Additional Examples of Chapter 11: DSP Algorithm Implementation

Example E11.1: Analyze the digital filter structure of Figure E11.1 and develop a set of time-domain equations in terms of the input x[n], output y[n], and the intermediate variables $w_k[n]$ in a sequential order. Does this set describe a valid computational algorithm? Justify your answer by developing a matrix representation of the digital filter structure and by examining the matrix **F**.

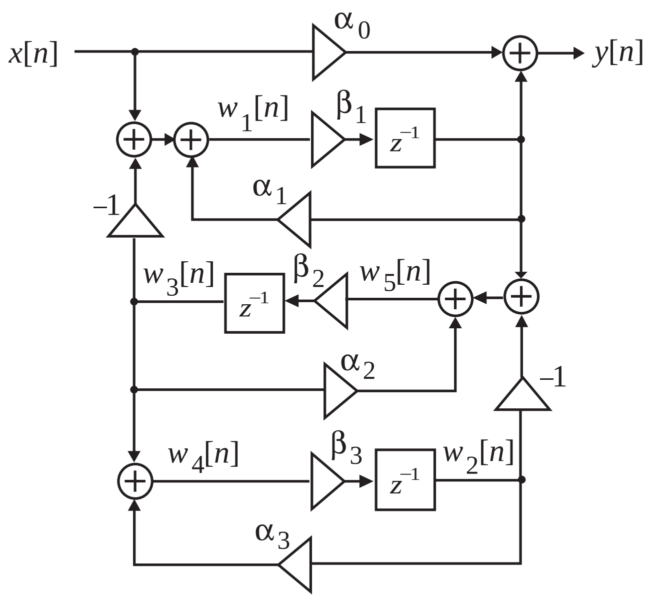


Figure E11.1

Answer: Analysis of Figure E11.1 yields

$w_1[n] = \alpha_1\beta_1 w_1[n-1] + x[n] - w_3[n],$
$w_2[n] = \beta_3 w_4[n-1],$
$w_3[n] = \beta_2 w_5[n-1],$
$w_4[n] = \alpha_3 w_2[n] + w_3[n],$
$w_5[n] = \alpha_2 w_3[n] - w_2[n] + \beta_1 w_1[n-1],$
$y[n] = \alpha_0 x[n] + \beta_1 w_1[n-1].$

In matrix form the above set of equations is given by:

$$\begin{bmatrix} w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \\ w_5[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_3 & 1 & 0 & 0 & 0 \\ 0 & -1 & \alpha_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \\ w_5[n] \\ y[n] \end{bmatrix} + \begin{bmatrix} \alpha_1\beta_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \beta_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \beta_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_1 & 0 & 0 & 0 & 0 & 0 \\ \beta_1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1[n-1] \\ w_2[n-1] \\ w_3[n-1] \\ w_4[n-1] \\ w_5[n-1] \\ y[n-1] \end{bmatrix} + \begin{bmatrix} x[n] \\ 0 \\ 0 \\ 0 \\ 0 \\ \alpha_0 x[n] \end{bmatrix}$$

Here the **F** matrix is given by

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_3 & 1 & 0 & 0 & 0 \\ 0 & -1 & \alpha_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Since the **F** matrix contains nonzero entries above the main diagonal, the above set of equations are not computable.

Example E11.2: Develop a computable set of time-domain equations describing the digital filter structure of Figure E11.1. Verify the computability condition by forming an equivalent matrix representation and by examining the matrix **F**.

Answer: A computable set of equations of the structure of Figure E11.1 is given by

$w_2[n] = \beta_3 w_4[n-1]$,
$w_3[n] = \beta_2 w_5[n-1]$,
$w_1[n] = \alpha_1\beta_1 w_1[n-1] - w_3[n] + x[n]$,
$w_4[n] = \alpha_3 w_2[n] + w_3[n]$,
$w_5[n] = \alpha_2 w_3[n] - w_2[n] + \beta_1 w_1[n-1]$,
$y[n] = \alpha_0 x[n] + \beta_1 w_1[n-1]$.

