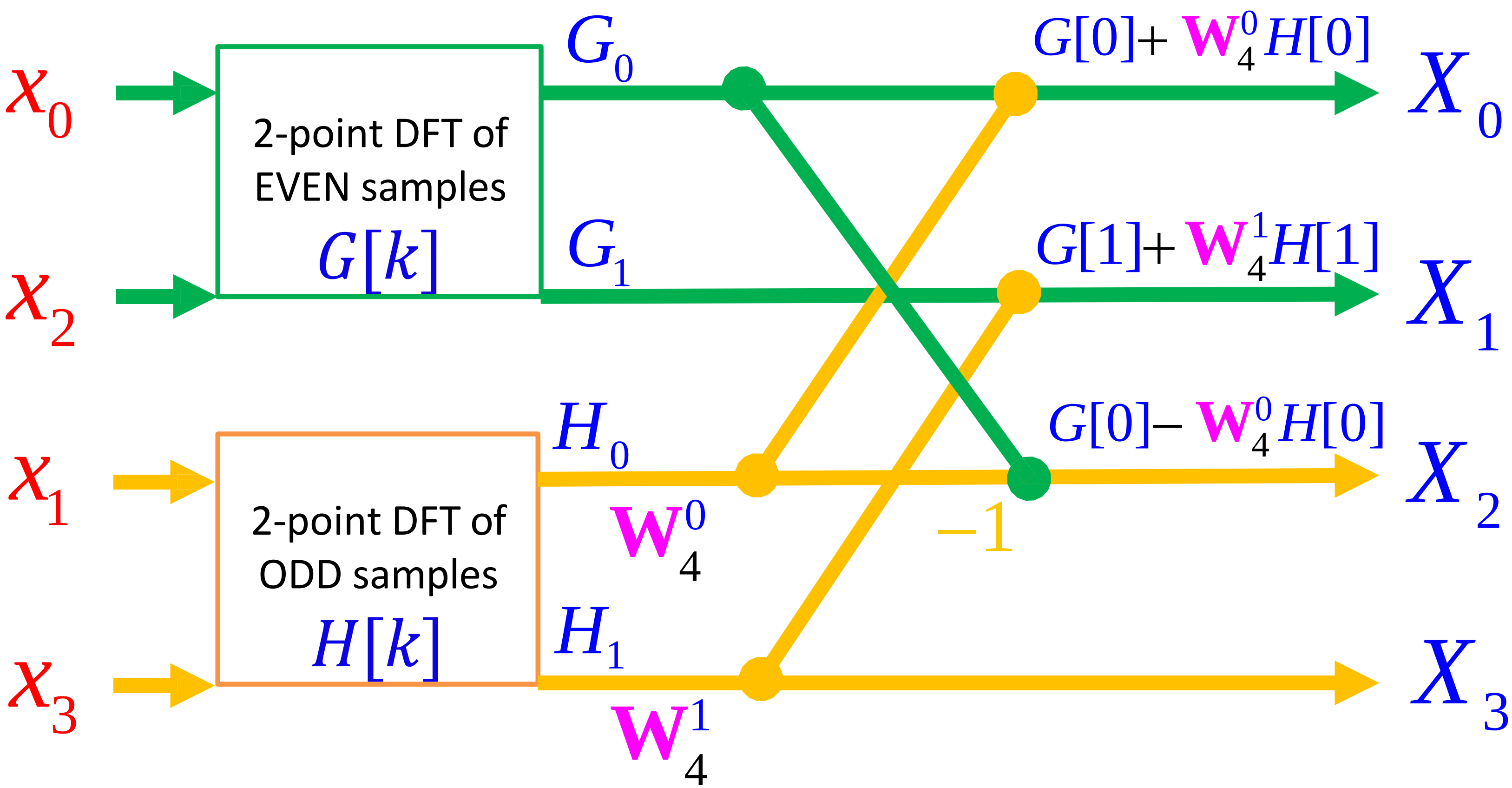
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - W_4^k H[k]$$

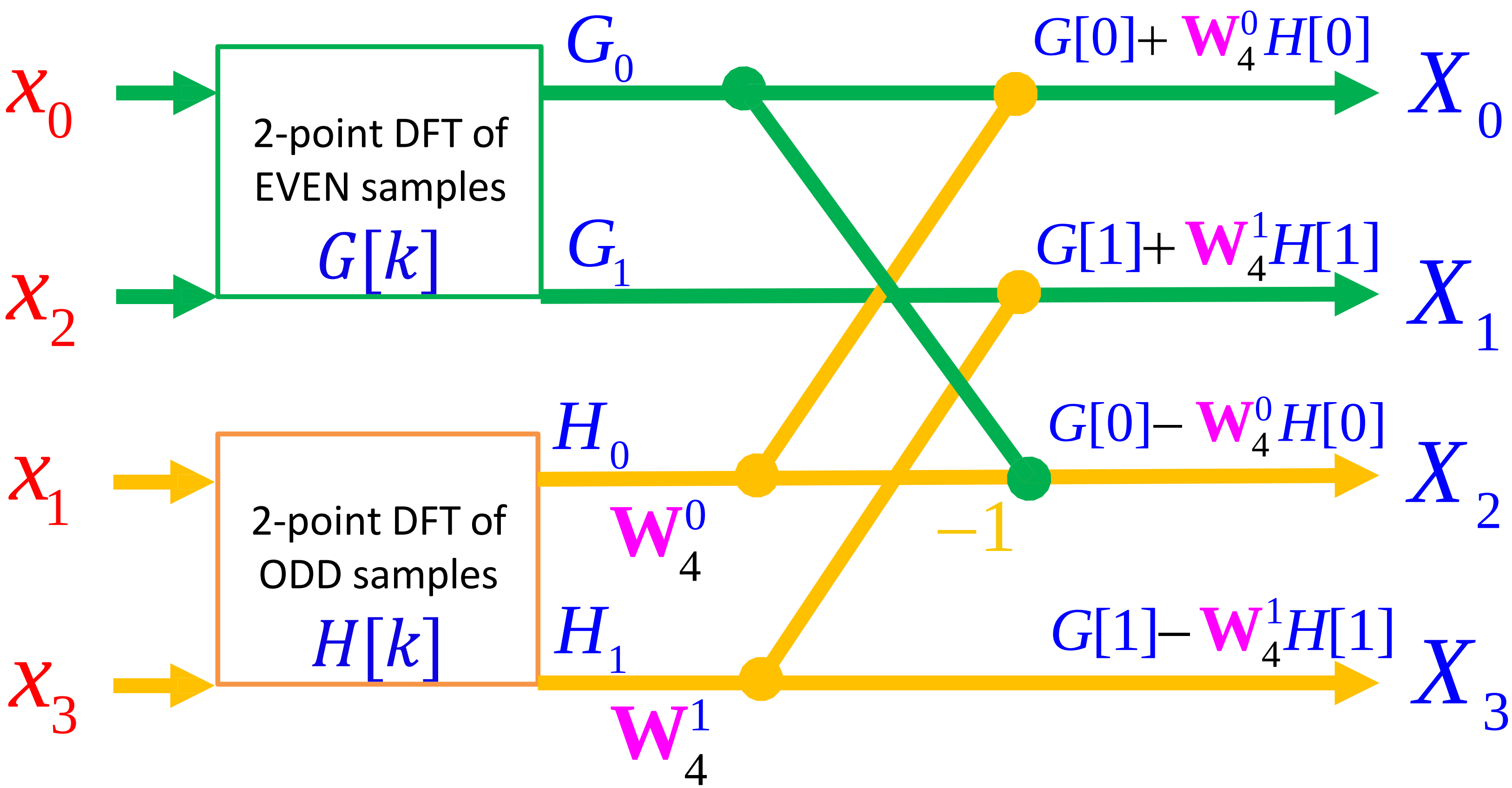
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - W_4^k H[k]$$

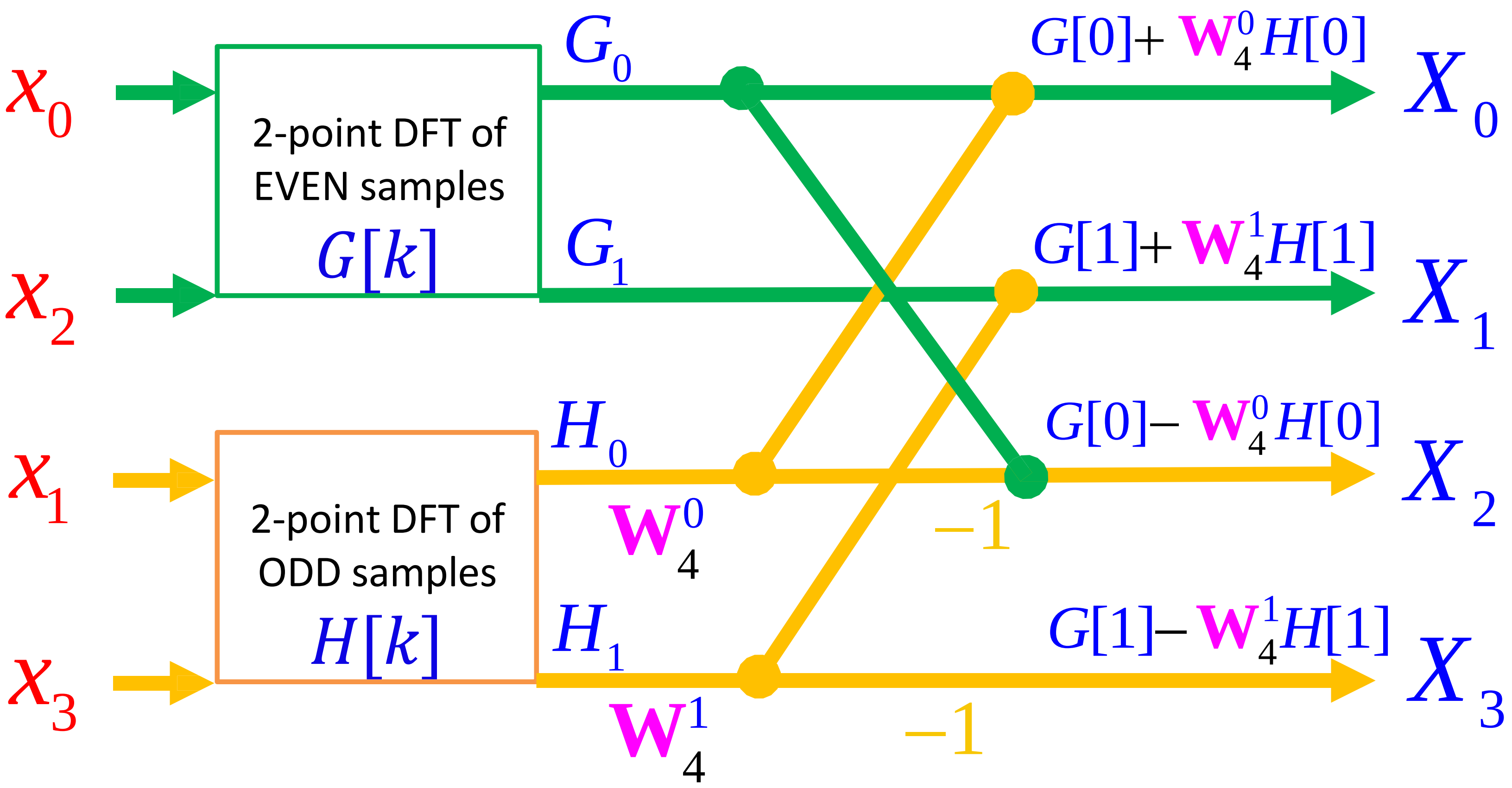
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - W_4^k H[k]$$

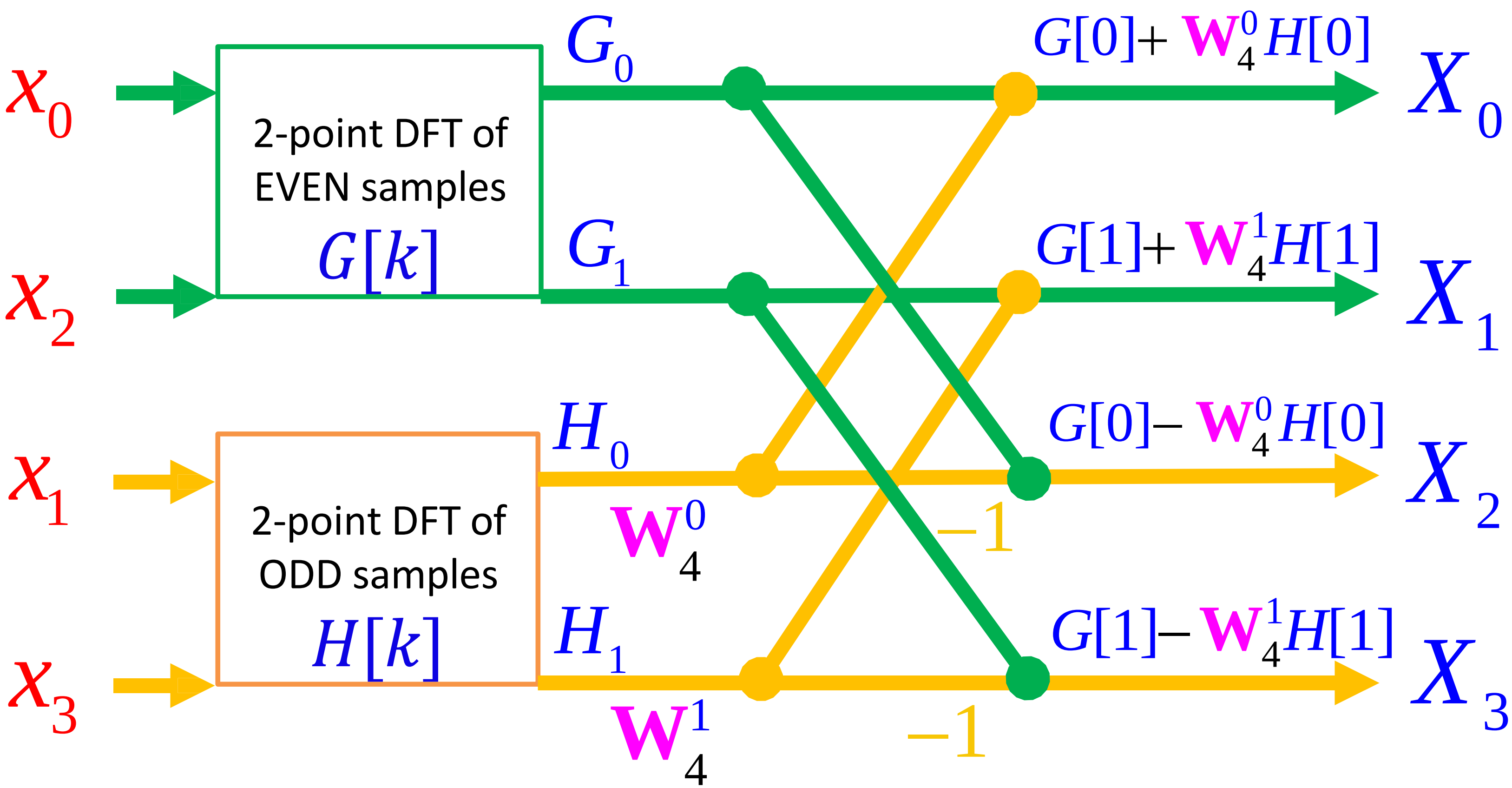
- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DITFFT

$$X[k] = G[k] - \mathbf{W}_4^k H[k]$$

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



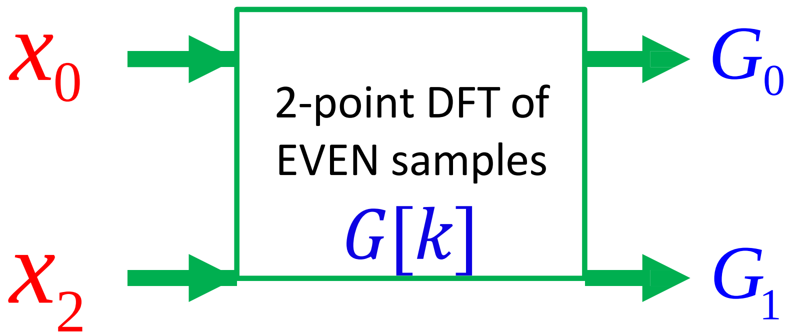
# 4-point DITFFT

# 4-point DITFFT

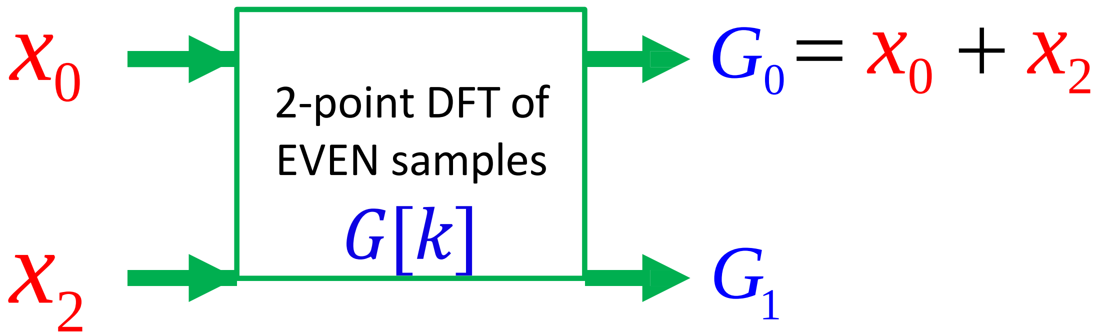
$x_0$

$x_2$

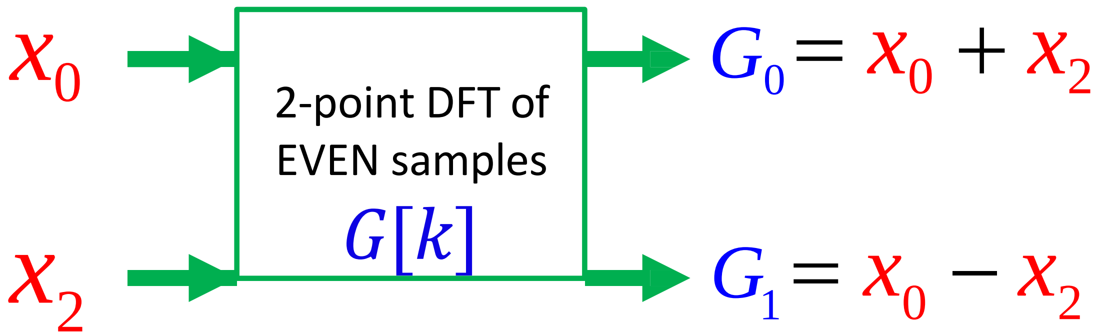
# 4-point DITFFT



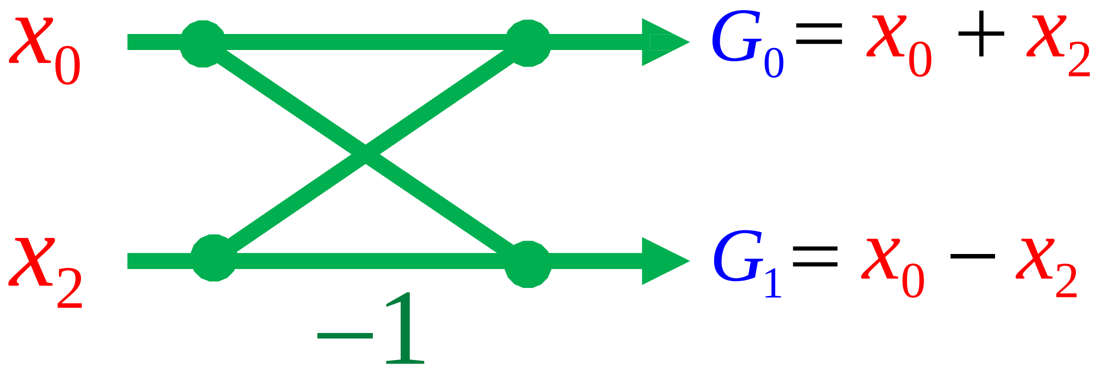
# 4-point DITFFT



# 4-point DITFFT



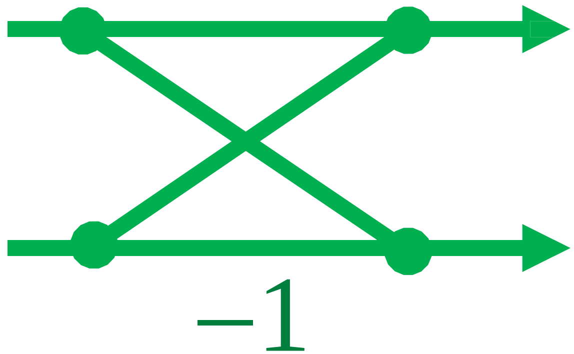
# 4-point DITFFT



# 4-point DITFFT

$x_0$

$x_2$



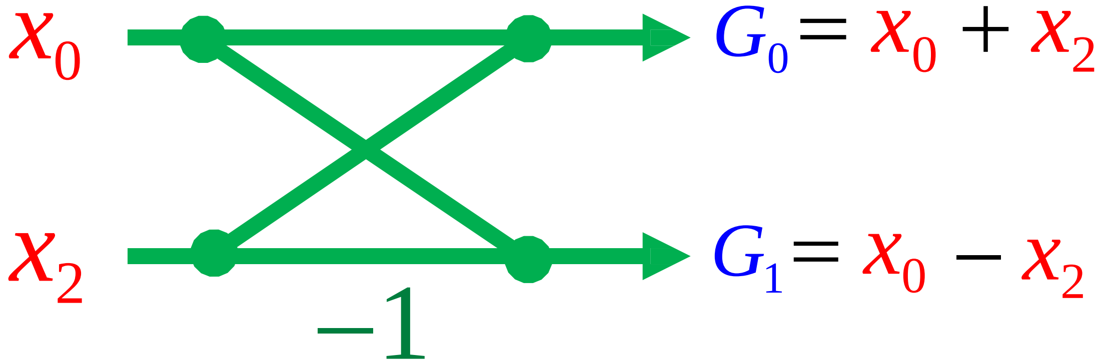

$$G_0 = x_0 + x_2$$

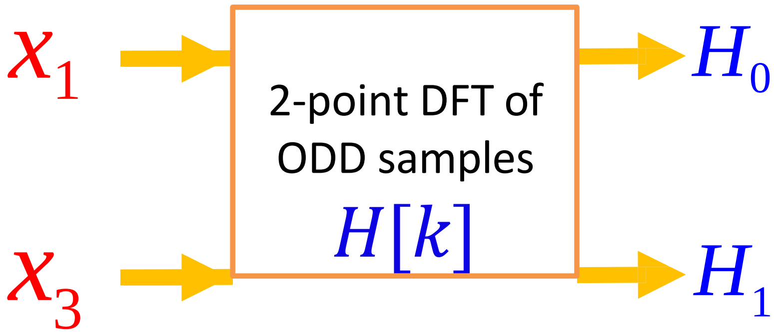
$$G_1 = x_0 - x_2$$

$x_1$

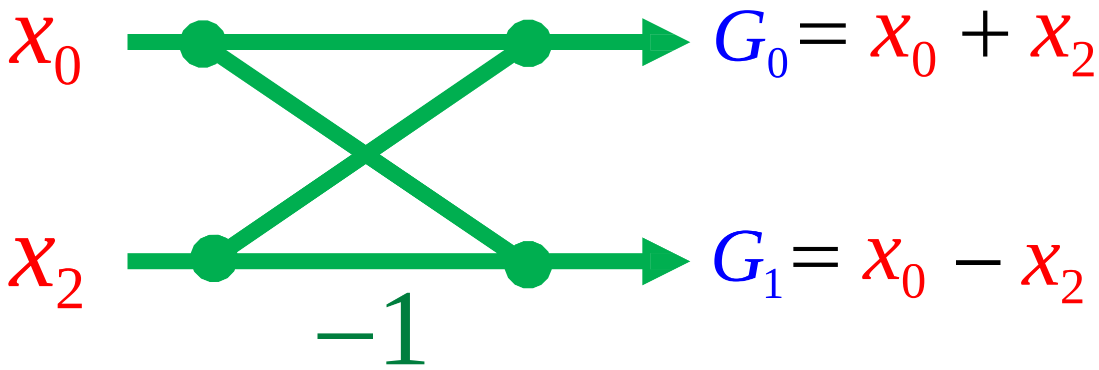
$x_3$

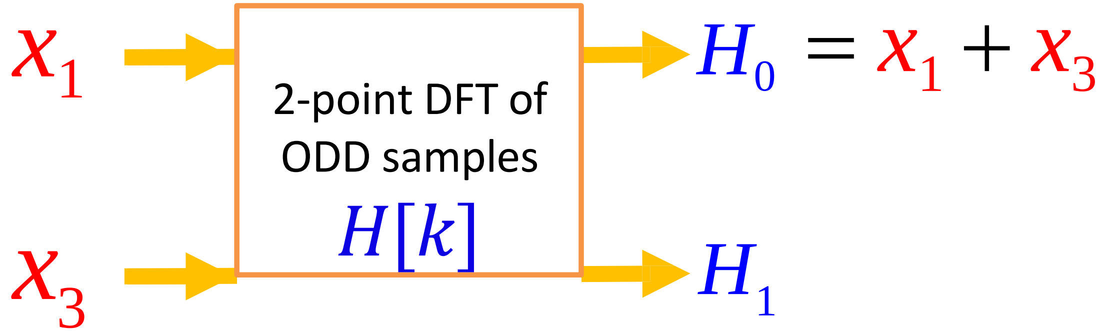
# 4-point DITFFT



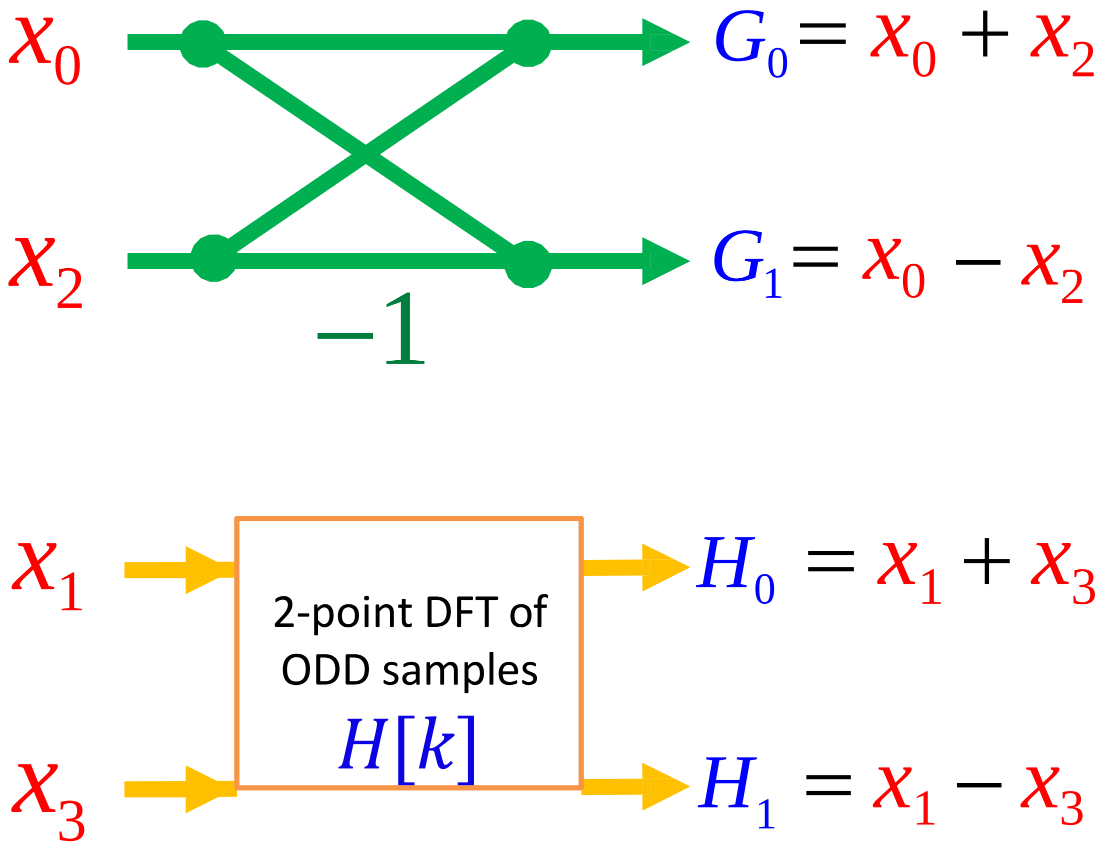



# 4-point DITFFT

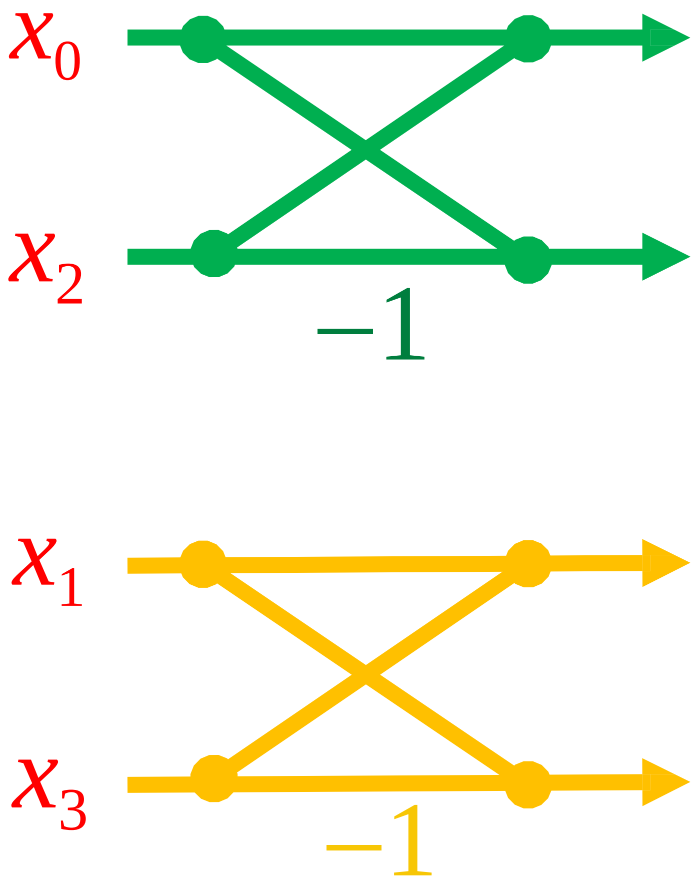


# 4-point DITFFT



# 4-point DITFFT




$$G_0 = x_0 + x_2$$

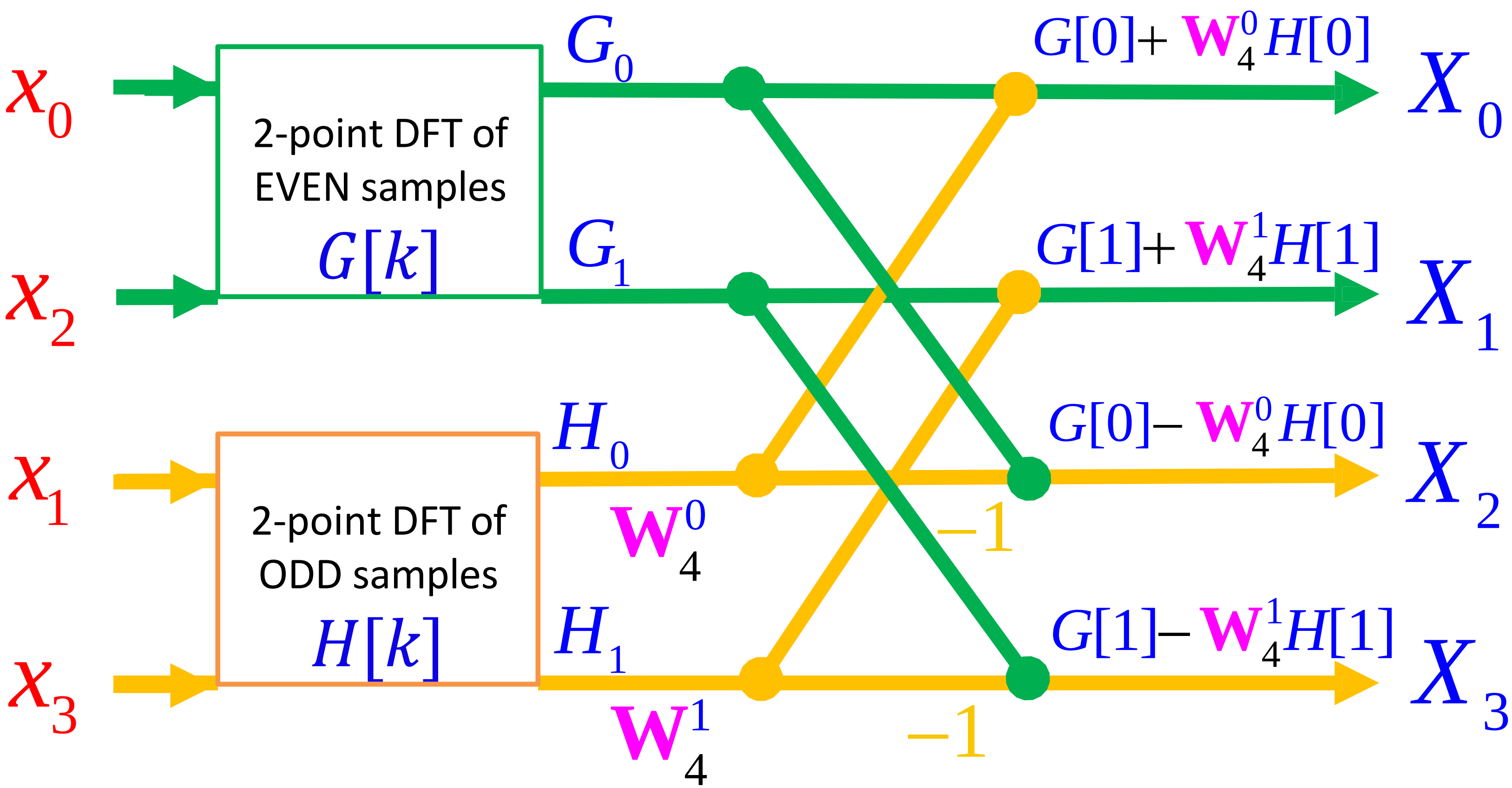
$$G_1 = x_0 - x_2$$

$$H_0 = x_1 + x_3$$

$$H_1 = x_1 - x_3$$

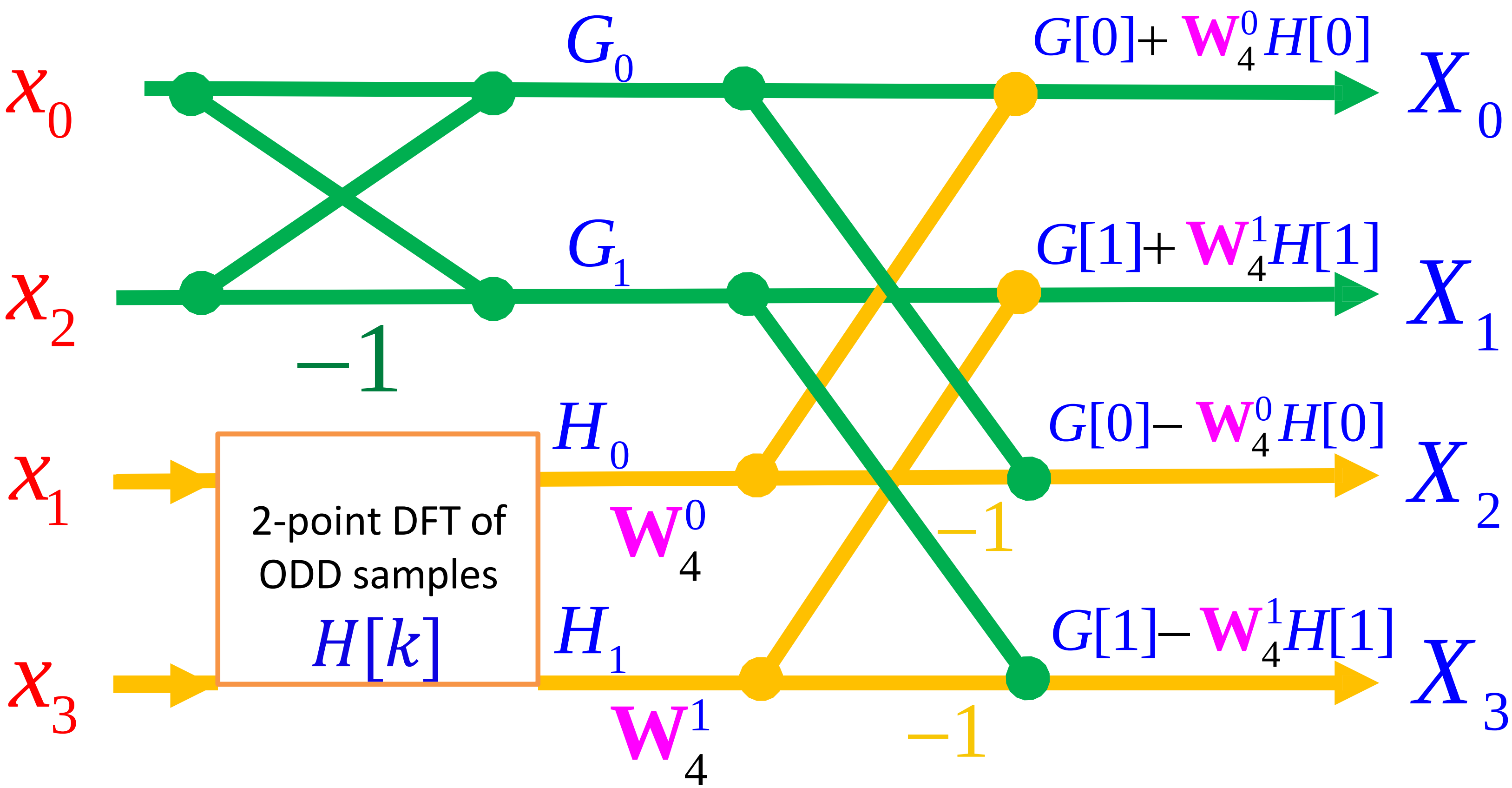
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



