Dr. Norbert Cheung's Series

in

Electrical Engineering

Level 4 Topic no: 14

Digital Filters Realization

Contents

1. Some background knowledge
2. FIR filters
3. Implementing a simple low pass FIR filter
4. IIR filters
5. FIR filter realization
6. IIR filter realization
7. Further reading

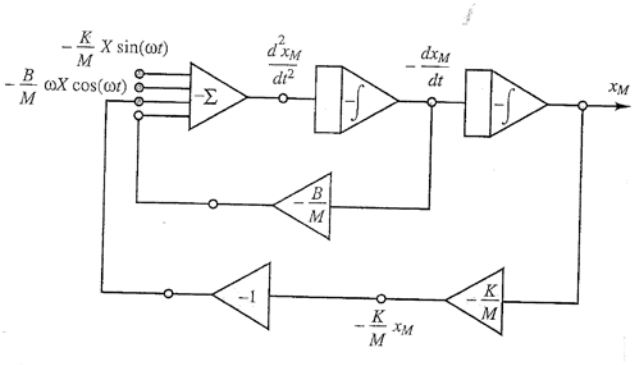
Reference:

"Modern Control Engineering", Ogata, Prentice Hall
"Simplifying Digital Signal Processing", Rajesh J Shah, Prompt Publication
"Introduction to Digital Signal Processing and Filter Design", B.A. Shenoi, Wiley

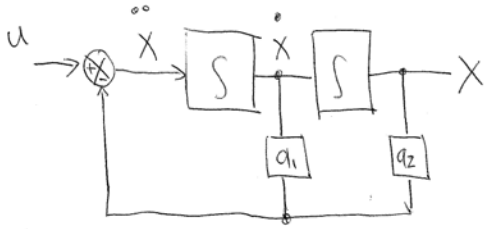
Email: norbert.cheung@polyu.edu.hk **Web Site:** www.ncheung.com

1. Some Background Knowledge

Recall the Analogue Computer case:



We redraw the circuit as simplified block diagram:

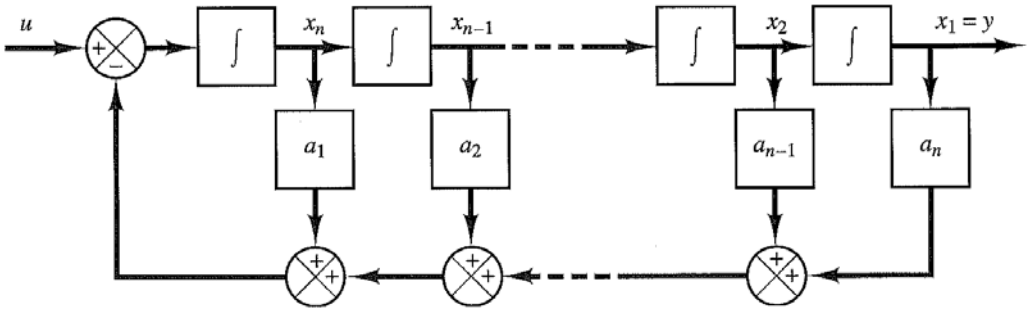


$$\ddot{X} = u - (a_1\dot{X} + a_2 X)$$

$$u = \ddot{X} + a_1\dot{X} + a_2 X$$

$$\frac{X}{u} = \frac{1}{S^2 + a_1 S + a_2}$$

This can be expressed as a general form:



$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_{n-1}\dot{y} + a_n y = u \qquad (3\text{–}34)$$

Let us define

$$x_1 = y$$
$$x_2 = \dot{y}$$
$$\vdots$$
$$x_n = y^{(n-1)}$$

Then Equation (3–34) can be written as

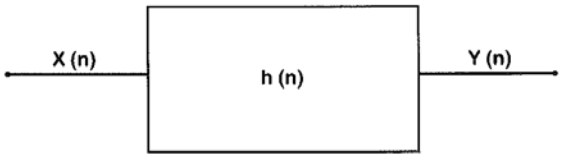
$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\vdots$$
$$\dot{x}_{n-1} = x_n$$
$$\dot{x}_n = -a_n x_1 - \cdots - a_1 x_n + u$$

Note that the state-space representation for the transfer function system

$$\frac{Y(s)}{U(s)} = \frac{1}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n}$$

