Lecture 22. *Thursday 17th December, 2020.*

0.0.1 Equivalent structures

Equivalent structures

Another important topic in digital filter design is the equivalence. Two digital filter structures are defined to be **equivalent** if they have the same transfer function.
In the following discussion we describe several methods for the generation of equivalent structures. However, we state now that a fairly simple way to do it from a given realization is via the **transpose operation**, which consists in:

Transpose operation

- reversing all the paths;
- replacing pick-off nodes by adders, and viceversa;
- interchanging the input and output nodes.

Let us consider an example of this technique. This is showed in Figure 1 along with the redrawn transposed structure.

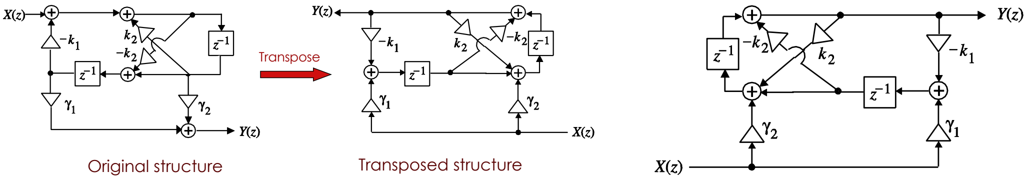

Figure 1: Example of the application of the transpose operation (left) and redrawn transposed structure (right).

We have to remark that all the other methods for developing equivalent structures are based on a specific algorithm for each structure. Moreover, there are literally an infinite number of equivalent structures realizing the same transfer function. Thus, it is impossible to develop all the equivalent realizations and in this course we will restrict our attention to a discussion of some commonly used structures.
Under infinite precision arithmetic, any given realization of a digital filter behaves identically to any other equivalent structure. However, in practice, due to the finite wordlength limitations, a specific realization behaves totally differently from its other equivalent realizations. Hence, it is important to choose a structure that has the least quantization effects when implemented. One way to arrive at such a structure is to determine a large number of equivalent structures, analyze the finite wordlength effects in each case, and select the one showing the least effects. In ceratin cases, it is possible to edvelop a structure that by construction has the least quantization effects. Here, we review some simple realizations that in many applications are quite adequate.

Equivalent structures and finite precision arithmetics

0.1 Basic FIR digital filter structures

Basic FIR digital filter structures

A **causal FIR filter of order** N is characterized by a transfer function $H(z)$ given by:

$$H(z) = \sum_{n=0}^{N} h[n] z^{-n} \tag{1}$$

which is a polynomial in z^{-1}. In the time-domain the input-output relation of the FIR filter in Eq. 1 is given by:

$$y[n] = \sum_{n=0}^{N} h[k] x[n-k] \tag{2}$$

0.1.1 Direct form FIR filter structures

Direct form FIR filter structures

It is important to remark that an FIR filter of order N is characterized by $N+1$ coefficients and, in general, requires $N+1$ multipliers and N two-input adders. Structures in which the multiplier coefficients are precisely the coefficients of the transfer function are called **direct form structures**.

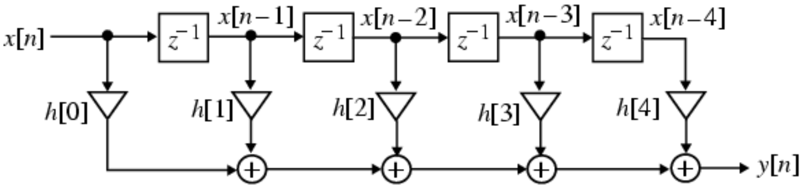

Figure 2: Example of FIR filter structure.

In particular, an analysis of the structure with $N=4$, as showed in Figure 2, yields:

$$y[n] = h[0]x[n] + h[1]x[n-1] + h[2]x[n-2] + h[3]x[n-3] + h[4]x[n-4] \tag{3}$$

Transversal filter

which is precisely of the form of the convolution sum description. The direct form structure in Figure 2 is also known as a **transversal filter**. The transpose of this direct form structure is showed in Figure 3. Both direct form structures are canonic with respect to the delays.