In matrix form the above set of equations is given by:

$$\begin{bmatrix} w_2[n] \\ w_3[n] \\ w_1[n] \\ w_4[n] \\ w_5[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ \alpha_3 & 1 & 0 & 0 & 0 & 0 \\ -1 & \alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_2[n] \\ w_3[n] \\ w_1[n] \\ w_4[n] \\ w_5[n] \\ y[n] \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \beta_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \beta_2 & 0 \\ 0 & 0 & \alpha_1\beta_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \beta_1 & 0 & 0 & 0 \\ 0 & 0 & \beta_1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1[n-1] \\ w_2[n-1] \\ w_3[n-1] \\ w_4[n-1] \\ w_5[n-1] \\ y[n-1] \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ x[n] \\ 0 \\ 0 \\ \alpha_0 x[n] \end{bmatrix}$$

Here the **F** matrix is given by

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ \alpha_3 & 1 & 0 & 0 & 0 & 0 \\ -1 & \alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Since the **F** matrix does not contain nonzero entries above the main diagonal, the new set of equations are computable.

Example E11.3: Analyze the digital filter structure of Figure E11.2 and develop a set of time-domain equations in terms of the input x[n], output y[n], and the intermediate variables $w_k[n]$ in a sequential order. Does this set describe a valid computational algorithm? Justify your answer by developing a matrix representation of the digital filter structure and by examining the matrix **F**.

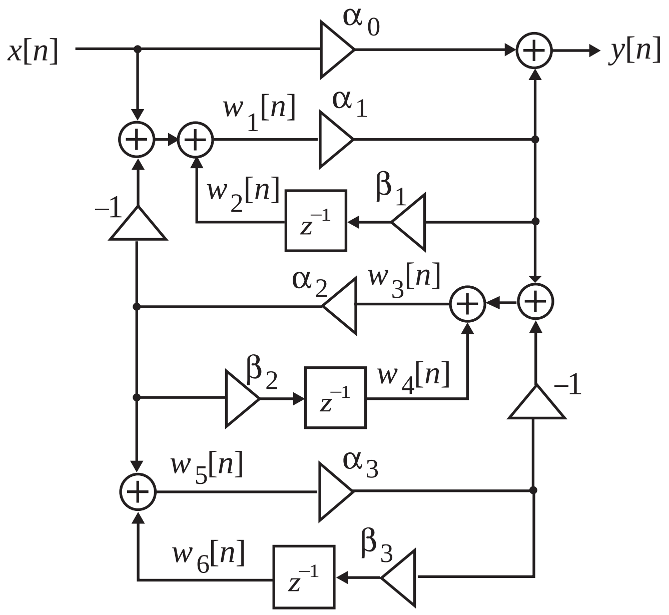


Figure E11.2

Answer: Analysis of Figure E11.2 yields

$$w_1[n] = x[n] - \alpha_2 w_3[n] + w_2[n],$$
$$w_2[n] = \alpha_1 \beta_1 w_1[n-1],$$
$$w_3[n] = \alpha_1 w_1[n] - \alpha_3 w_5[n] + w_4[n],$$
$$w_4[n] = \alpha_2 \beta_2 w_3[n-1],$$
$$w_5[n] = \alpha_2 w_3[n] + w_6[n],$$
$$w_6[n] = \alpha_3 \beta_3 w_5[n-1],$$
$$y[n] = \alpha_0 x[n] + \alpha_1 w_1[n].$$

In matrix form

$$\begin{bmatrix} w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \\ w_5[n] \\ w_6[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} 0 & 1 & -\alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & 0 & 0 & 1 & -\alpha_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \\ w_5[n] \\ w_6[n] \\ y[n] \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1\beta_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2\beta_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_3\beta_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1[n-1] \\ w_2[n-1] \\ w_3[n-1] \\ w_4[n-1] \\ w_5[n-1] \\ w_6[n-1] \\ y[n-1] \end{bmatrix} + \begin{bmatrix} x[n] \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \alpha_0 x[n] \end{bmatrix}$$