2. FIR filters

The FIR filters are also called *non-recursive* filters. The recursive filters use present and past values to perform computations. The past values used are the previous inputs and outputs. Since digital filters use microprocessors, the past values are stored in the memory. Non-recursive filters use a finite number of the past input samples, and only have feedforward circuits and no feedback circuitry. The best way to understand the concepts of an FIR filter is by representing it either in a graphical or mathematical form. There are several ways an FIR filter can be represented. These differ-



An expression for the FIR filter is given as follows:

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

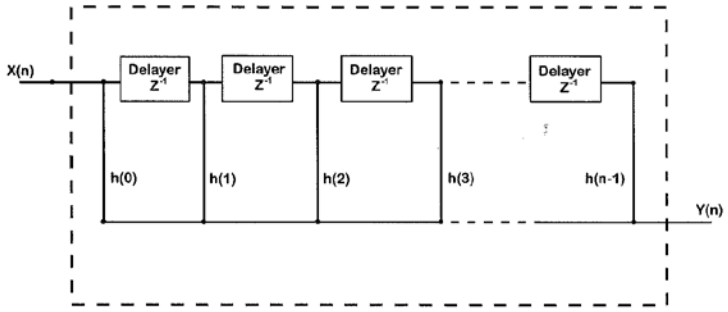
To find the transfer function, we do the following:

$$h(n) = \frac{y(n)}{x(n)}$$

Taking the z-transform of this expression, we get:

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{k=0}^{M-1} h(k)z^{-k}$$

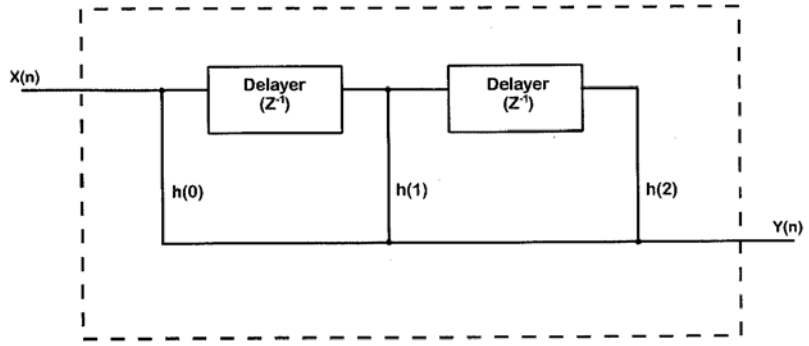
The term z^{-1} represents a delayer. To represent the FIR filter in a direct form, we use the above expressions to derive the structure as shown in *Figure 8-6*.



This is called a *direct-form* FIR filter. The above figure shows a filter of *n* length, where n is a finite number.

For example, for a filter of length 3, we write the expression as follows:

$$y(n) = \sum_{k=0}^{2} h(k)x(n-k)$$

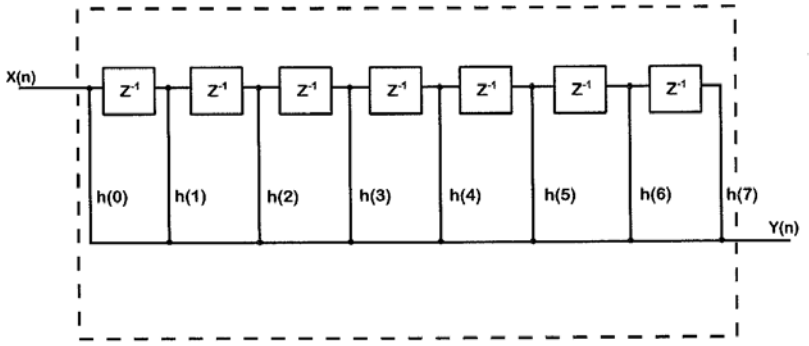


Another example:

For a filter of length 8, we write the expression as follows:

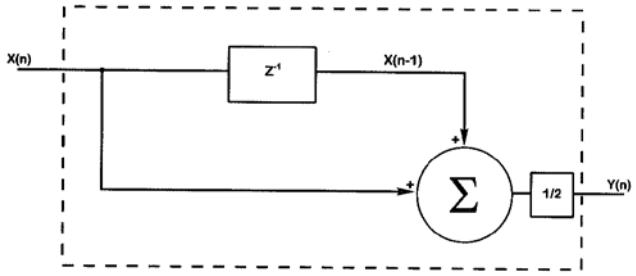
$$y(n) = \sum_{k=0}^{7} h(k)x(n-k)$$

The graphical representation of the above expression is given in *Figure 8-8.*



3. Implementing a simple Low Pass Filter

$$y(n) = \frac{x(n) + x(n-1)}{2}$$



The transfer function of the above network is given as follows:

$$H(z) = \frac{1 + z^{-1}}{2}$$

To see the frequency response, we substitute $e^{-j\omega T}$ for z. Therefore, the above expression becomes:

$$H(e^{j\omega T}) = \frac{1 + e^{-j\omega T}}{2}$$

Manipulating the above expression to rewrite it in the more recognizable form:

$$H(e^{j\omega T}) = \frac{1 + e^{-j\omega T}}{2} = e^{-j\omega T/2}\left[\frac{e^{-j\omega T/2} + e^{+j\omega T/2}}{2}\right]$$

The expression within the parentheses above can be rewritten using *Euler's Identity*, which says the following:

$$\cos(x) = \frac{e^{-jx} + e^{+jx}}{2}$$

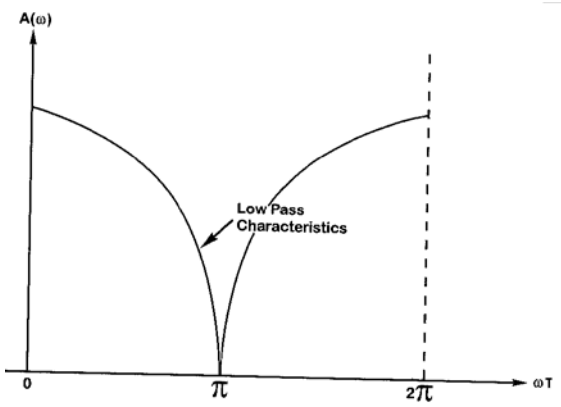
$$\sin(x) = \frac{e^{-jx} - e^{+jx}}{2j}$$

Therefore, rewriting the transfer function becomes:

$$H(e^{j\omega T}) = e^{-j\omega T/2}\cos\left(\frac{\omega T}{2}\right)$$

To find the amplitude of the above expression, we write it as follows:

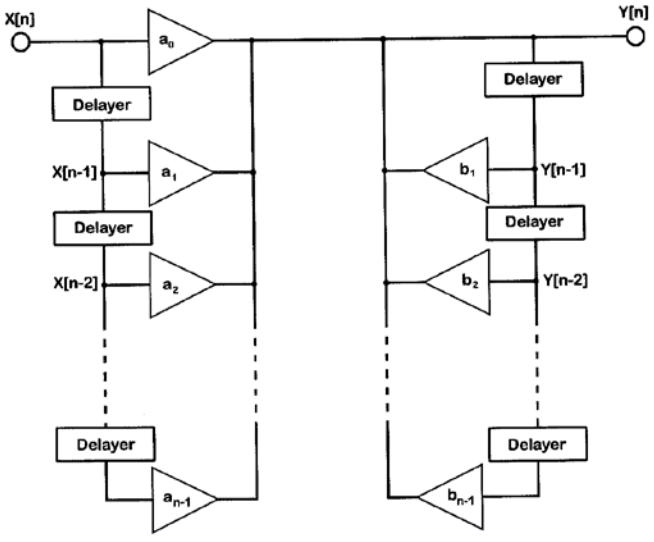
A(w) = ½cos (wT/2)½



4. IIR Filters

IIR (Infinite Impulse Response) filters are recursive filters. Unlike non-recursive filters, these filters use feedback to obtain the past values of outputs and inputs. An expression for an IIR filter is:

$$y(n) = \sum_{k=0}^{M-1} a(k)x(n-k) + \sum_{k+1}^{N-1} b(k)y(n-k)$$



An example of realizing low-pass characteristics using an IIR filter is given as follows:

$$y(n) = a\,y(n-1) + x(n)$$

$$H(z) = \frac{1}{1 - az^{-1}}$$

To find the frequency response of the above expression, we make the following substitution:

$$\text{let } z = e^{j\omega T}$$

Therefore, the above expression becomes:

$$H(e^{j\omega T}) = \frac{1}{1 - ae^{-j\omega T}}$$

When manipulating the above expression to obtain the amplitude, we get the following:

$$A(\omega) = \left[\frac{1}{\sqrt{1 - 2a\cos\omega T + a^2}}\right]^{1/2}$$

Plotting the above expression as shown in *Figure 8-13*, we see that we have realized a low-pass response using an IIR filter.