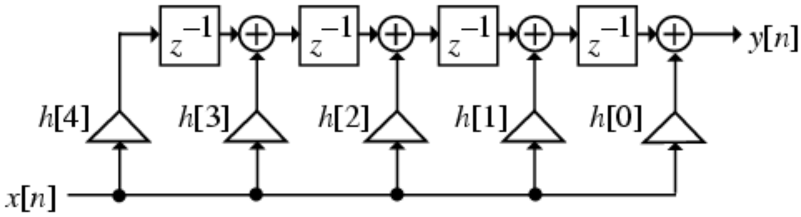

Figure 3: Transpose of the direct form structure in Figure 2.

0.1.2 Cascade form FIR filter structures

Cascade form FIR filter structures

A higher-order FIR transfer function can also be realized as a **cascade** of second-order FIR sections and possibly a first-order section. To this end, we express $H(z)$ as:

$$H(z) = h[0]\prod_{i=1}^{k}\left(1+\beta_{1,k}z^{-1}+\beta_{2,k}z^{-2}\right) \tag{4}$$

where $k=\frac{N}{2}$, if N is even, or $k=\frac{N+1}{2}$, if N is odd, with $\beta_{2k}=0$. A cascade realization for $N=6$ is showed in Figure 4. Each second-order section in the structure can also be realized in the transposed direct form.

0.1.3 Polyphase FIR structures

Polyphase decomposition and FIR structures

The **polyphase decomposition** of $H(z)$ leads to a parallel form structure. To illustrate this approach, let us consider a causal FIR transfer function $H(z)$ with $N=8$:

$$H(z) = \sum_{\ell=0}^{8} h[\ell]z^{-\ell} \tag{5}$$

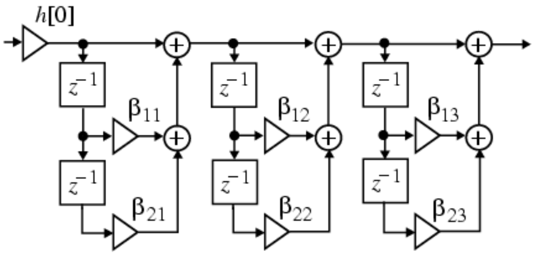

Figure 4: Example of cascade realization for $N = 6$ of an FIR filter.

$H(z)$ can be expressed as a sum of two terms, with one term containing the even-indexed coefficients and the other containing the odd-indexed ones:

$$\begin{aligned} H(z) &= \sum_{\ell=0}^{4} h[2\ell]z^{-2\ell} + \sum_{\ell=1}^{4} h[2\ell-1]z^{-(2\ell-1)} \\ &= \sum_{\ell=0}^{4} h[2\ell]z^{-2\ell} + z^{-1}\sum_{\ell=0}^{3} h[2\ell+1]z^{-2\ell} \qquad (6) \end{aligned}$$

By using the notation:

$$E_0(z) = h[0] + h[2]z^{-1} + h[4]z^{-2} + h[6]z^{-3} + h[8]z^{-4} \qquad (7)$$
$$E_1(z) = h[1] + h[3]z^{-1} + h[5]z^{-2} + h[7]z^{-3} \qquad (8)$$

we can express $H(z)$ as:

$$H(z) = E_0(z^2) + z^{-1}E_1(z^2) \qquad (9)$$

2-branch polyphase decomposition

The decomposition in Eq. 9 is more commonly known as the **2-branch polyphase decomposition**. A realization of $H(z)$ based on the latter is thus showed in Figure 5.

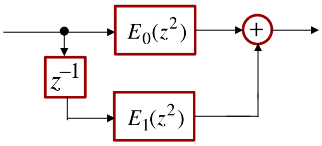

Figure 5: Example of structure of 2-branch polyphase decomposition.