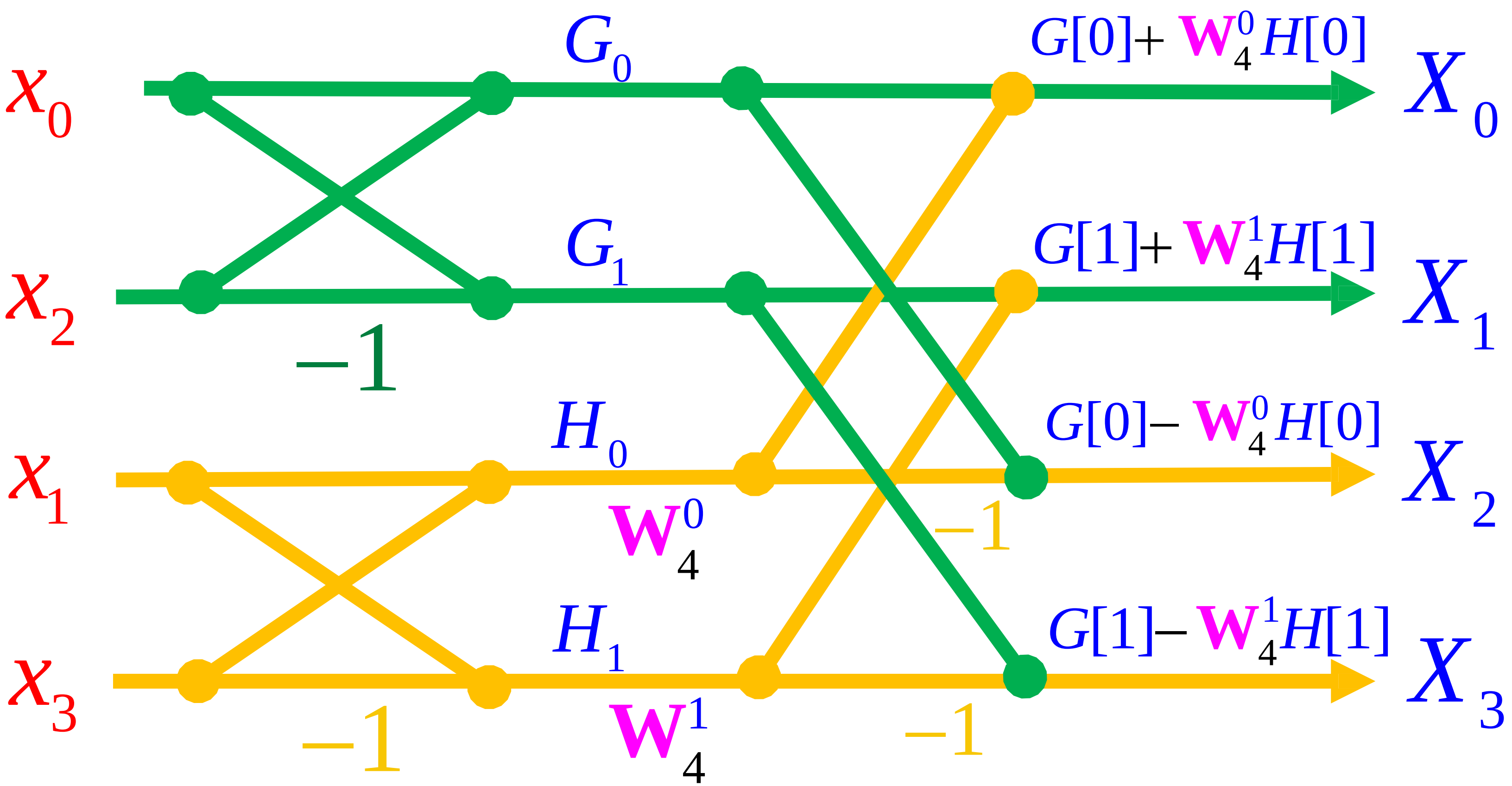
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



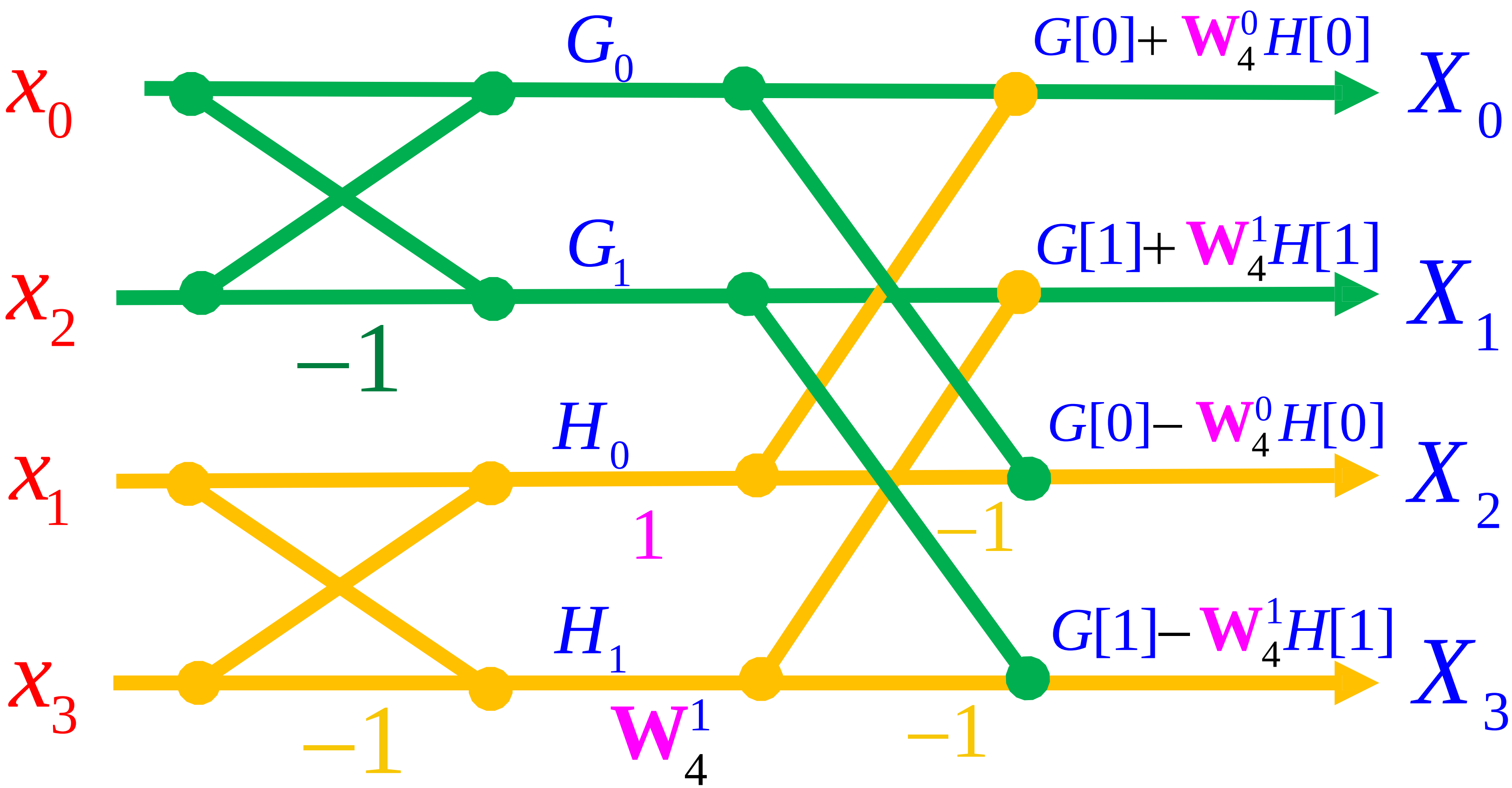
# 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



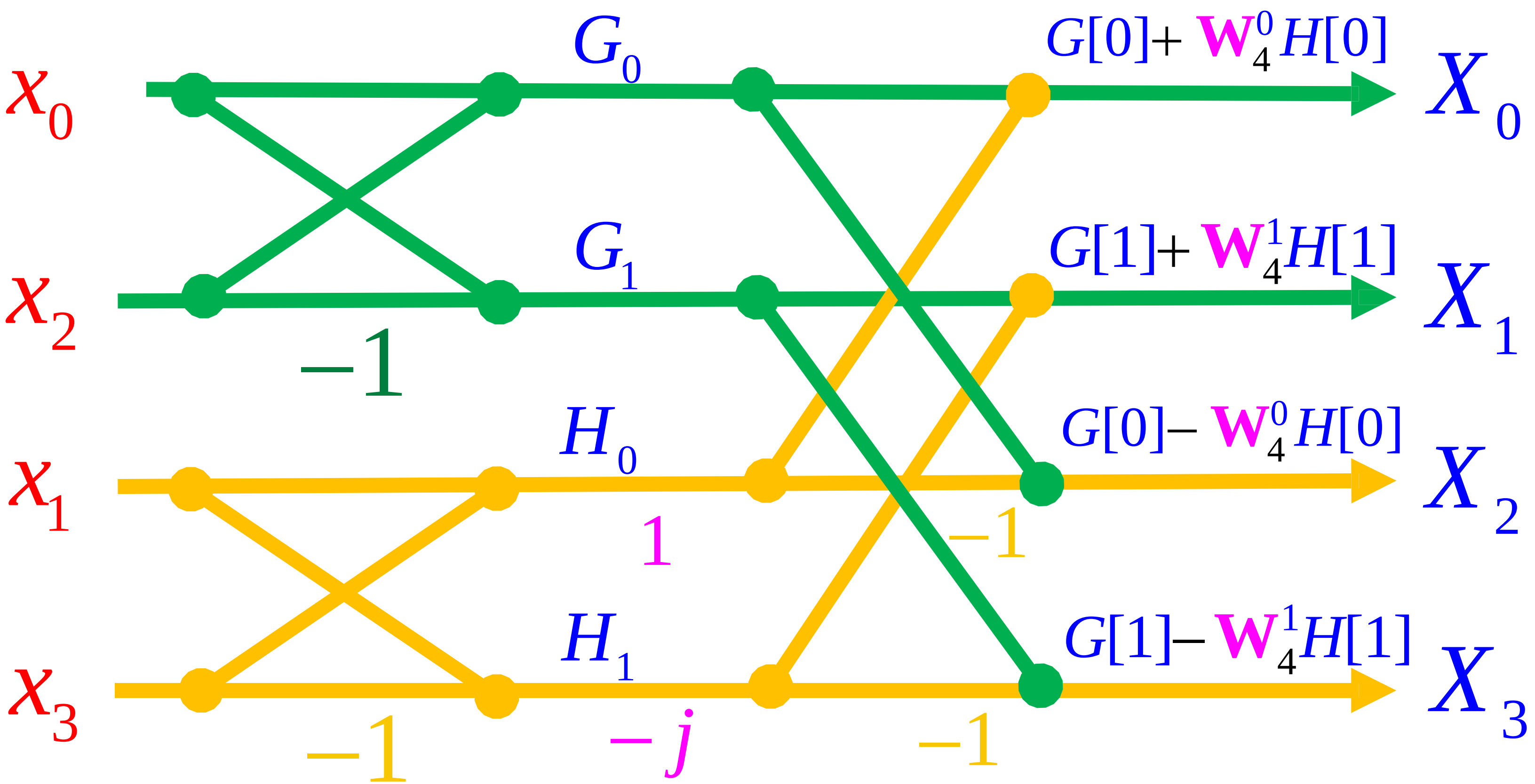
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



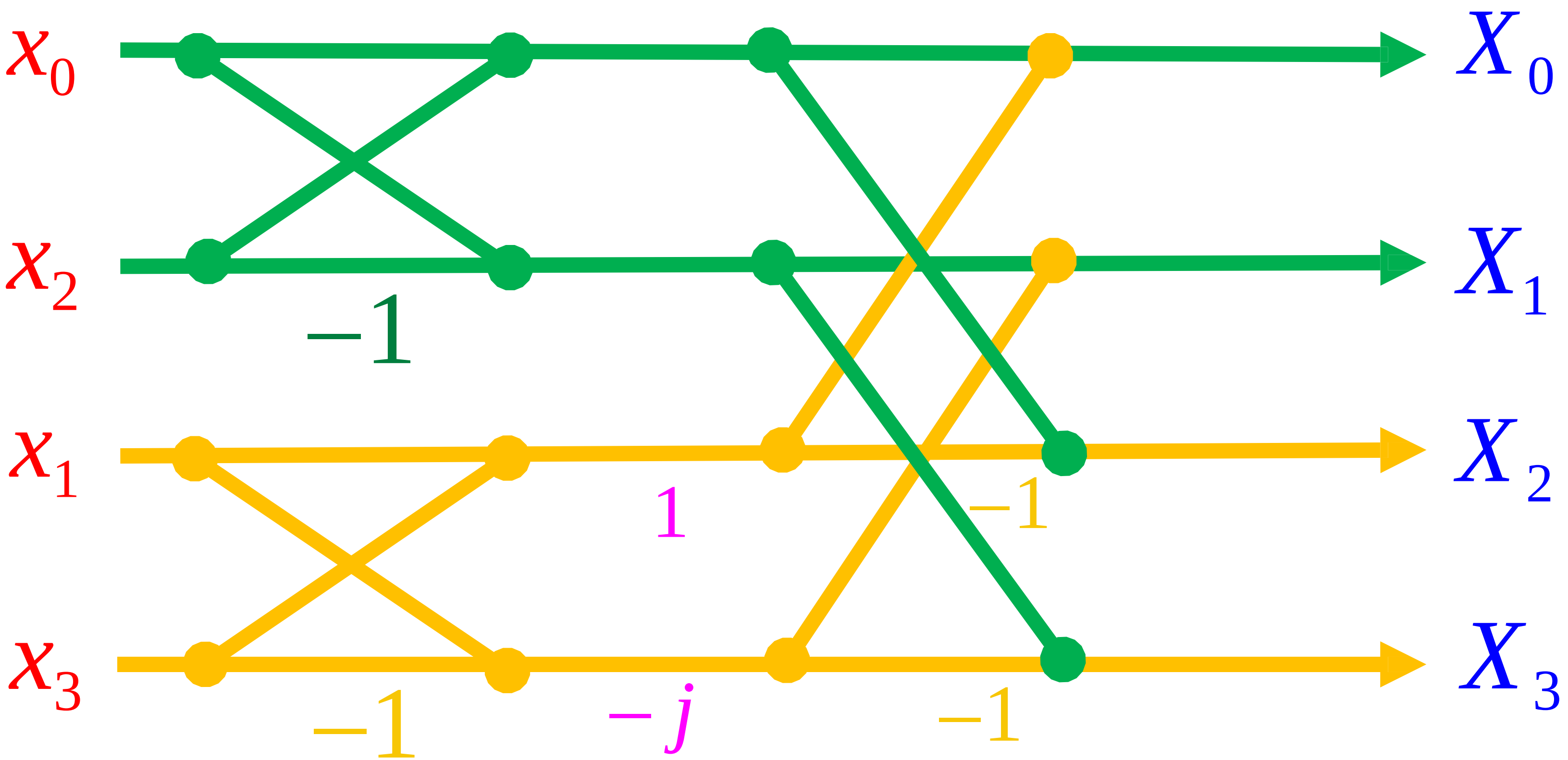
## 4-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\}$ and corresponding DFT $X(k) = \{X_0, X_1, X_2, X_3\}$



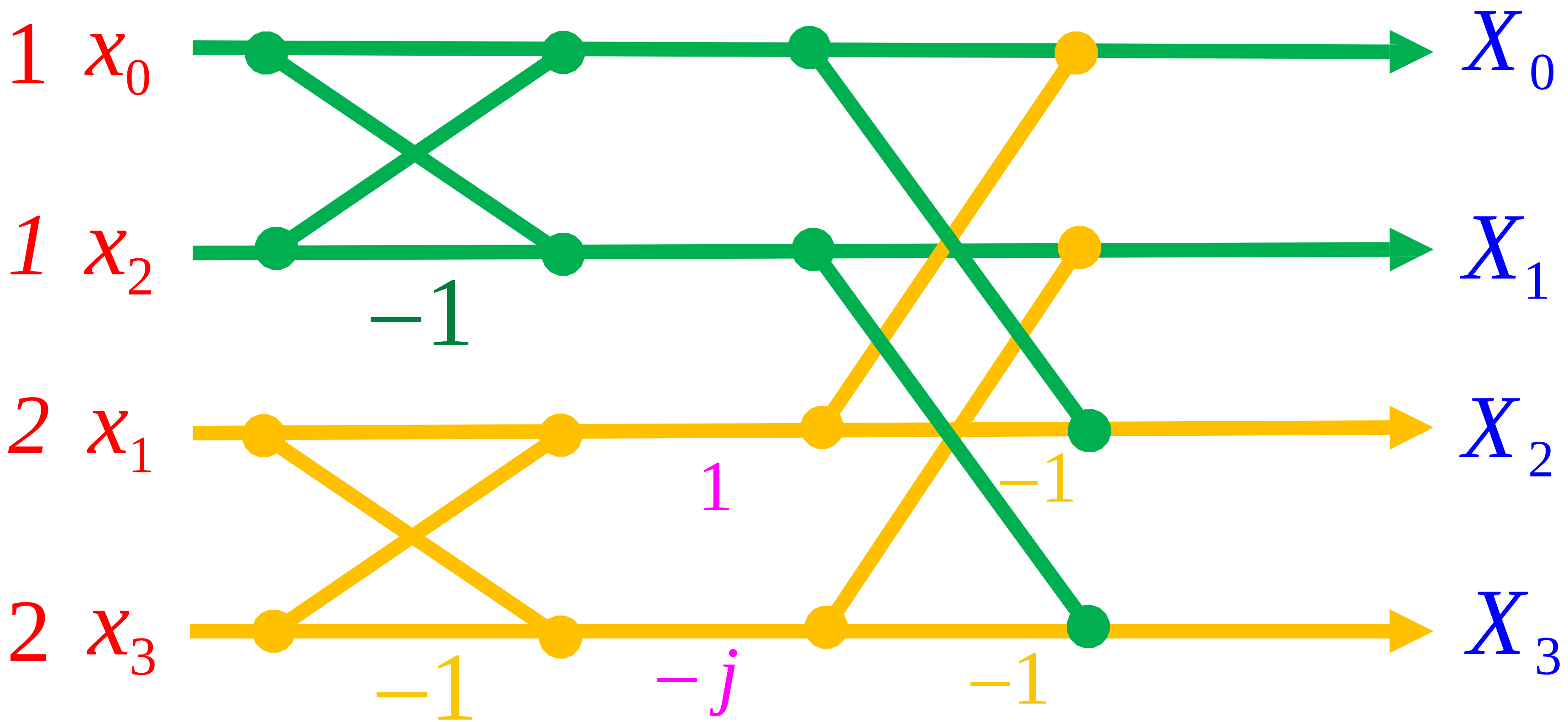
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT (DEC18, 4marks)



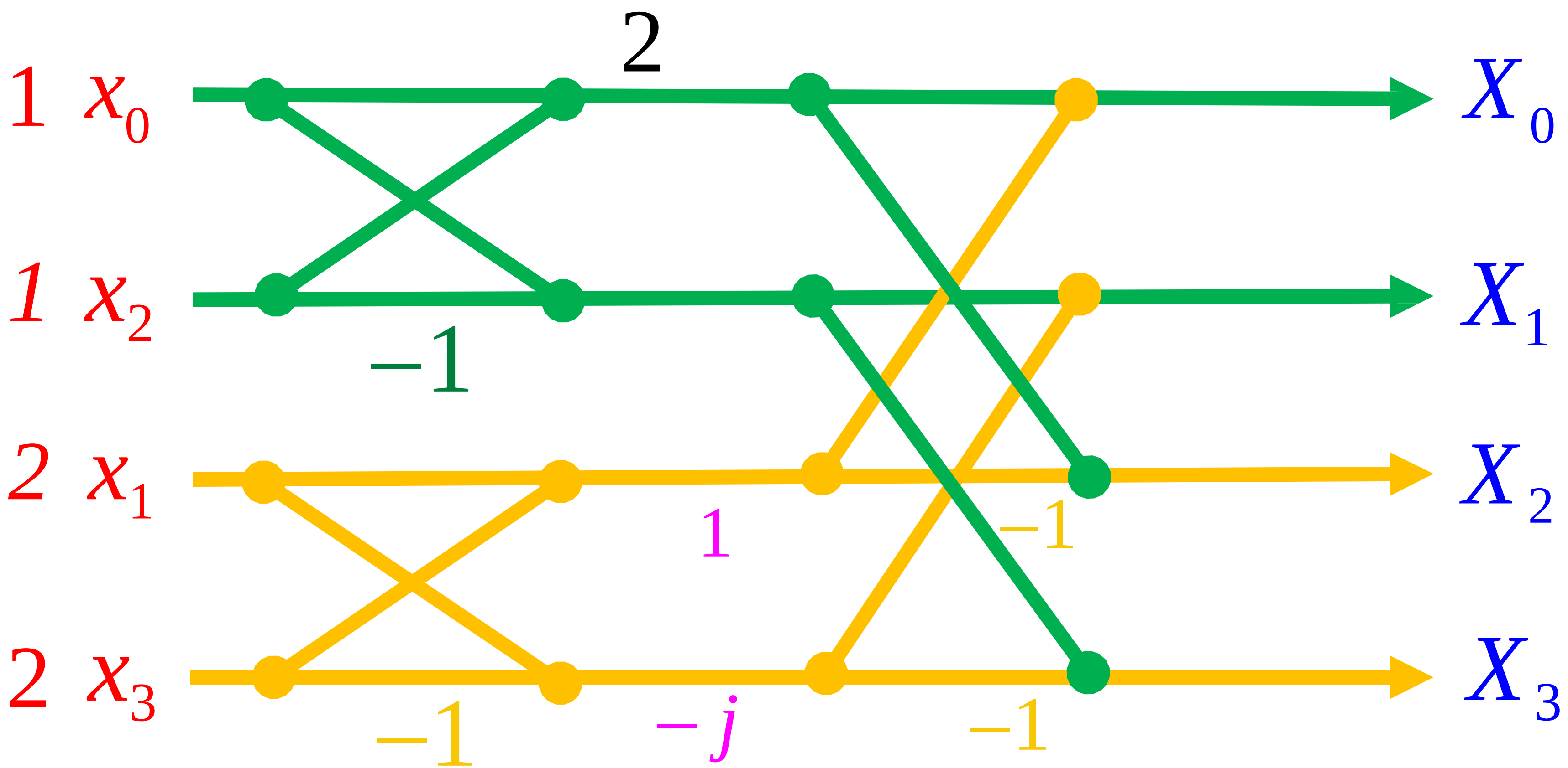
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT



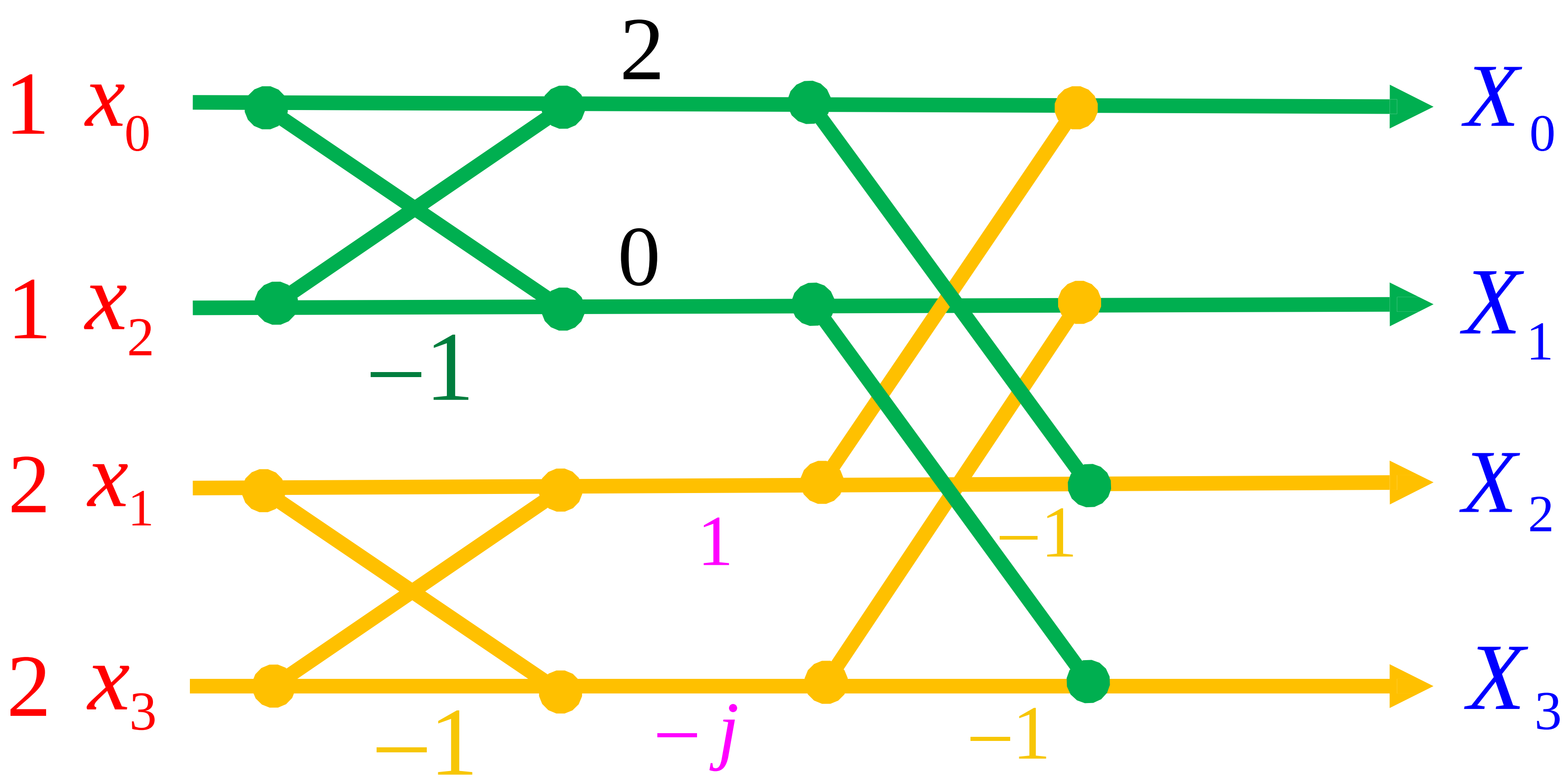
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT



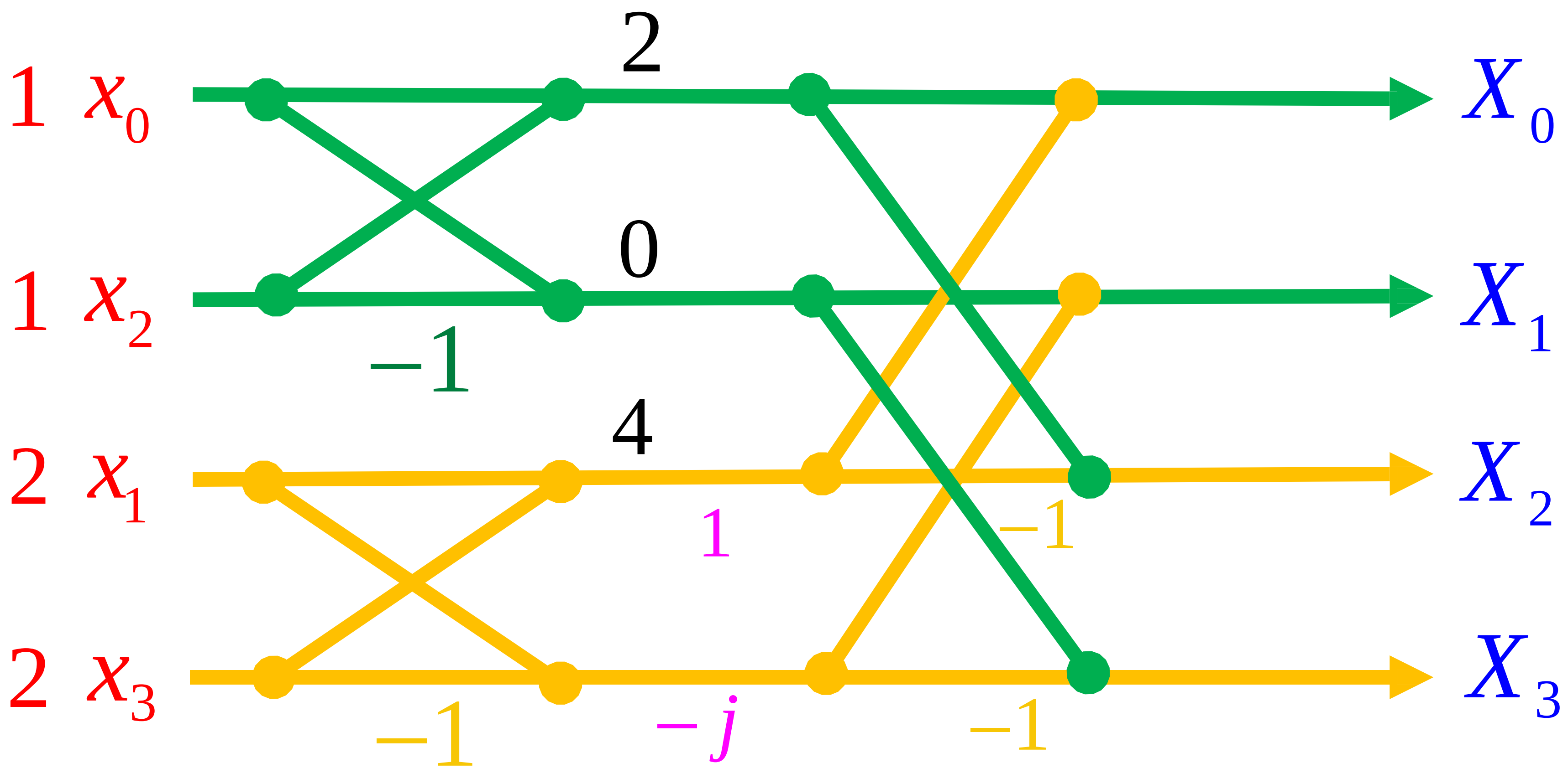
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT

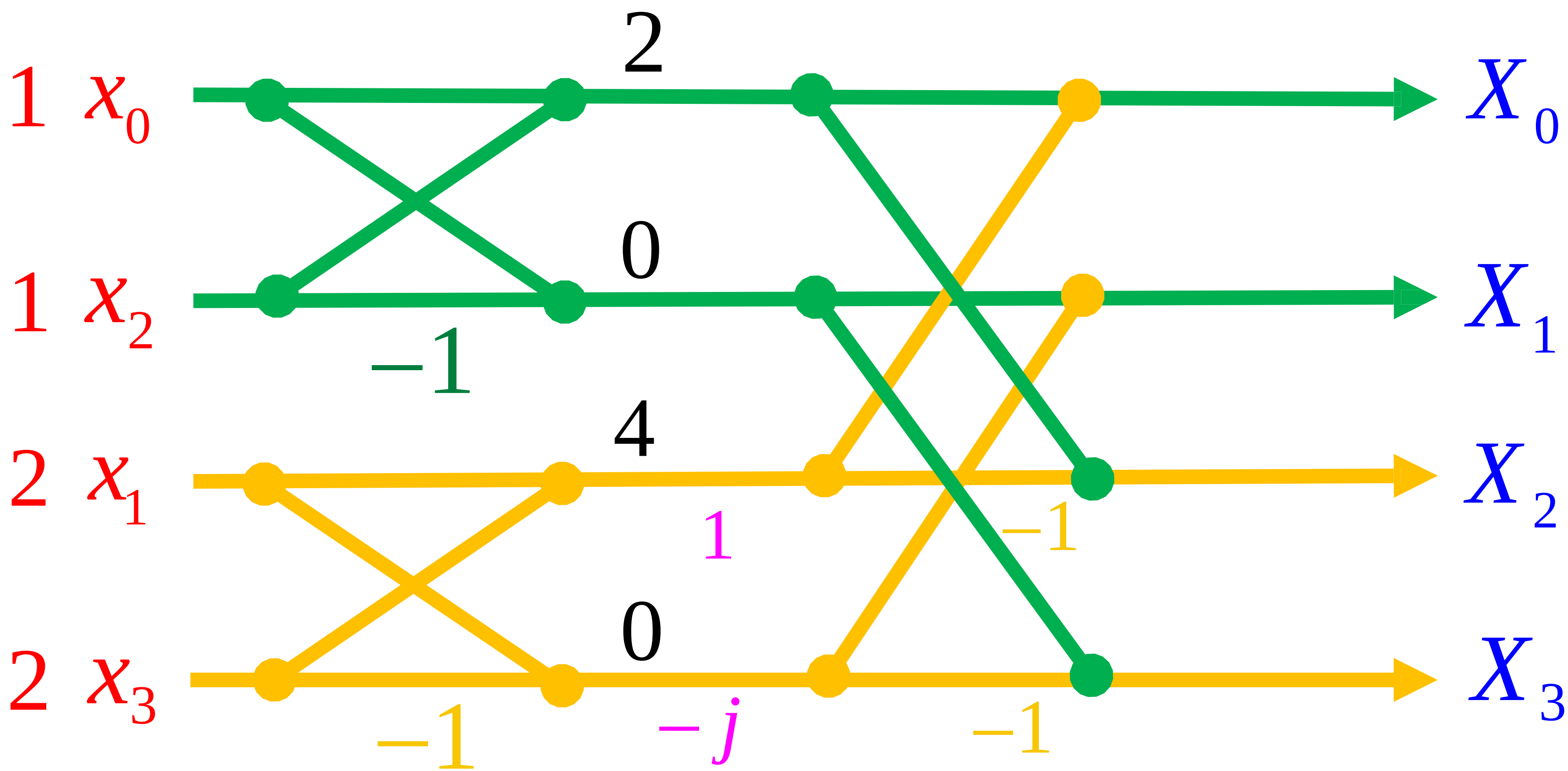


## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT

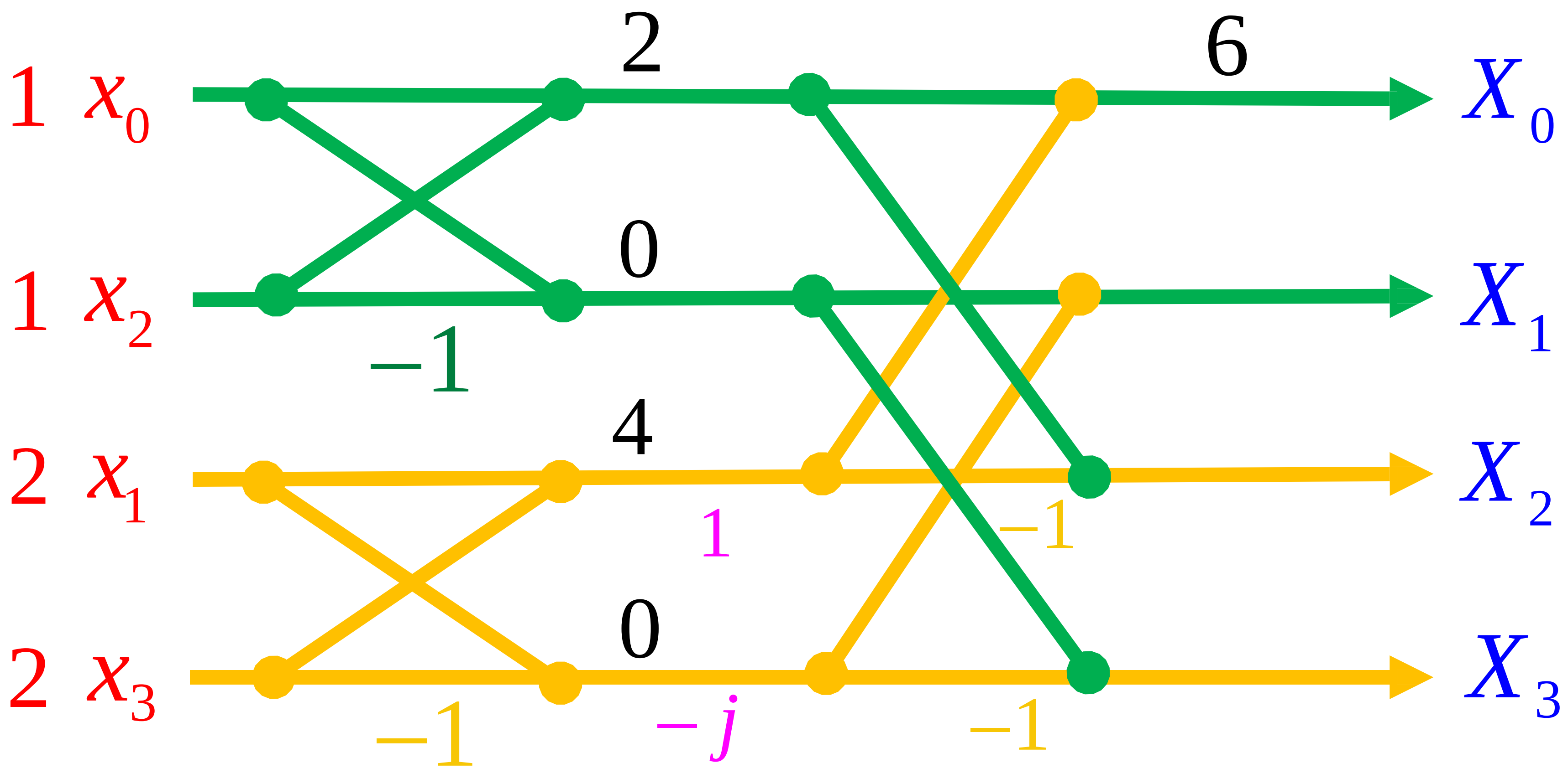


- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT



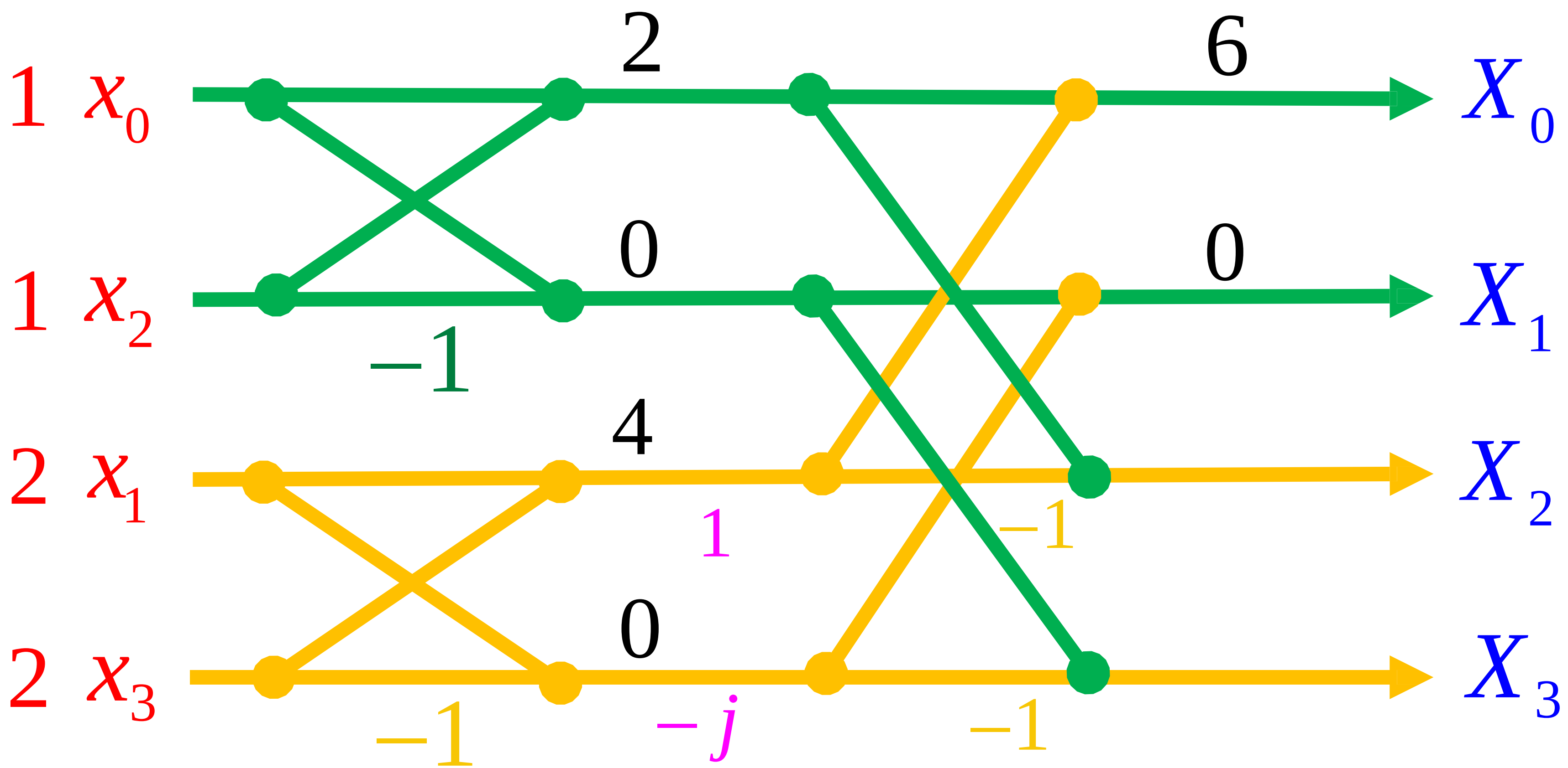
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT



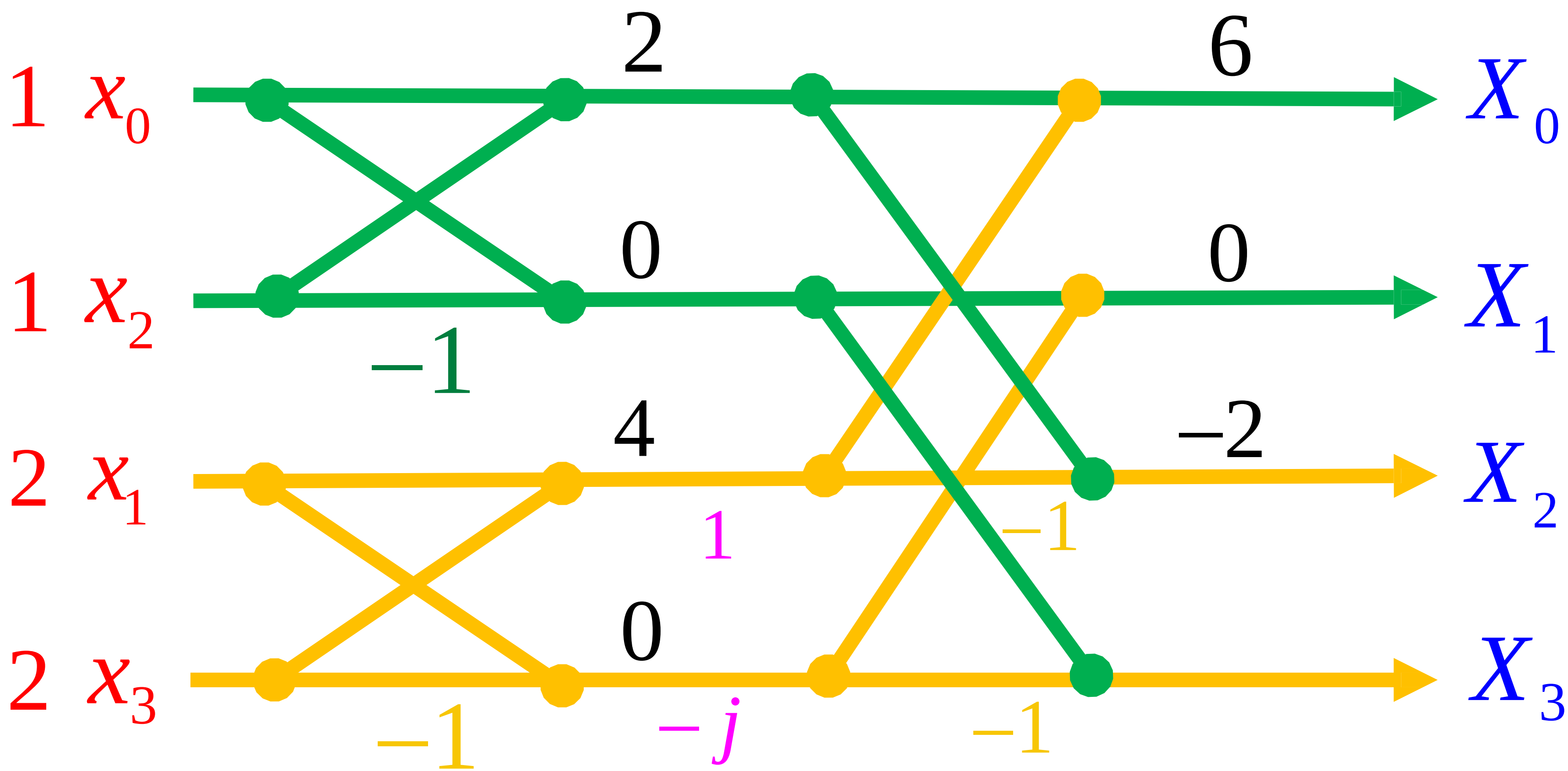
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT)



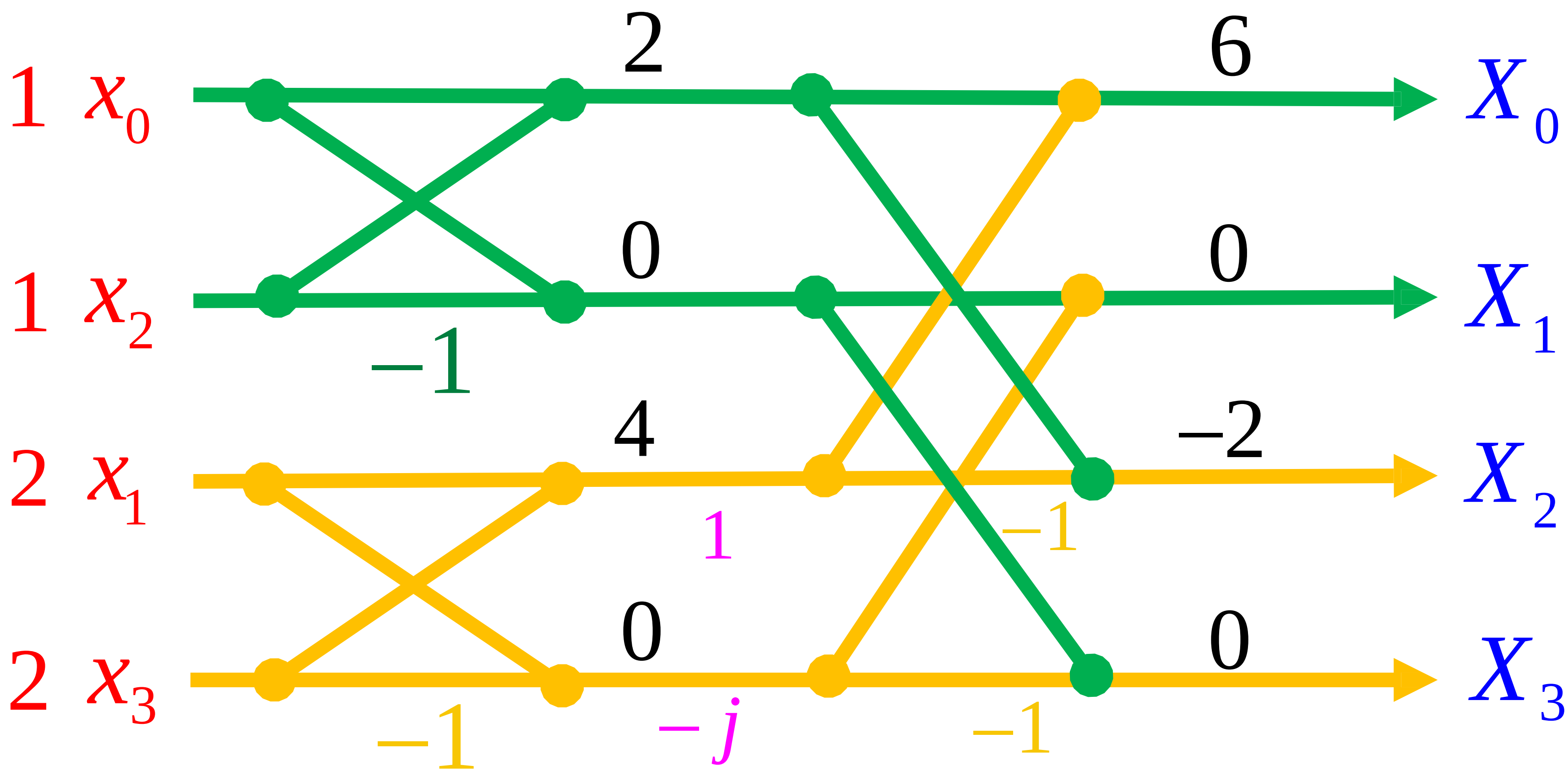
## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT

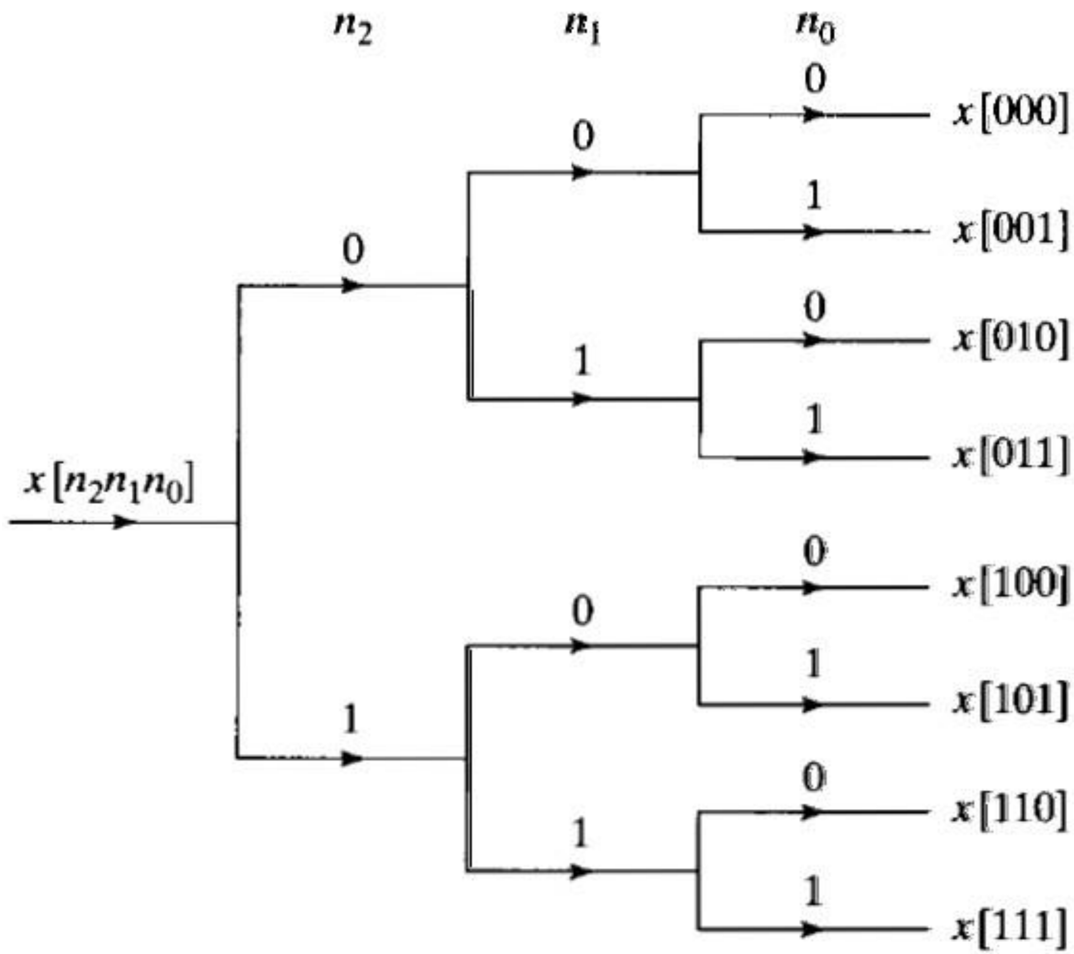


## Example

- Find DFT of $x[n] = \{1, 2, 1, 2\}$ using DITFFT



# Bit-reversed order




$$X_0[000] = x[000]$$
$$X_0[001] = x[100]$$
$$X_0[010] = x[010]$$
$$X_0[011] = x[110]$$
$$X_0[100] = x[001]$$
$$X_0[101] = x[101]$$
$$X_0[110] = x[011]$$
$$X_0[111] = x[111]$$

Tree diagram depicting normal order sorting *[Source: Chapter 9, Digital Signal Processing* by Prokakis]

# 8-point DITFFT

- Let $x[n] = \{x_0,\ x_1\ , x_2\ , x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

$x_0$

$x_4$

$x_2$

$x_6$

$x_1$

$x_5$

$x_3$

$x_7$

# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

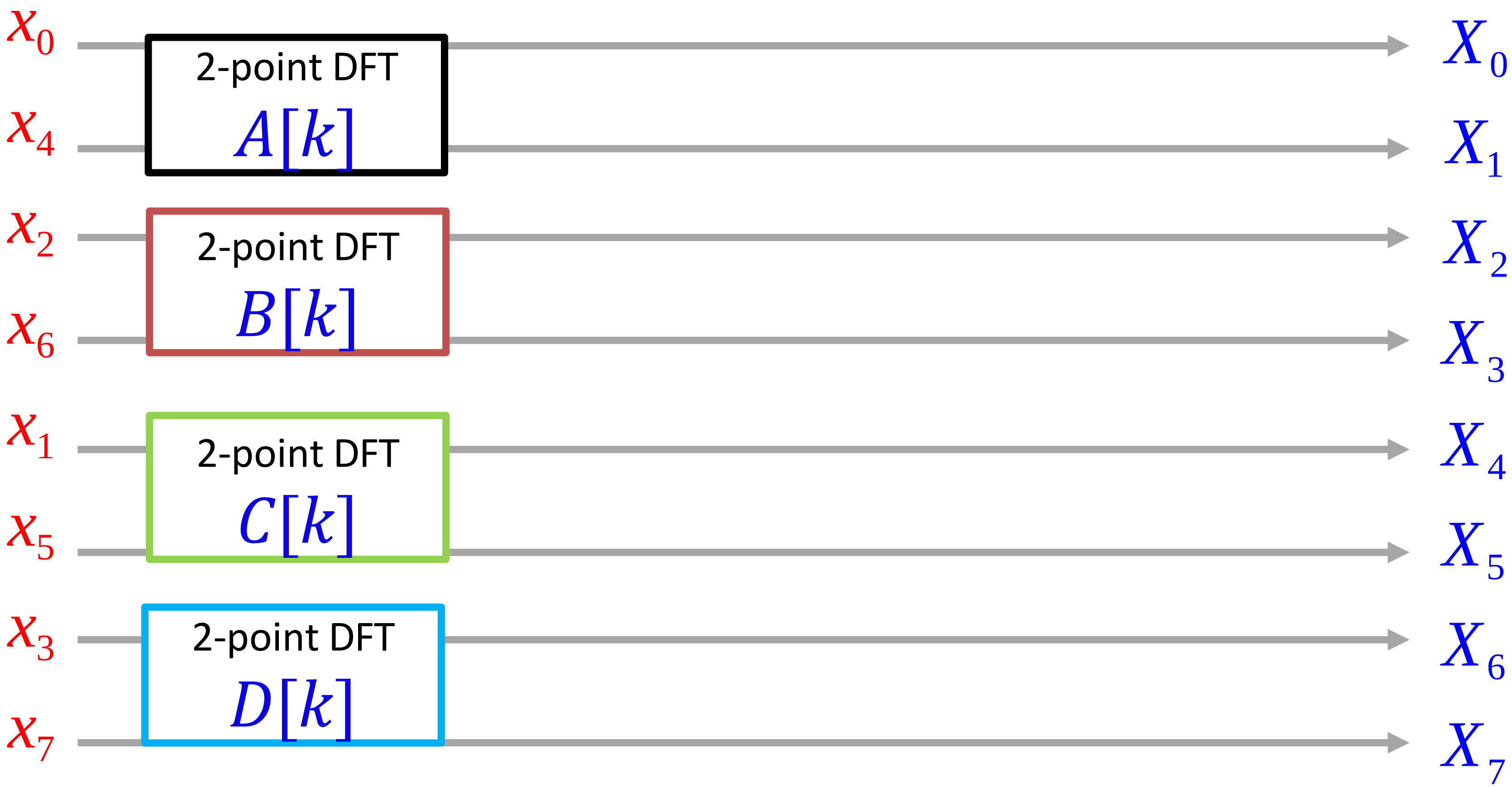
| | |
|---|---|
| $x_0$ | $X_0$ |
| $x_4$ | $X_1$ |
| $x_2$ | $X_2$ |
| $x_6$ | $X_3$ |
| $x_1$ | $X_4$ |
| $x_5$ | $X_5$ |
| $x_3$ | $X_6$ |
| $x_7$ | $X_7$ |

## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

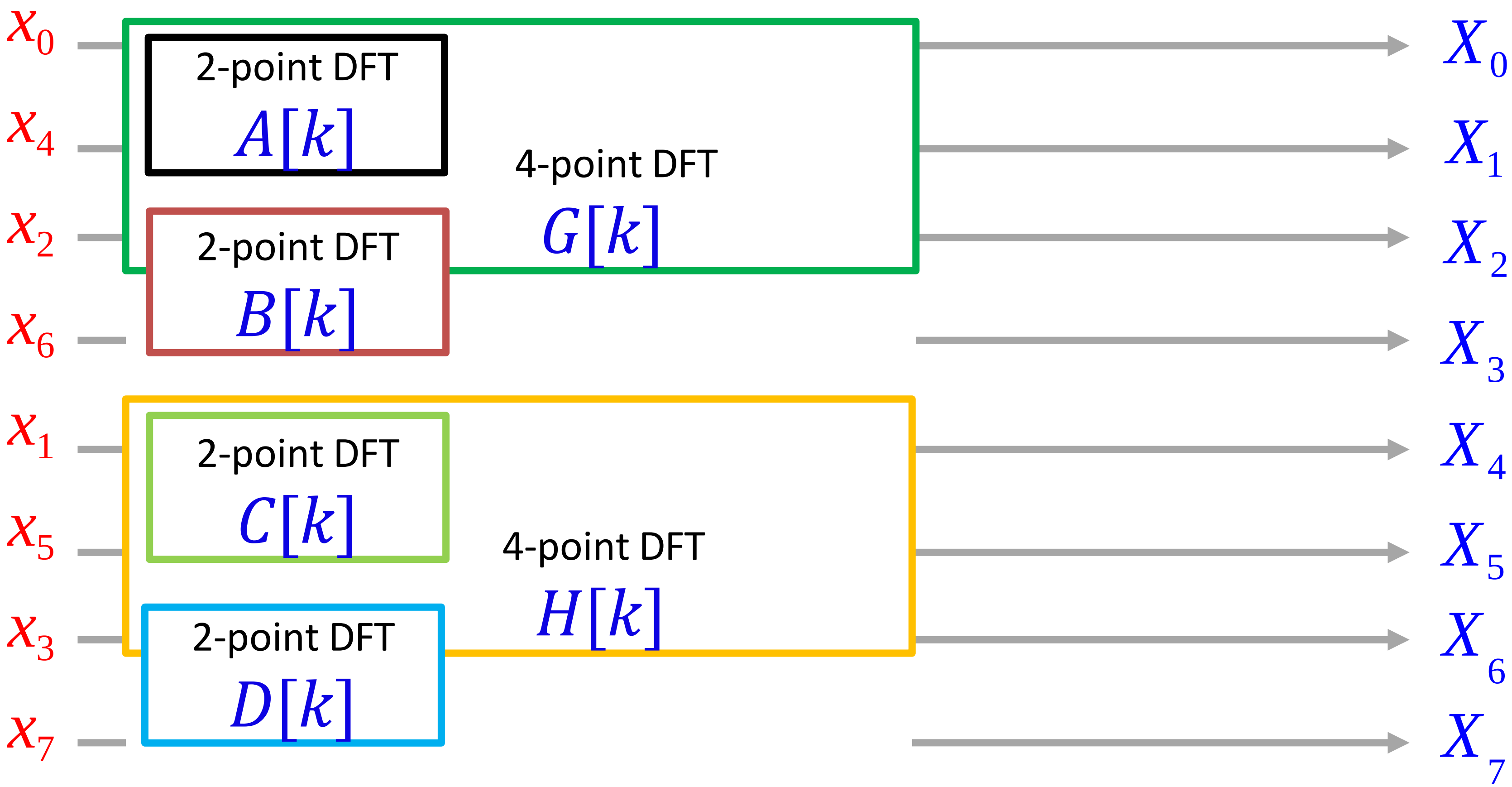
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



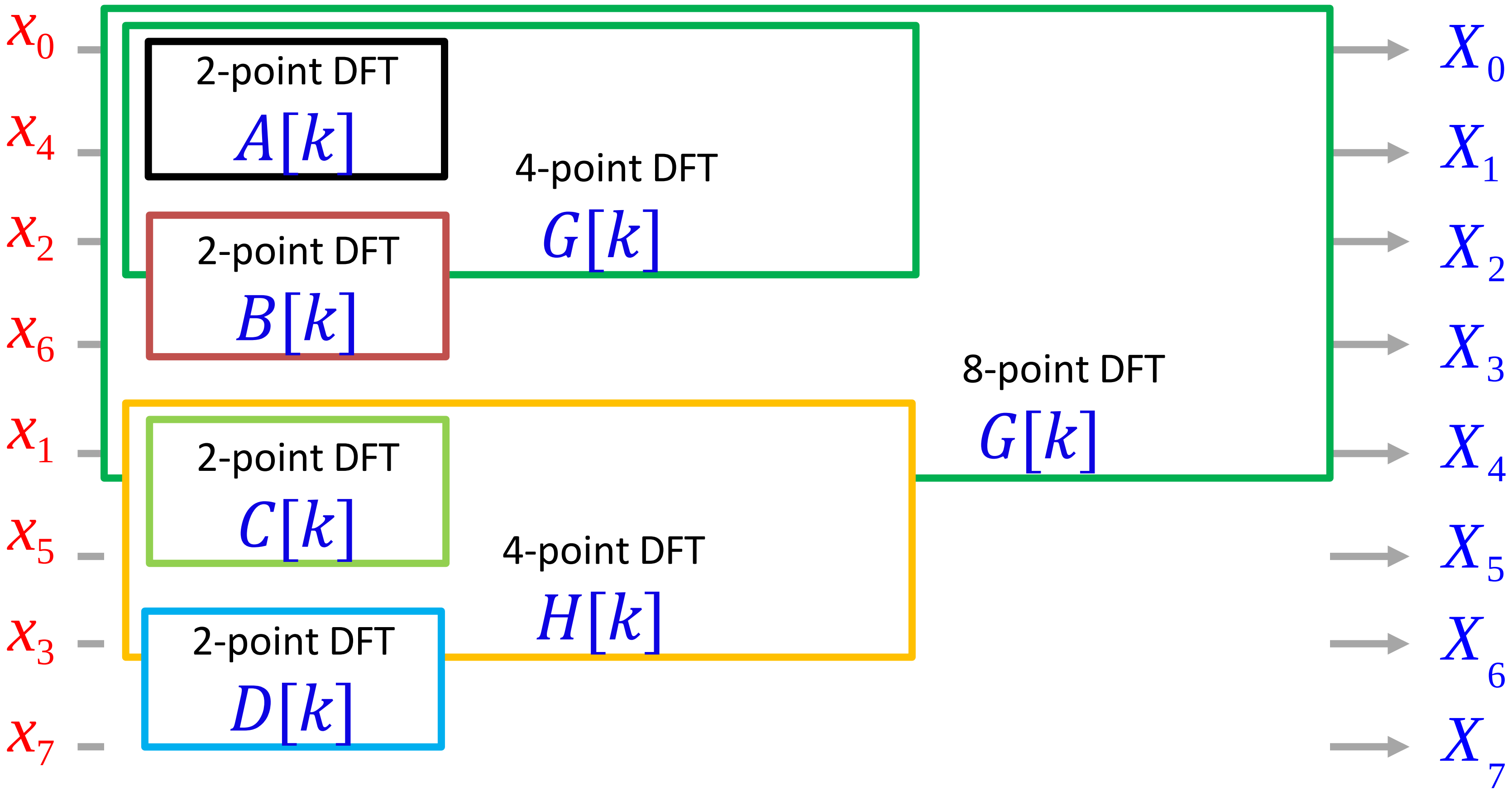
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



# 8-point DITFFT

- Let $x[n] = \{x_0,\ x_1\,, x_2\ , x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

$x_0$

$x_4$

$x_2$

$x_6$

$x_1$

$x_5$

$x_3$

$x_7$

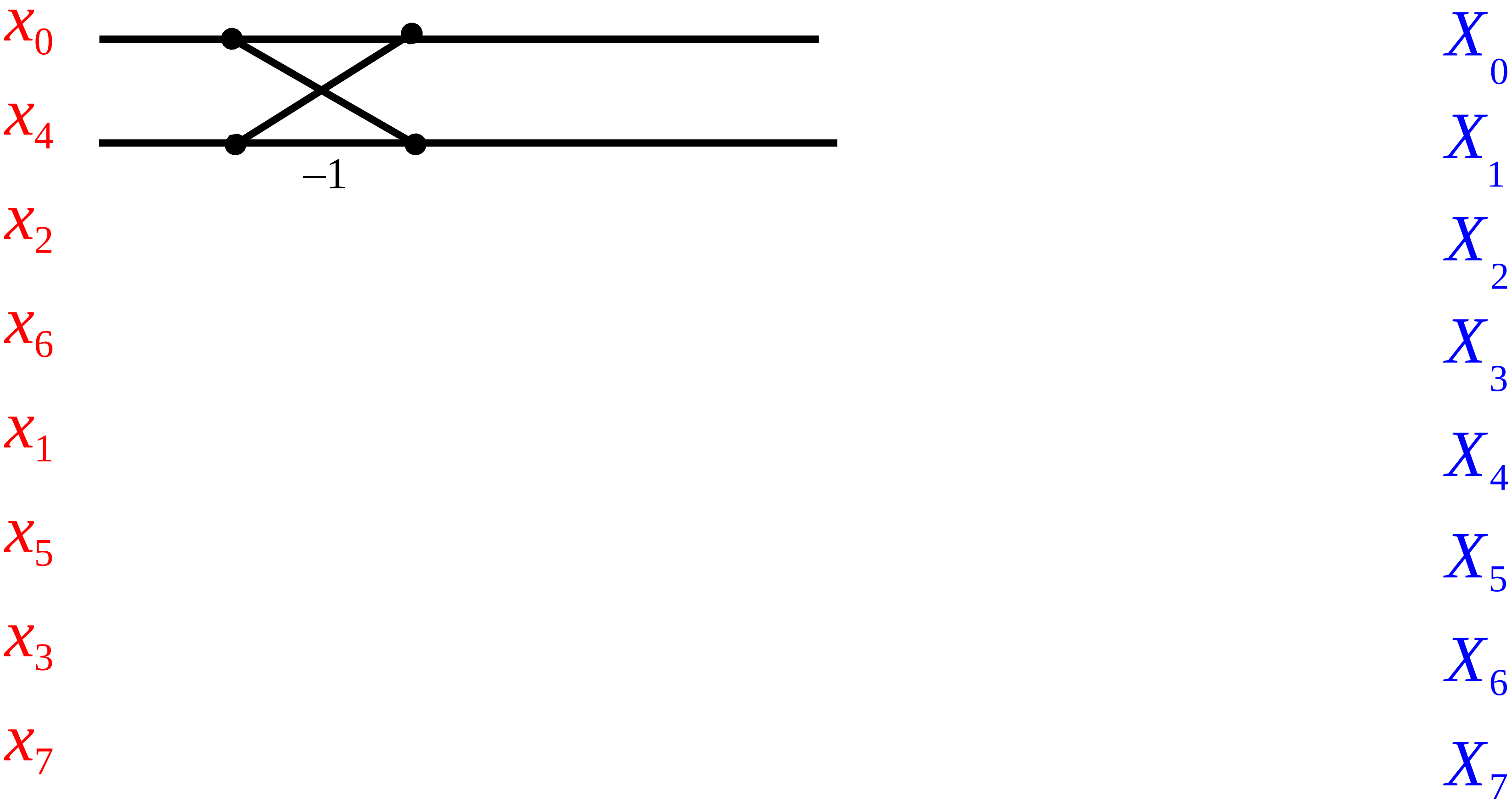
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$

| | |
|---|---|
| $x_0$ | $X_0$ |
| $x_4$ | $X_1$ |
| $x_2$ | $X_2$ |
| $x_6$ | $X_3$ |
| $x_1$ | $X_4$ |
| $x_5$ | $X_5$ |
| $x_3$ | $X_6$ |
| $x_7$ | $X_7$ |

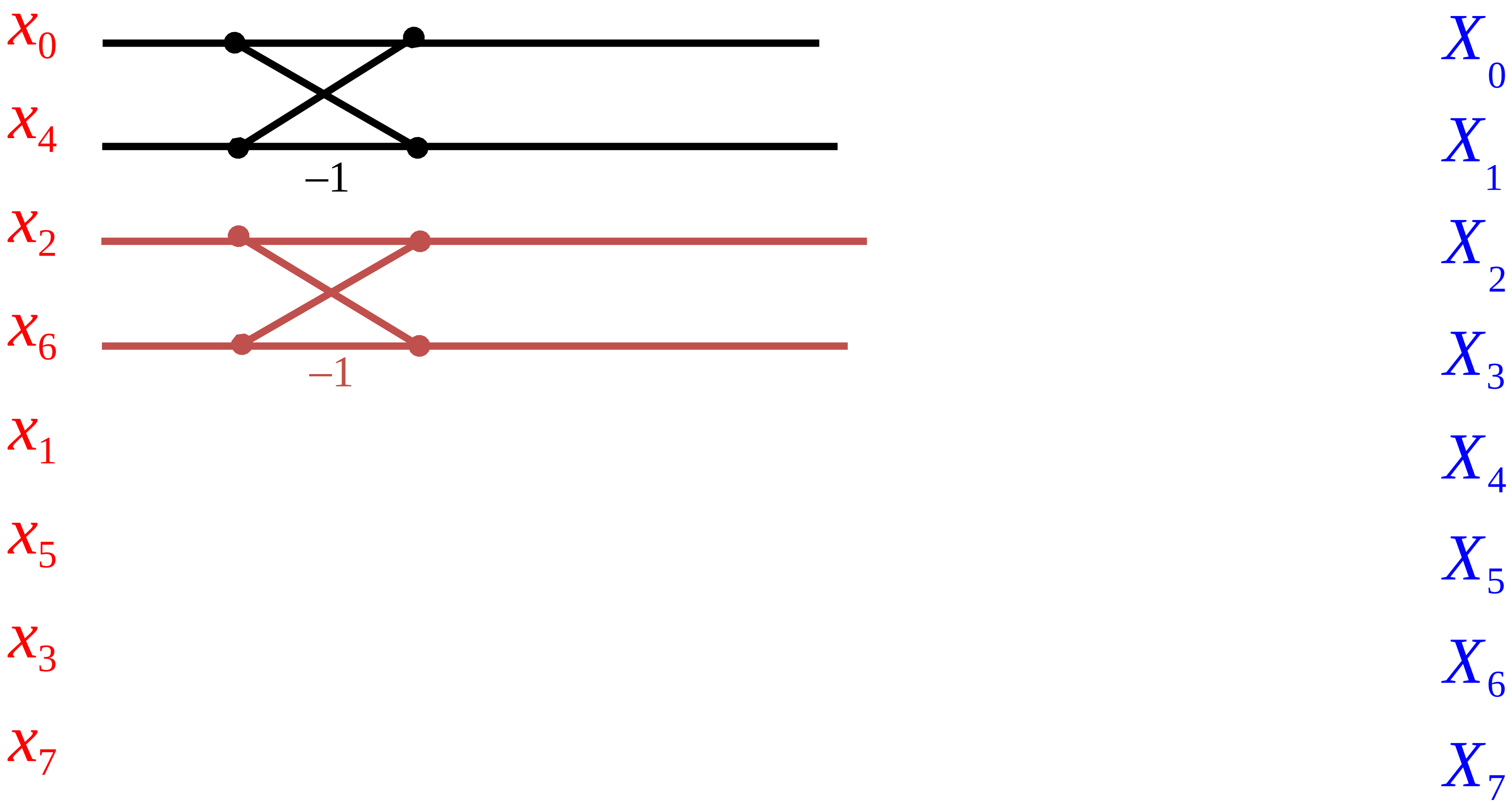
# 8-point DITFFT

- Let $x[n] = \{x_0,\ x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



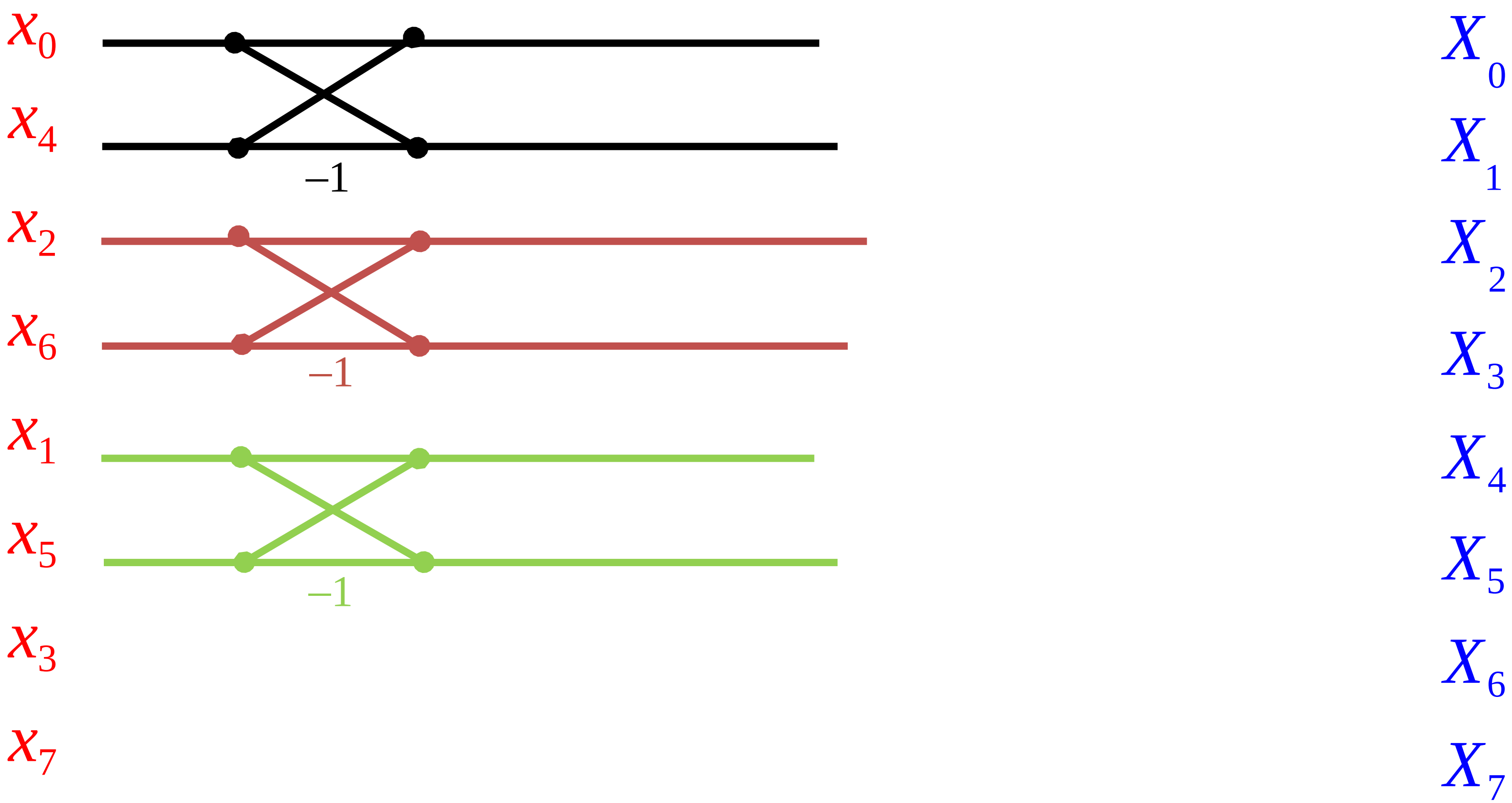
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



