inmos

digital filtering with the IMS A100

8.1 Introduction

When an analogue signal is sampled in time, the sampled signal is referred to as a discrete-time signal. If each sample in this discrete-time signal is also quantised in amplitude, (e.g. represented by an arbitrary n-bit number), then it is usually referred to as a digital signal. In the subject of digital filtering it is these types of signals which are processed and operated on. The fact that the digital signals are quantised both in time and amplitude gives one greater control over the processing as compared to analogue signal processing.

In these application notes the concept of the digital filtering is first introduced. This is done by starting from a simple RC analogue filter and deriving a corresponding digital filter. The classification of digital filters is then summarized, followed by giving a summary of techniques applicable to filter design using the IMS A100 device.

8.2 From analogue to digital

Figure 8.1a shows a simple first-order RC filter. The simple differential equation describing this circuit in terms of its input and output voltages is:

$$v_0(t) + RC\frac{dv_0(t)}{dt} = v_i(t) \tag{1}$$

where $v_0(t)$ and $v_i(t)$ are analogue output and input voltage waveforms. In the analogue world both input and output voltages are continuous-time waveforms and the complexity of the solution would depend on the input voltage function $v_i(t)$. Given an input waveform $v_i(t)$, the solution can be obtained using:

(i) Standard mathematical techniques which solve the differential equation and obtain the output waveform in closed form.

(ii) Numerical techniques which calculate the approximate output waveform in a digital computer. This would necessitate the sampling of the input and output waveforms.

The second method above provides the basis for digital filtering techniques. Consider that the input and the output voltages are sampled with a sampling interval T such that $v_i(nT)$ and $v_o(nT)$ represent the values of $v_i(t)$ and $v_o(t)$ at time $t = nT$.

If T is sufficiently small then the derivative $\frac{dv_o(t)}{dt}$ at time $t = NT$ can be approximated by:

$$\frac{dv_o(nT)}{dt} \simeq \frac{v_o(nT) - v_o((n-1)T)}{T} \tag{2}$$

substituting this in equation (1) we obtain:

$$v_o(nT) + \frac{RC}{T}v_o(nT) - \frac{RC}{T}v_o((n-1)T) = v_i(nT) \tag{3a}$$

Equation (3a) is a linear difference equation that approximates the differential equation (1). Equation (3a) can be rewritten as:

$$v_o(nT) = \frac{1}{1+(RC/T)}v_i(nT) + \frac{(RC/T)}{1+(RC/T)}v_o((n-1)T) \tag{3b}$$

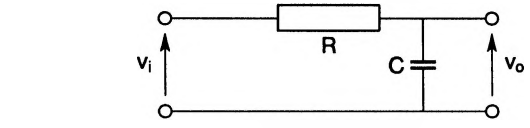
This is now a recursion formula in which the present input sample and the previous output sample are used to calculate the present output sample. The notation can be simplified to:

$$v_o(n) = b_0 v_i(n) + a_1 v_o(n) \tag{4a}$$

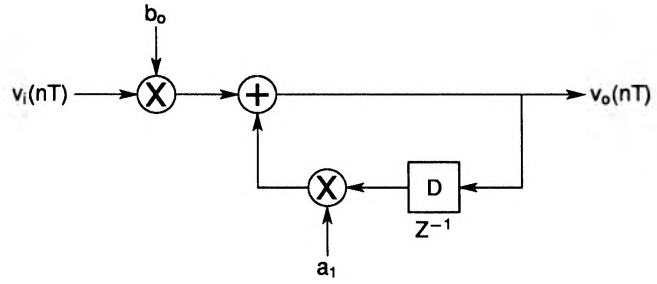
where $b_0 = \frac{1}{1+(RC/T)}$ and $a_1 = \frac{(RC/T)}{1+(RC/T)}$.

The signal-flow diagram for this filter is shown in figure 8.1b. The block labelled 'D' represents a delay equal to one sampling period T. In digital filter notations a delay of n sampling periods is usually denoted by z^{-n}. Therefore a delay of one sampling period can be represented by z^{-1}.

It is important to note that a common element in all filter structures is the concept of storage. In the analogue RC filter (figure 8.1a) the storage is present in the form of a capacitor and in its digital equivalent (figure 8.1b) the storage takes the form of a delay stage. In fact the storage element is the essential ingredient for any filter whether analogue or a digital. This is because filters are used to operate on the signal 'changes' and as such they need to have some knowledge of the history of the signal to allow them to perform their function.



(a) Analogue RC filter



(b) Discrete-time version of (a)

Figure 8.1 Analogue RC filter and its discrete-time equivalent

An important characteristic feature of any filters is its so called 'impulse response'. This is defined as the output waveform of the filter when a unity impulse is applied to the input. Using equation (4a) and assuming a unity impulse as the input waveform i.e.

$$\begin{aligned} v_i(0) &= 1 \\ v_i(n) &= 0 \quad for \quad n > 0 \end{aligned}$$

then the output sequence would be:

$$b_0,\ a_1 b_0,\ a_1^2 b_0, \ldots\ldots,\ a_1^n b_0, \ldots\ldots\ldots$$

or in short

$$v_0(n) = a_1^n b_0$$

It should be noted that the above impulse response has, in theory, infinite length. This is due to the recursive nature of this particular filter structure. This types of filters are often referred to as infinite-impulse-response (IIR) filters.

An alternative way of looking at the filter in this example is to use equation (4a) in successive substitutions i.e.

$$\begin{aligned} v_o(n) &= b_0 v_i(n) + a_1 v_o(n-1) \\ &= b_0 v_i(n) + a_1[b_0 v_i(n-1) + a_1 v_o(n-2)] \\ &= b_0 v_i(n) + a_1 b_0 v_i(n-1) + a_1^2[b_0 v_i(n-2) + a_1 v_o(n-3)] \\ &= \ldots\ldots \\ &= \ldots\ldots \\ &= b_0 v_i(n) + a_1 b_0 v_i(n-1) + a_1^2 b_0 v_i(n-2) + a_1^3 b_0 v_i(n-3) + \ldots\ldots \end{aligned} \qquad (4b)$$

Equation (4b) expresses the output waveform as a linear combination of input samples only, but this involves infinite number of input samples. Notice also that the coefficients b_0 and a_1 have positive values less than unity (R and C are assumed to be finite and non-zero). This means that in equation (4b) the coefficients decrease for older input samples. It may therefore be reasonable to assume that these coefficients approximate to zero beyond a certain point. In this way only a finite number of terms would be involved in equation (4b), or in other words, the infinite impulse response is approximated by a finite impulse response since it decays rapidly to zero. This modified filter with its finite duration impulse response falls in the category of FIR (Finite-Impulse Response) filters. In the next section these concepts are generalized.

8.3 Digital filter classifications

Linear difference equations, similar to equation (4a & 4b) are the basis for the theory of digital filters. The general difference equation can be expressed as:

$$y(n) + \sum_{m=1}^{M} a_m y(n-m) = \sum_{k=0}^{N} b_k x(n-k) \tag{5}$$

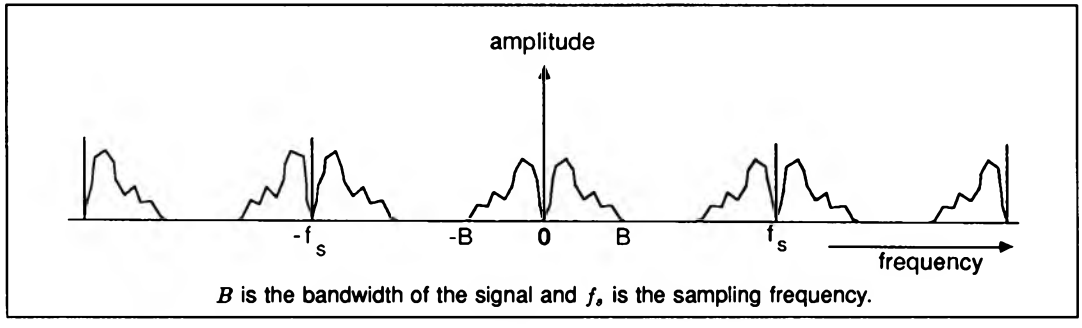
Where the x and y sequences are the input and the output of the filter and a_m's and b_k's are the coefficients of the filter.