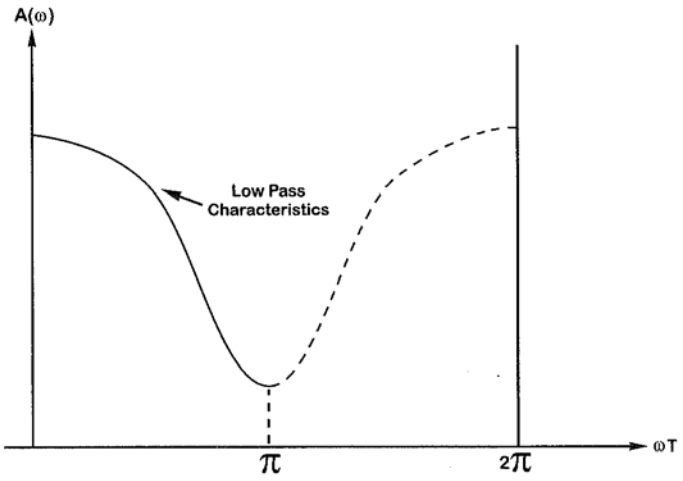


Figure 8-13. Low-pass characteristics using an IIR filter.

5. FIR filter realization

Example 6.1: Direct Form

Given the transfer function of an FIR filter as $H(z) = \sum_{n=0}^{M} h(n)z^{-n}$, let us consider its equivalent algorithm for the output, for example, when $M = 4$:

$$y(n) = h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) + h(3)x(n-3) + h(4)x(n-4) \quad (6.5)$$

We have already discussed one structure employed to implement this algorithm in Chapter 5, and because the coefficients of the multipliers in it are directly available as the coefficients $h(n)$ in $H(z)$, it is called the *direct form I structure* and is shown in Figure 6.1.

Whenever we have a structure to implement an FIR or an IIR filter, an equivalent structure can be obtained as its transpose by the following operations:

1. Interchanging the input and the output nodes
2. Replacing adders by pickoff nodes and vice versa
3. Reversing all paths

Using these operations, we get the transpose of the structure of Figure 6.1 as Figure 6.2. This is known as *direct form II structure*; remember that this (direct form II) structure will be called *direct form I transposed structure* in the next chapter.

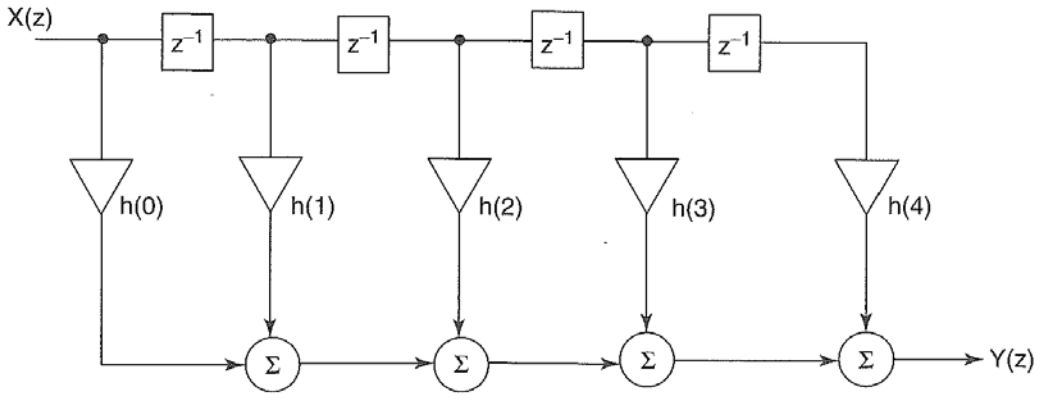


Figure 6.1 Direct form I of an FIR filter.