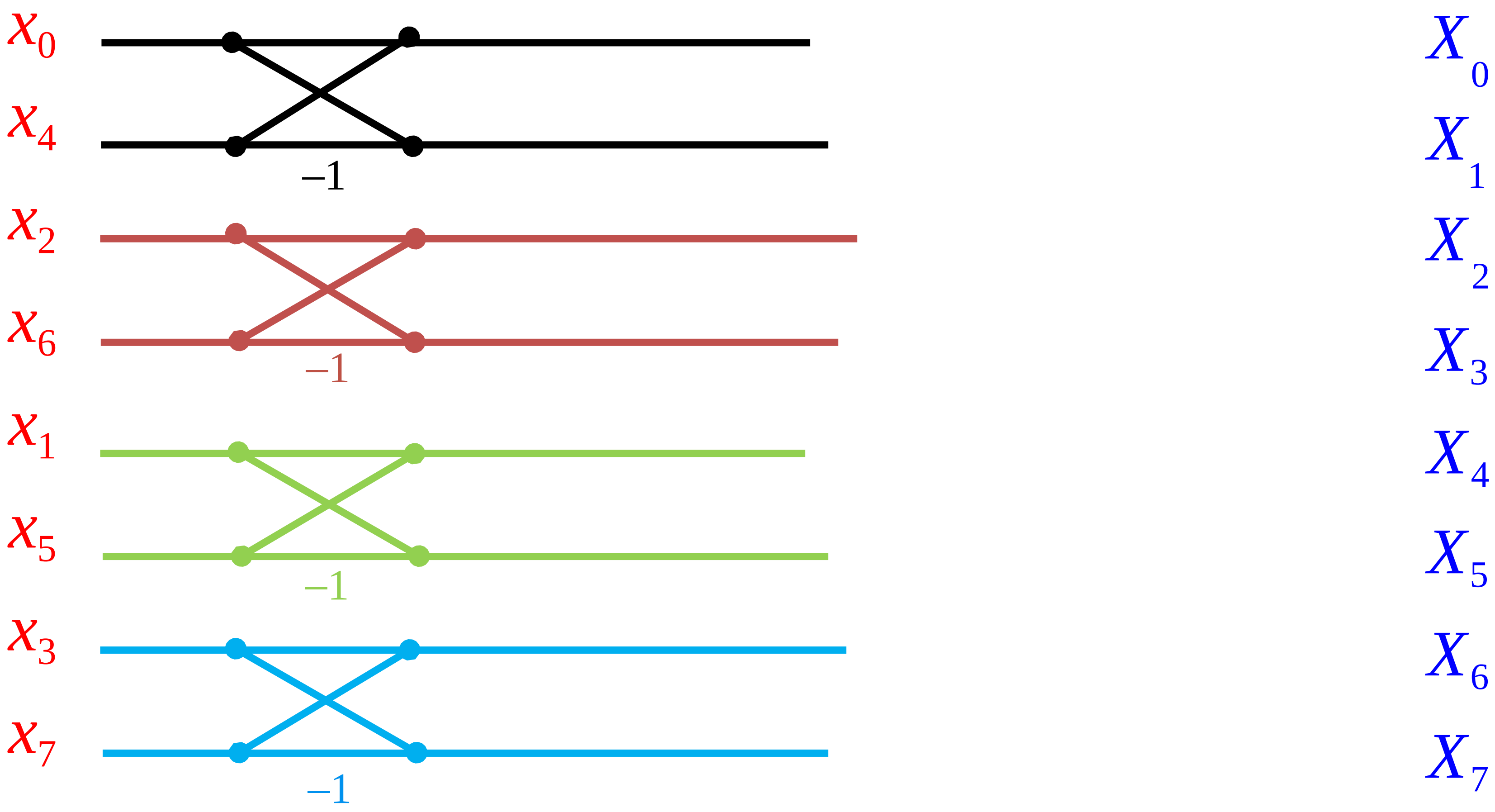
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



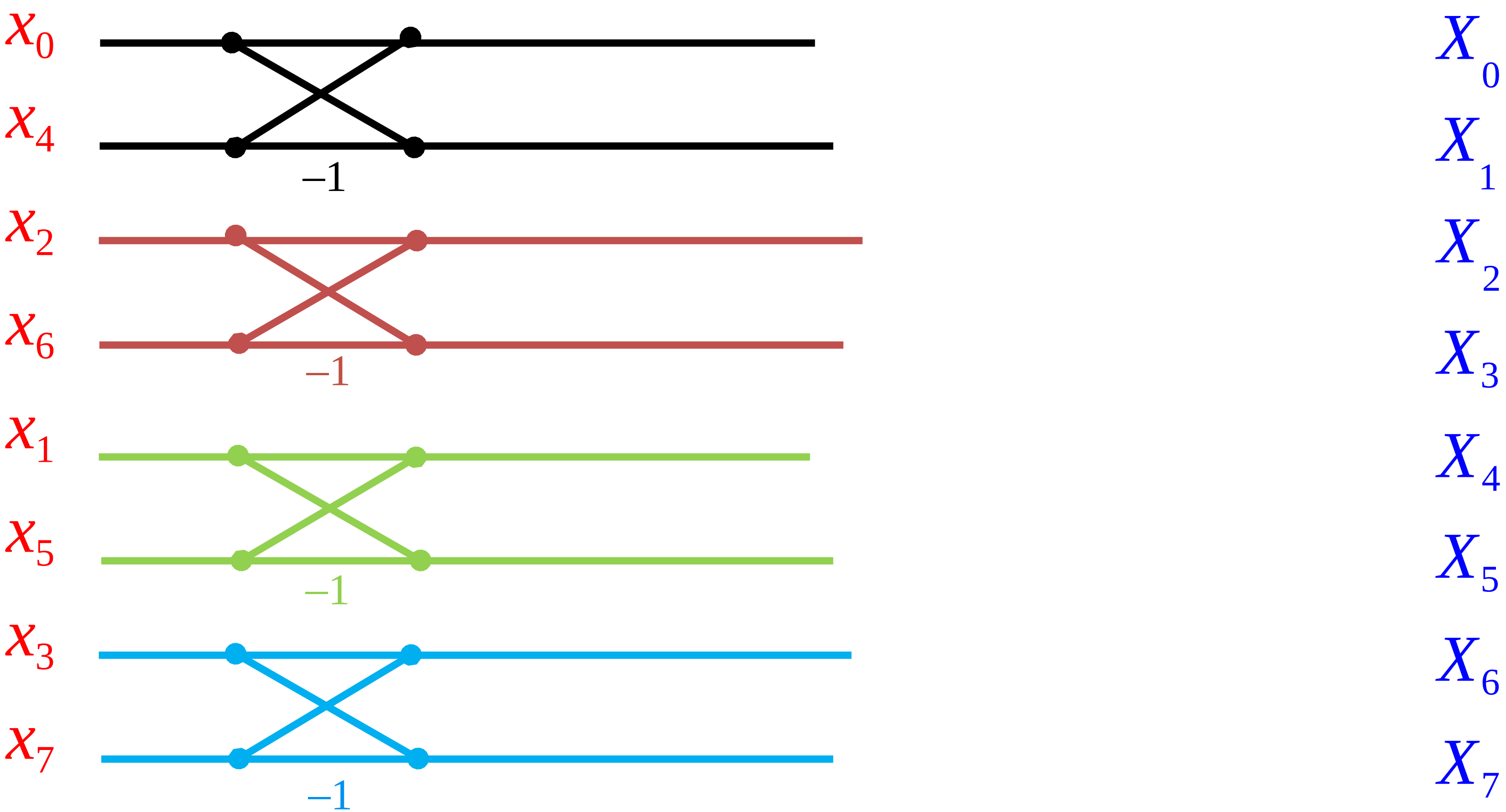
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



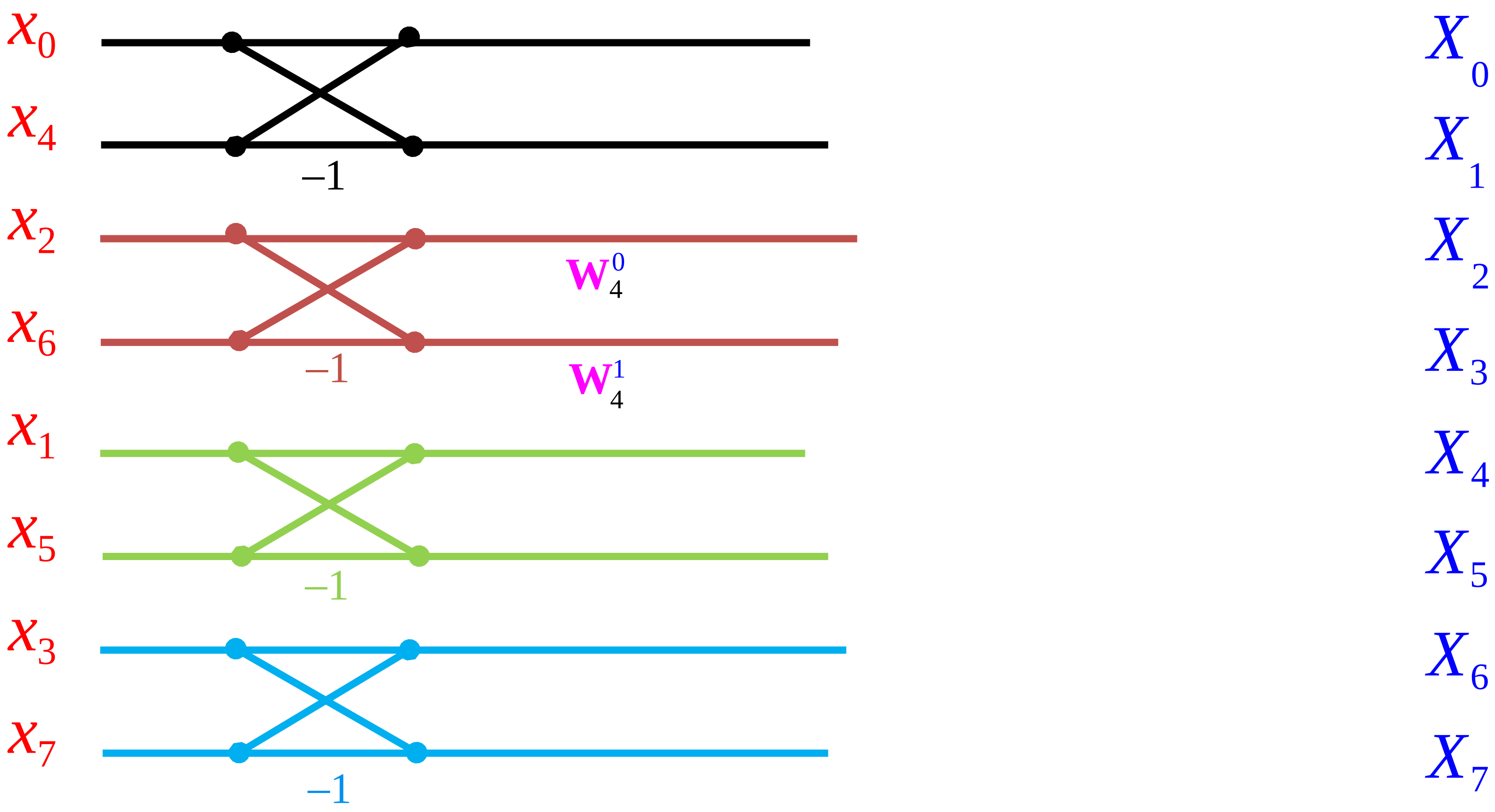
## 8-point DITFFT

- Let $x(n) = x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



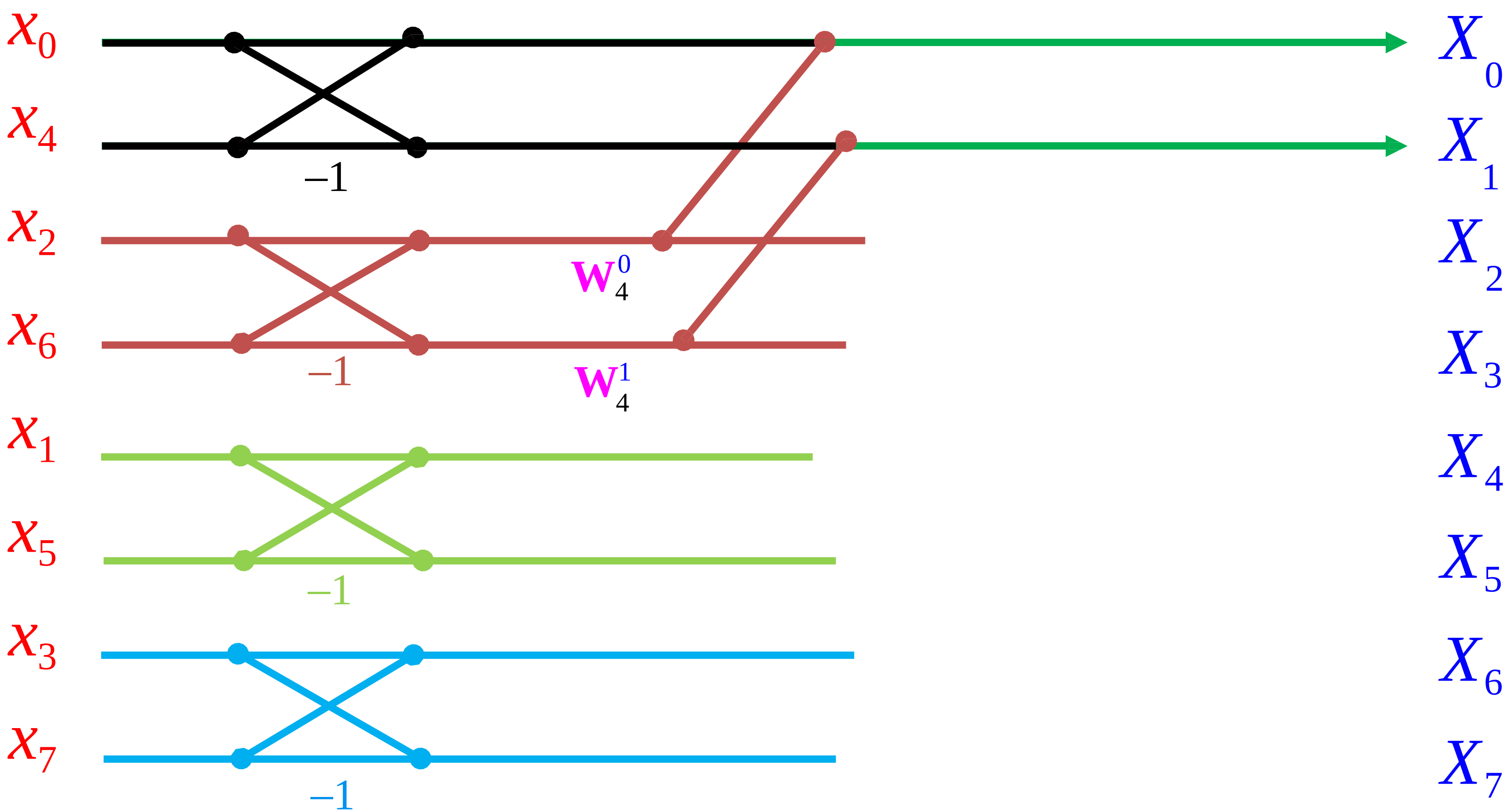
## 8-point DITFFT

- Let $x(n) = x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



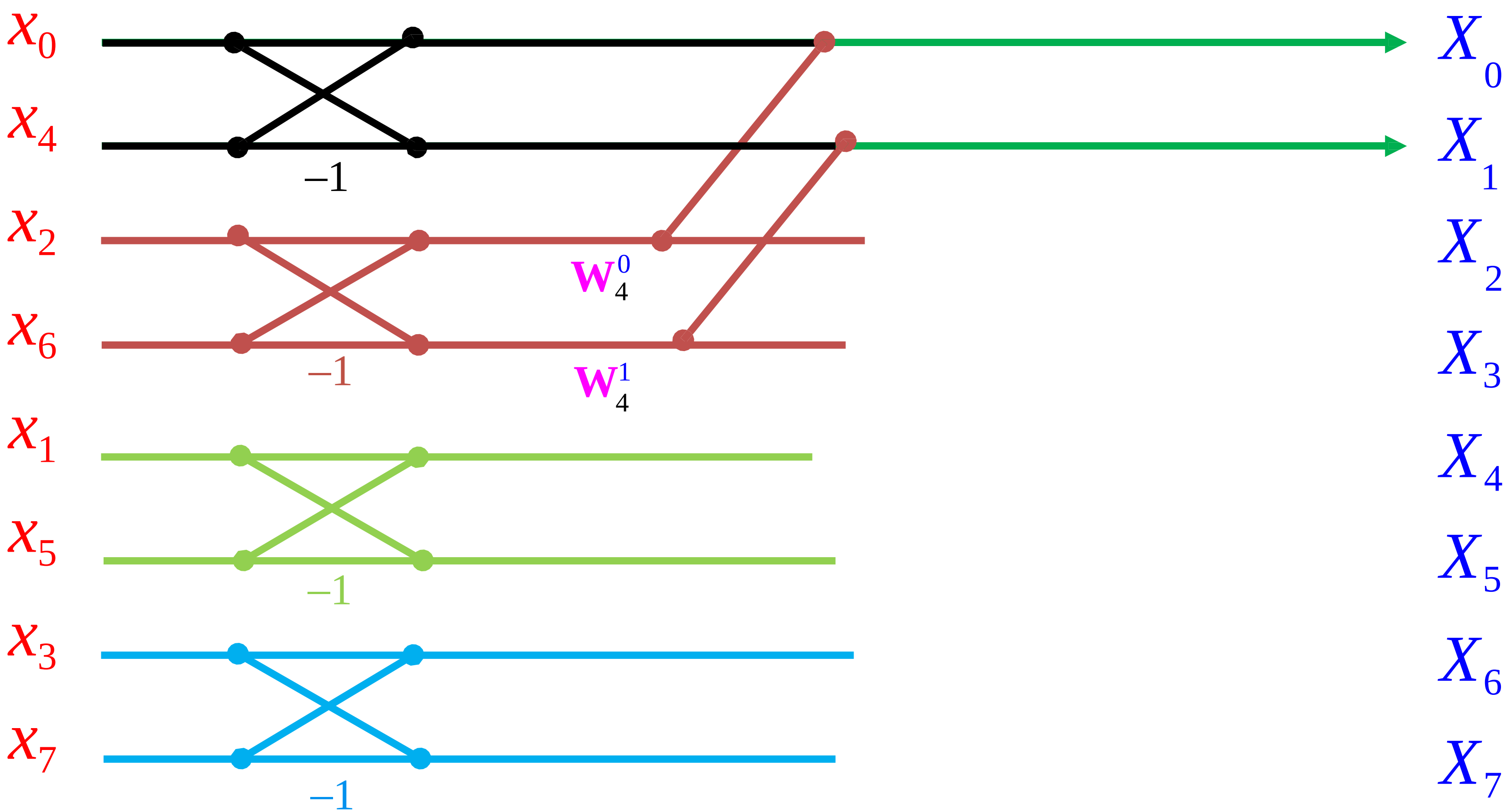
# 8-point DITFFT

- Let $x\ n\ = \ x_0,\ x_1,\ x_2\ ,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



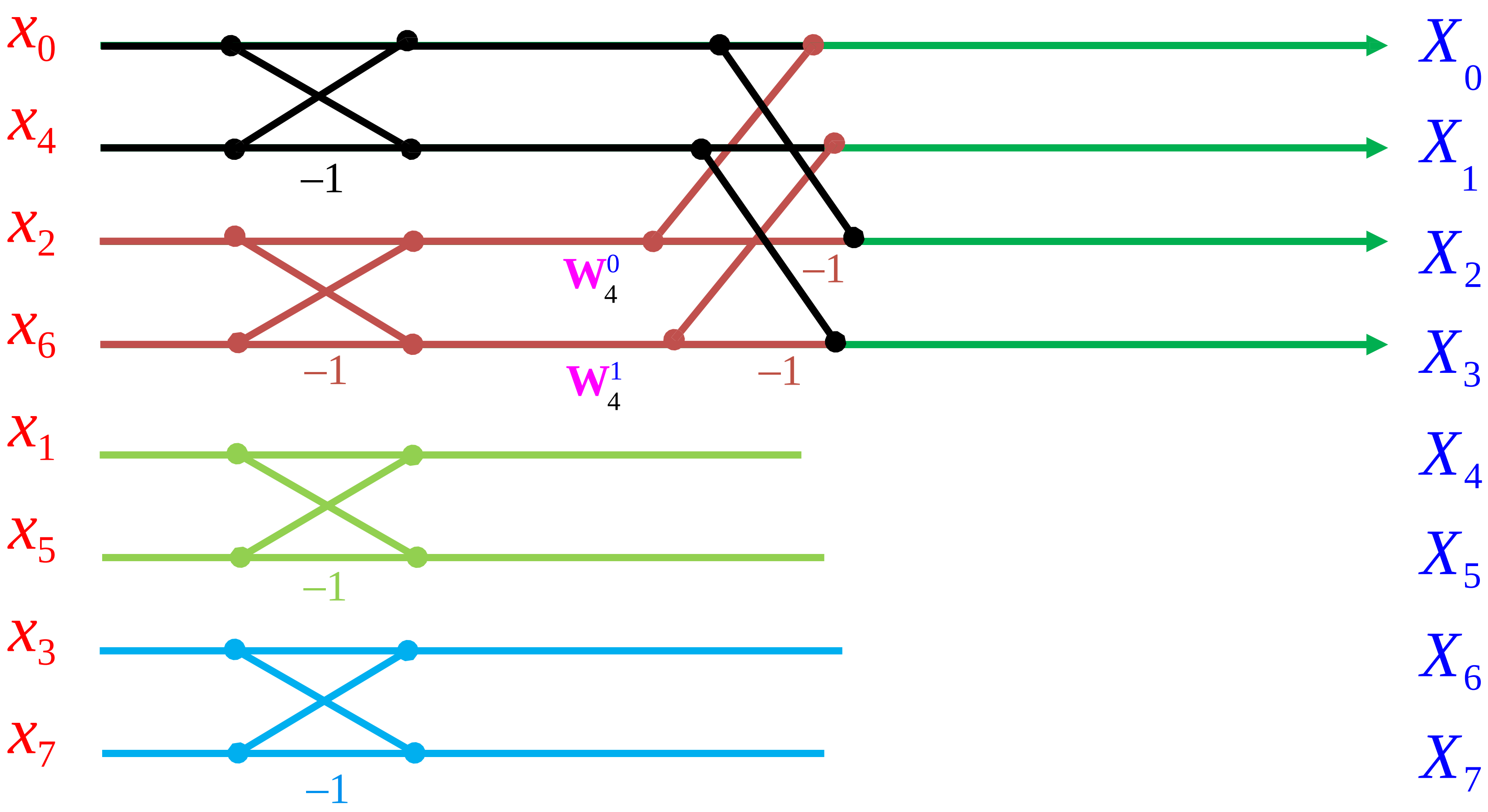
## 8-point DITFFT

- Let $x(n) = x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



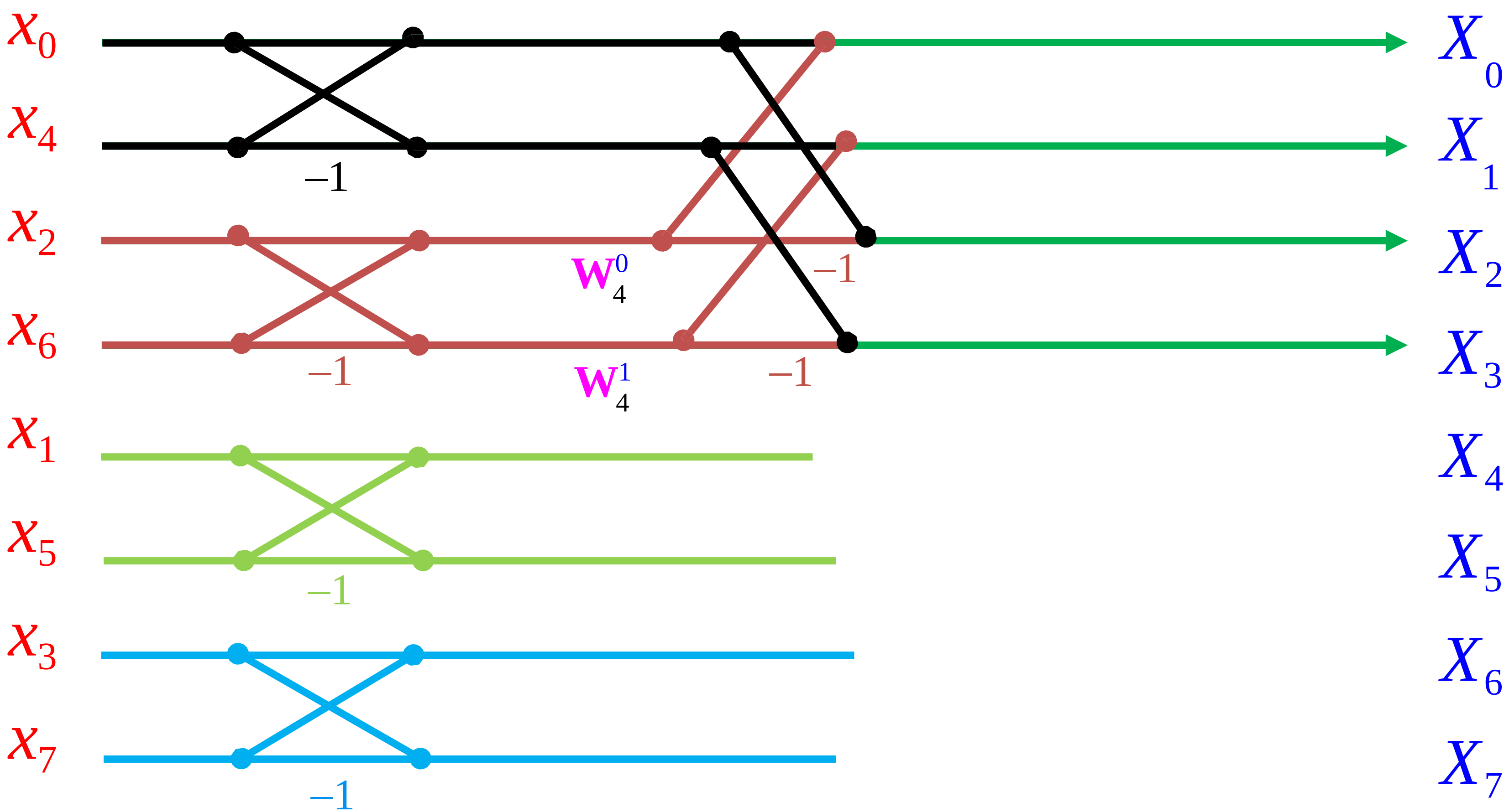
## 8-point DITFFT

- Let $x\ n\ = \ x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



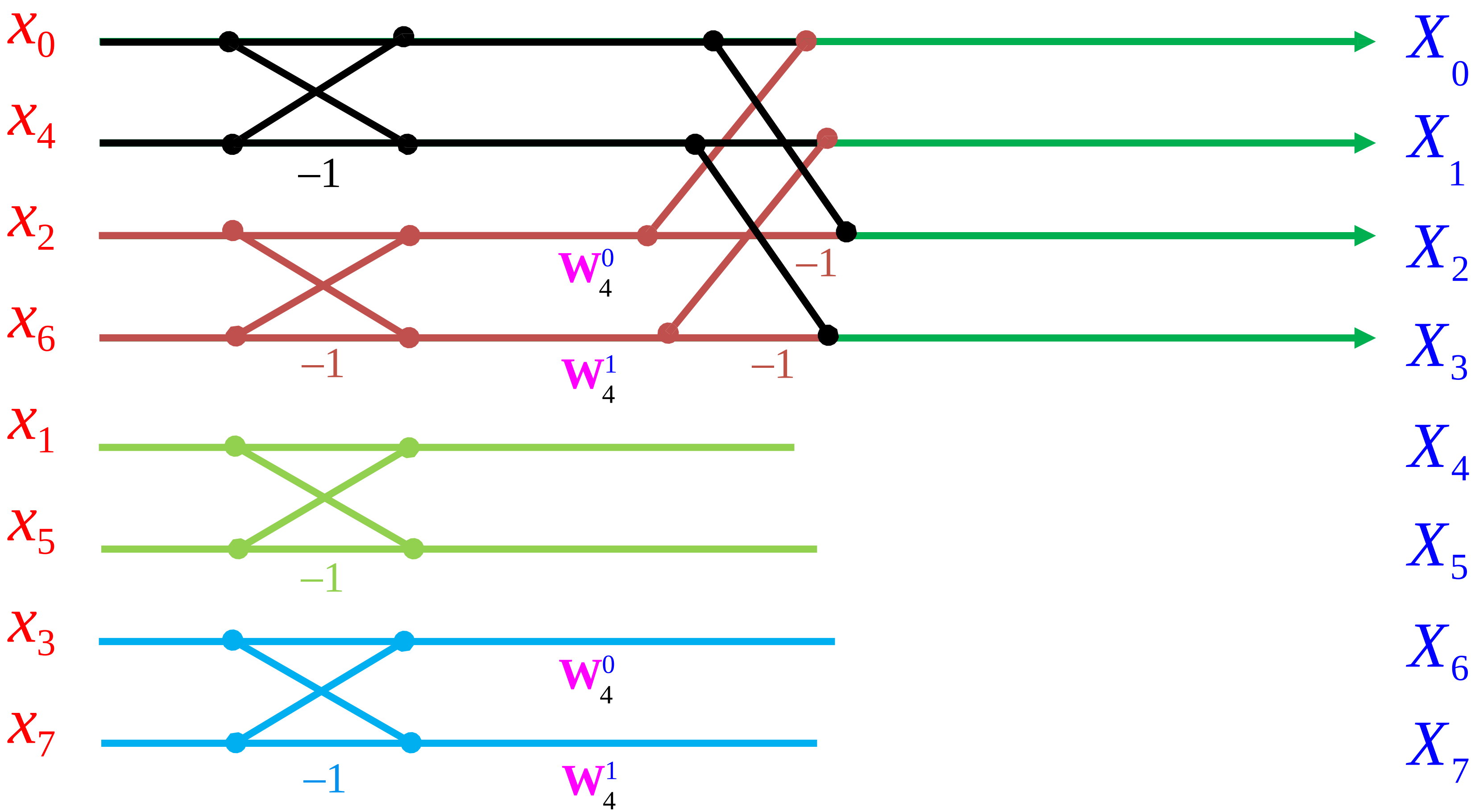
## 8-point DITFFT

- Let $x(n) = x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



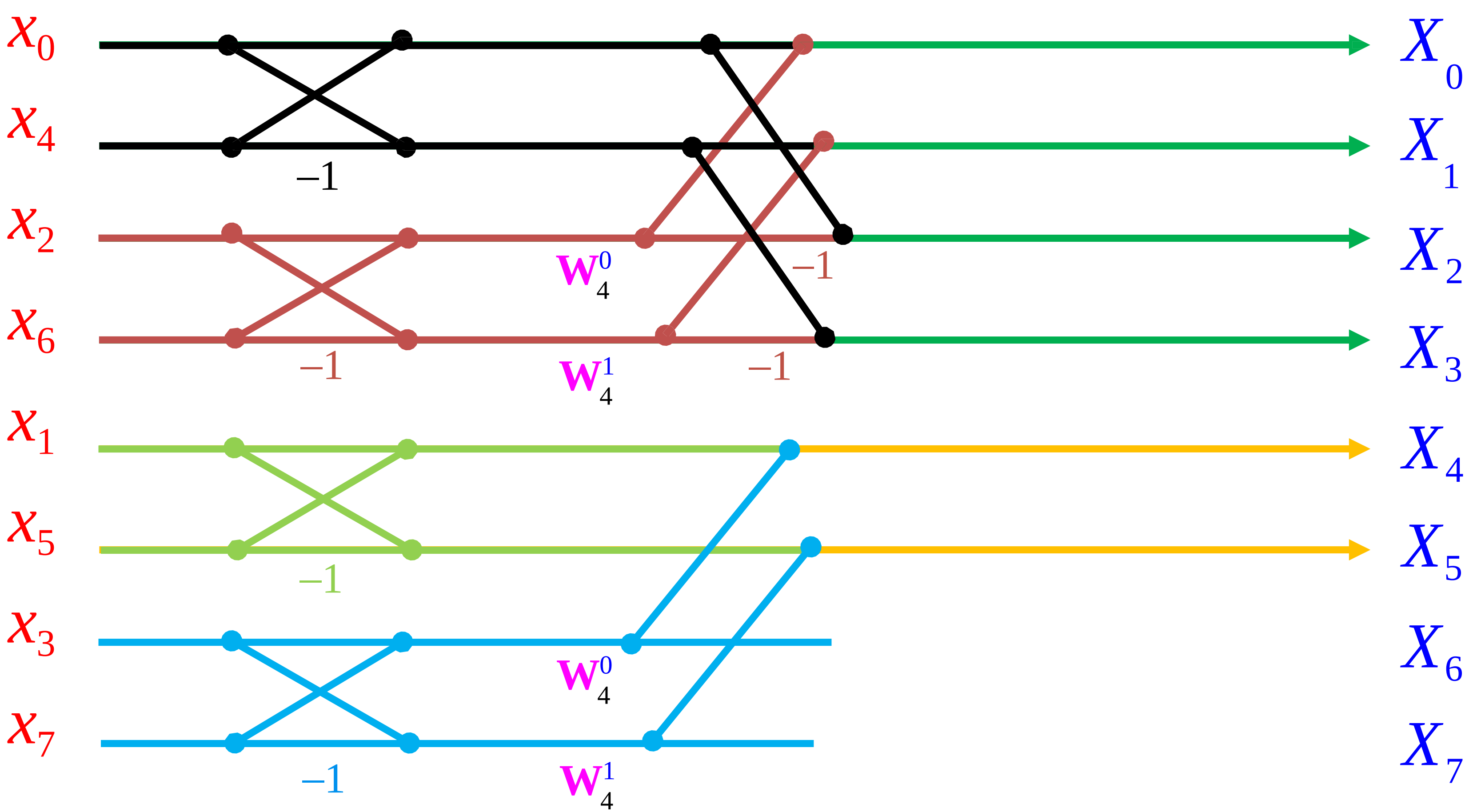
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



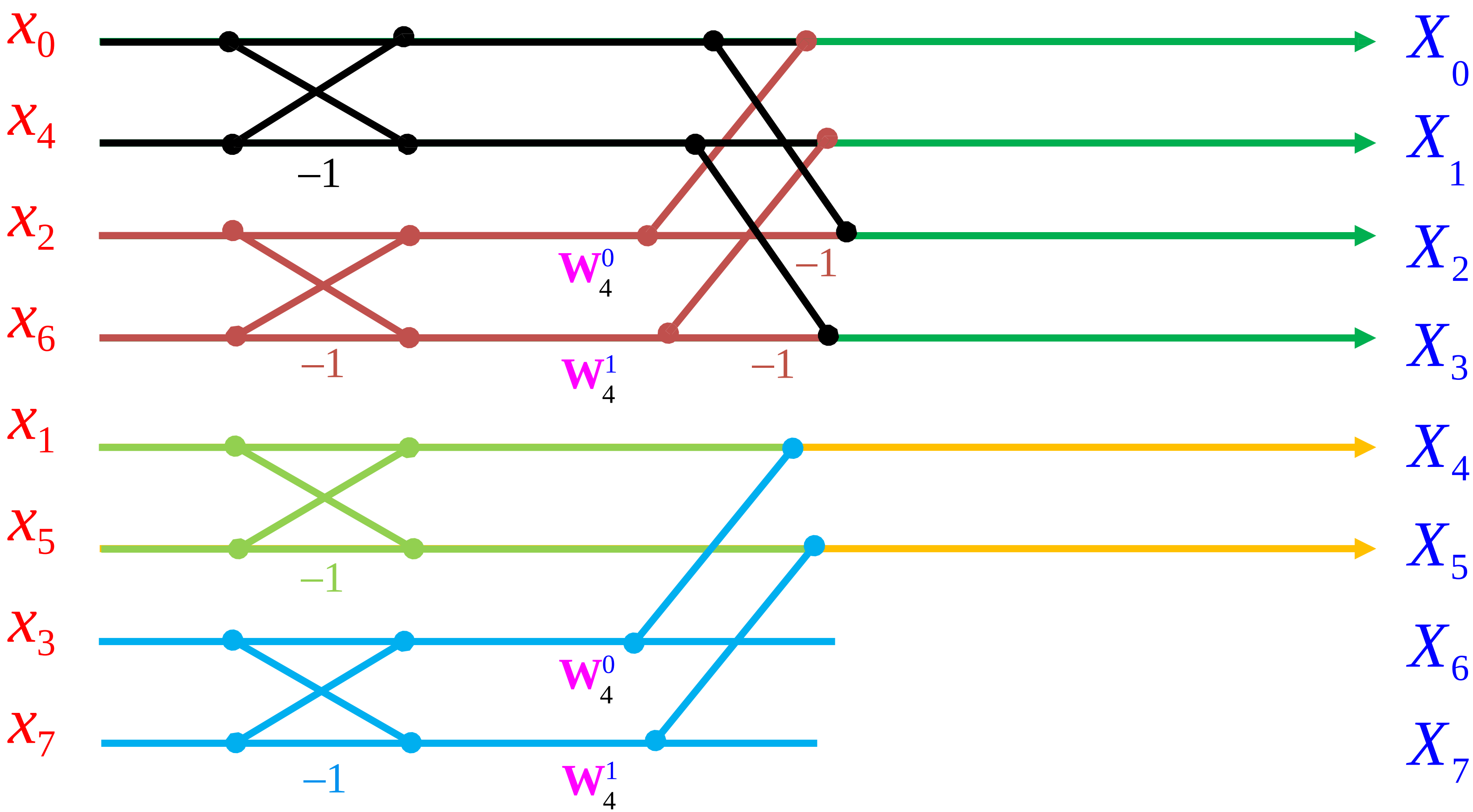
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