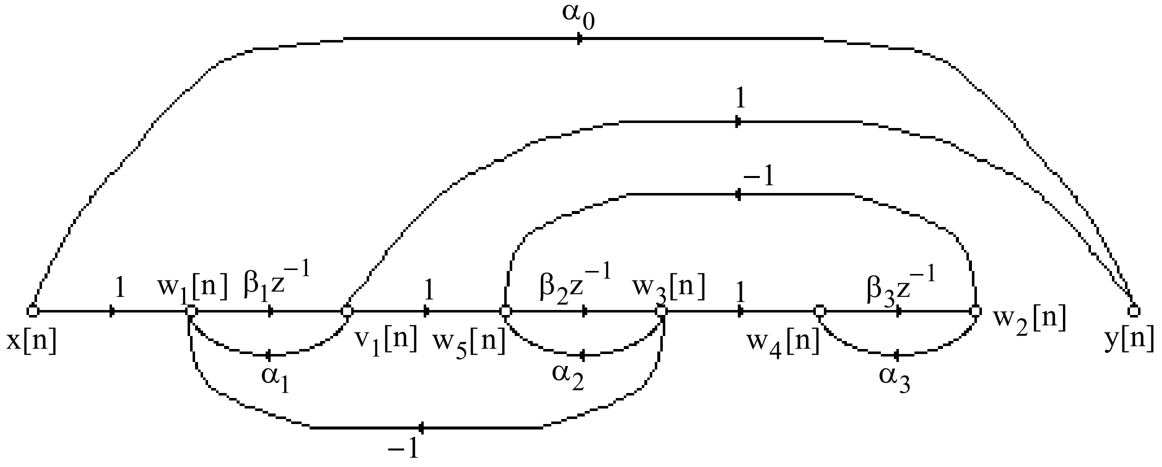
Here the **F** matrix is given by $\mathbf{F} = \begin{bmatrix} 0 & 1 & -\alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & 0 & 0 & 1 & -\alpha_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

Since the **F** matrix contains nonzero entries above the main diagonal, the above set of equations are not computable.

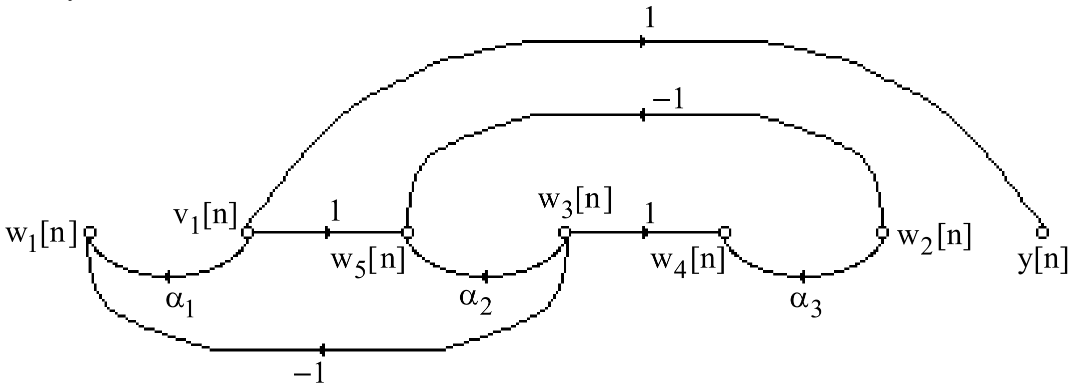
Example E11.4: Develop the precedence graph of the digital filter structure of Figure E11.1, and investigate its realizability. If the structure is found to be realizable, then from the precedence graph, determine a valid computational algorithm describing the structure.

Answer: The signal-flow graph representation of the structure of Figure E11.1 is shown below:

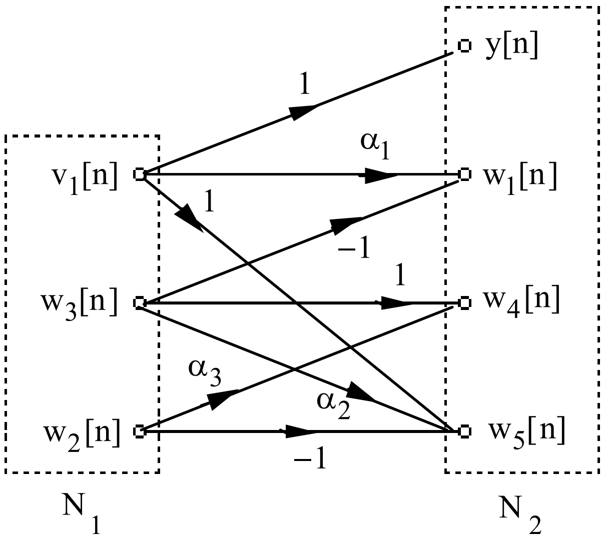
Additional Examples of Chapter 11: DSP Algorithm Implementation