In a similar manner, by grouping the terms in the original expression for $H(z)$, we can re-express it in the form:

$$H(z) = E_0(z^3) + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \qquad (10)$$

where now:

$$E_0(z) = h[0] + h[3]z^{-1} + h[6]z^{-2} \qquad (11)$$
$$E_1(z) = h[1] + h[4]z^{-1} + h[7]z^{-2} \qquad (12)$$
$$E_2(z) = h[2] + h[5]z^{-1} + h[8]z^{-2} \qquad (13)$$

3-branch polyphase decomposition

The decomposition of $H(z)$ in Eq. 10 is more commonly known as the **3-branch polyphase decomposition**, and a realization of $H(z)$ based on the latter is thus showed in Figure 6.

General case of polyphase decomposition

In the general case, an ***L*-branch polyphase decomposition** of an FIR transfer

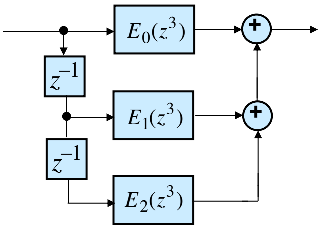

Figure 6: Example of structure of 3-branch polyphase decomposition.

function of order N is of the form:

$$H(z) = \sum_{m=0}^{L-1} z^{-m} E_m(z^L) \tag{14}$$

where:

$$E_m(z) = \sum_{n=0}^{\lfloor \frac{N+1}{L} \rfloor} h[Ln+m]z^{-m} \tag{15}$$

with $h[n] = 0$ for $n > N$.
We have to remark that the subfilters $E_m(z^L)$ in the polyphase realization of an FIR transfer function are also FIR filters and they can be realized using any methods described so far. However, to obtain a canonic realization of the overall structure, the delays in all the subfilters must be shared. For example, a canonic realization of a length-9 FIR transfer function is obtained using **delay sharing** is showed in Figure 7.

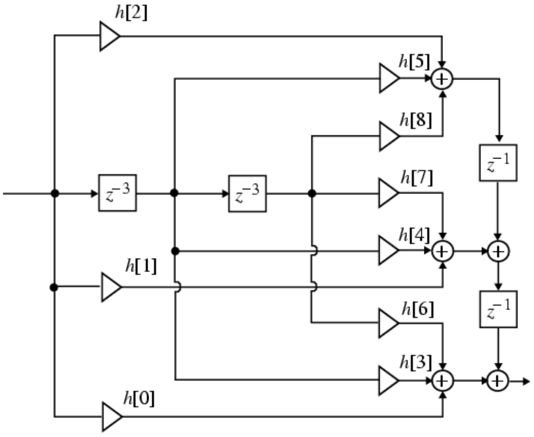

Figure 7: Example of the structure of a length-9 FIR transfer function with delay sharing.

0.1.4 Linear-phase FIR structures

Symmetry properties for structure simplification

The symmetry (or antisymmetry) property of a linear-phase FIR filter can be exploited to reduce the number of multipliers into almost half of that in the direct form implementations. For example, we consider a length-7 Type 1 FIR transfer function with a symmetric impulse response:

$$H(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[2]z^{-4} + h[1]z^{-5} + h[0]z^{-6} \tag{16}$$

If we rewrite $H(z)$ in the form:

$$H(z) = h[0]\big(1 + z^{-6}\big) + h[1]\big(z^{-1} + z^{-5}\big) + h[2]\big(z^{-2} + z^{-4}\big) + h[3]z^{-3} \tag{17}$$

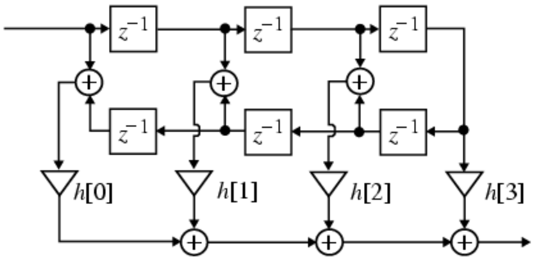

Figure 8: Example of the structure of a length-7 Type 1 FIR transfer function with a symmetric impulse response.