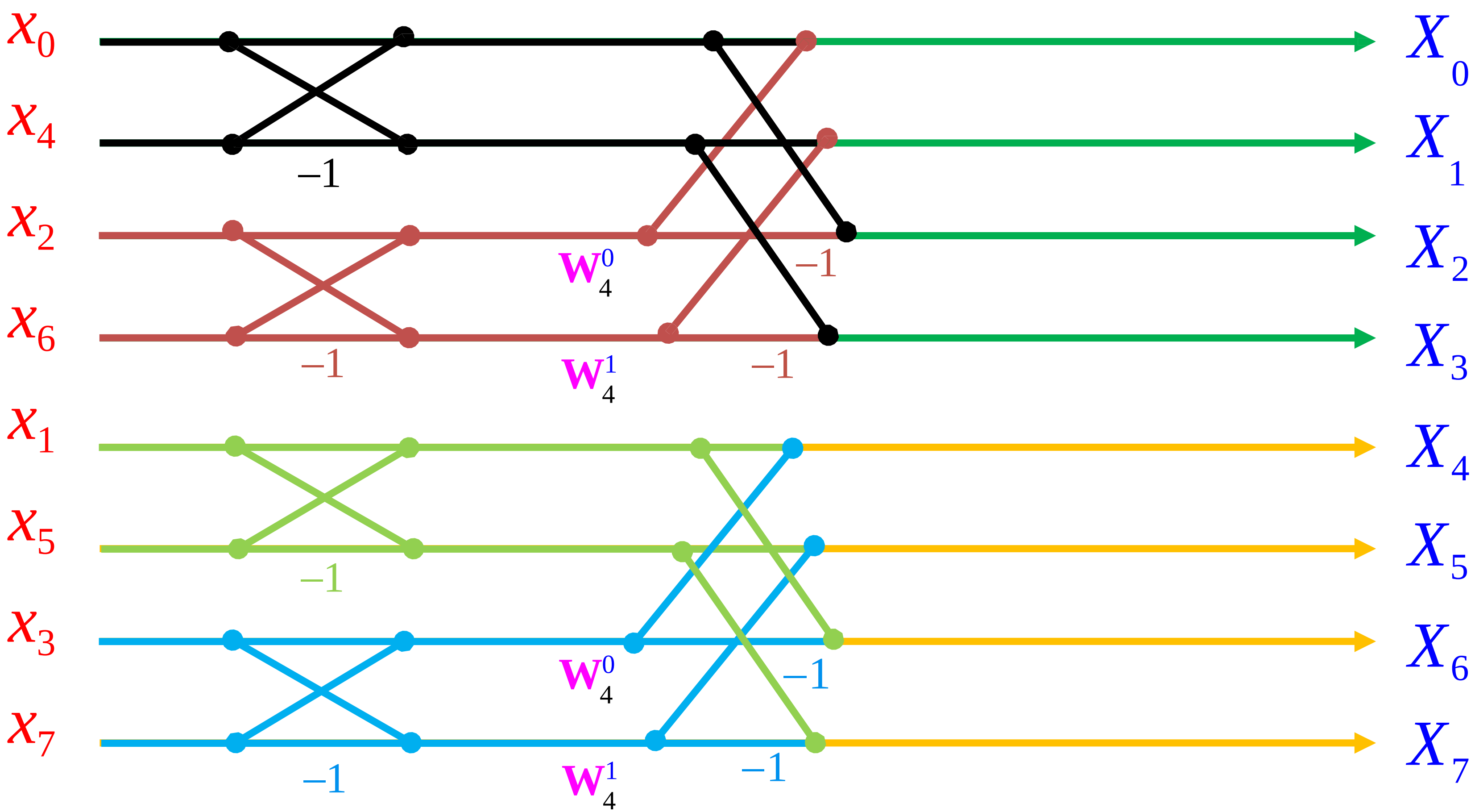
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



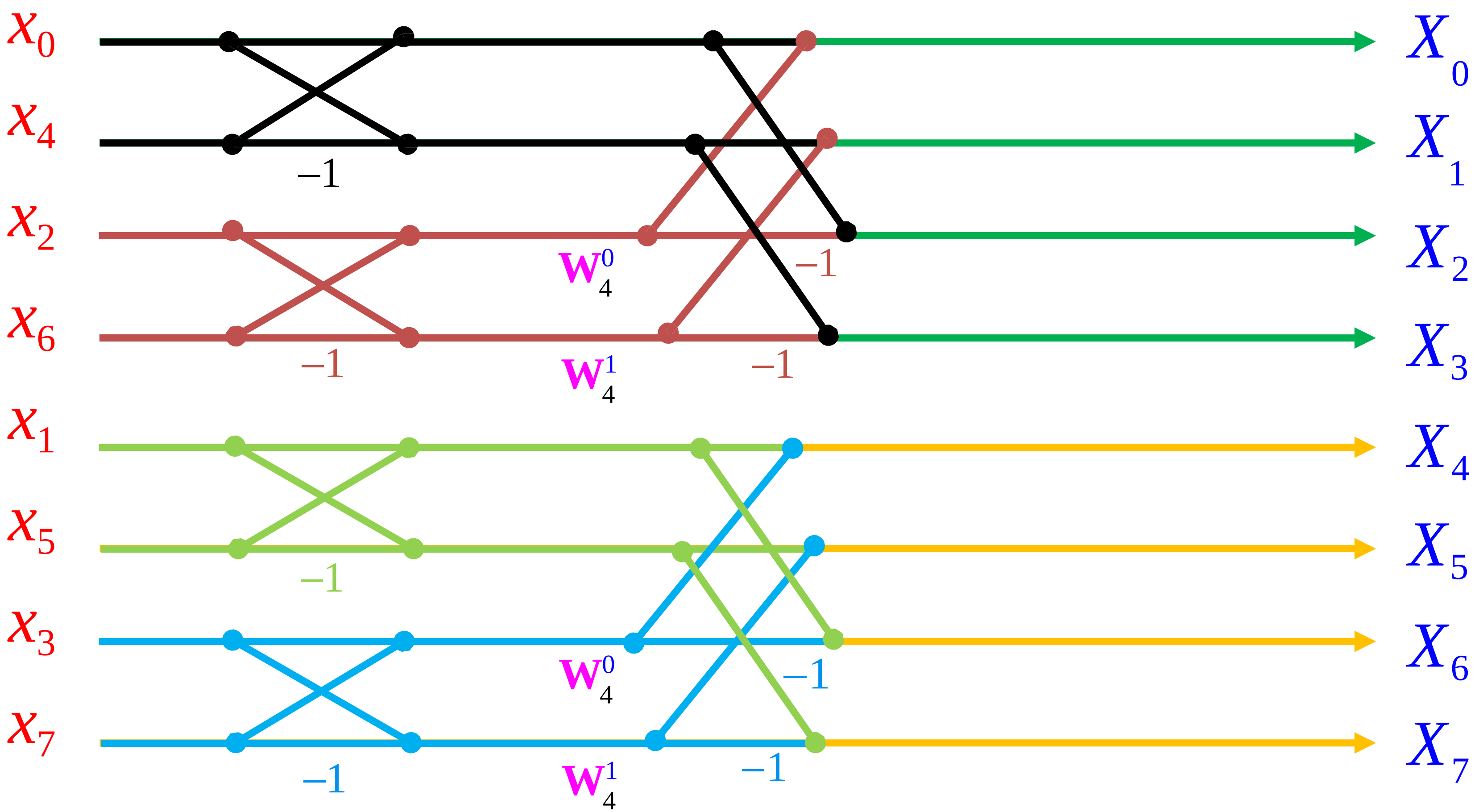
# 8-point DITFFT

- Let $x[n] = \{x_0,\ x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



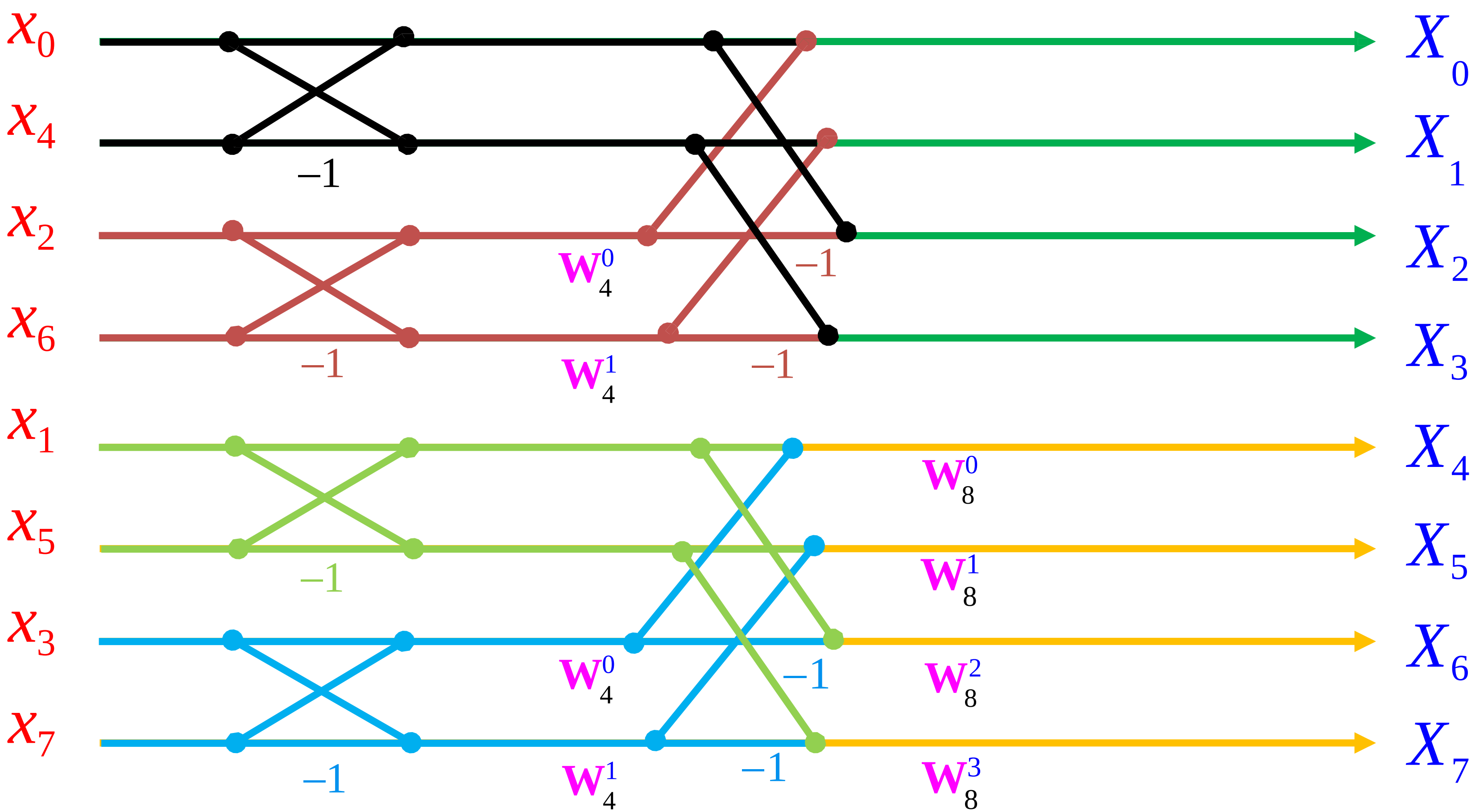
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



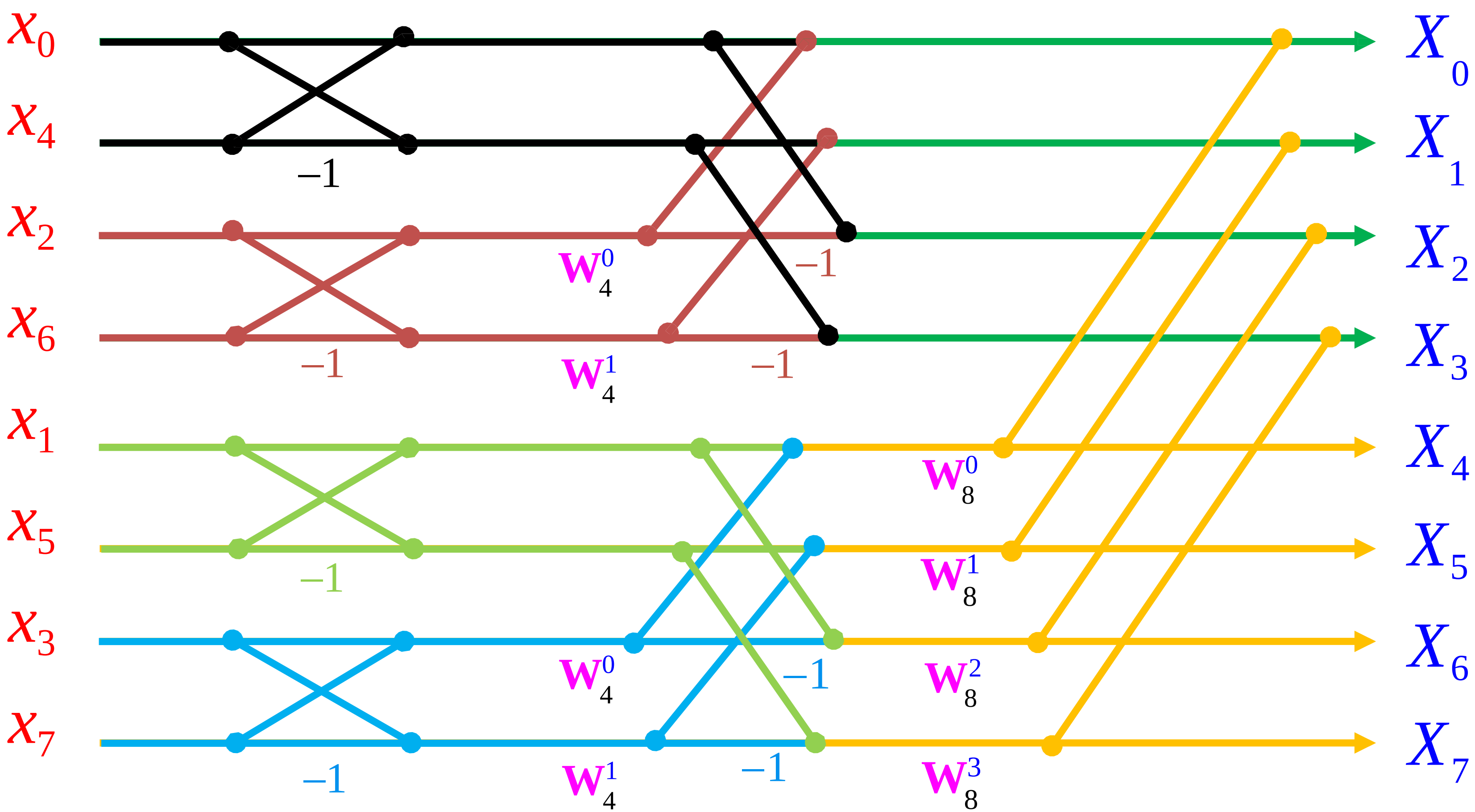
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



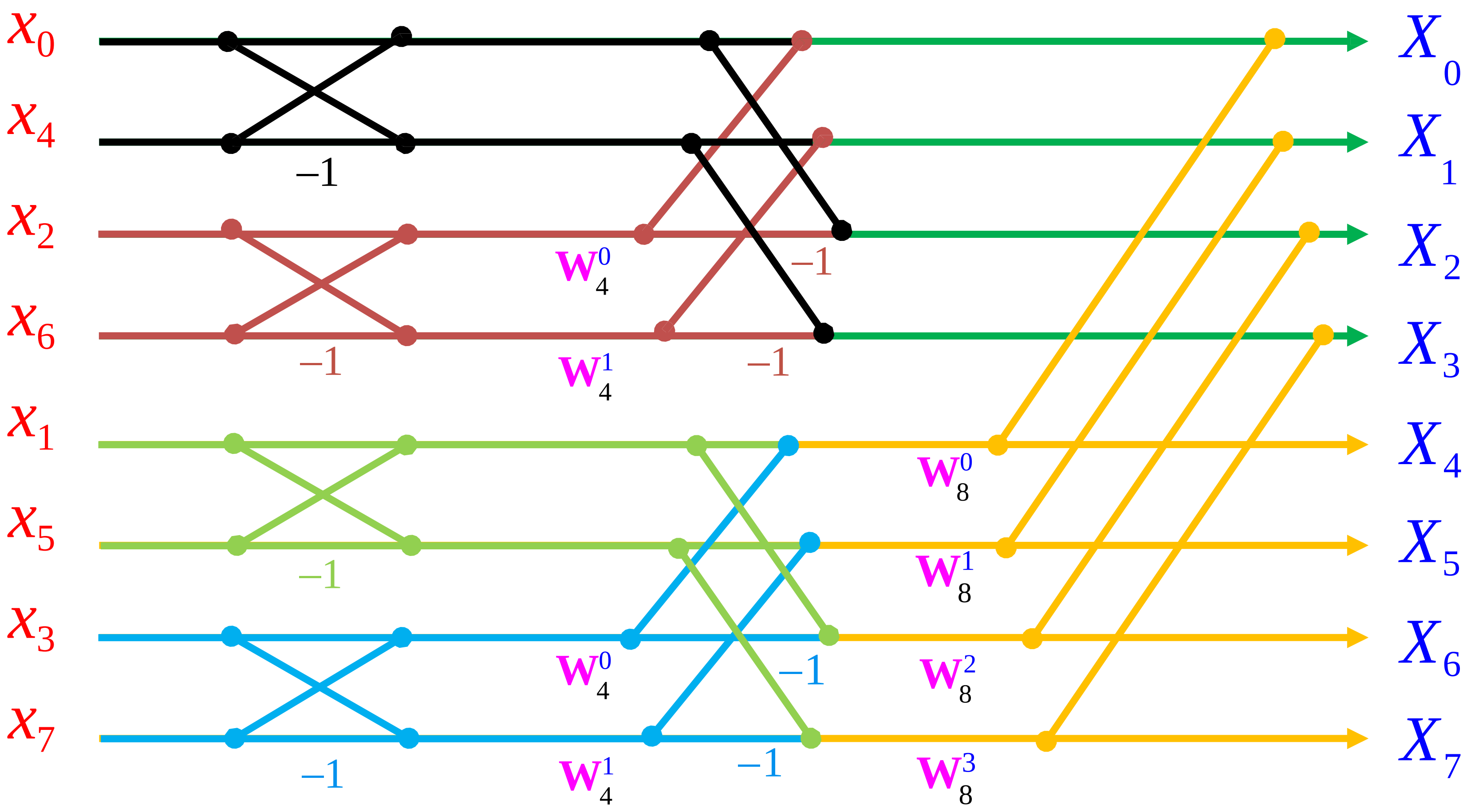
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



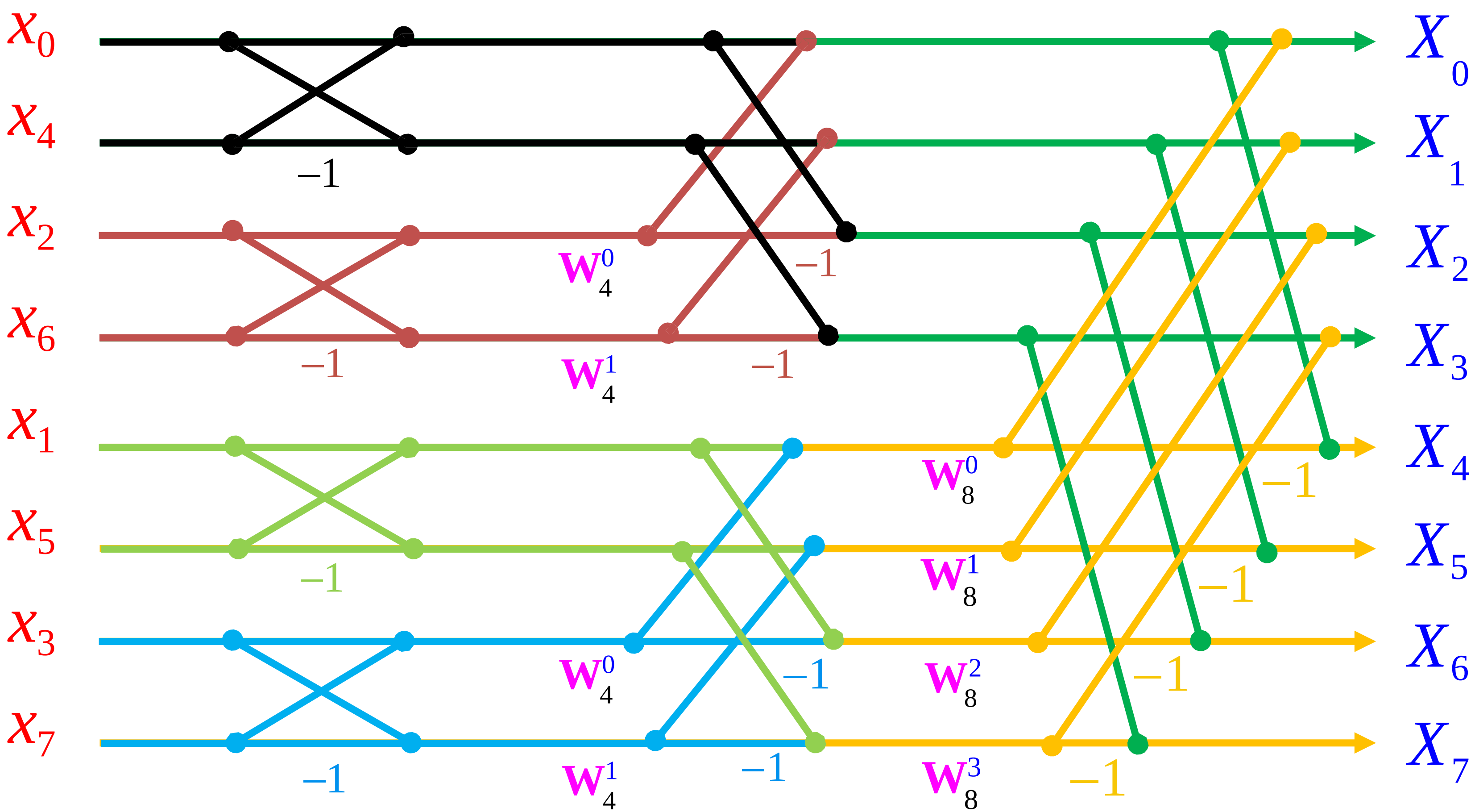
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



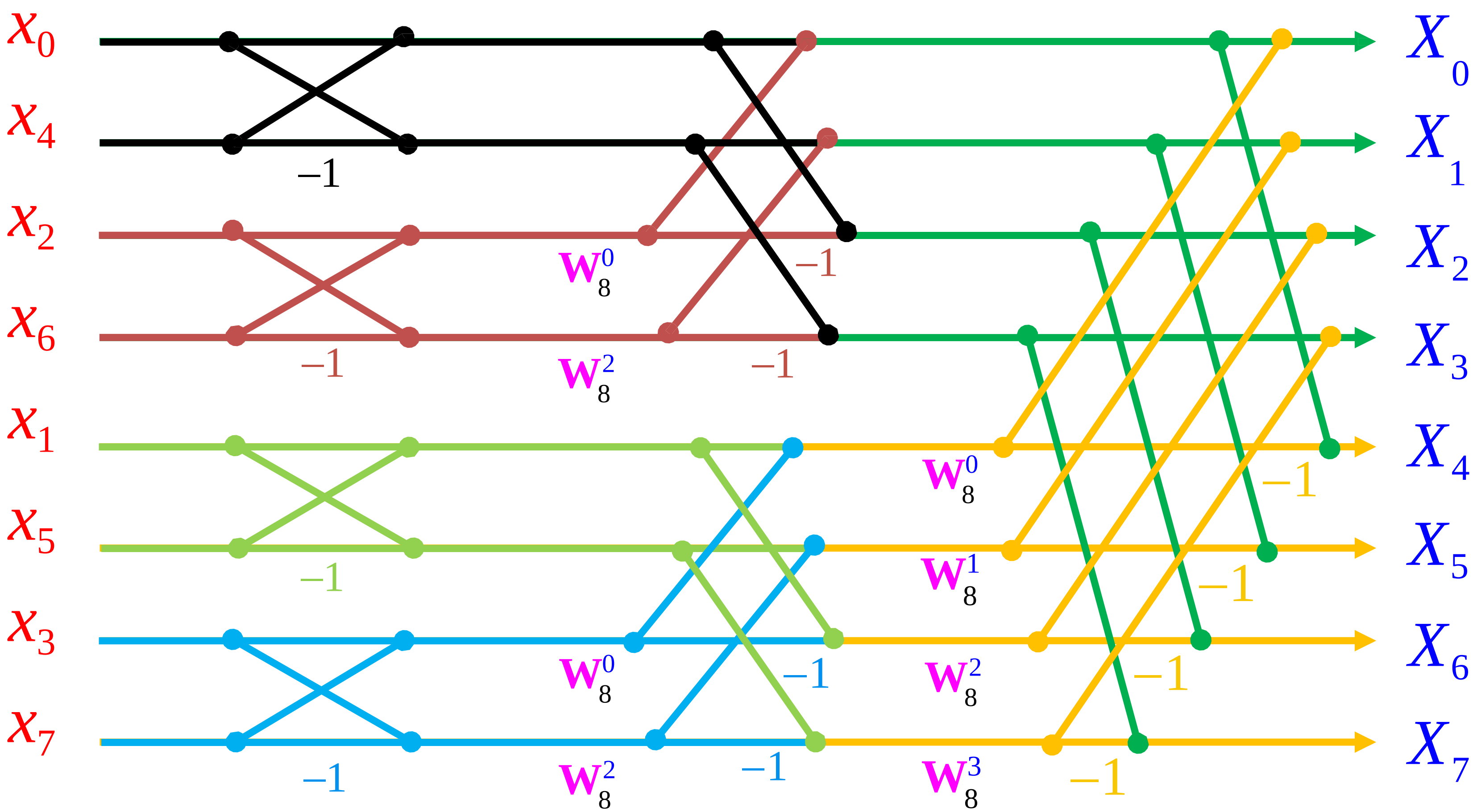
# 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



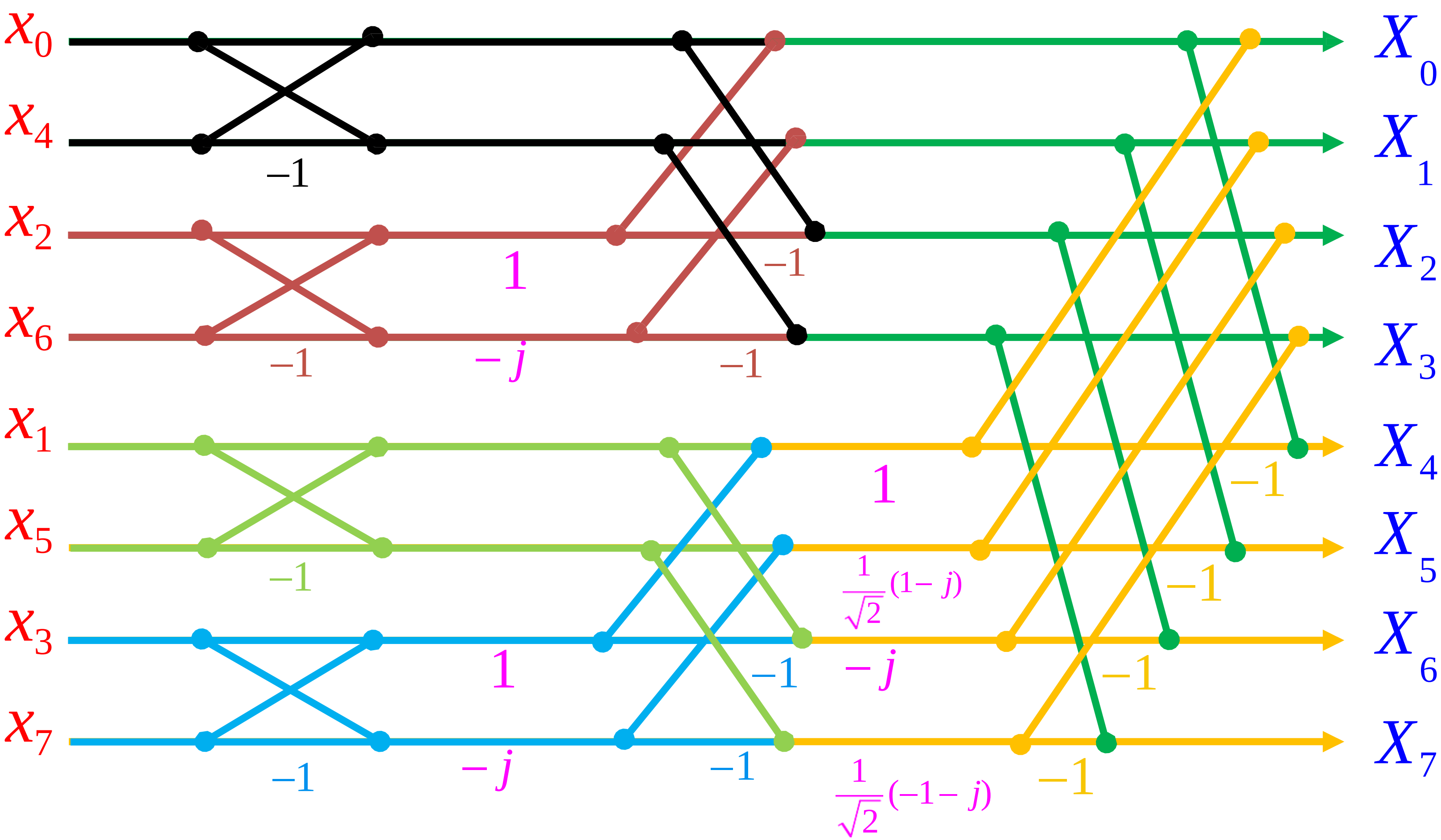
## 8-point DITFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



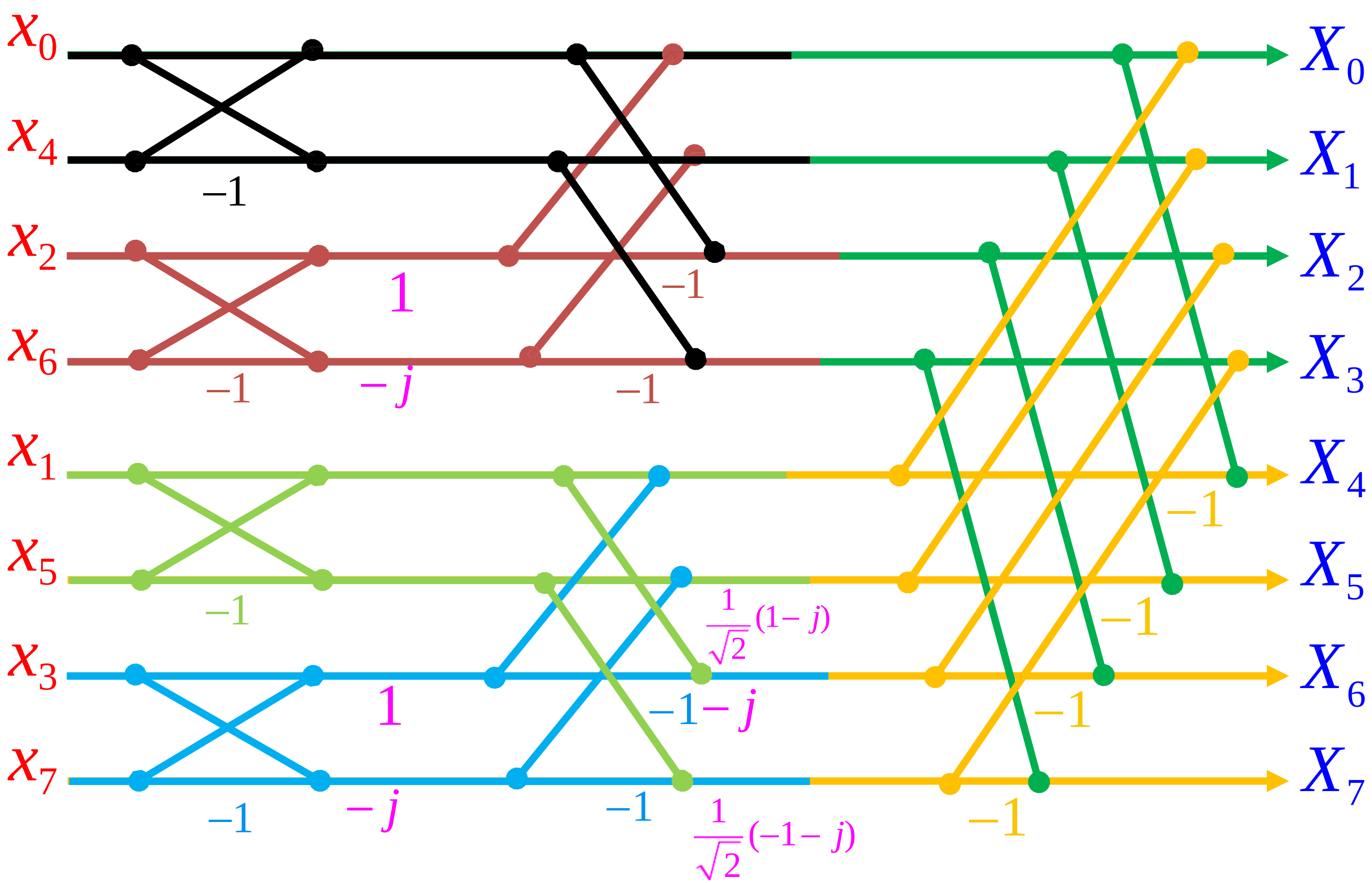
# 8-point DITFFT

- Let $x[n] = \{x_0,\ x_1, x_2\ , x_3, x_4, x_5, x_6, x_7\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7\}$



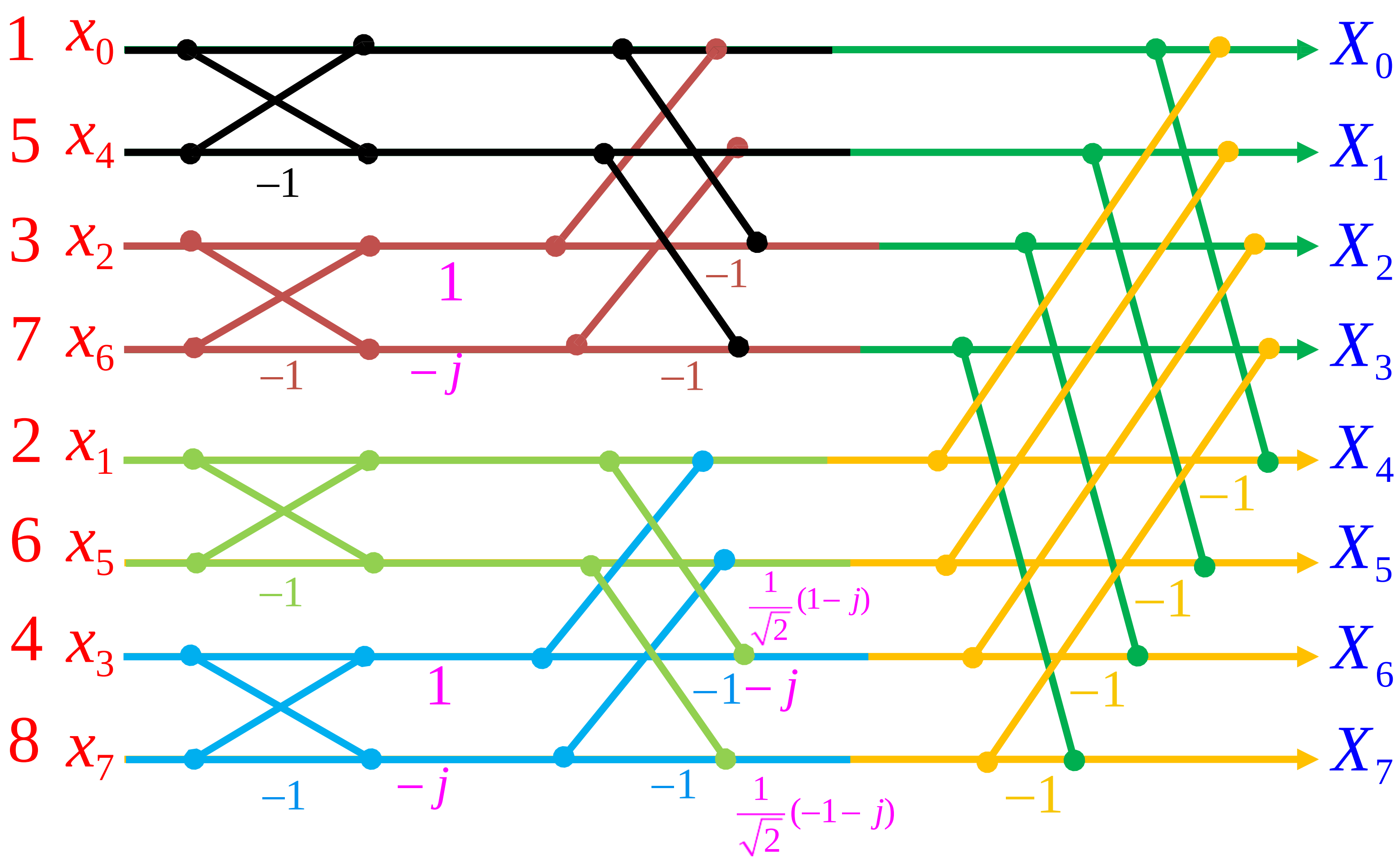
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm. (May19,10 marks)