The reduced signal-flow graph obtained by removing the branches going out of the input node and the delay branches is as indicated below:



From the above signal-flow graph we arrive at its precedence graph shown below:

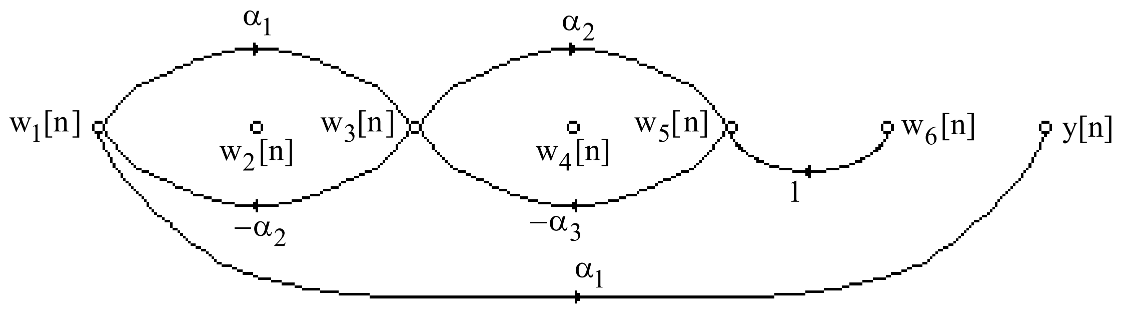


In the above precedence graph, the set N_1 contains nodes with only outgoing branches and the final set N_2 contains nodes with only incoming branches. As a result, the structure of Figure E11.1 has no delay-free loops. A valid computational algorithm by computing the node variables in set N_1 first in any order followed by computing the node variables in set N_2 in any order. For example, one valid computational algorithm is given by

$v_1[n] = \beta_1 w_1[n-1]$,
$w_3[n] = \beta_2 w_5[n-1]$,
$w_2[n] = \beta_3 w_4[n-1]$,
$w_1[n] = \alpha_1 v_1[n] - w_3[n] + x[n]$,
$w_4[n] = \alpha_3 w_2[n] + w_3[n]$,
$w_5[n] = \alpha_2 w_3[n] - w_2[n] + \beta_1 v_1[n]$,
$y[n] = \alpha_0 x[n] + v_1[n]$.

Example E11.5: Develop the precedence graph of the digital filter structure of Figure E11.2, and investigate its realizability. If the structure is found to be realizable, then from the precedence graph, determine a valid computational algorithm describing the structure.

Answer: The reduced signal-flow graph obtained by removing the branches going out of the input node and the delay branches from the signal-flow graph representation of the structure of Figure E11.2 is as indicated below:



The only node with outgoing branch is $w_6[n]$ and hence it is the only member of the set N_1. Since it is not possible to find a set of nodes N_2 with incoming branches from N_1 and all other branches being outgoing, the structure of Figure E11.2 has delay-free loops and is therefore not realizable.

Example E11.5: Determine the transfer function H(z) of a third-order causal IIR digital filter whose first 10 impulse response samples are given by

$$\{h[n]\} = \{2, \quad -5, \quad 6, \quad -2, \quad -9, \quad 18, \quad -7, \quad -31, \quad 65, \quad -30\}.$$

Additional Examples of Chapter 11: DSP Algorithm Implementation

Answer: $H(z) = \dfrac{p_0 + p_1 z^{-1} + p_2 z^{-2} + p_3 z^{-3}}{1 + d_1 z^{-1} + d_2 z^{-2} + d_3 z^{-3}}$. The equation corresponding to Eq. (11.16) is given by

$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ -5 & 2 & 0 & 0 \\ 6 & -5 & 2 & 0 \\ -2 & 6 & -5 & 2 \\ -9 & -2 & 6 & -5 \\ 18 & -9 & -2 & 6 \\ -7 & 18 & -9 & -2 \end{bmatrix} \begin{bmatrix} 1 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix}.$$ Therefore, from Eq. (11.20) we have