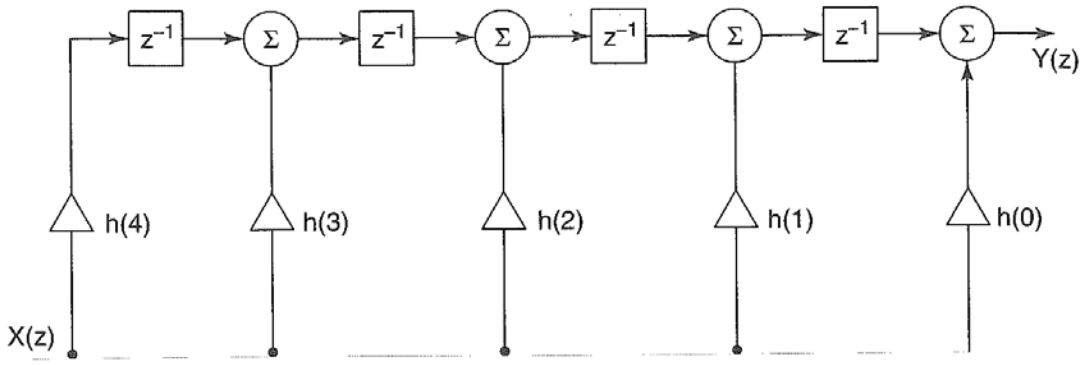


Figure 6.2 Direct form II of an FIR filter.

Example 6.6

If we consider a type II FIR filter of order 7, its transfer function is given by

$$H_4(z) = h(0) + h(1)z^{-1} + h(2)z^{-2} + h(3)z^{-3} + h(3)z^{-4} + h(2)z^{-5} + h(1)z^{-6} + h(0)z^{-7} \quad (6.14)$$

$$= h(0)(1 + z^{-7}) + h(1)(z^{-1} + z^{-6}) + h(2)(z^{-2} + z^{-5}) + h(3)(z^{-3} + z^{-4}) \quad (6.15)$$

This is realized by the canonic circuit shown in Figure 6.10, thereby reducing the total number of multipliers from 7 to 4. Similar cost saving is achieved in the realization of FIR filters with antisymmetric coefficients.

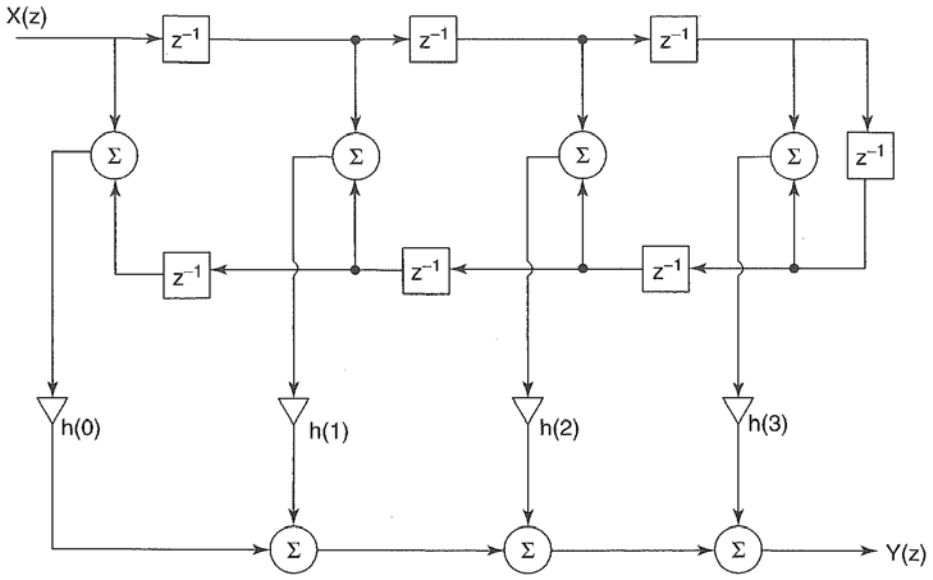


Figure 6.10 Direct-form structure of type II linear phase FIR filter function $H_4(z)$.