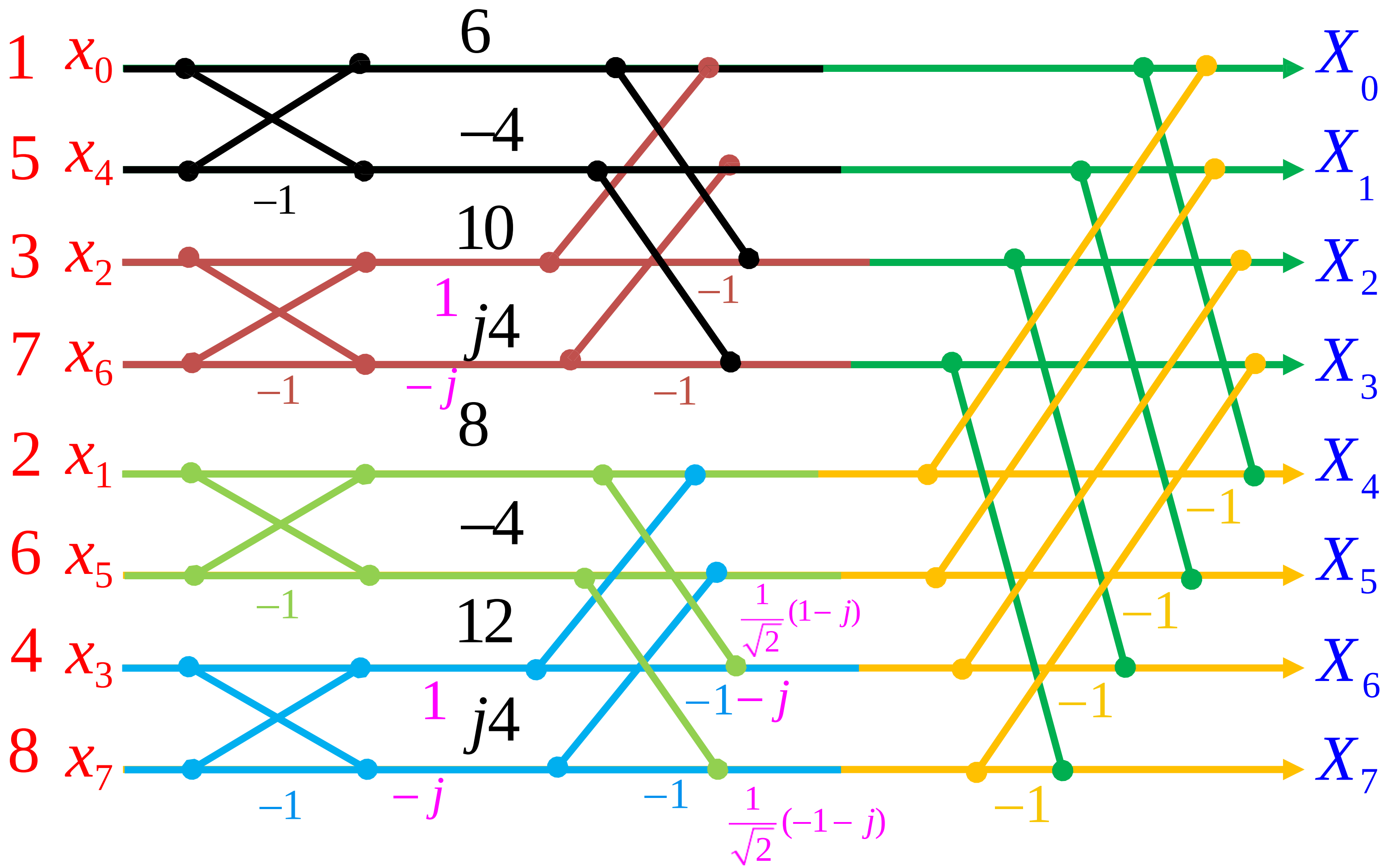
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



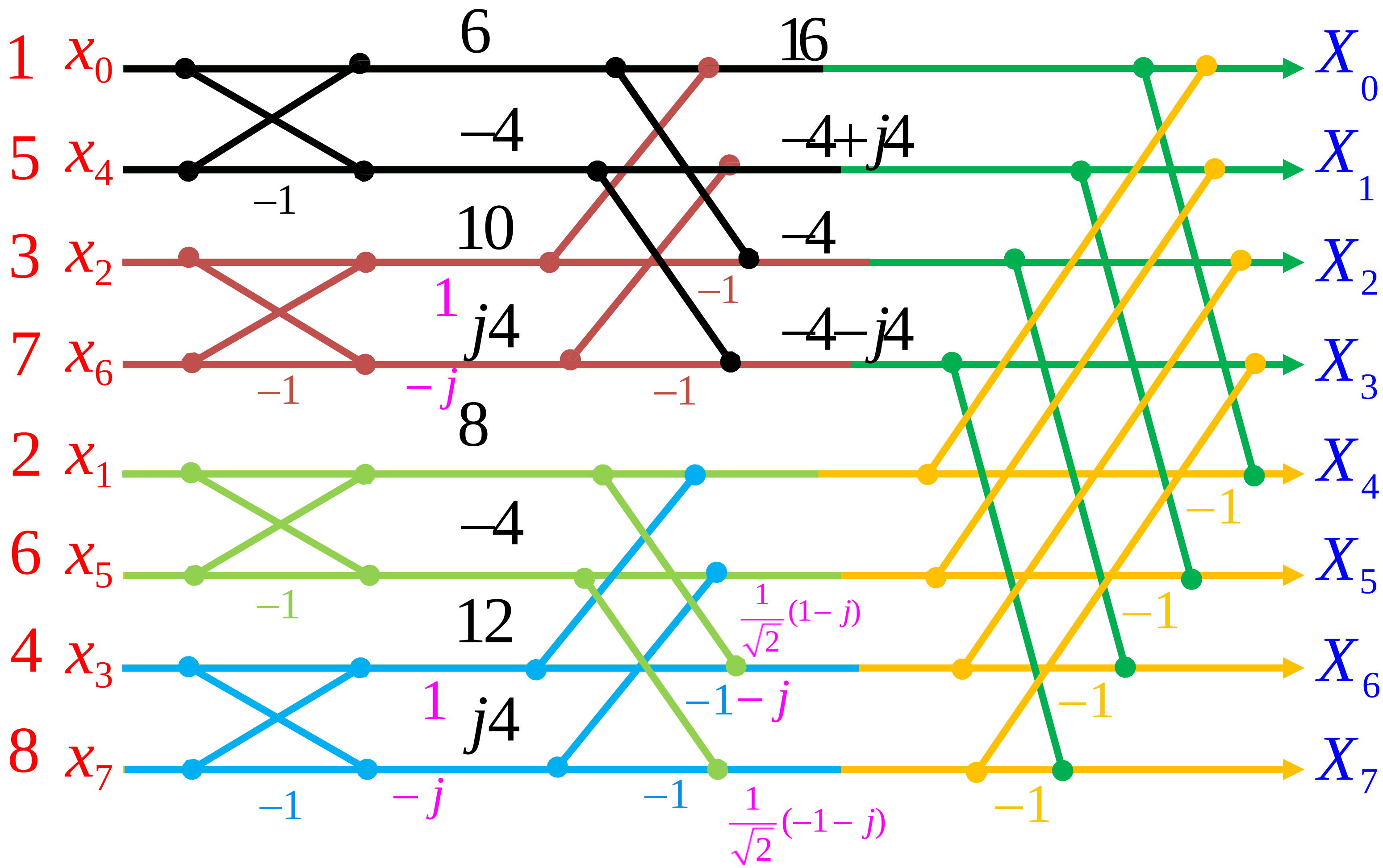
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



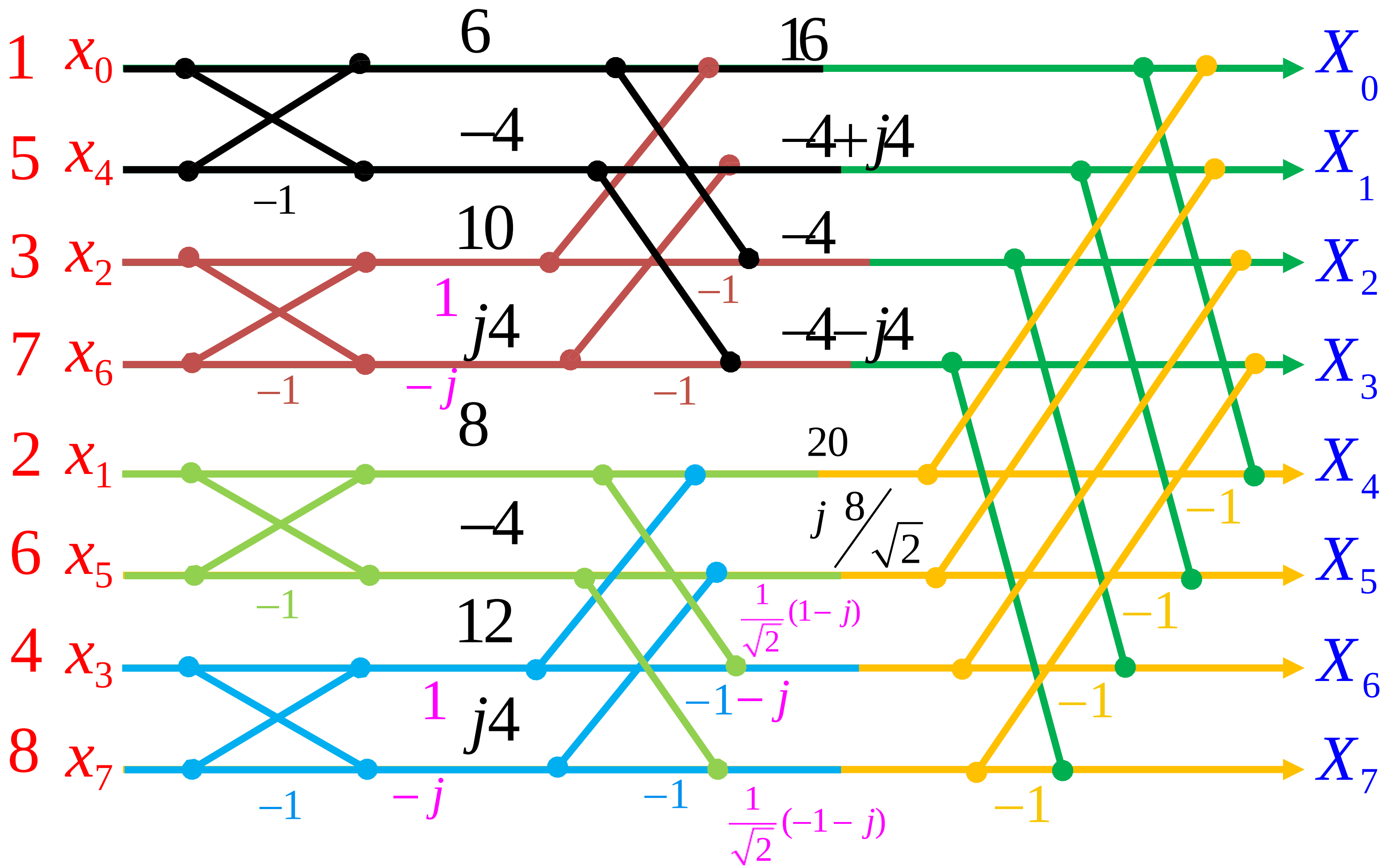
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



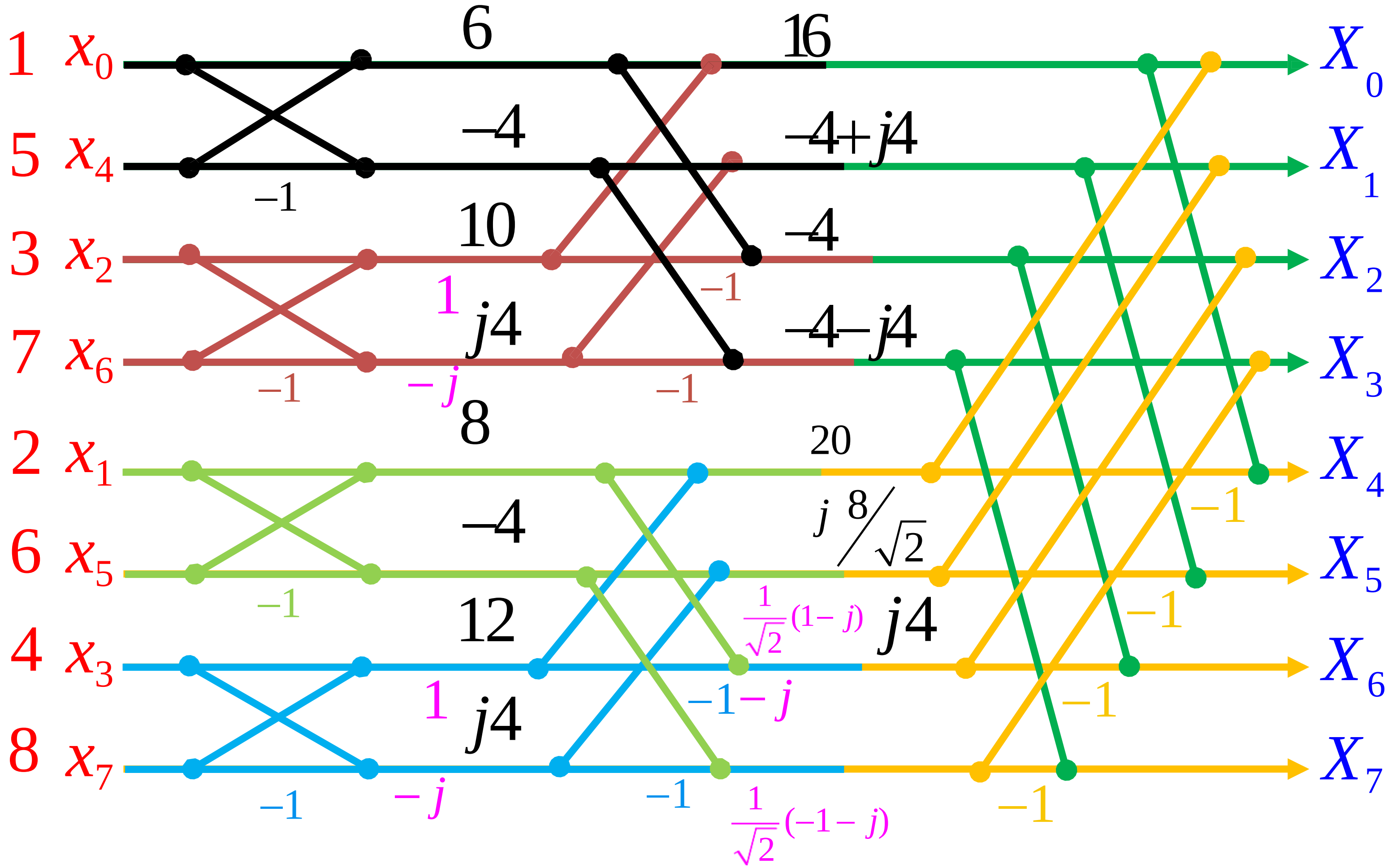
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



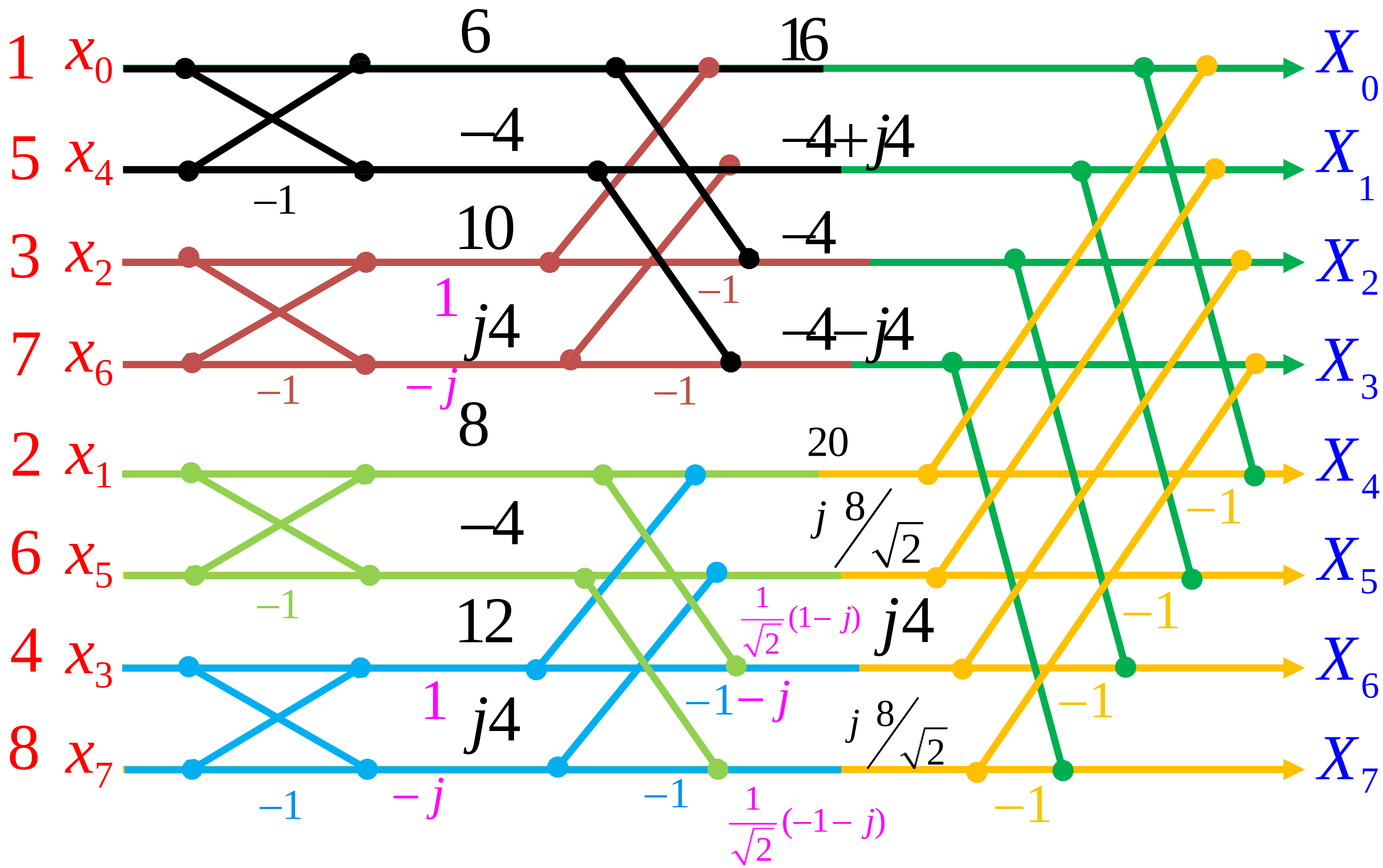
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



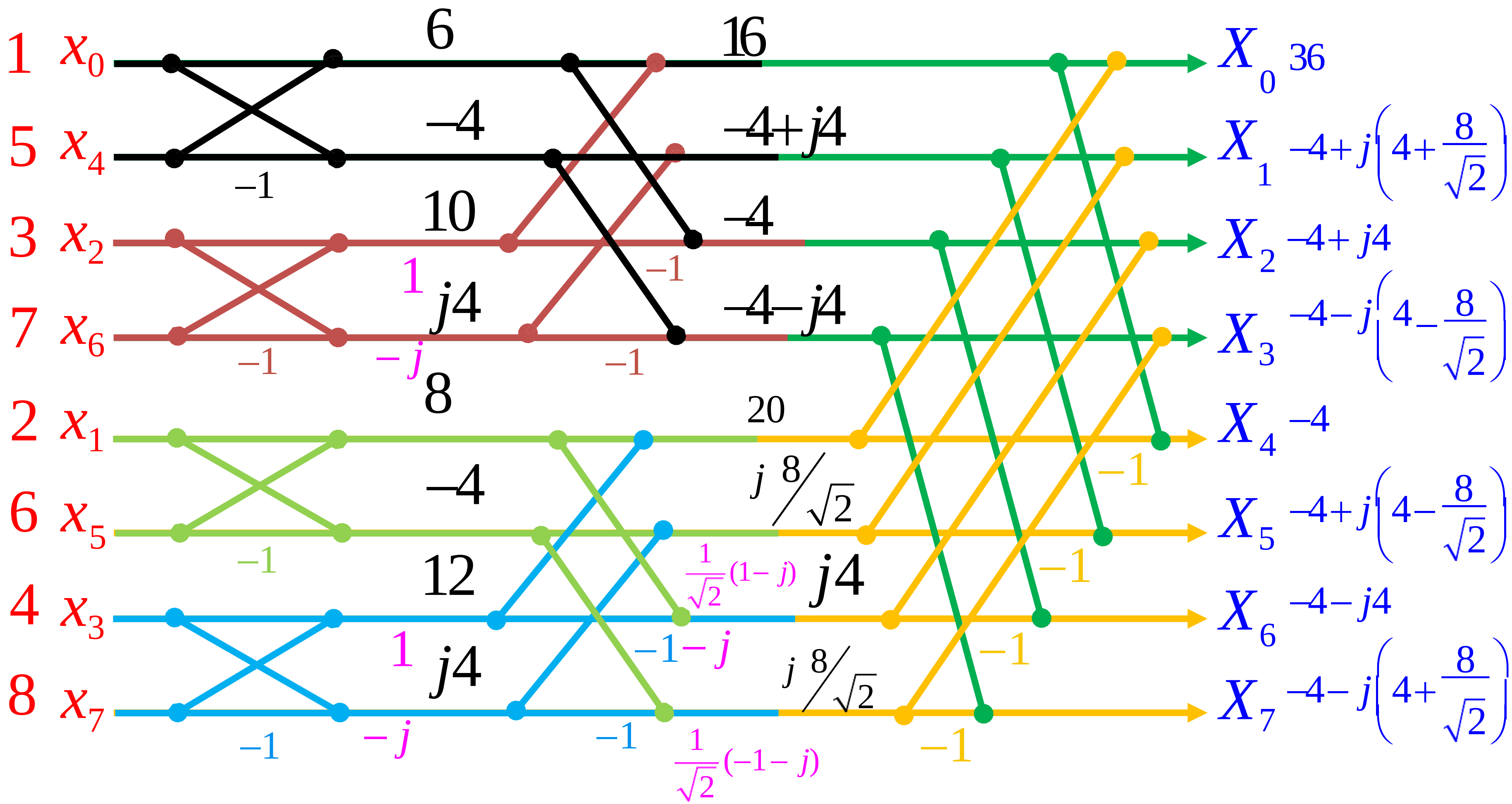
## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.

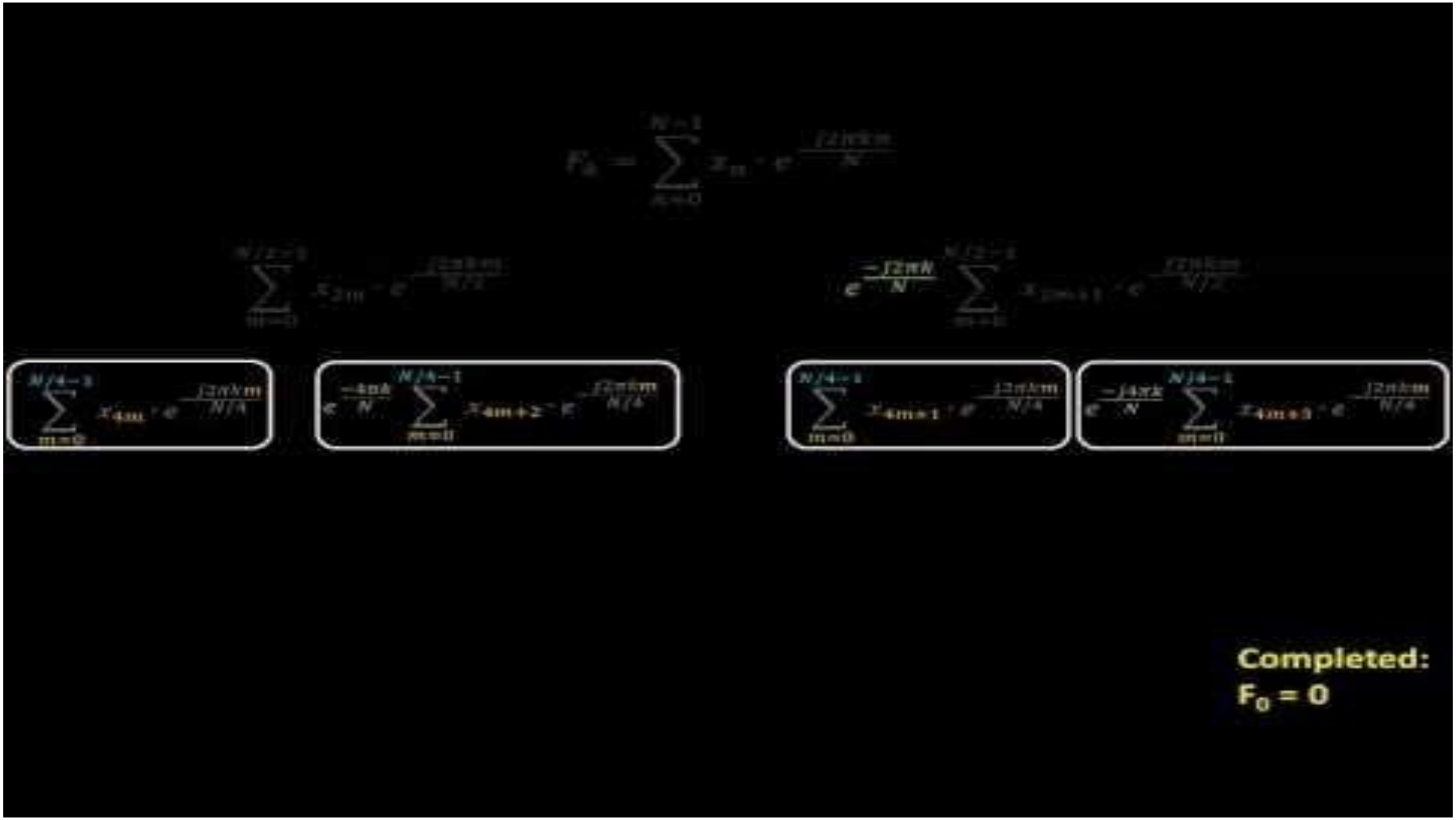


## Example

Compute DFT of $x[n] = \{1,2,3,4,5,6,7,8\}$ using DITFFT algorithm.



# DIT-FFT




[FFT Algorithm: Step-by-Step]

# Decimation-In-Frequency FFT Algorithm

# Decimation-In-Frequency FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

# Decimation-In-Frequency FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

# Decimation-In-Frequency FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

- Separating $X[k]$ into two $(N/2)$-point sequences consisting of the even-indexed points in $X[k]$ and the odd-indexed points in $X[k]$.
- Even indexed $X[2r]$ and odd indexed $X[2r+1]$ each $\boldsymbol{N/2}$ long

# Decimation-In-Frequency FFT Algorithm

- Sequence $x[n]$ is decomposed into successively smaller subsequences.
- Considering the special case of ***N* an integer power of 2**, i.e., $N = 2^v$.

$$X[k] = \sum_{n=0}^{N-1} x[n] \, \mathbf{W}_N^{kn} \qquad k, n = 0, 1, \ldots, N-1$$

- Separating $X[k]$ into two $(N/2)$-point sequences consisting of the even-indexed points in $X[k]$ and the odd-indexed points in $X[k]$.
- Even indexed $X[2r]$ and odd indexed $X[2r+1]$ each $\boldsymbol{N/2}$ long

$$X[2r] = \sum_{n=0}^{N-1} x[n] \, \mathbf{W}_N^{2rn} \qquad r = 0, 1, \ldots, (N/2) - 1$$

# Decimation-In-Frequency FFT Algorithm

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{2rn} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{2rn}$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{2rn} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{2rn}$$

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \left(\mathbf{W}_N^{2}\right)^{rn} + \sum_{n=0}^{N/2-1} x[n+(N/2)]\, \left(\mathbf{W}_N^{2}\right)^{r[n+(N/2)]}$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{2rn} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{2rn}$$

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \left(\mathbf{W}_N^{2}\right)^{rn} + \sum_{n=0}^{N/2-1} x[n+(N/2)]\, \left(\mathbf{W}_N^{2}\right)^{r[n+(N/2)]}$$

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{2rn} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{2rn}$$

$$X[2r] = \sum_{n=0}^{N/2-1} x[n]\, \left(\mathbf{W}_N^2\right)^{rn} + \sum_{n=0}^{N/2-1} x[n+(N/2)]\, \left(\mathbf{W}_N^2\right)^{r[n+(N/2)]}$$

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

- $(N/2)$-point DFT of the $(N/2)$ -point sequence obtained by adding the first half and the last half of the input sequence.
  - Adding the two halves of the input sequence represents **time aliasing**

# Decimation-In-Frequency FFT Algorithm

# Decimation-In-Frequency FFT Algorithm

$$X[2r+1]=\sum_{n=0}^{N-1}x[n]\,\mathbf{W}_N^{(2r+1)n}\qquad r=0,1,\ldots,(N/2)-1$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r+1] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n} \qquad r = 0,1,\ldots,(N/2)-1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{(2r+1)n} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n}$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r+1] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n} \qquad r = 0,1,\ldots,(N/2)-1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{(2r+1)n} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n}$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{(2r+1)n} + \sum_{n=0}^{N/2-1} x[n+(N/2)]\, \mathbf{W}_N^{(2r+1)[n+(N/2)]}$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r+1]=\sum_{n=0}^{N-1} x[n]\, W_N^{(2r+1)n} \qquad r=0,1,\ldots,(N/2)-1$$

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

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

$$X[2r+1]=\sum_{n=0}^{N/2-1} \left(x[n]-x[n+(N/2)]\right) W_N^{(2r+1)n} \qquad r=0,1,\ldots,\left(\frac{N}{2}\right)-1$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r+1] = \sum_{n=0}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n} \qquad r = 0,1,\ldots,(N/2)-1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{(2r+1)n} + \sum_{n=N/2}^{N-1} x[n]\, \mathbf{W}_N^{(2r+1)n}$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} x[n]\, \mathbf{W}_N^{(2r+1)n} + \sum_{n=0}^{N/2-1} x[n+(N/2)]\, \mathbf{W}_N^{(2r+1)[n+(N/2)]}$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left(x[n] - x[n+(N/2)]\right) \mathbf{W}_N^{(2r+1)n} \qquad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left(x[n] - x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn}\, \mathbf{W}_N^{n}$$

# Decimation-In-Frequency FFT Algorithm

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left( x[n] + x[n+(N/2)] \right) W_{N/2}^{rn} \quad r = 0,1,\ldots, \frac{N}{2} - 1$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots,\ \frac{N}{2}\ -1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) \mathbf{W}_{N/2}^{rn} \mathbf{W}_{N}^{n} \quad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots,\ \frac{N}{2} - 1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) \mathbf{W}_{N/2}^{rn} \mathbf{W}_{N}^{n} \quad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

- Let two $(N/2)$-point sequences be generated as,

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) W_{N/2}^{rn} \quad r = 0,1,\ldots,\frac{N}{2}-1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) W_{N/2}^{rn} W_{N}^{n} \quad r = 0,1,\ldots,\left(\frac{N}{2}\right)-1$$

- Let two $(N/2)$-point sequences be generated as,

$$g[n] = x[n] + x[n+(N/2)] \qquad h[n] = x[n] - x[n+(N/2)]$$

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots, \frac{N}{2} - 1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) \mathbf{W}_{N/2}^{rn} \mathbf{W}_{N}^{n} \quad r = 0,1,\ldots, \left(\frac{N}{2}\right) - 1$$

- Let two $(N/2)$-point sequences be generated as,

$$g[n] = x[n] + x[n+(N/2)] \qquad h[n] = x[n] - x[n+(N/2)]$$

- Separating $G[k]$ and $H[k]$ into $(N/2)$-point sequences consisting of the even-indexed points and the odd-indexed points

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots, \frac{N}{2} - 1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) \mathbf{W}_{N/2}^{rn} \mathbf{W}_{N}^{n} \quad r = 0,1,\ldots, \left(\frac{N}{2}\right) - 1$$

- Let two $(N/2)$-point sequences be generated as,

$$g[n] = x[n] + x[n+(N/2)] \qquad h[n] = x[n] - x[n+(N/2)]$$

- Separating $G[k]$ and $H[k]$ into $(N/2)$-point sequences consisting of the even-indexed points and the odd-indexed points
- Repeating the procedure to calculate smaller FFTs.
  - Repeat until 2-point FFT.

# Decimation-In-Frequency FFT Algorithm

$$X[2r] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+(N/2)]\right) \mathbf{W}_{N/2}^{rn} \quad r = 0,1,\ldots, \frac{N}{2} - 1$$

$$X[2r+1] = \sum_{n=0}^{N/2-1} \left( x[n] - x\left[n + \frac{N}{2}\right]\right) \mathbf{W}_{N/2}^{rn} \mathbf{W}_{N}^{n} \quad r = 0,1,\ldots, \left(\frac{N}{2}\right) - 1$$

- Let two $(N/2)$-point sequences be generated as,

$$g[n] = x[n] + x[n+(N/2)] \qquad h[n] = x[n] - x[n+(N/2)]$$

- Separating $G[k]$ and $H[k]$ into $(N/2)$-point sequences consisting of the even-indexed points and the odd-indexed points
- Repeating the procedure to calculate smaller FFTs.
    - Repeat until 2-point FFT.