As mentioned earlier the notation z^{-1} is often used to denote a delay equal to one sampling period. In the theory of the dicrete-time signals, the concept of z has been developed further and is referred to as the z-transform. This is a discrete-time version of the well known Laplace transform (sometimes referred to as the s-transform) which is mainly used for dealing with continuous signals. In the s-domain a delay of T seconds corresponds to e^{-sT}. Therefore the two variables s and z are related by:

$$z^{-1} = e^{-sT} \tag{6}$$

where T is the sampling period.

In the s-domain the spectrum of a signal with a bandwidth B and sampled at a frequency f_s, is periodic with a period equal to f_s. This is depicted in figure 8.2. This periodicity in the spectrum of a sampled signal is the basic reason behind the Nyquist criterion which requires a minimum sampling frequency of twice the signal bandwidth (i.e. $f_{s_{min}} = 2 \times B$), in order to avoid aliasing effects.



B is the bandwidth of the signal and f_s is the sampling frequency.

Figure 8.2 Spectrum of a sampled signal

Equation (6) allows a mapping between the two domains. Part of the imaginary axis between $-\frac{f_s}{2}$ to $+\frac{f_s}{2}$, in the s-plane, is mapped into a unit circle in the z-domain as shown in figure 8.3. The fact that the imaginary axis in the s-plane is mapped onto a circle is a consequence of the periodic nature of the spectrum. As shown in figure 8.3, the left-hand half of the s-plane (between $-\frac{f_s}{2}$ and $+\frac{f_s}{2}$) is mapped onto the inside of the unit circle, while the right-hand half is mapped onto the outside of the circle.

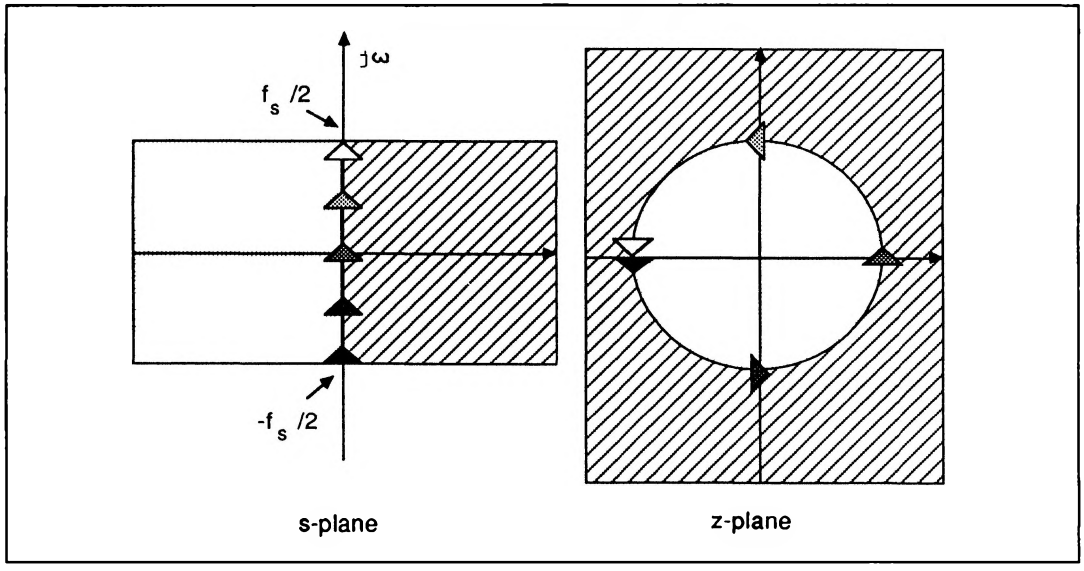


Figure 8.3 Relationship between the s-domain and the z-domain

As in the analogue design (s-domain) where a pole in the wrong place, i.e. in the right-half plane, indicates instability, in the case of discrete-time signals (z-domain) a pole outside the unit circle causes instabilities. In both cases zeroes can be anywhere.

Using the z-transform notation, the general linear equation (5) can be expressed as:

$$Y(z)(1 + \sum_{m=1}^{M} a_m z^{-m}) = X(z) \sum_{k=o}^{N} b_k z^{-k} \tag{7}$$

Where $X(z)$ and $Y(z)$ are the z-transforms of the input and output waveforms. The discrete-time (or digital) transfer function of the general filter is thus given by:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} b_k z^{-k}}{1 + \sum_{m=1}^{M} a_m z^{-m}} \tag{8}$$

In terms of realization, digital filters are classified into nonrecursive and recursive types. The nonrecursive structure contains only feed-forward paths and as such all the a_m terms (equation (8)) are zero. This means that for the nonrecursive filters the output is a sum of linearly weighted present and a number of past samples of the input signal as shown in figure 8.4. Referring to equation (8), for the nonrecursive filters the transfer function has only zeroes and as such is always stable.

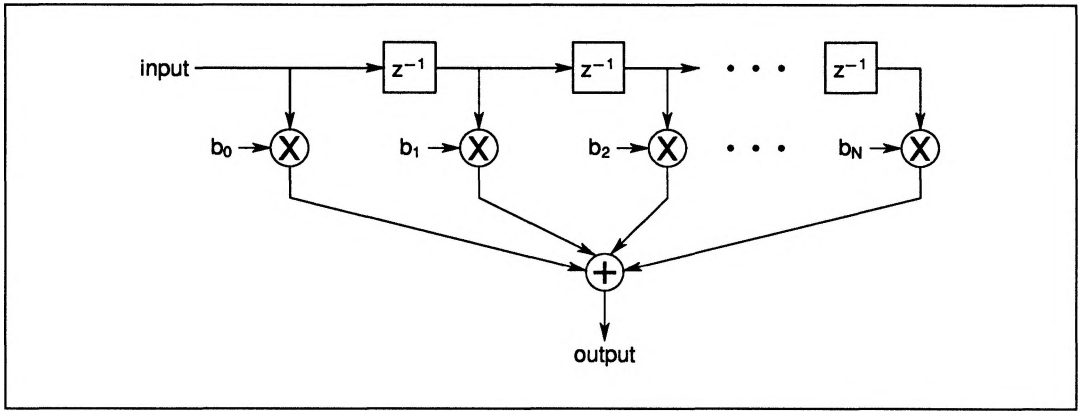


Figure 8.4 Nonrecursive digital filter structure

In the recursive filters on the other hand some or all of the a_m terms are non-zero resulting in the presence of both poles and zeroes in the transfer function. Figure 8.5 shows the general recursive filter structure. Figure 8.6 shows an alternative structure for the same transfer function with a reduced number of delay stages.