6. IIR filter realization

Example 6.7: Direct Forms

The transfer function (6.1) of an IIR filter is the ratio of a numerator polynomial to a denominator polynomial. First we decompose it as the product of an all-pole function $H_1(z)$ and a polynomial $H_2(z)$

$$H(z) = \frac{\sum_{n=0}^{M} b(n)z^{-n}}{1 + \sum_{n=1}^{N} a(n)z^{-n}} \tag{6.16}$$

$$= H_1(z)H_2(z) = \left[\frac{1}{1 + \sum_{n=1}^{N} a(n)z^{-n}}\right]\left[\sum_{n=0}^{M} b(n)z^{-n}\right] \tag{6.17}$$

and construct a cascade connection of an FIR filter $H_2(z)$ and the all-pole IIR filter $H_1(z)$. Again we select an example to illustrate the method. Let $H_2(z) = b_0 + b(1)z^{-1} + b(2)z^{-2} + b(3)z^{-3}$ and

$$H_1(z) = \frac{1}{1 + a(1)z^{-1} + a(2)z^{-2} + a(3)z^{-3}}$$

The realization of $H_1(z)$ in direct form I is shown in Figure 6.11 as the filter connected in cascade with the realization of the FIR filter $H_2(z)$ also in direct form I structure. The structure for the IIR filter is also called a *direct form I* because the gain constants of the multipliers are directly available from the coefficients of the transfer function.

We note that $H_1(z) = V(z)/X(z)$ and $H_2(z) = Y(z)/V(z)$. We also note that the signals at the output of the three delay elements of the filter for $H_1(z)$ are

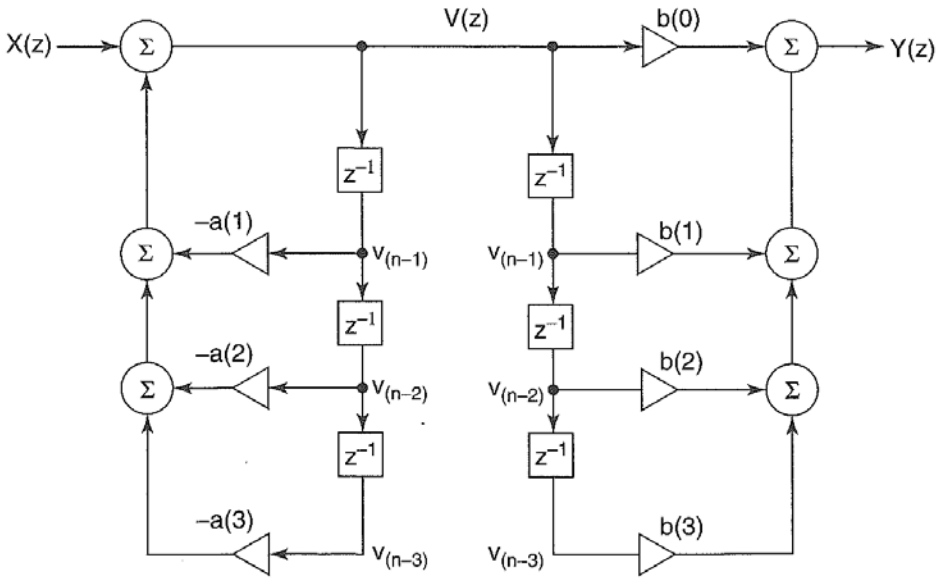


Figure 6.11 Direct form I of an IIR filter.

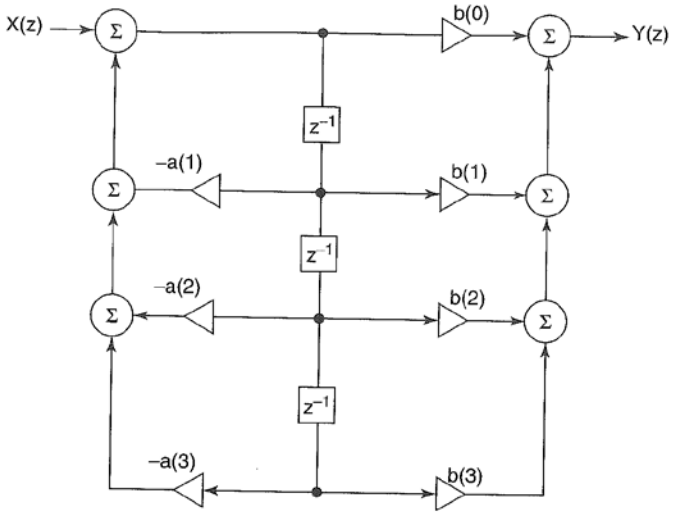


Figure 6.12 Direct form II structure of an IIR filter.

the same as those at the output of the three delay elements of filter $H_2(z)$. Hence we let the two circuits share one set of three delay elements, thereby reducing the number of delay elements. The result of merging the two circuits is shown in Figure 6.12 and is identified as the direct form II realization of the IIR filter. Its transpose is shown in Figure 6.13. Both of them use the minimum number of delay elements equal to the order of the IIR filter and hence are canonic realizations.