# 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

# 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$

$x_1$

$x_2$

$x_3$

## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$ $\quad X_0$

$x_1$ $\quad X_2$

$x_2$ $\quad X_1$

$x_3$ $\quad X_3$

## 4-point DIFFFT

- Let $x[n] = \{x_0,\ x_1\,, x_2\ , x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$$x_0 \longrightarrow X_0$$

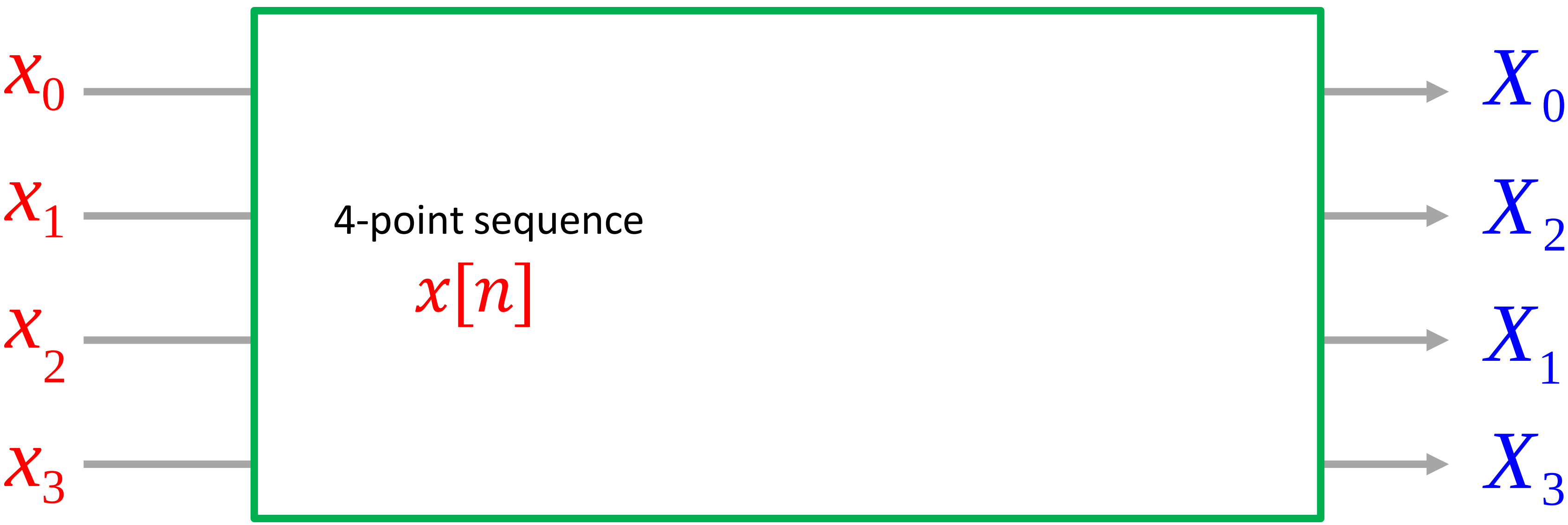
$$x_1 \longrightarrow X_2$$

$$x_2 \longrightarrow X_1$$

$$x_3 \longrightarrow X_3$$

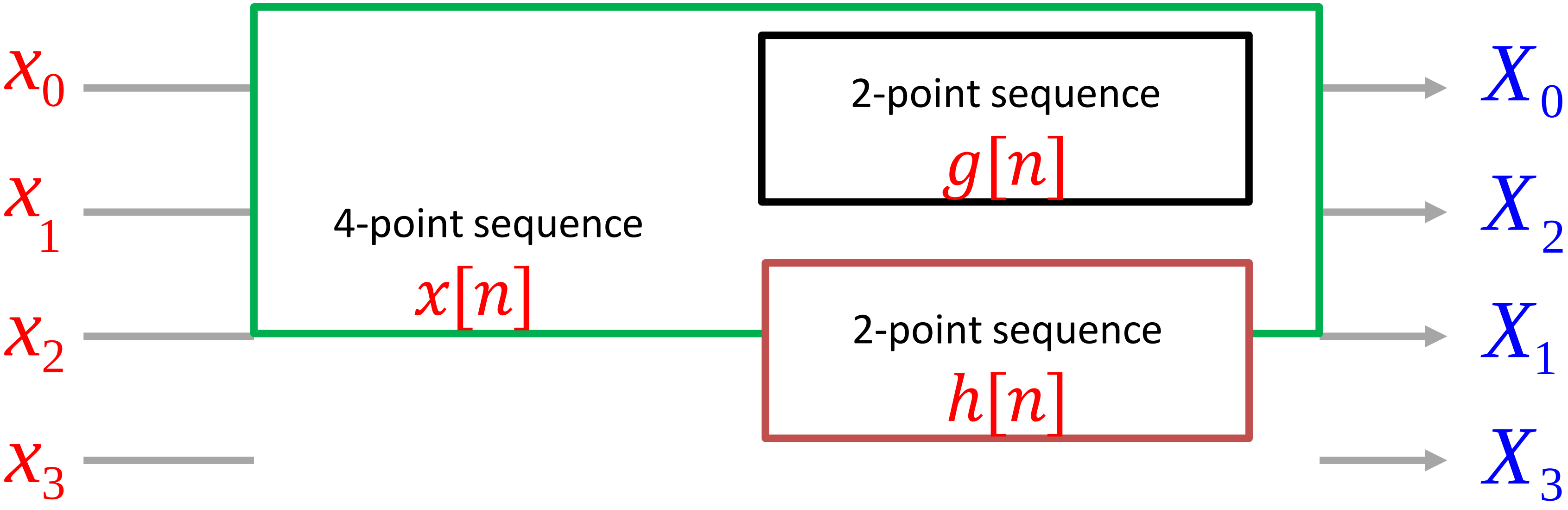
## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$



# 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

# 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$

$x_1$

$x_2$

$x_3$

## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$x_0$ $X_0$

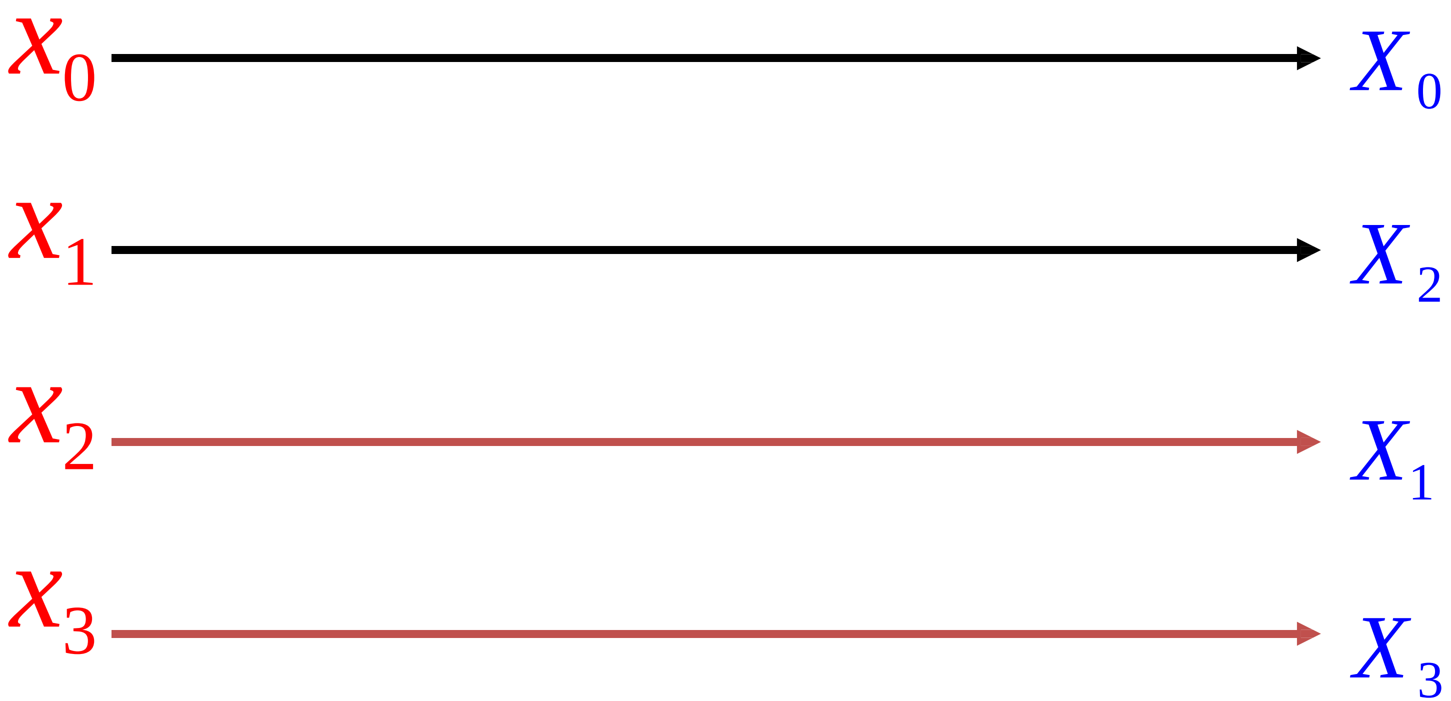
$x_1$ $X_2$

$x_2$ $X_1$

$x_3$ $X_3$

## 4-point DIFFFT

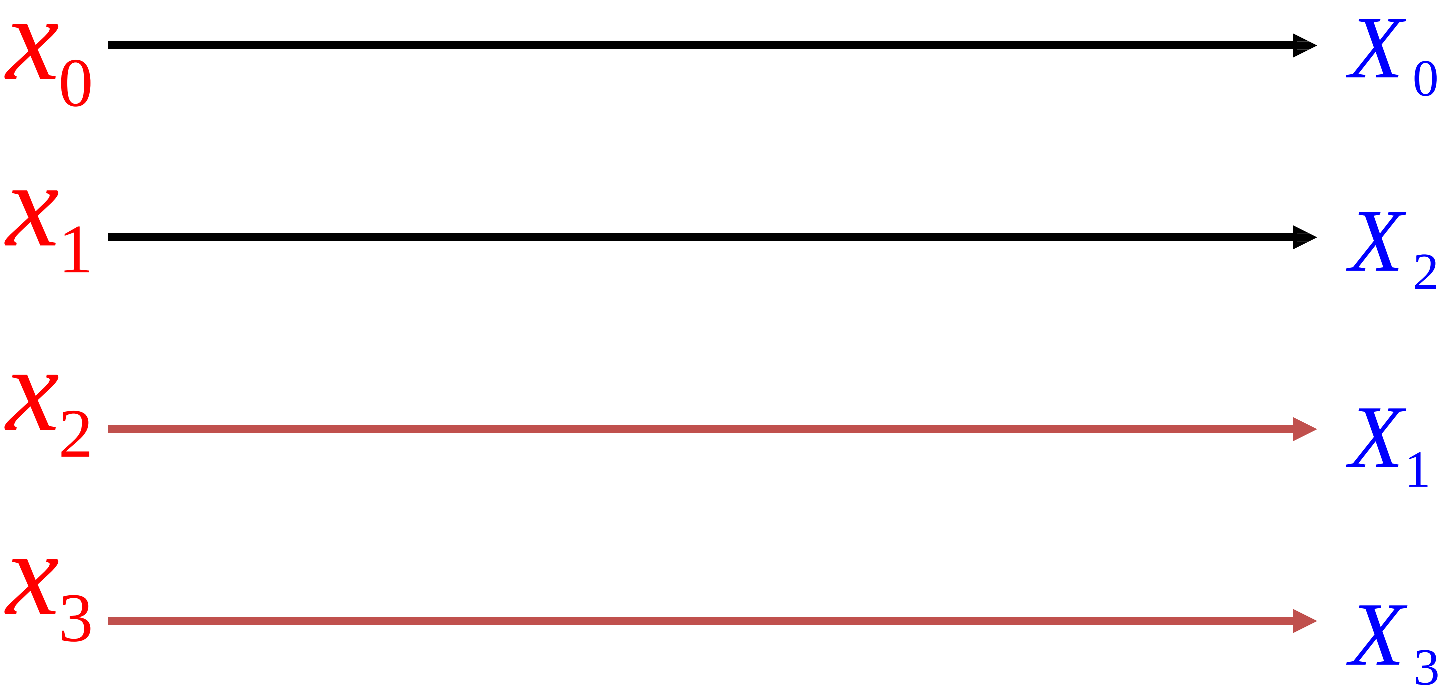
- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

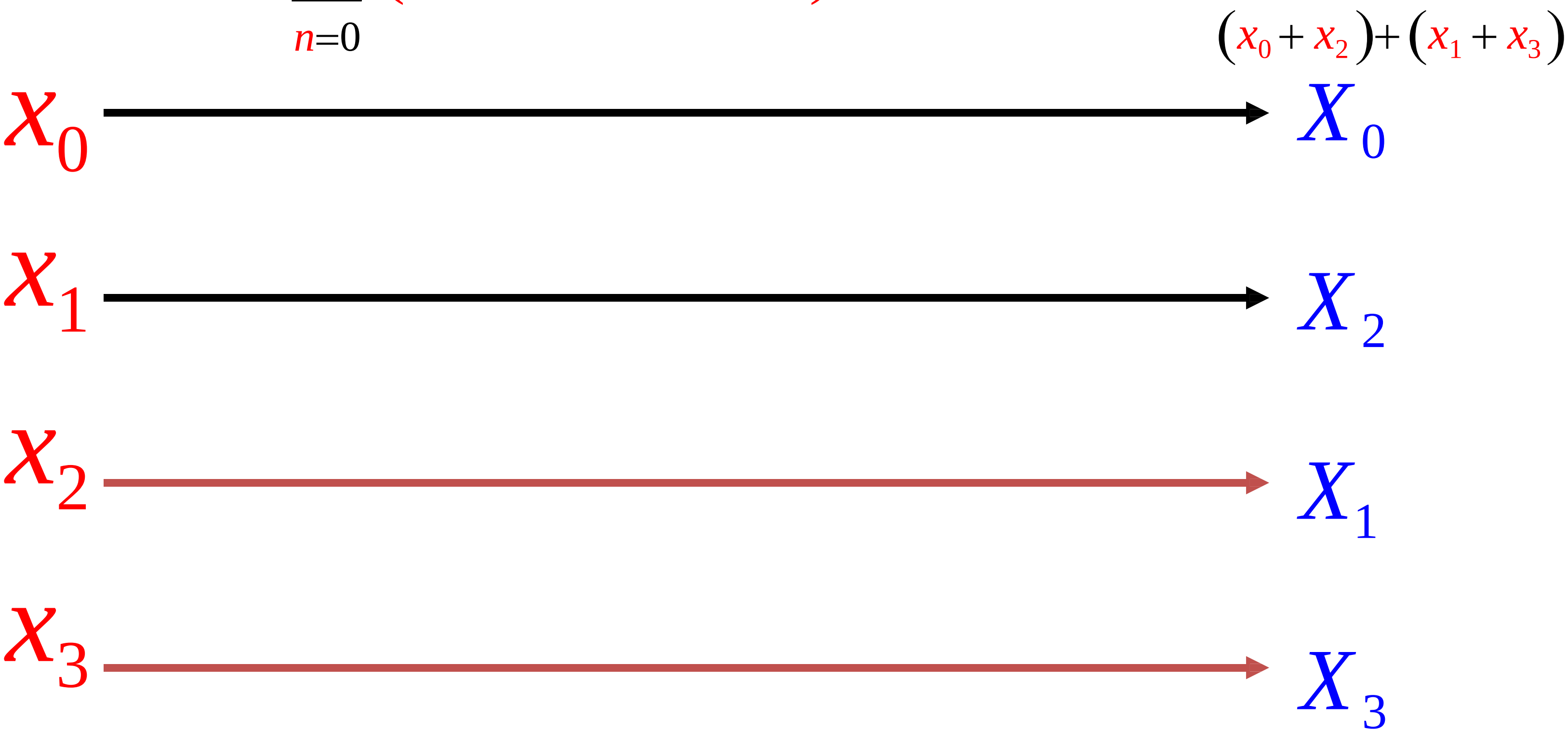
$$X[2r] = \sum_{n=0}^{1} \left( x[n] + x[n+2] \right) \mathbf{W}_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

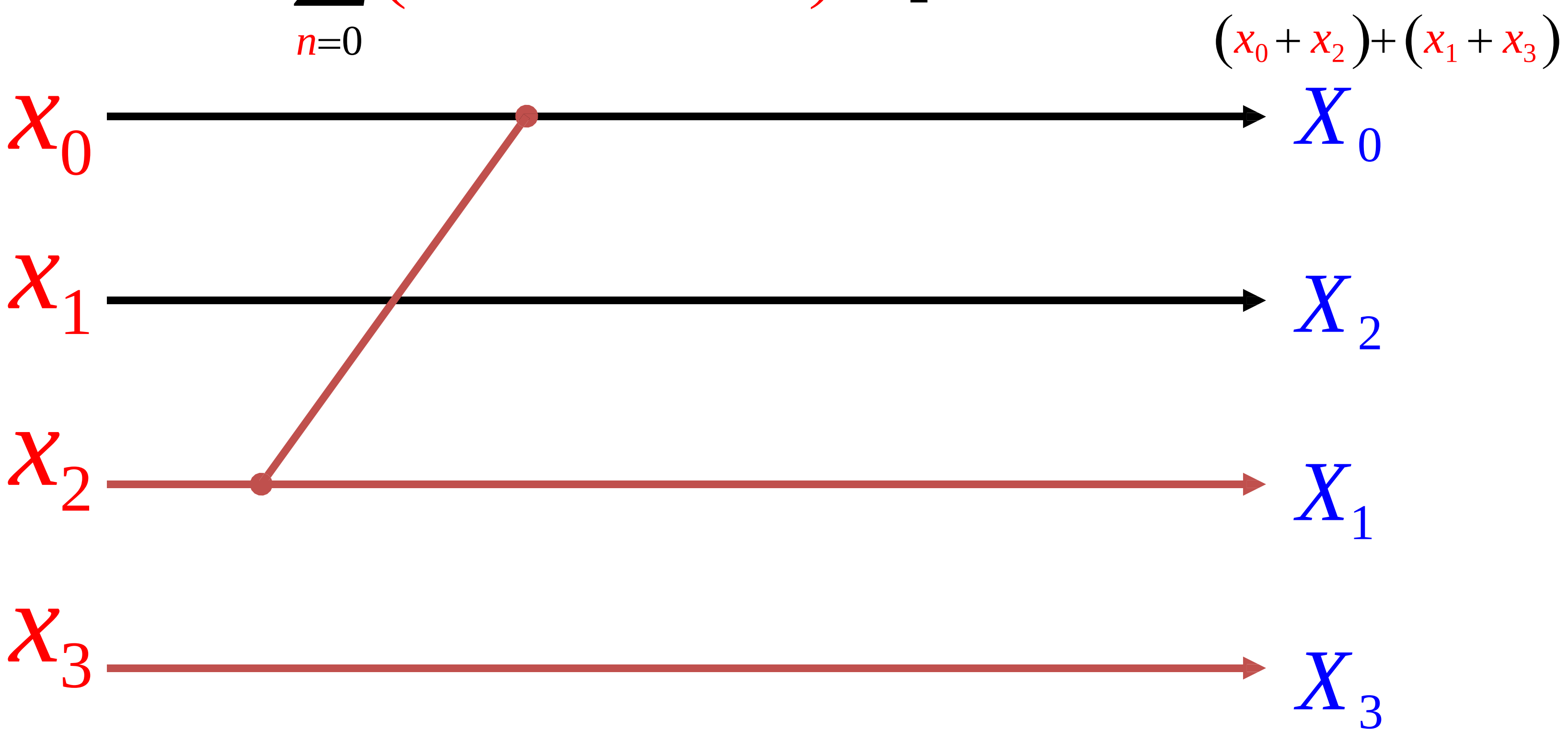
$$X[2r] = \sum_{n=0}^{1} \left( x[n] + x[n+2] \right) W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

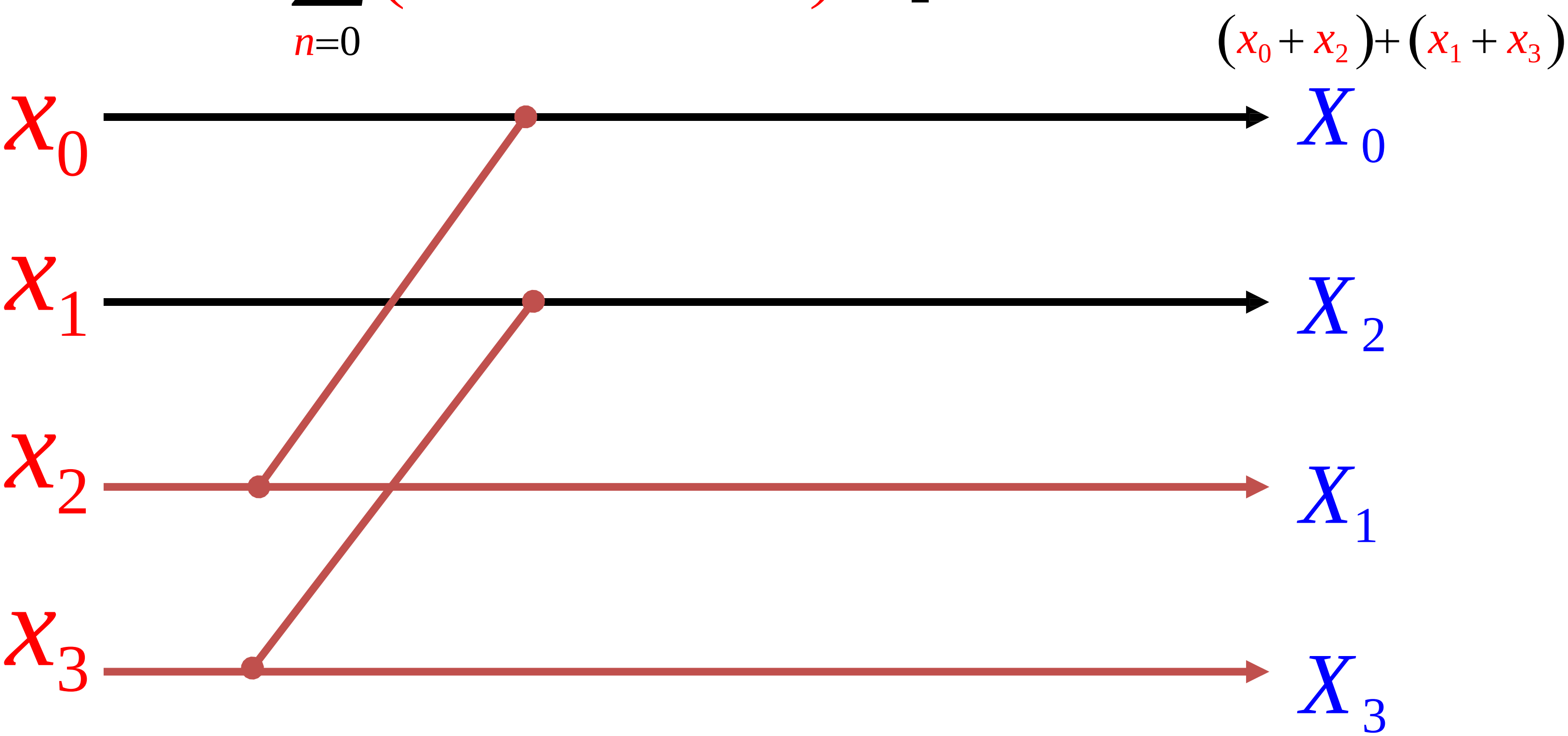
$$X[2r] = \sum_{n=0}^{1} \left( x[n] + x[n+2] \right) W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

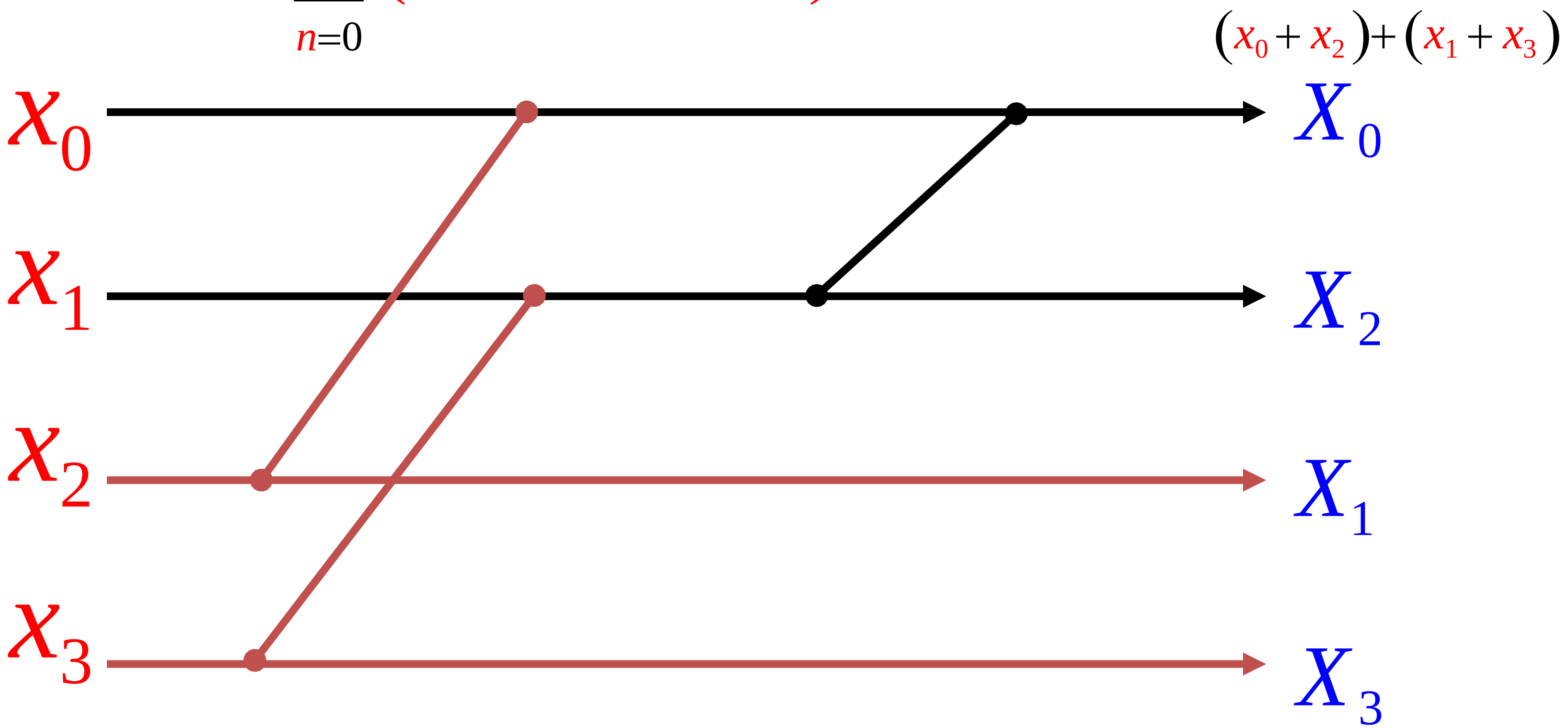
$$X[2r] = \sum_{n=0}^{1} \left( x[n] + x[n+2] \right) W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

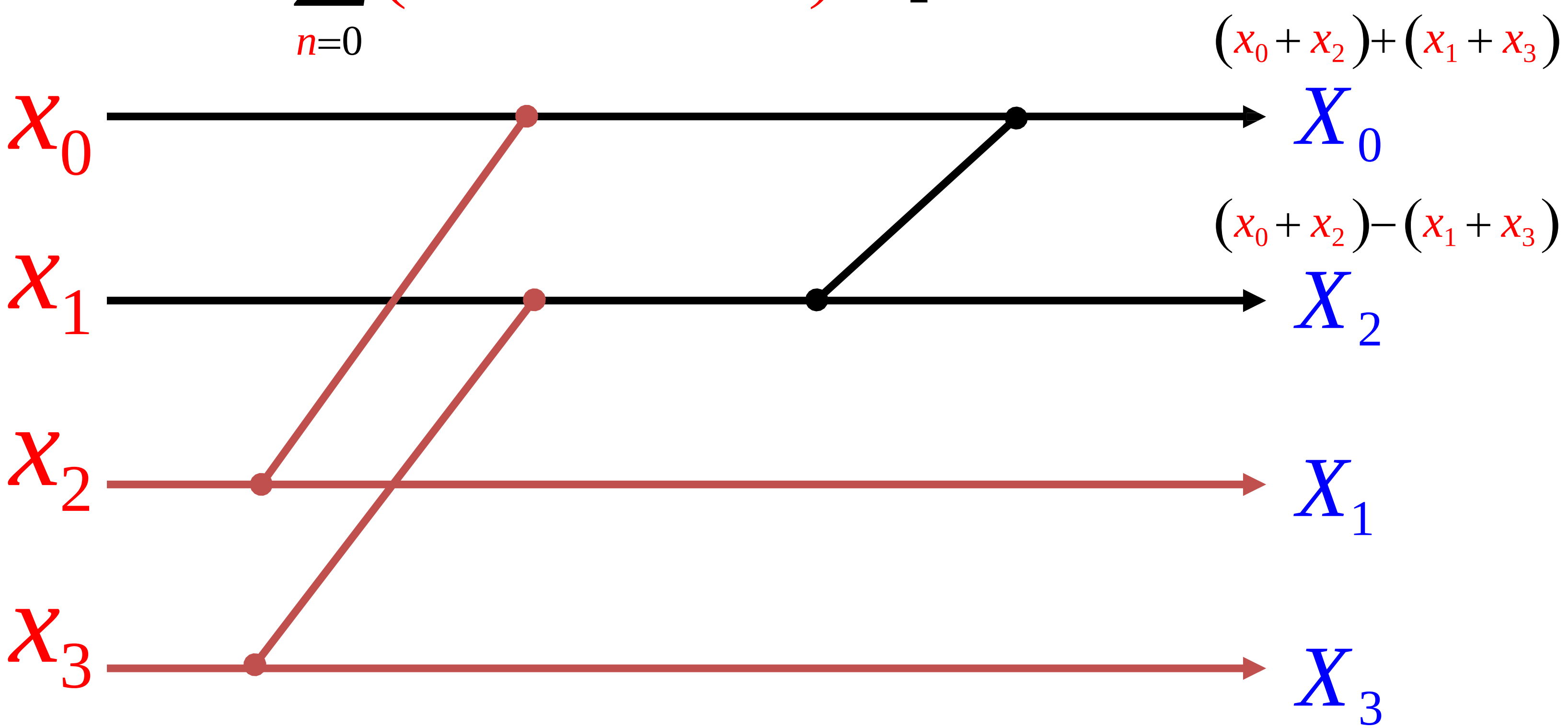
$$X[2r] = \sum_{n=0}^{1}\left(x[n] + x[n+2]\right)W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

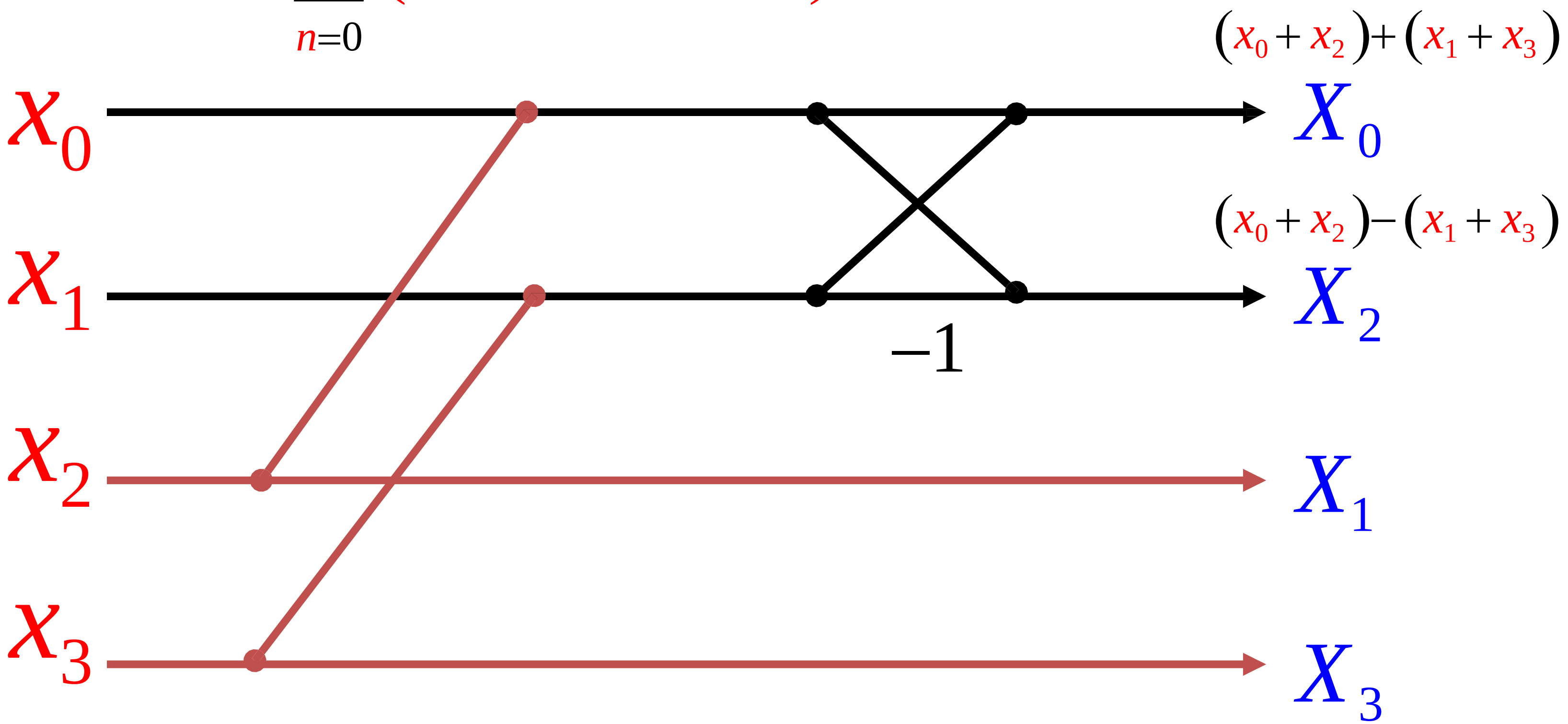
$$X[2r] = \sum_{n=0}^{1} \left(x[n] + x[n+2]\right) W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

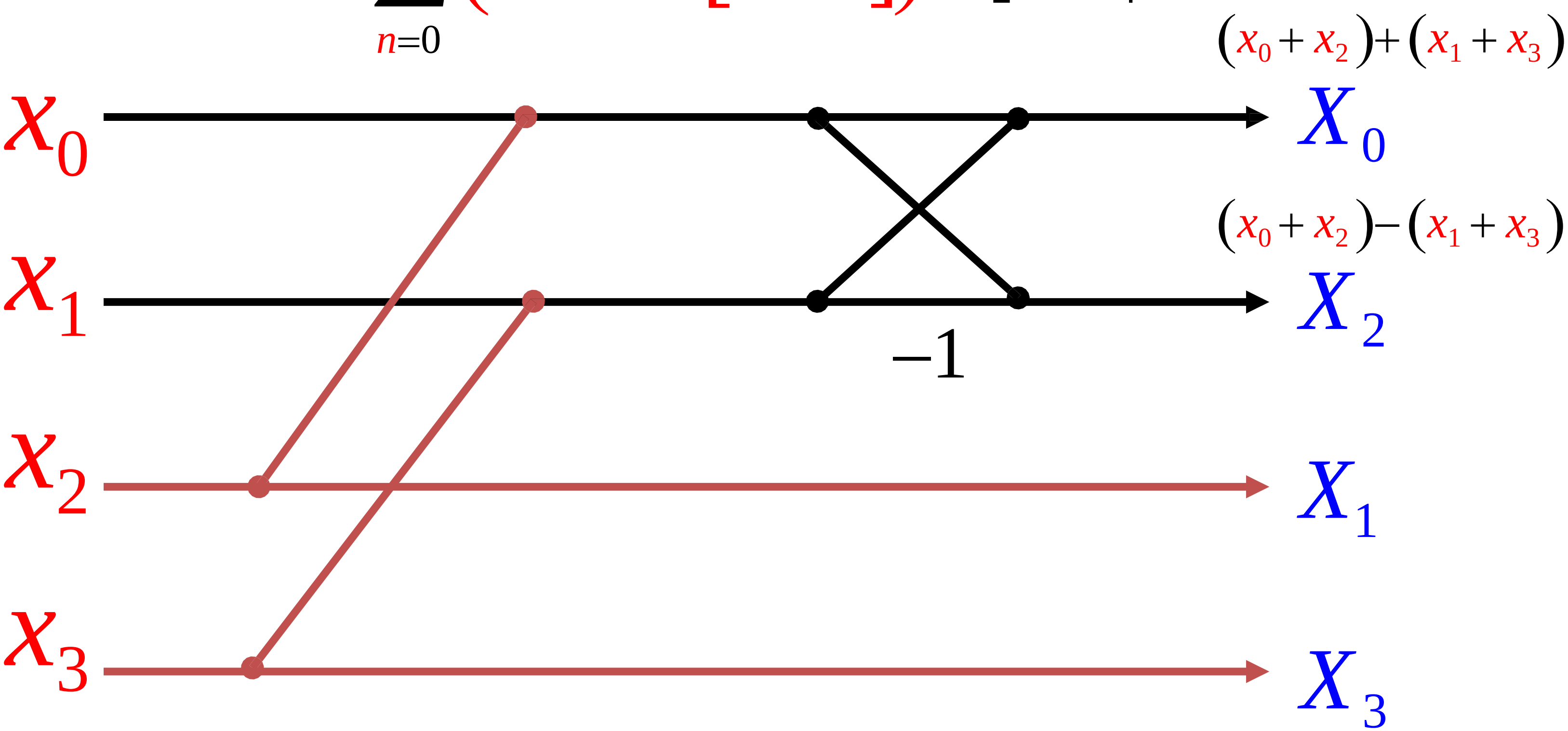
$$X[2r] = \sum_{n=0}^{1} \left(x[n] + x[n+2]\right) W_2^{rn} \qquad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0,\ x_1, x_2\ , x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

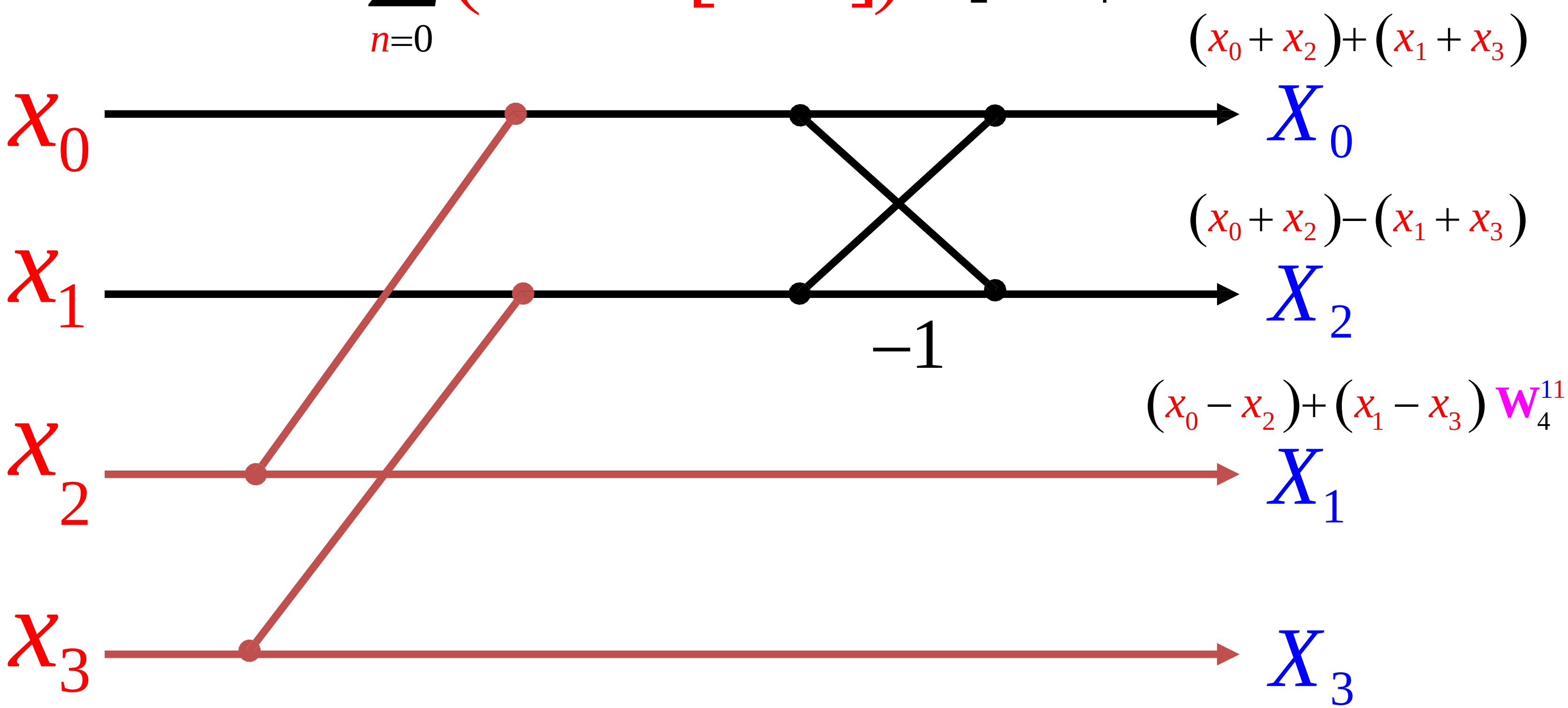
$$X[2r+1] = \sum_{n=0}^{1}\left(x[n] - x[n+2]\right)\mathbf{W}_2^{rn}\mathbf{W}_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