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = -\begin{bmatrix} -2 & 6 & -5 \\ -9 & -2 & 6 \\ 18 & -9 & -2 \end{bmatrix}^{-1} \begin{bmatrix} -9 \\ 18 \\ -7 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$$ and from Eq. (11.15) we have

$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ -5 & 2 & 0 & 0 \\ 6 & -5 & 2 & 0 \\ -2 & 6 & -5 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 2 \\ -3 \end{bmatrix}.$$ Hence, $H(z) = \dfrac{2 - z^{-1} + 2z^{-2} - 3z^{-3}}{1 + 2z^{-1} + 3z^{-2} + z^{-3}}$.

Example E11.6: A sequence y[n] is to be formed by a linear convolution of a length-8 sequence x[n] with a length-6 sequence h[n]. To determine y[n], we can follow one of the following methods:
Method #1: Direct implementation of the linear convolution.
Method #2: Implementation of the linear convolution via a single circular convolution.
Method #3: Implementation of the linear convolution using a radix-2 FFT algorithm.
Determine the least number of real multiplications needed in each of the above methods. For the radix-2 FFT algorithm, do not include in the count multiplication by ± 1, $\pm j$, and W_N^0.

Answer: Method #1: Total # of real multiplications required

$$= 2\left(\sum_{n=1}^{N} n\right) + N(L - N - 1) = 2\left(\sum_{n=1}^{6} n\right) + 6(8 - 6 - 1) = 48.$$

Method #2: Total # of real multiplications required $= 13^2 = 169$.

Method #3: Linear convolution via radix-2 FFT - The process involves computing the 16-point FFT G[k] of the length-16 complex sequence $g[n] = x_e[n] + jh_e[n]$ where $x_e[n]$ and $h_e[n]$ are length-16 sequences obtained by zero-padding x[n] and h[n], respectively. Then recovering the 16-point DFTs, $X_e[k]$ and $H_e[k]$, of $x_e[n]$ and $h_e[n]$, respectively, from G[k]. Finally, the IDFT of the product $Y[k] = X_e[k] \cdot H_e[k]$ yields y[n].

Now, the first stage of the 16-point radix-2 FFT requires 0 complex multiplications, the second stage requires 0 complex multiplications, the third stage requires 4 complex multiplications, and the multiplications.

# of complex mult. to implement G[k] = 10

# of complex mult. to recover $X_e[k]$ and $H_e[k]$ from G[k] = 0

# of complex mult. to form $Y[k] = X_e[k] \cdot H_e[k] = 16$

# of complex mult. to form the IDFT of Y[k] = 10

Hence, the total number of complex mult. = 36

A direct implementation of a complex multiplication requires 4 real multiplications resulting in a total of 4×36 = 144 real multiplications for Method #3. However, if a complex multiply can be implemented using 3 real multiplies (see Problem 11.13), in which case Method #3 requires a total of 3×36 = 108 real multiplications.

Example E11.7: An input sequence x[n] of length 1024 is to be filtered using a linear-phase FIR filter $h[n]$ of length 34. This filtering process involves the linear convolution of two finite-length sequences and can be computed using the overlap-add algorithm discussed in Section 5.10.2 where the short linear convolutions are performed using the DFT-based approach of Figure 5.13 with the DFTs implemented by the Cooley-Tukey FFT algorithm.

(a) Determine the appropriate power-of-2 transform length that would result in a minimum number of multiplications and calculate the total number of multiplications that would be required.

(b) What would be the total number of multiplications if the direct convolution method is used?

Answer: (a) Since the impulse response of the filter is of length 34, the transform length N should be greater than 34. If L denotes the number of input samples used for convolution, then L = N – 33. So for every L samples of the input sequence, an N-point DFT is computed and multiplied with an N-point DFT of the impulse response sequence h[n] (which needs to be computed only once), and finally an N-point inverse of the product sequence is evaluated. Hence, the total number R_M of complex multiplications required (assuming N is a power-of-2) is given by

$$R_M = \left\lceil \frac{1024}{N-33} \right\rceil \left(N \log_2 N + N \right) + \frac{N}{2} \log_2 N.$$

It should be noted that in developing the above expression, multiplications due to twiddle factors of values ± 1 and $\pm j$ have not been excluded. The values of R_M for different values of N are as follows:

For N = 64, R_M = 15,424
For N = 128, R_M = 11,712
for N = 256, R_M = 12,544

$$\text{for } N = 512, \quad R_M = 17{,}664$$

Hence, N = 128 is the appropriate choice for the transform length requiring 14,848 complex multiplications or equivalently, $11{,}712 \times 3 = 35{,}136$ real multiplications.