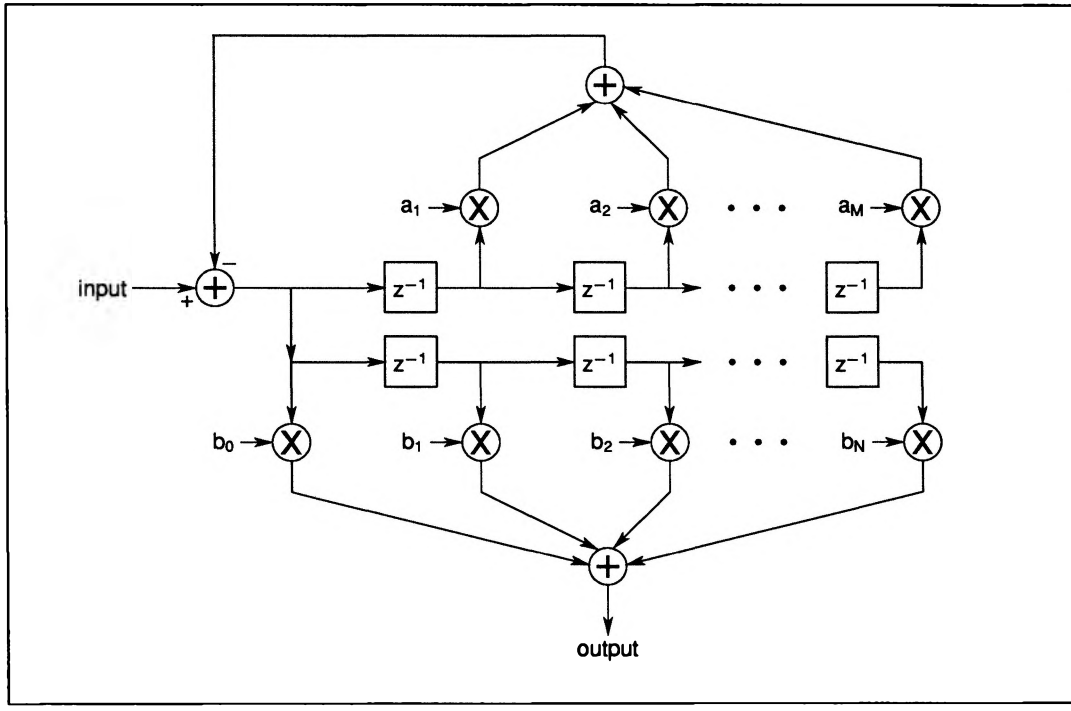


Figure 8.5 Recursive (IIR) digital filter structure

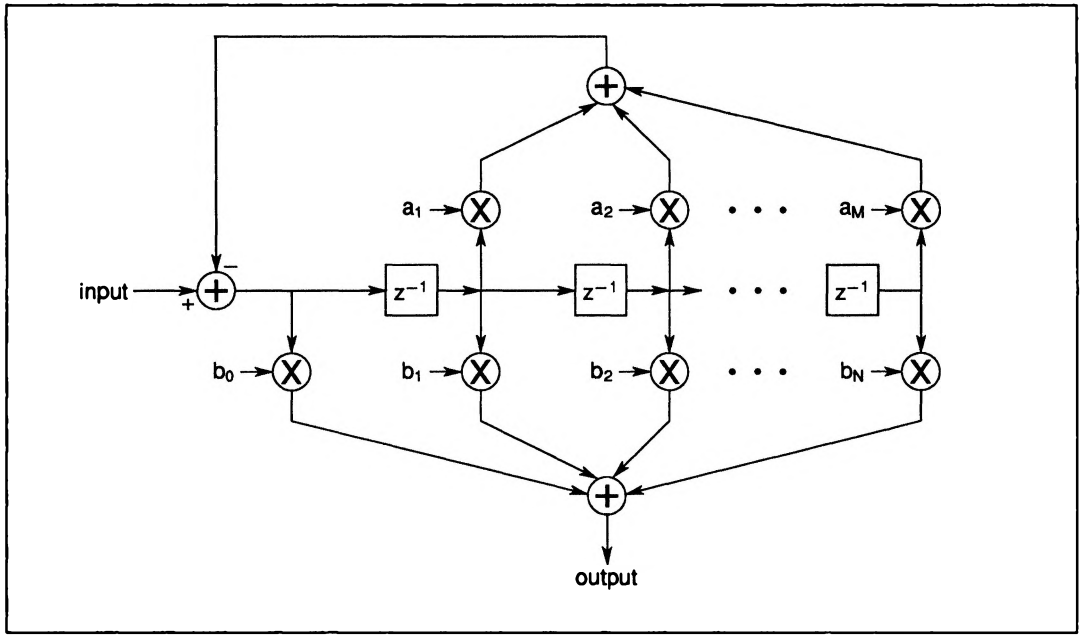


Figure 8.6 Alternative recursive (IIR) digital filter structure with reduced number of delay stages

Digital filters are also classified in terms of their impulse responses. In this classification those filters with a finite duration impulse response are referred to as FIR filters and those with an infinite duration impulse response are called IIR filters. The simplest FIR filter realization is in the nonrecursive form. For example in figure 8.4, if a unit impulse is clocked through the filter, the sequence,

$$b_0, b_1, b_2, ...b_N, 0, 0, 0, 0, 0, ...0, 0, 0 \tag{9}$$

will be output. Notice that the response consists of a sequence of samples corresponding to the filter coefficients followed by zeroes, i.e. the nonrecursive structure is an FIR filter. On the other hand the impulse response of the recursive structure (figures 8.5 & 8.6), because of the feedback paths, is infinite in duration, making the configuration an IIR filter.

8.4 Digital filter design

Digital filter design methods can be divided into two categories:

(a) Design techniques suitable for FIR filters.

(b) Design techniques suitable for IIR filters.

In both cases the requirement is simply the choice of filter coefficients in such a way that the specification for the required transfer function is met. The IMS A100 can be used to implement high performance FIR filters directly. It can also be used to implement IIR filters, although the general problems associated with IIR filter design are then introduced. In this section a brief comparison between FIR and IIR filters is given and some of their associated design techniques are summarized. Where necessary the IMS A100 implementation issues are also discussed.

8.4.1 Comparison between FIR and IIR filters

FIR filters, because of their finite-impulse response have no counterparts among analogue filters and as such can implement transfer functions which cannot be realized in the analogue world. One such property

is the excellent linear-phase characteristic which can easily be realized with FIR filters. Since a linear-phase response corresponds to only a fixed delay, attention can be focussed on approximating the desired magnitude response without concern for the phase. The design techniques for FIR filters are generally simpler than those for IIR filters, and as there are no feedback paths in an FIR filter, the stability of the filter is guaranteed. Also FIR filters have been employed, and algorithms have been developed, for adaptive processing while the use of IIR filters in these types of systems is not common.

IIR filters on the other hand have infinite impulse responses and thus their design can be closely related to analogue filter design. IIR filters in general require fewer stages compared to FIR filters but their stability is not unconditional and great care should be taken to insure stability. Furthermore IIR filters do not generally result in linear-phase characteristics which is important in many applications.

8.4.2 Basic design parameters

In digital filter design, for the reason of convenience, the frequency axis is usually normalised with respect to the sampling frequency f_s. For example for a filter with an actual pass-band cut-off frequency of $20kHz$, a stop-band cut-off frequency of $30kHz$ and a sampling frequency of $100kHz$ we have:

The normalised pass-band cut-off frequency $f_{pb} = \frac{20}{100} = 0.2$

The normalised stop-band cut-off frequency $f_{sb} = \frac{30}{100} = 0.3$

As shown in figure 8.7 the useful frequency axis (normalised) extends from 0.0 to 0.5, because the Nyquist sampling theorem requires a signal to be sampled at more than twice its highest frequency. This means that the ratio of the frequency of any component in the signal to the sampling frequency must always be less than 0.5.