we obtain the realization showed in Figure 8.
A similar decomposition can be applied to a Type 2 FIR transfer function. For example, let us consider a length-8 Type 2 FIR transfer function expressed as:

$$H(z) = h[0]\left(1 + z^{-7}\right) + h[1]\left(z^{-1} + z^{-6}\right) + h[2]\left(z^{-2} + z^{-5}\right) + h[3]\left(z^{-3} + z^{-4}\right) \quad (18)$$

The corresponding realization is showed in Figure 9.

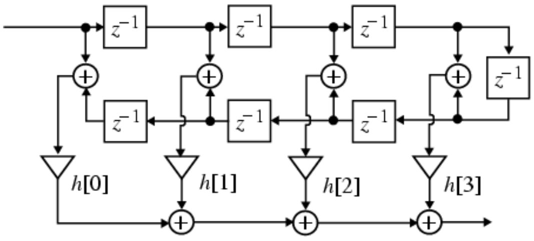

Figure 9: Example of the structure of a length-8 Type 2 FIR transfer function with a symmetric impulse response.

From the previous structures, we observe that:

- the Type 1 linear-phase structure for a length-7 FIR filter requires 4 multipliers, whereas a direct form realization requires 7 of them;
- the Type 2 linear-phase structure for a length-8 FIR filter requires 4 multipliers, whereas a direct form realization requires 8 of them.

Similar savings occur in the realization of Type 3 and 4 linear-phase FIR filters, with antisymmetric impulse responses.

0.1.5 Tapped delay line

Tapped delay line

A **tapped delay line (TDL)** is a delay line with at least one **"tap"**. The ladder extracts a signal ouput from somewhere withing the delay line, optionally scales it, and usually sums with other taps to form an output signal. A tap may be **interpolating** or **non-interpolating**. The ladder extracts the signal at some fixed integer delay relative to the input.
In this context, in some applications such as musical and sound processing, FIR filter structures of the form in Figure 10 are employed.
the structure consists of a chain of $M_1 + M_2 + M_3$ unit delays with taps at the input, at the end of the first M_1 delays, at the end of the next M_2 delays, and at the output. Signals at these taps are then multiplied by constants α_0, α_1, α_2 and α_3, then they are added to form the output. The direct form FIR structure in Figure 11 is seen to be a special case of the tapped delay line, where there is a tap after each unit delay.

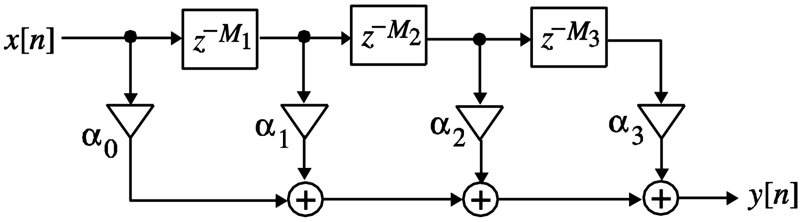

Figure 10: Example of structure of a tapped delay line.

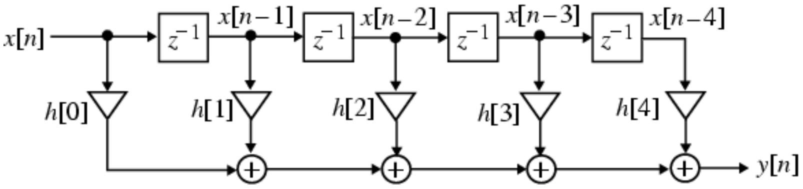

Figure 11: Special case of the tapped delay line in Figure 10, with a tap after each unit delay.