Since the first stage of the FFT calculation process requires only multiplications by ± 1, the total number of complex multiplications for N = 128 is actually

$$R_M = \left\lceil \frac{1024}{N-33} \right\rceil \left(N \log_2 N + N \right) + \frac{N}{2} \log_2 N - \frac{N}{2} = 11{,}648$$

or equivalently, $11{,}648 \times 3 = 34{,}944$ real multiplications.

(b) For direct convolution, # of real multiplications =

$$2\left(\sum_{n=1}^{N} n \right) + N(L - N - 1) = 2\left(\sum_{n=1}^{34} n \right) + 34(1024 - 34 - 1) = 34{,}816.$$

Example E11.8: Develop the index mapping for implementing an N-point DFT X[k] of a length-N sequence x[n] using the Cooley-Tukey FFT algorithm for N = 12.

Answer: N = 12. Choose $N_1 = 4$ and $N_2 = 3$. Thus,

$$n = n_1 + 4n_2, \begin{cases} 0 \le n_1 \le 3 \\ 0 \le n_2 \le 2 \end{cases}, \text{ and } k = 3k_1 + k_2, \begin{cases} 0 \le k_1 \le 3 \\ 0 \le k_2 \le 2 \end{cases}.$$

$n_2 \backslash n_1$	0	1	2	3
0	x[0]	x[1]	x[2]	x[3]
1	x[4]	x[5]	x[6]	x[7]
2	x[8]	x[9]	x[10]	x[11]

$k_2 \backslash k_1$	0	1	2	3
0	X[0]	X[3]	X[6]	X[9]
1	X[1]	X[4]	X[7]	X[10]
2	X[2]	X[5]	X[8]	X[11]

Example E11.9: Develop the index mapping for implementing an N-point DFT X[k] of a length-N sequence x[n] using the prime factor algorithm for N = 12.

Answer: N = 12. Choose $N_1 = 4$ and $N_2 = 3$.

$$A = 3, \; B = 4, \; C = 3 < 3^{-1} >_4 = 9, \; D = 4 < 4^{-1} >_3 = 4.$$

$$n = < 3n_1 + 4n_2 >_{12}, \begin{cases} 0 \le n_1 \le 3 \\ 0 \le n_2 \le 2 \end{cases} \qquad k = < 9k_1 + 4k_2 >_{12} \begin{cases} 0 \le k_1 \le 3 \\ 0 \le k_2 \le 2 \end{cases}.$$

$n_2 \backslash n_1$	0	1	2	3
0	x[0]	x[3]	x[6]	x[9]
1	x[4]	x[7]	x[10]	x[1]
2	x[8]	x[11]	x[2]	x[5]

$k_2 \backslash k_1$	0	1	2	3
0	X[0]	X[9]	X[6]	X[3]
1	X[4]	X[1]	X[10]	X[7]
2	X[8]	X[5]	X[2]	X[11]

Additional Examples of Chapter 11: DSP Algorithm Implementation

---

Example E11.10: Develop a scheme to compute the 3072-point DFT of a sequence of length 3072 using 512-point FFT modules and complex multiplications and additions. Show the scheme in block diagram form. How many FFT modules and complex multiplications and additions are needed for the overall computation?