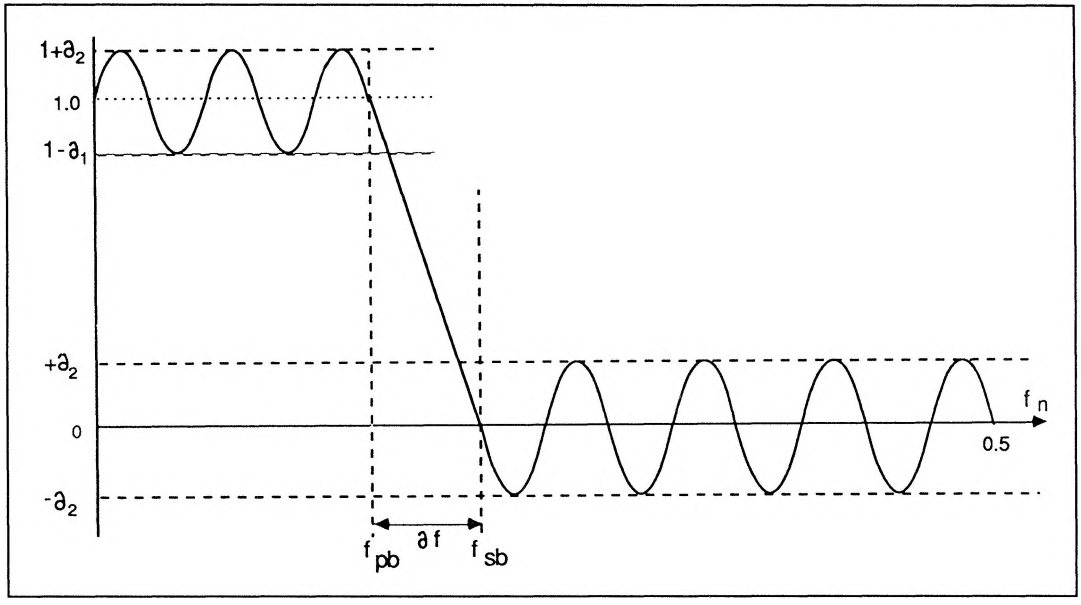


Figure 8.7 Specification parameters for a low pass filter.
Similar parameters exist for high pass and band pass filters.

Referring to figure 8.7, the pass-band and the stop-band ripples are usually expressed in dBs i.e:

pass-band ripple $(dB) = 20\log_{10}(1+\delta_1)$

stop-band ripple $(dB) = -20\log_{10}(\delta_2)$.

The parameters f_{pb}, f_{sb}, δ_1, δ_2 and the sampling frequency define the basic specification of a filter prior to its design.

8.4.3 Design techniques suitable for FIR filters

As mentioned earlier one of the major advantages of FIR filters is the ease with which linear-phase behaviour can be obtained from these types of filters. Before summarizing the design techniques for FIR filters let us briefly consider the necessary conditions for linear-phase behaviour. It can readily be shown that in order to obtain an FIR filter with a linear-phase characteristic, the following condition has to be met (references 1 & 2):

$$\begin{aligned} h(i) &= \pm h(N-i) \quad for \quad 0 \le i \le N \\ &= 0 \quad otherwise \end{aligned} \tag{10}$$

This condition requires that the the impulse response of the FIR filter, $h(i)$, to have either positive or negative symmetry.

In the case of positive symmetry the frequency response will be of the form

$$H(e^{j\omega T}) = A(\omega T)e^{-j\omega TN/2} \tag{11a}$$

where $A(\omega T)$ is a real function of ω. Notice that the phase is a linear function of frequency. These types of filters are appropriate for frequency selective filters.

In the case of negative symmetry the filter transfer function will have the following form:

$$H(e^{j\omega T}) = jB(\omega T)e^{-j\omega TN/2} \tag{11b}$$

Again $B(\omega T)$ is a real function of ω. Note that the phase is again linear with frequency, but we also have a j term which indicates an extra phase shift of $\frac{\pi}{2}$. These types of frequency responses are required to realise approximate differentiators and Hilbert transforms which implement a $\frac{\pi}{2}$ phase shift over a specified frequency range.

There are essentially three well-established classes of design methods for (linear phase) FIR filters which are:

(i) window method

(ii) frequency sampling

(iii) optimal design (Remez Exchange Algorithm)

Each one of these techniques has its own merits and the choice of which would depend on the application requirements and the design time involved.

Window method

This is the most straight-forward approach to the design of FIR filters. In this method having defined an ideal frequency-response function, the corresponding ideal impulse response is determined by evaluating the inverse Fourier transform of the ideal frequency response. In the selection of the ideal frequency response, the linear phase condition may or may not be applied depending on the application.

As mentioned earlier because digital filters deal with signals sampled at a frequency f_s,it therefore follows that this frequency response is periodic in frequency with a period equal to f_s (Nyquist theorm). It is therefore possible to relate the impulse response and the frequency response of a digital filter via the following Fourier pairs:

$$H(\omega) = \sum_{n=-\infty}^{+\infty} h(n)e^{-jn\omega T} \tag{12}$$

$$h(n) = \frac{1}{\omega_s} \int_{-\frac{\omega_s}{2}}^{\frac{+\omega_s}{2}} H(\omega)e^{jn\omega T} d\omega \tag{13}$$

where ω_s, is the sampling frequency in radians/s and T is the sampling period. Having defined an ideal frequency response, $H(\omega)$, equation (13) can be used to obtain the impulse response, $h(n)$, of the filter. As an example consider the ideal low-pass frequency response characteristics with a cut-off frequency ω_c as shown in figure 8.8a. Using equation (13), and equating $H(\omega)$ to 1.0 for $-\omega_c \leq \omega \leq +\omega_c$ and to zero elsewhere, we can calculate the impulse response $h(n)$ which is given by:

$$h(n) = \frac{\omega_c T}{\pi} \frac{\sin(n\omega_c T)}{(n\omega_c T)} \tag{14}$$

where $-\infty < n < +\infty$. This impulse response is shown in figure 8.8b. There are two problems associated with this impulse response obtained in this way:

(i) The filter impulse response is infinite in duration and as such an FIR filter of infinite length is required (remember as discussed earlier for FIR filter the impulse response sample values are effectively the filter coefficients).

(ii) The filter is unrealizable since the impulse response begins at $-\infty$, indicating that no finite amount of delay can make the impulse response realizable.

One way to obtain an FIR filter which approximates the required frequency response is to truncate the infinite impulse response at $n = \pm\frac{N}{2}$, (see figure 8.8c), and shift the impulse response to the right to avoid negative time (figure 8.8d). This would result in a realizable FIR filter with $N + 1$ coefficients which are equal to the impulse response samples.

The problem with this direct truncation of the impulse response is that it results in a fixed amount of overshoot (approximately 9%) before and after the discontinuity in the frequency response. In the literature this problem is referred to as the Gibbs phenomenon. For this reason, direct truncation is not often a reasonable way of designing FIR filters.