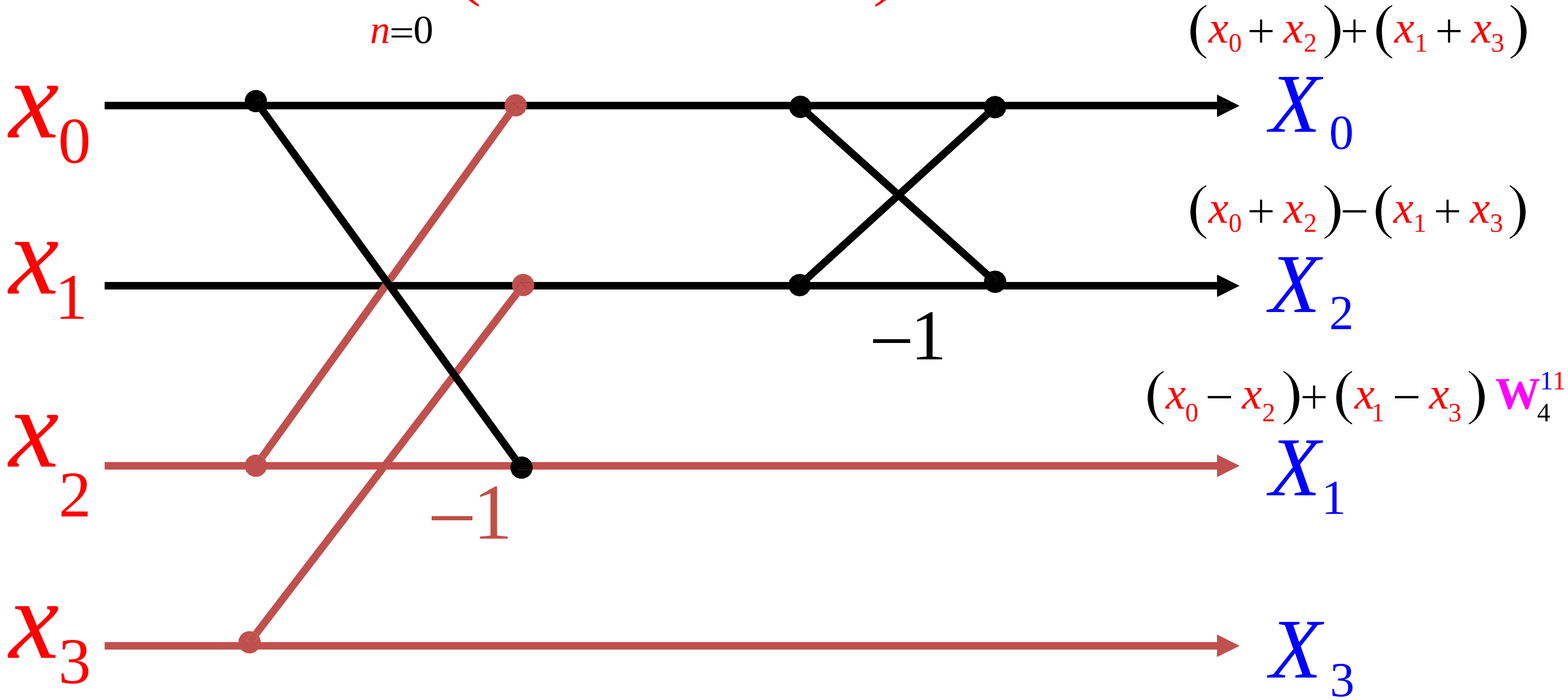
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

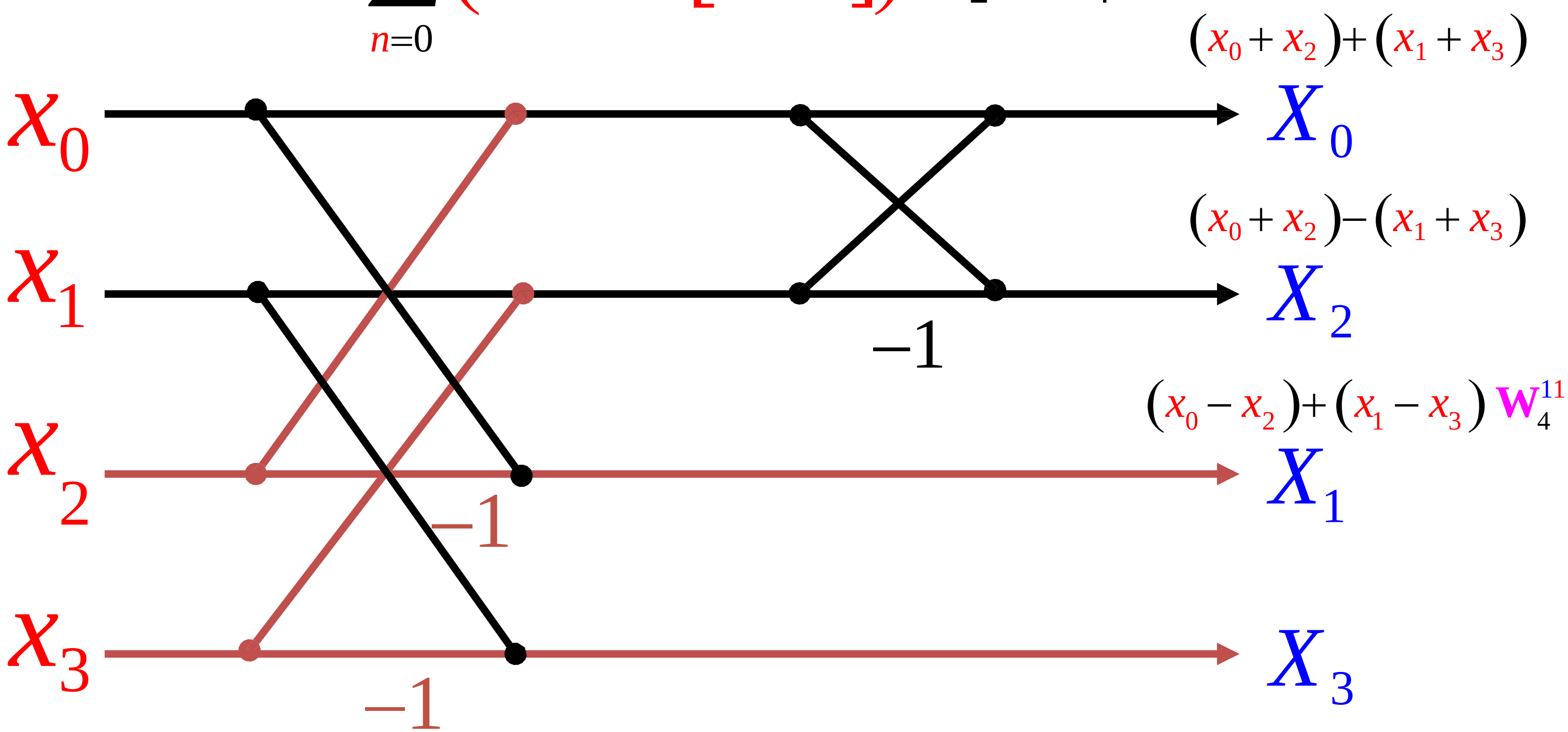
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

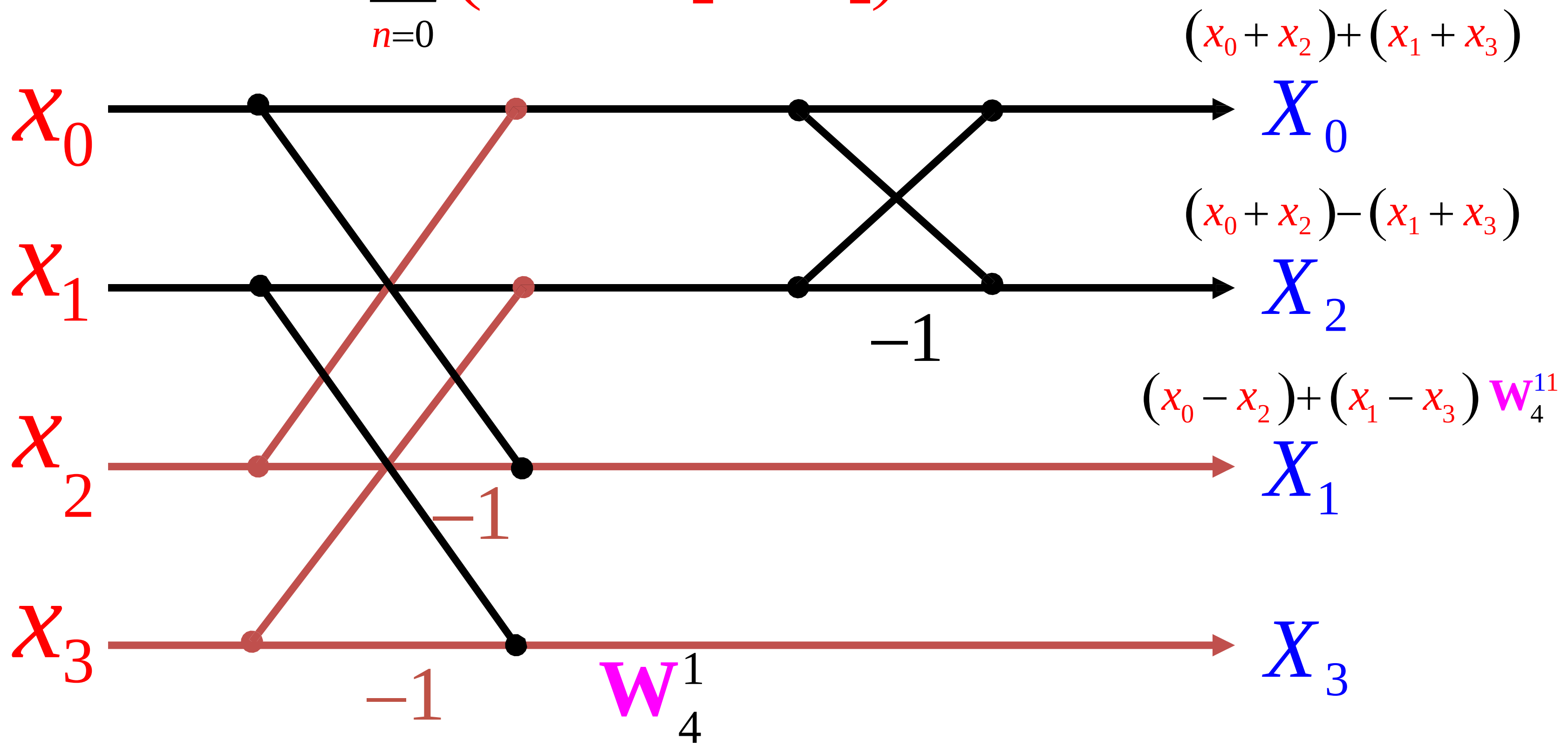
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

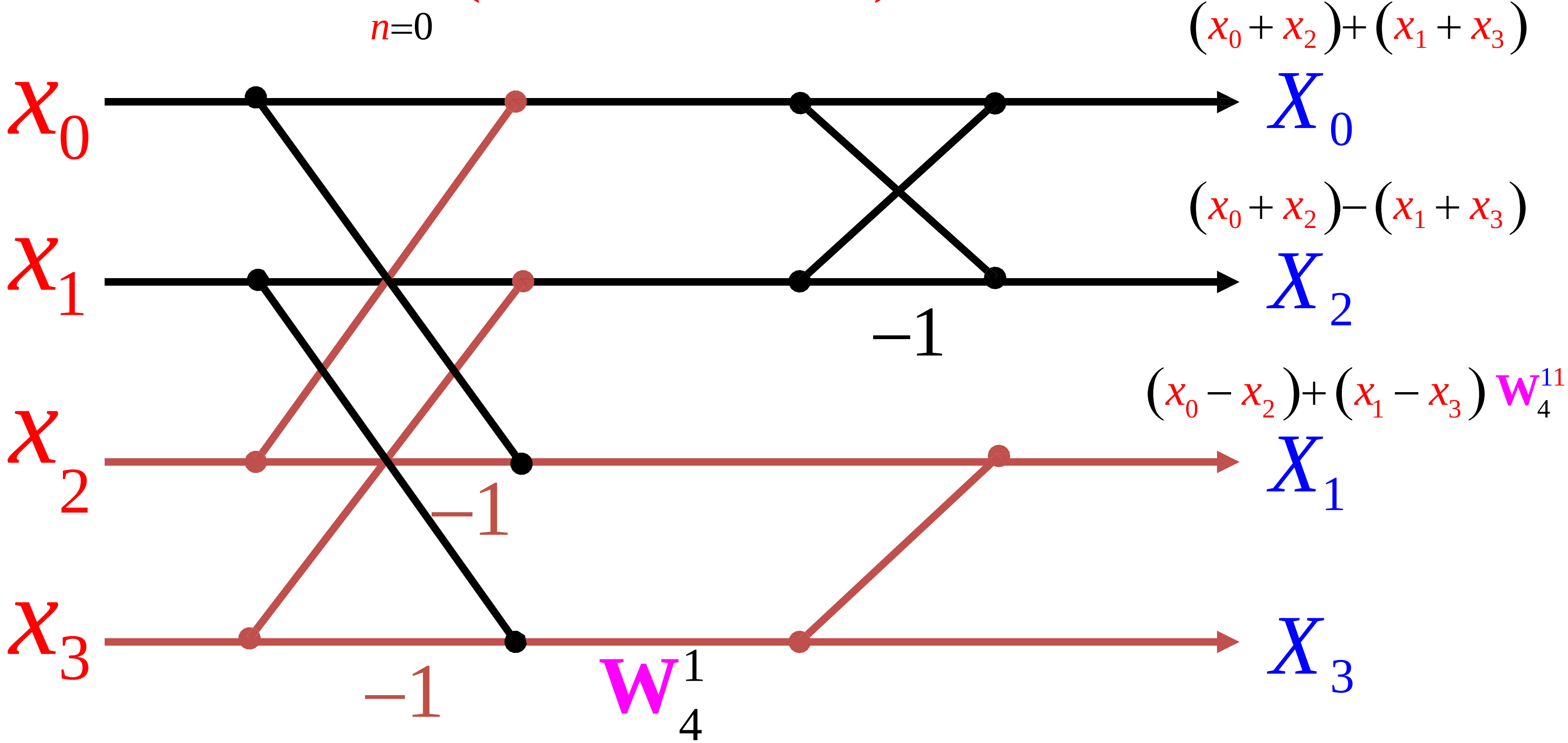
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

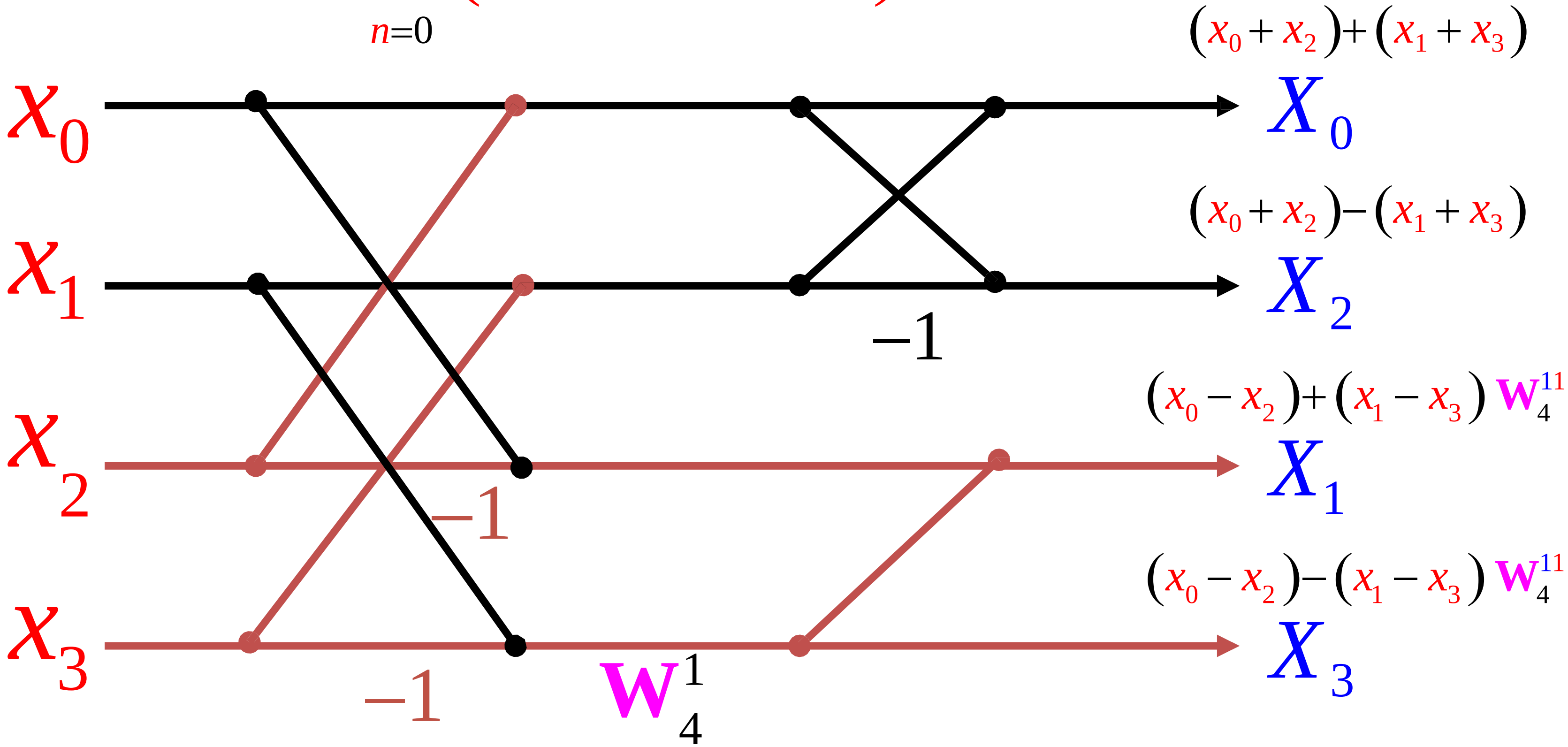
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

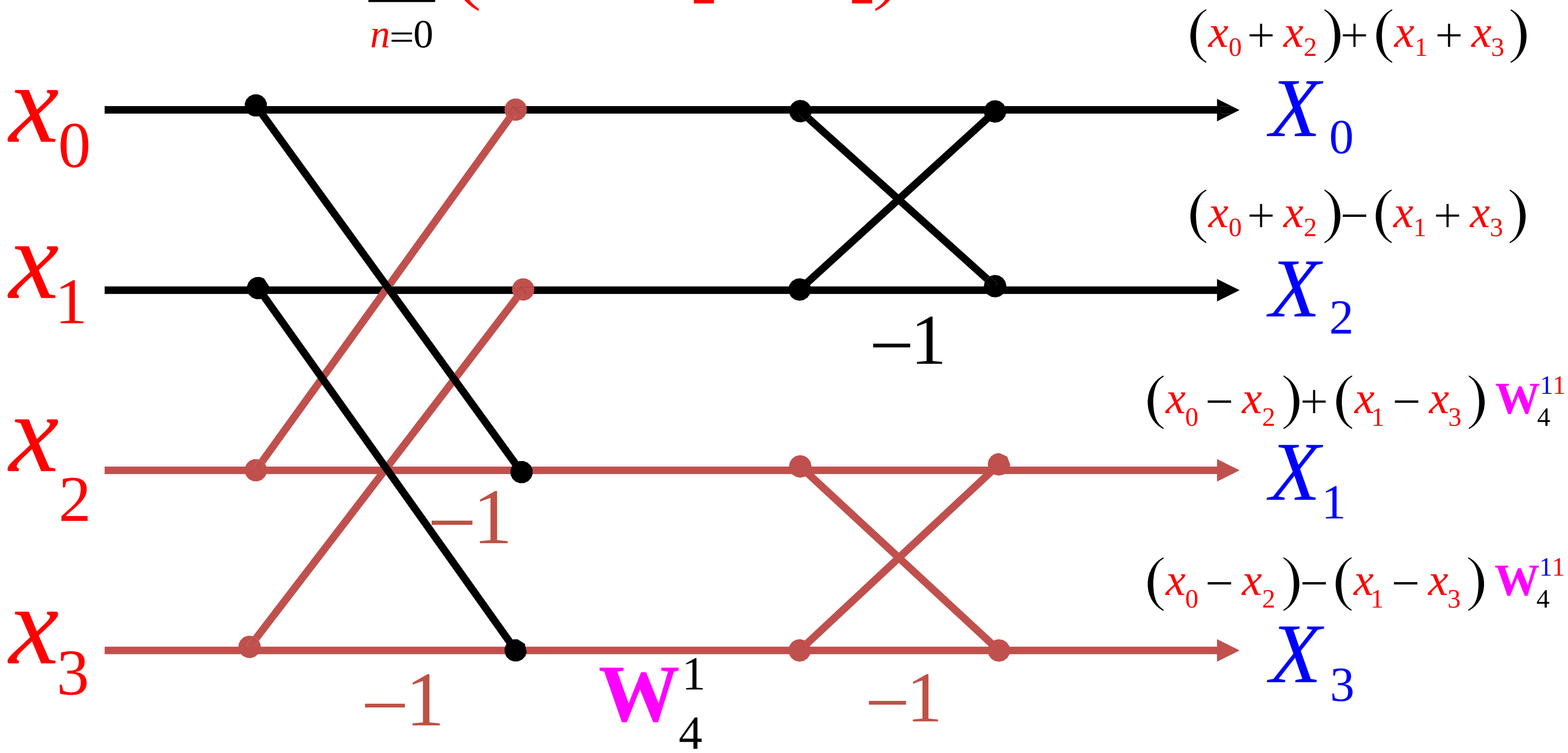
$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



## 4-point DIFFFT

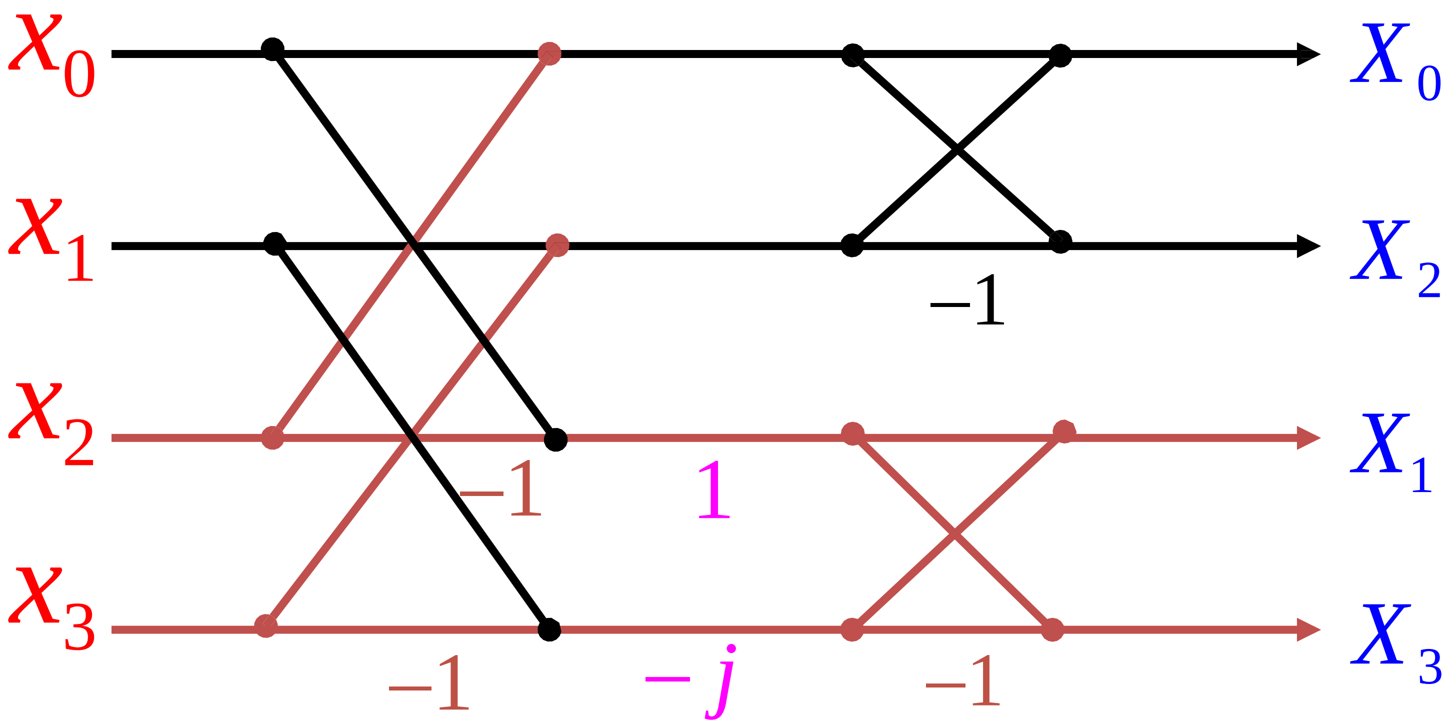
- Let $x[n] = \{x_0, x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$

$$X[2r+1] = \sum_{n=0}^{1} \left(x[n] - x[n+2]\right) W_2^{rn} W_4^{n} \quad r = 0,1$$



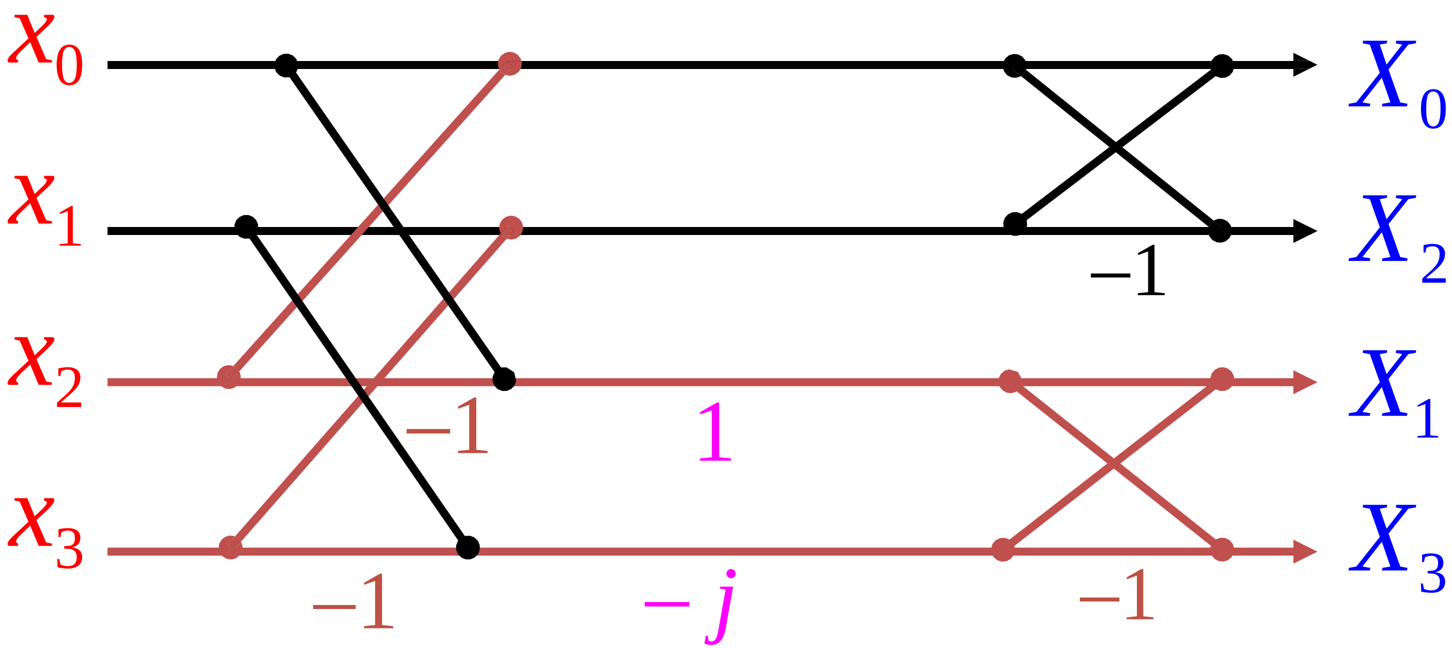
## 4-point DIFFFT

- Let $x[n] = \{x_0,\ x_1, x_2, x_3\} \xrightarrow{DFT} X(k) = \{X_0, X_1, X_2, X_3\}$



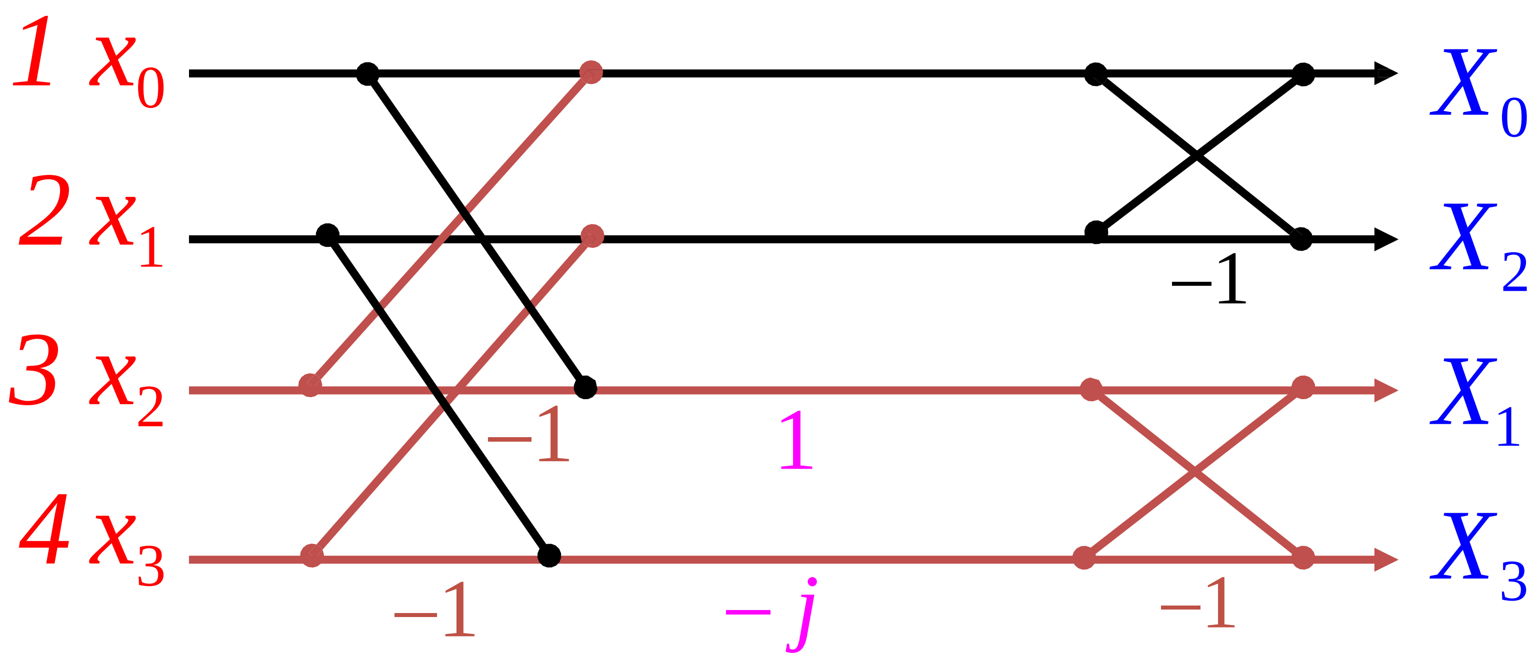
## Example

Compute DFT of $x[n] = \{1,2,1,2\}$ using DIFFFT algorithm.



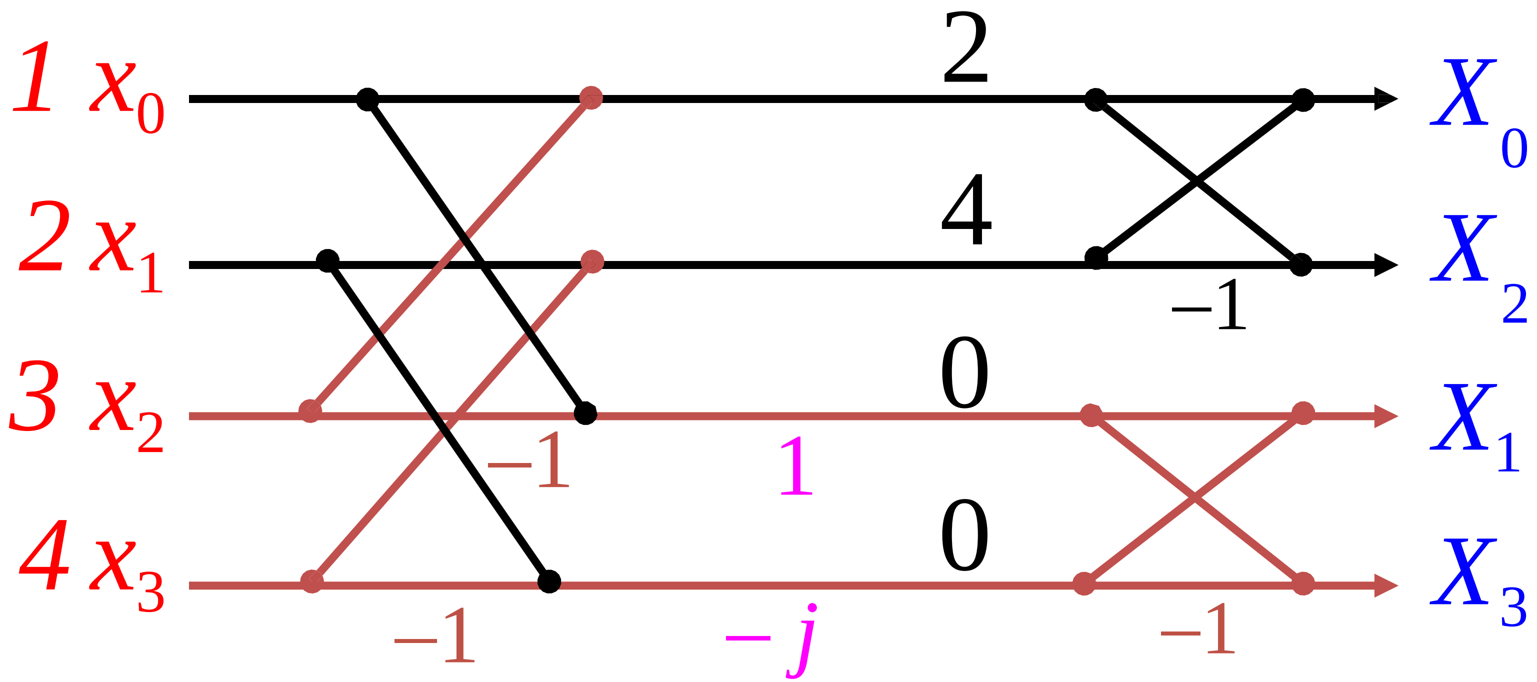
## Example

Compute DFT of $x[n] = \{1,2,1,2\}$ using DIFFFT algorithm.



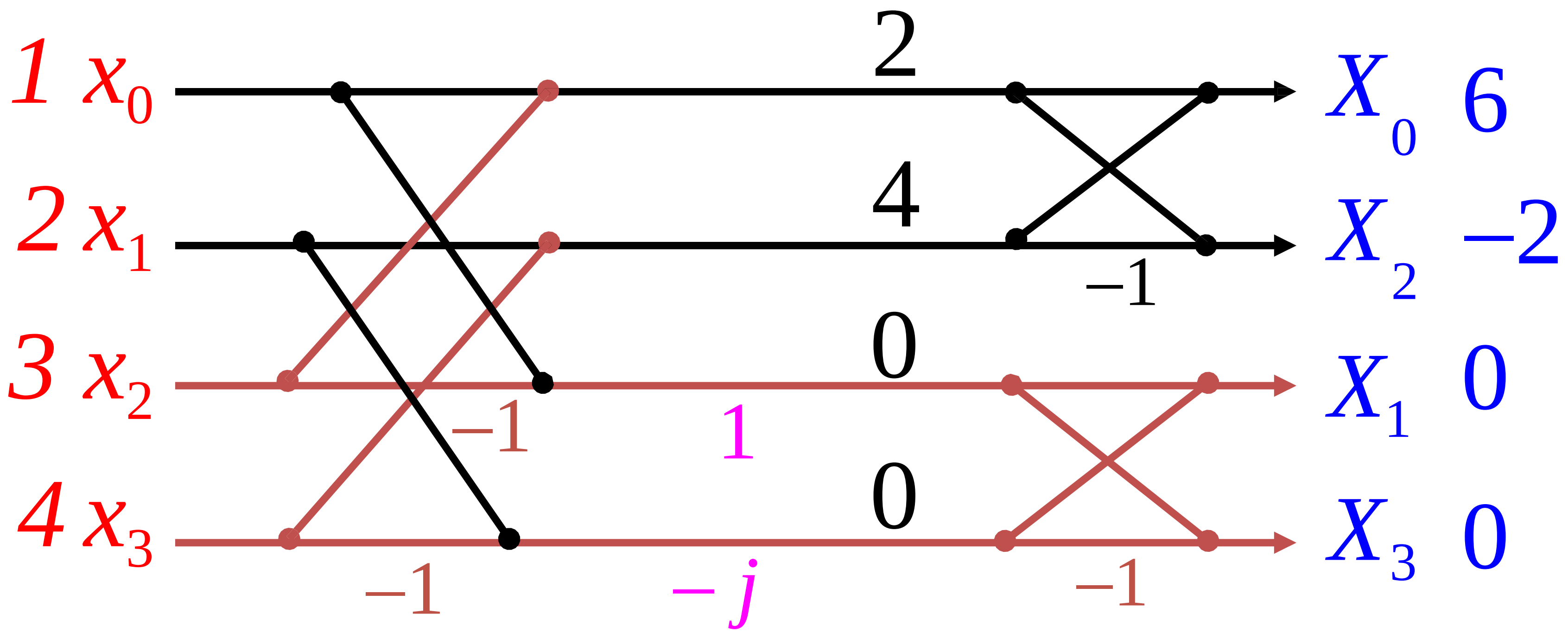
## Example

Compute DFT of $x[n] = \{1,2,1,2\}$ using DIFFFT algorithm.



## Example

Compute DFT of $x[n] = \{1,2,1,2\}$ using DIFFFT algorithm.



# Thank You