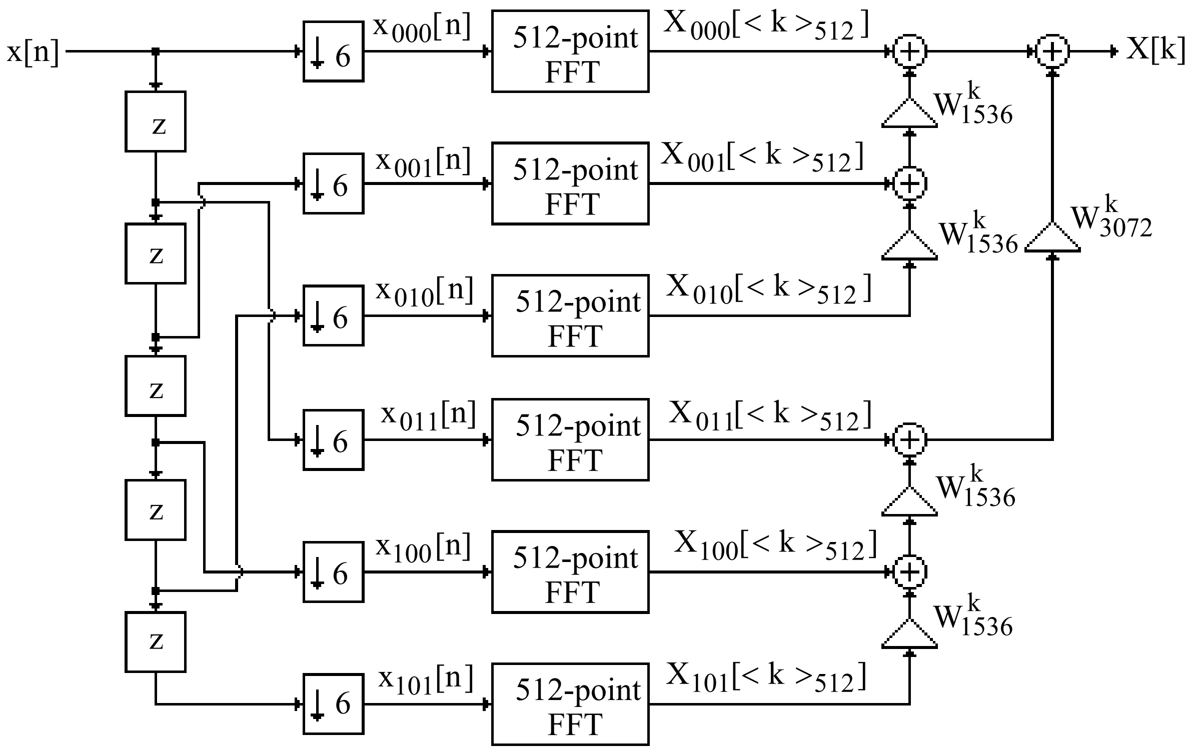
Answer: Note that $3072 = 512 \times 6$. Now an N-point DFT , with N divisible by 6, can be computed as follows:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} = X_0[\langle k\rangle_{N/6}] + W_N^k \cdot X_1[\langle k\rangle_{N/6}] + W_N^{2k} \cdot X_2[\rangle k\rangle_{N/6}]$$
$$+ W_N^{3k} \cdot X_3[\langle k\rangle_{N/6}] + W_N^{4k} \cdot X_4[\langle k\rangle_{N/6}] + W_N^{5k} \cdot X_5[\langle k\rangle_{N/6}], \text{ where}$$

$$X_\ell[\langle k\rangle_{N/6}] = \sum_{r=0}^{\frac{N}{6}-1} x[6r+\ell] W_{N/6}^{rk}, \; 0 \le \ell \le 5.$$ For N = 3072, we thus get

$$X[k] = X_0[\langle k\rangle_{512}] + W_{3072}^k \cdot X_1[\langle k\rangle_{512}] + W_{3072}^{2k} \cdot X_2[\langle k\rangle_{512}]$$
$$+ W_{3072}^{3k} \cdot X_3[\langle k\rangle_{512}] + W_{3072}^{4k} \cdot X_4[\langle k\rangle_{512}] + W_{3072}^{5k} \cdot X_5[\langle k\rangle_{512}], \text{ where}$$

$$X_\ell[\langle k\rangle_{512}] = \sum_{r=0}^{511} x[6r+\ell] W_{512}^{rk}, \; 0 \le \ell \le 5.$$