The frequency response of a truncated time series can be improved considerably by using a window function, $w(n)$, which modifies the impulse response to $w(n) \times h(n)$. In the previous example the window was simply a rectangular window. Figure 8.9 shows the application of a different window function to the example of the ideal low-pass filter. Figure 8.9a shows the ideal infinite duration impulse response. Figure 8.9b shows the window function and figure 8.9c shows the impulse response after the application of the window function. Figure 8.9d shows the shifted impulse response which avoids unrealizable negative delays. The filter coefficients (b_k's) correspond to the sample values of this modified impulse response which is now finite and realizable. Several window functions have been suggested in the literature some of which are:

(i) Hamming window

(ii) Hanning window

(iii) Kaiser window

(iv) Dolph-Chebyshev window

(v) Blackman window

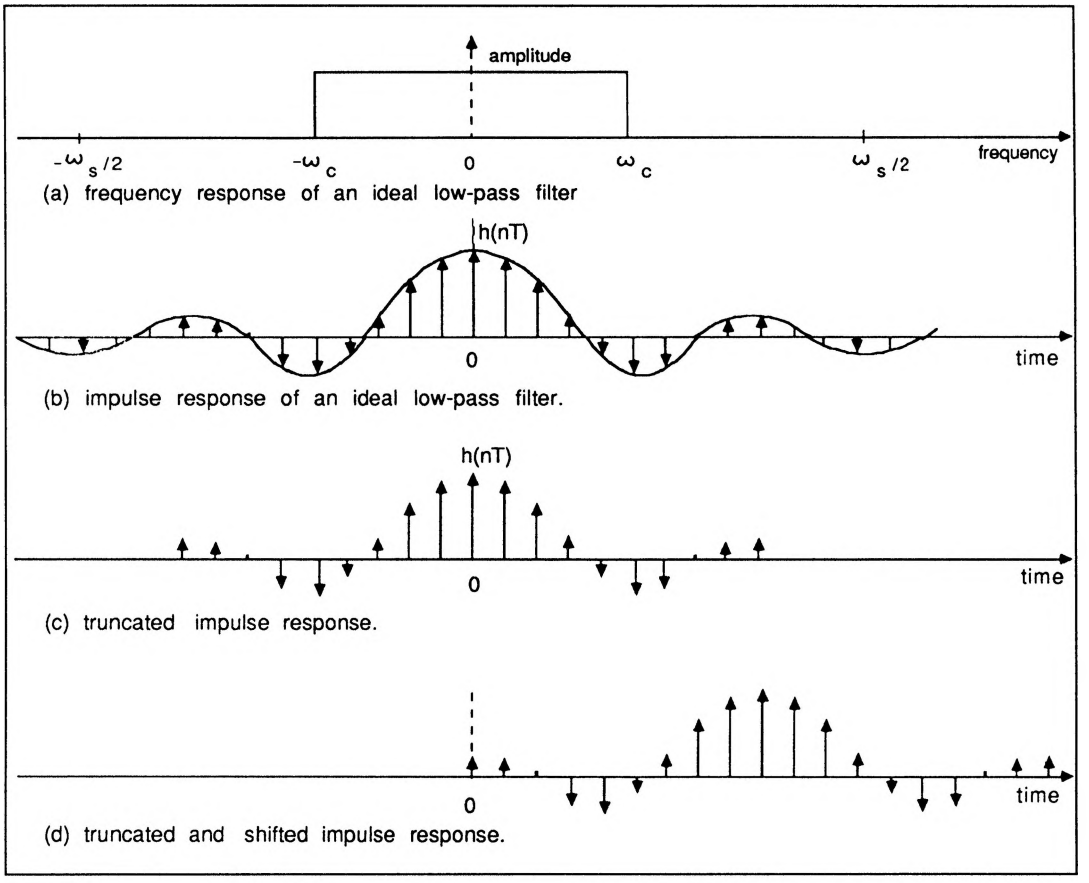


Figure 8.8

The generalized Hamming window function is given by:

$$\begin{aligned} w_H(n) &= \alpha + (1-\alpha)\cos\left(\tfrac{2\pi n}{N}\right) \quad for \quad -\left(\tfrac{N-1}{2}\right) \le n \le -\left(\tfrac{N-1}{2}\right) \\ &= 0 \qquad\qquad otherwise \end{aligned} \tag{15}$$

where $0 \le \alpha \le 1$. If $\alpha = 0.54$ the window is called a Hamming window, and if $\alpha = 0.50$ it is called a Hanning window.

For the Hamming window the main lobe of the frequency response is twice the width of that of the simple rectangular window. The amplitudes of the ripples of the Hamming window frequency response are considerably smaller than those of the rectangular window. For the rectangular window the peak side lobe (in the stop band) is only 14dB below the main-lobe (pass-band) peak. For the Hamming window the peak side lobe ripple is about 40dB below the pass band peak. Furthermore for the Hamming window 99.96% of the spectral energy is in the main-lobe peak.

Another family of windows are those proposed by Kaiser:

$$\begin{aligned} W_K(n) &= \frac{I_0(\beta\sqrt{1-[2n/(N-1)]^2})}{I_0(\beta)} \quad for \quad -\left(\tfrac{N-1}{2}\right) \le n \le -\left(\tfrac{N-1}{2}\right) \\ &= 0 \qquad\qquad otherwise \end{aligned} \tag{16}$$

Where I_0 is the modified Bessel function of the first kind. The parameters β is used to specify the main-lobe width and the side-lobe level of the frequency response. β is usually specified to have a value between 4

and 9. This range of β corresponds to a range of side-lobe peaks of 3.1% to 0.047% of the main-lobe peak. The Kaiser window is essentially an optimum window in the sense that it is a finite duration sequence that has the minimum spectral energy beyond some specified frequency. For the Kaiser window the width of main lobe is almost three times that of the rectangular window, while the peak side lobe in the stop band is 57dB below the pass-band peak. The side-lobe ripple envelope decays to 94dB below the pass-band peak at half the sampling frequency.

The Dolph-Chebyshev window function has the minimum width of the main lobe in its frequency response for a given peak value of side-lobe ripple. For this window the stop-band ripples all have the same amptitude. Recursive equations exist which allow this window function to be evaluated.

References 1 and 2 contain further information on this design method and the associated window functions.