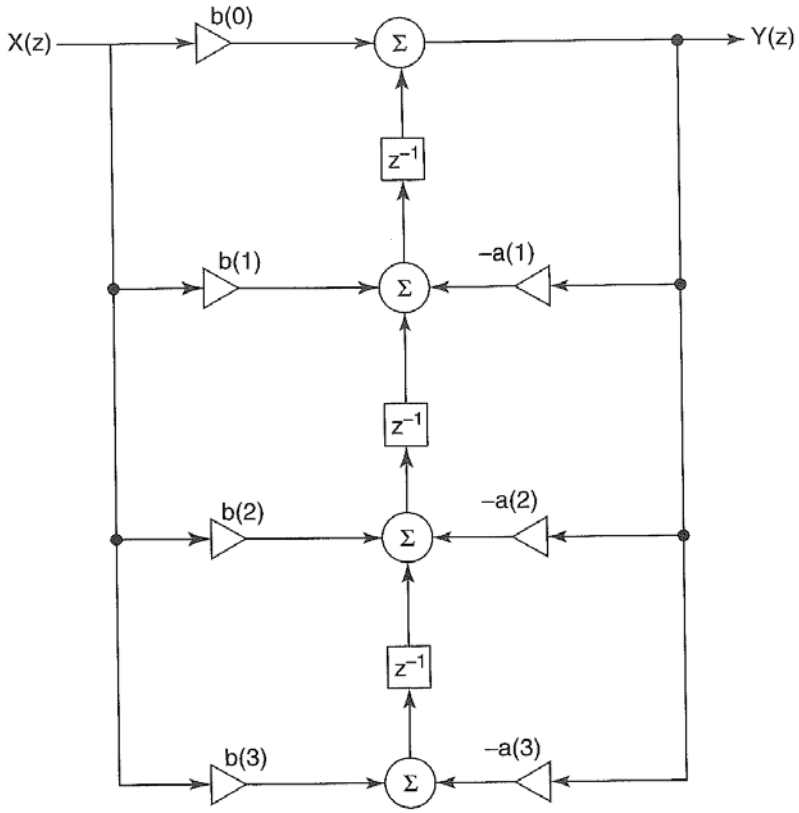


Figure 6.13 Direct form II transposed structure of an IIR filter.

7. Further Reading

Extract from a conference paper submitted to the IEEE International Conference on Multimedia Technology (ICMT), 2011. From the School of Information Engineering, Inner Mongolia University of Science & Technology, Baotou, China

Implement method of the FIR Filter

FIR digital filter can adopt two ways of software and hardware to complete.

Software implementation make used of the computer's memory, arithmetic logic unit and controller to compile the computer code, the code can accomplish the filtering operation, present some company detrude package of signal filtering used different languages, it can achieved more easily filter operation, but software implementation is slow, it is difficult to real-time signal processing and more for teaching and research.

Hardware implementation is design a dedicated digital filter, it can be completed with DSP (digital signal processing) processors, specific application signal processor (ASIC) and FPGA/CPLD completed.

1.1 Using DSP processors

The main operational unit of DSP (digital signal processing) processor is a multiply accumulator, a multiply-accumulate operations can be completed in one machine cycle, it have a unique address and reverse the cycle of addressing capability and can quickly complete the filter design, but DSP can only provide the hardware optimization to Some fixed operators, the system of DSP is still a serial instruction execution system, and optimization of these operations is not fixed to meet the needs of all algorithms.

1.2 Using specific application signal processor (ASIC) to achieve

ASIC is a hardware chip designed to dedicated signal processor or application for a fixed algorithm specially.

It is fast, small, security and high performance, but the design cycle is long, so non-high -volume applications possess a single features and expensive, flexibility of the system design is poor.

1.3 Using FPGA implementation

Programmable Logic Devices -FPGA adopt LCA (Logic Cell Array) structure, the internal structure include three parts of configurable logic block (CLB), input-output module (lOB) and. FPGA not only inherited the advantage of ASIC in the large-scale and highly integrated degree, but also overcome the shortcoming of common ASIC that design cycle is long, big investment and poor flexibility.

--- END ---