Additional Examples of Chapter 11: DSP Algorithm Implementation

Now an N-point FFT algorithm requires $\frac{N}{2}\log_2 N$ complex multiplications and $N\log_2 N$ complex additions. Hence, an $\frac{N}{6}$ – point FFT algorithm requires $\frac{N}{12}\log_2\left(\frac{N}{6}\right)$ complex multiplications and $\frac{N}{6}\log_2\left(\frac{N}{6}\right)$ complex additions. In addition, we need $5\times N$ complex multiplications and $5\times N$ complex additions to compute the N-point DFT X[k]. Hence, for N = 3072, the evaluation of X[k] using 6 (512)-point FFT modules requires $\frac{N}{12}\log_2\left(\frac{N}{6}\right) + 5\times N =$ $256\times\log_2(512)+5\times 3072 = 17{,}664$ complex multiplications and $\frac{N}{6}\log_2\left(\frac{N}{6}\right) + 5\times N$ $512\times\log_2(512)+5\times 3072 = 19{,}968$ complex additions.

It should be noted that a direct computation of the 3072-point DFT would require 9,437,184 complex multiplications and 9,434,112 complex additions.

Example E11.11: A 1024-point DFT of a length-1000 sequence x[n] is to be computed. How many zero-valued samples should be appended to x[n] prior to the computation of the DFT? What are the total number of complex multiplications and additions needed for the direct evaluation of all DFT samples? What are the total number of complex multiplications and additions needed if a Cooley-Tukey type FFT is used to compute the DFT samples?

Answer: (a) # of zero-valued samples to be added is 1024 – 1000 = 24.

(b) Direct computation of a 1024-point DFT of a length-1000 sequence requires $(1000)^2 = 1{,}000{,}000$ complex multiplications and $999\times 1000 = 999{,}000$ complex additions.

(c) A 1024-point Cooley-Tukey type FFT algorithm requires $512\times\log_2(1024) = 5{,}120$ complex multiplications and $1024\times\log_2(1024) = 10{,}240$ complex additions.

Example E11.12: Determine the 9-bit sign-magnitude, ones'-complement, and two's-complement representations of the negative decimal fractions -0.7734375_{10}.

Answer: (i) Signed-magnitude representation = $1_\Delta 11000110$

(ii) Ones'-complement representation = $1_\Delta 00111001$

(iii) Two's-complement representation = $1_\Delta 00111010$

Example E11.13: Develop the signed-digit (SD) representation of the binary number $0_\Delta 11101101$.

Answer: SD-representation = $0_\Delta 00\bar{1}10\bar{1}1\bar{1}$,

---

Additional Examples of Chapter 11: DSP Algorithm Implementation

Example E11.14: Perform the binary addition $0_\Delta 10101 + 0_\Delta 01111$.

Answer:

$$\begin{array}{cccccccc l}
 & 1 & & 1 & 1 & 1 & 1 & & \leftarrow \text{carry} \\
 & 0 & \Delta & 1 & 0 & 1 & 0 & 1 & \\
+ & 0 & \Delta & 0 & 1 & 1 & 1 & 1 & \\
\hline
 & 1 & \Delta & 0 & 0 & 1 & 0 & 0 &
\end{array}$$

As the sign bit is a 1 there has been an overflow and the sum is not correct.

Example E11.15: Evaluate the difference $0_\Delta 10101 - 0_\Delta 01111$ by performing the binary addition of a positive fraction and a negative number represented in two's-complement form.

Answer: The difference of the two positive binary fractions $0_\Delta 10101 - 0_\Delta 01111$ can be carried out as an addition of the positive binary fraction $0_\Delta 10101$ and the two's-complement representation of $-0_\Delta 01111$ which is given by $1_\Delta 10001$. The process is illustrated below:

$$\begin{array}{ccccccccc l}
 & & 1 & & & & & 1 & & \leftarrow \text{carry} \\
 & & 0 & \Delta & 1 & 0 & 1 & 0 & 1 & \\
+ & & 1 & \Delta & 1 & 0 & 0 & 0 & 1 & \\
\hline
 & 1 & 0 & \Delta & 0 & 0 & 1 & 1 & 0 & \\
 & \uparrow & & & & & & & & \\
 & \text{drop} & & & & & & & &
\end{array}$$

The extra bit 1 on the left of the sign bit is dropped resulting in $0_\Delta 00110$ which is the correct difference.