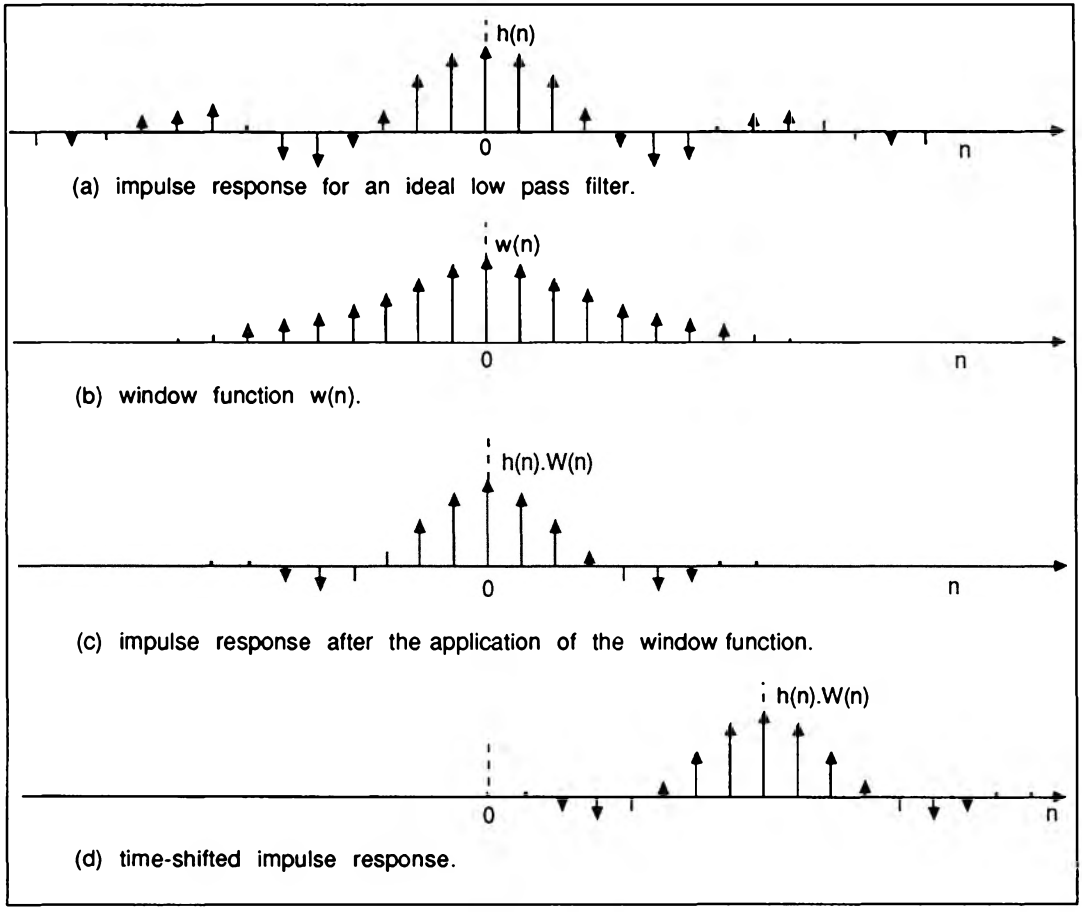


Figure 8.9

Frequency sampling technique

This technique is less common than the other two design methods, however for the sake of completeness it is briefly mentioned here.

The basic idea behind this technique is that the given (desired) frequency response is approximated by sampling it at N equally-spaced points along the frequency axis between 0 and f_s (corresponding to N samples on the unit circle in the z-plane). An N-point inverse DFT is then performed on these N frequency

samples to give N samples of the impulse response $h(n)$ which corresponds to the filter coefficients. The z-transform of the filter impulse response is then given by

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

Substituting $e^{j\omega T}$ for z, the resulting frequency response of the filter may be evaluated which would be an approximation of the desired frequency response. The approximation error would be exactly zero at points where the desired frequency response was sampled and would be finite between them. This process is depicted in figure 8.10.

To reduce these approximation errors a number of frequency samples (particularly those in the transition band between band-pass and band-stop regions, i.e. points T_1, T_2, T_3 and T_4 in figure 8.10) can be made unconstrained variables. The values of these unconstrained variables are then optimised using computer optimisation techniques involving linear-programming methods. This involves the solution of a set of linear unequalities in the unconstrained frequency samples. In this way, by adjusting the frequency sample values at T_1, T_2, T_3 and T_4, considerable ripple cancellation, both in the pass-band and stop-band, can be achieved resulting in very good filter characteristics. The detail of these techniques are beyond the objectives of this application note, however interested readers can refer to reference 1 for further information.

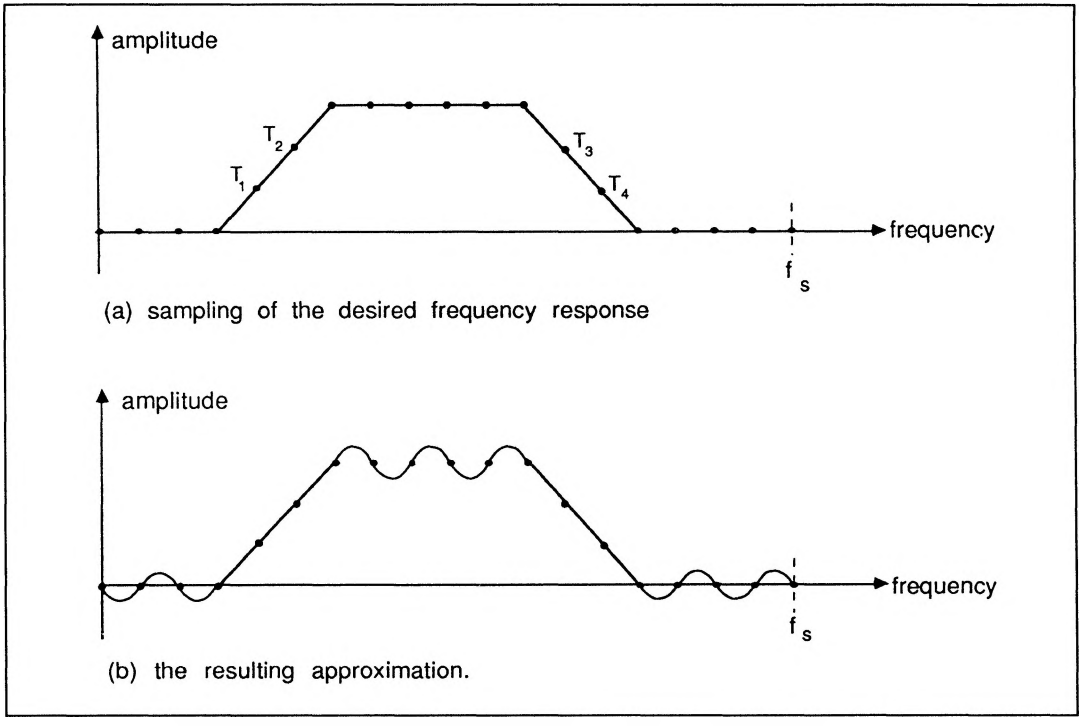


Figure 8.10

Optimal filter design – (Remez exchange algorithm)

In the frequency sampling technique, discussed in the previous section, some degree of improvement in the filter characteristics is obtained by allowing only a few of the frequency samples to be adjusted via a linear-programming technique.

An even more powerful technique which results in truly optimal filters, in the sense of having the sharpest transition between pass bands and stop bands (for a given filter length and a given approximation error) has been formulated based on the so-called Chebyshev approximations. Computer optimisation techniques based on linear programming have been developed (references 3, 4, 5 & 6) which allowed engineers to design optimal FIR filters with a minimum amount of knowledge about the actual optimisation algorithm. These iterative algorithms are based upon the principles of the Remez exchange algorithm. This algorithm yields optimal filters that satisfy the so-called minimax error criterion (reference 1), where for a given number of coefficients, the filter minimizes the maximum ripple amplitude in the pass band. The implications of this optimal design are:

(a) The Remez exchange algorithm results in an FIR filter with the smallest number of coefficients satisfying the required specification.

(b) The pass-band ripple components all have the same magnitude and need not be equal to the stop-band ripples, but their ratio must be specified.

The input to the Remez exchange program usually includes the type of filter (frequency selective filters, differentiators and Hilbert transform filters), normalised stop-band and pass-band edges, the desired minimum stop-band attenuations, the maximum pass-band ripple and the ratio of the pass-band to stop-band ripples.

The output of the program include estimated filter length, and impulse response (filter coefficients). It also includes first pass computed values for design parameters, such as pass-band ripple, stop-band attenuation. If the computed values do not satisfy the design requirements, the filter length may be increased slightly and the program is run again. Interested readers can find copies of this program in references 1, 2 & 4.

Implementing FIR filters with the IMS A100

The coefficient word size in the IMS A100 can be programmed to be 4, 8, 12 or 16 bits. Having calculated the filter coefficients using one of the techniques described earlier, these coefficients are then expressed in a 4, 8, 12 or 16-bit format, depending on the required accuracy. The filter can then be implemented by simply loading these coefficients into the IMS A100 coefficient memories. If the number of coefficients (filter stages) required is less than or equal to 32, a single IMS A100 would be sufficient, any unused coefficient locations being set to zero. If however, more than 32 coefficients are involved a number of IMS A100 devices can be cascaded to obtain the required filter order. Alternatively it is possible to partition a long FIR transfer function into product terms where each term has an order equal or less than 32. Then, using a single IMS A100, the data can be recirculated through the same device with different coefficients (associated with each term in the transfer function) for each circulation. In this way a very long FIR filter can be implemented with a single device at the expense of a reduction in the data rate.

The IMS A100 can be cascaded very easily, without the need for any external components, to obtain high order filters with a high degree of accuracy. The device has a versatile architecture which allows it to be used in various system configurations. The coefficients can be programmed via a standard memory interface, while the input and output data can be communicated either via the memory interface or dedicated I/O ports. Figure 8.11 shows some of the possible system configurations for the IMS A100. In this diagram the interface between the host and the IMS A100 consists of data and address buses of the processor plus standard memory-type control signals such as R/W, CE and CS. In figure 8.11a the host processor controls the filter coefficients, while the actual data to be processed is supplied directly from an A/D to IMS A100. In this example the filtered output is fed directly to a D/A. Using the IMS A100 and a host processor it is possible to supply the input data to the device and also to collect the filtered samples via the memory interface. This allows system configuration such as those shown in figures 8.11b&c. In figure 8.11b the host processor receives the input data from a peripheral such as an A/D and writes it (may be after some preprocessing) into the data-input register (DIR) of the IMS A100. The filtered output sample is also collected by the host via the memory interface and output (possibly after post processing) to a peripheral such as a D/A. Figure 8.11c shows a configuration where the IMS A100 is used purely as a signal processing accelerator to the host. Numerous other configurations are possible including integrating an IMS A100 into existing microprogrammed systems in order to improve the overall system performance.

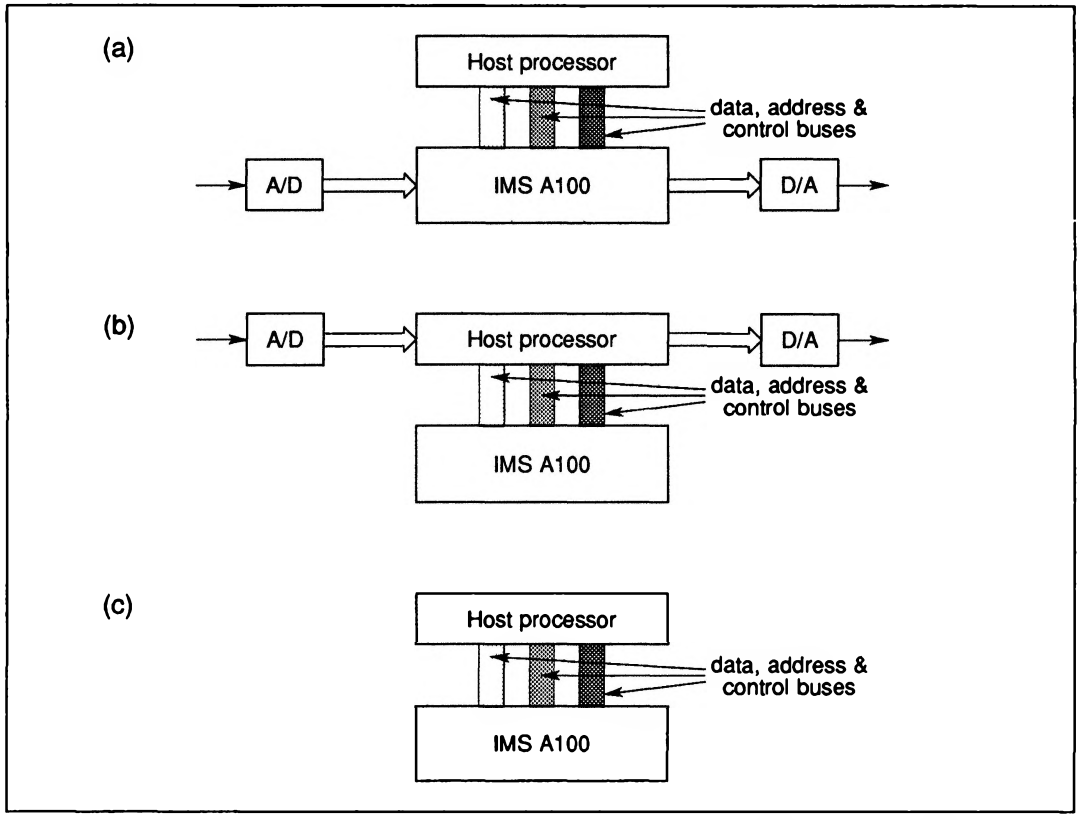


Figure 8.11 Possible system configurations using the IMS A100 in digital filtering applications

As mentioned earlier large numbers of the IMS A100 devices can be cascaded to construct FIR filters of a high order. The cascading does not involve any external components and is simply a matter of connecting the output of the previous device to the cascade input of the next chip and joining the data input ports together (if they are being used rather than the memory interface). In normal operation the cascade input of the first device should be grounded. Figure 8.12a shows this cascading arrangement for two IMS A100 devices and figure 8.12b depicts the block diagram of a system consisting of a host processor and two cascaded devices. In the latter case the data-input register (DIR) of both devices should be associated with the same address in the host's address space; and one of the devices should be selected as a master to generate the GO signal (see product data sheet for further detail).

Another important feature of the IMS A100 is a selector that is incorporated after the multiply-accumulator array. As discussed in the data sheet, the 32 multiply and accumulation in the array are performed to a precision of 36 bits which ensures that no intermediate overflows occur. The output selector can then be used to select and round a 24-bit word from this 36 bit result. This selection and rounding can be programmed to start from bits 7, 11, 15 or 20 and the selected word is sign extended if needed. One particularly useful selection is available when the input data and coefficients are in the form of 16 bit two's complement numbers normalised to between +1 and −1. In this case, if the selection is taken to start from bit 15, the output will have the same format as the input data (i.e. normalised to between +1 and −1).

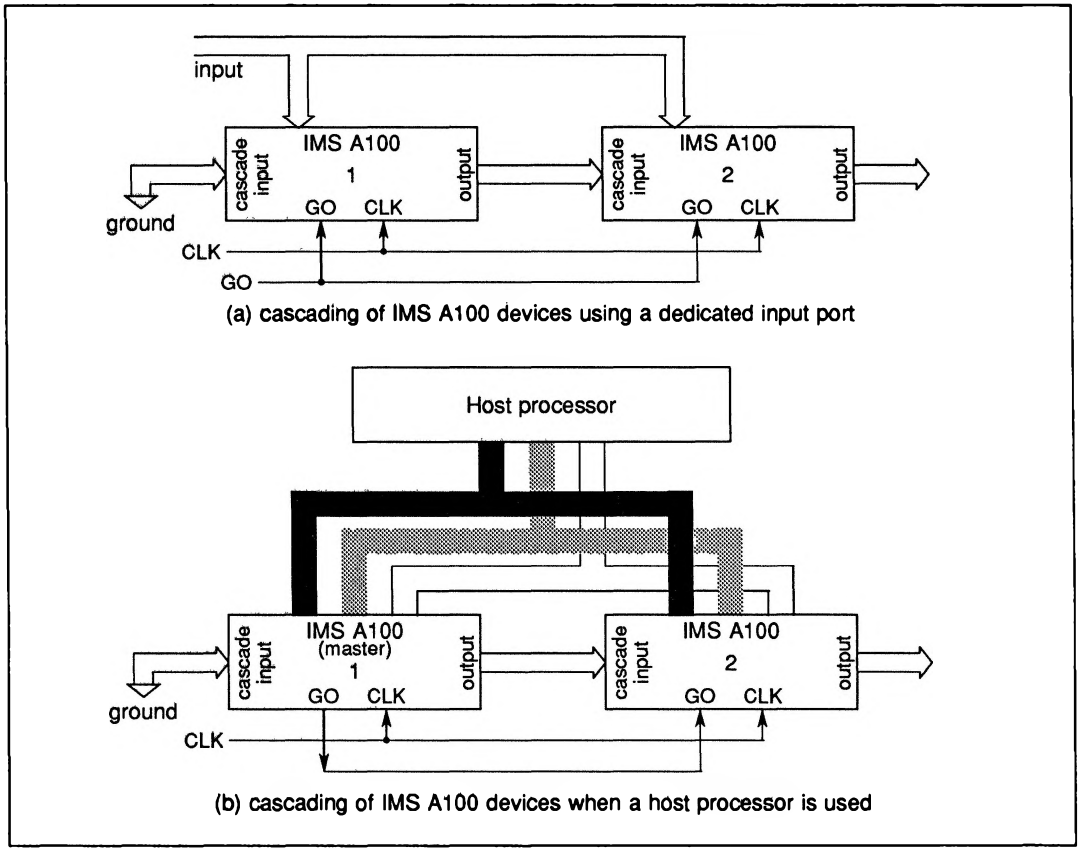


Figure 8.12 Cascading IMS A100 devices

8.4.4 The IMS A100 and IIR filters

Although the IMS A100 is designed primarily for FIR type filter implementations, it can also be used in realizing IIR filters. Referring to figure 8.5 it can be seen that two IMS A100 devices can be used to implement an IIR filter of order 32 or less in the direct form. One chip performing the calculation in the feed-forward path while the other does the feed-back path. Note that in figure 8.5 the output of the feed-back filter has to be combined with the input sequence in a subtractor and fed into the input of the second chip. This subtraction can be performed either by the host processor controlling the two IMS A100s or by an external adder.

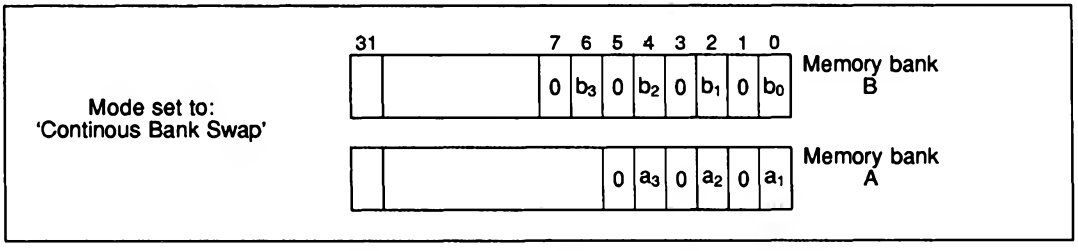


Figure 8.13 Coefficient memory allocation for IIR filter implementation

A simpler and more elegant technique to implement IIR filters using IMS A100 is to make use of the continuous bank swap feature on the IMS A100 coefficient memories. This allows a single IMS A100 to be sufficient for the implementation of IIR filters whose order is less than or equal to 16. (Before describing how this can be achieved it is worth noting that IIR filters generally require considerably fewer stages than their FIR counterparts, and as such a 16th order IIR filter implementable on a single IMS A100 can be considered as having quite a high order). Figure 8.13 shows the coefficient memory allocations in this approach, where a's and b's are the feedback and feedforward coefficients of the IIR filter respectively (see figures 8.5 & 8.6) and are loaded by the host processor. Note that in figure 8.13 alternate coefficients are set to zero in the two memory banks. The chip is also set to the continuous bank swap mode so that in one cycle the feedback coefficients (a's) and in the next cycle the feedforward coefficients (b's) are used in the calculation. It will be shown in the following paragraphs that if the difference between data samples and alternate output samples are written to the data input register of the IMS A100, then the remaining output samples would correspond to the correct filter output. The sequence of operations is as follows:

The host starts the filter operation by writing the first data value, x_0, to the data input register of the IMS A100. Remembering that the coefficient allocation is as shown in figure 8.13, the first output of the device would be a_1x_0. Referring to figure 8.6, it can readily be seen that this is indeed the feed back contribution needed to be subtracted from the next data sample x_1. The host reads this value (a_1x_0) from the data output registers (DOH and/or DOL) and stores it and then writes x_0, for a second time, to the IMS A100 input. This time the coefficient memory banks would have been swapped and the output would correspond to b_0x_0 which can readily be confirmed to be the first correct filter output (see figure 8.6). The host then reads this result as the first valid sample of the filtered output.

Next the host subtracts the feedback factor, read in earlier (a_0x_0), from the second data sample x_1, and writes the difference to the input register of the IMS A100. Remembering that the memory banks are automatically swapped every cycle, the corresponding ouput of the IMS A100 will be:

$$a_2x_0 + a_1(x_1 - a_1x_0)$$

Referring to figure 8.6 you should be able to confirm that this value corresponds to the feedback contribution needed for the third input sample. The host reads this value and stores it and as before writes the input value ($x_1 - a_1x_0$) to the IMS A100 input register for a second time. This will yield the second valid filtered sample i.e:

$$b_1x_0 + b_0(x_1 - a_1x_0) \tag{17}$$

The process is then continued in the same manner. The output of the IMS A100 will alternate between the feedback contribution and the filtered output samples. It should be emphasized that although the host is performing a single subtraction for every output value, it is the IMS A100 device which is performing the bulk of the processing. Having established how the IMS A100 can be configured to implement IIR filters, the next section deals with some of the design techniques that are used for determining the IIR filter coefficients.

8.4.5 Summary of the IIR filter design techniques

The problem of designing recursive filters is one of determining the feedforward and feedback coefficients (i.e. b_n's and a_m's in equation (8). The design techniques for IIR filters can be categorised into two basic groups:

(i) Indirect approaches.

(ii) Direct approaches.

Indirect approaches for the design of IIR filters

As mentioned earlier digital recursive filters are closely related to conventional analogue filters. In the indirect method this similarity is exploited and the digital filter coefficients are determined from a suitable analogue filter, using some form of transformation technique. In other words the indirect approach uses the wealth of knowledge already available on analogue filters (such as Butterworth, Chebyshev and Elliptic filters) and develops a corresponding recursive digital filter. This method involves the following two steps:

(1) the determination of a suitable analogue filter transfer function $H(s)$

(2) transformation and digitization of this analogue filter

Some of the most popular design techniques falling into the indirect category are:

(a) Impulse-invariant transformation.

(b) Bilinear z-transform.

(c) Matched z-transform.

These three techniques can be employed to derive recursive digital filters from conventional analogue filter structures. Before discussing these three techniques the basic characteristics of the common analogue filters, from which IIR filters are derived, will be briefly reviewed. The starting point in the indirect IIR design techniques is often one of the following analogue filter types.

1 **Butterworth filters:** These filters are characterised by the property that their magnitude characteristic is maximally flat at the origin of the s-plane. Butterworth filters are specified by their magnitude-square functions i.e:

$$|H(s)|^2 = \frac{1}{1+(\frac{s}{s_c})^{2n}} \tag{18}$$

The pole locations in the s-plane are equally spaced around a circle of radius ω_c ($s_c = j\omega_c$). These filters have a monotonically decreasing amplitude function with a roll-off of approximately $6n$ dB/decade. Figure 8.14 shows the overall amplitude response of this type of filter.

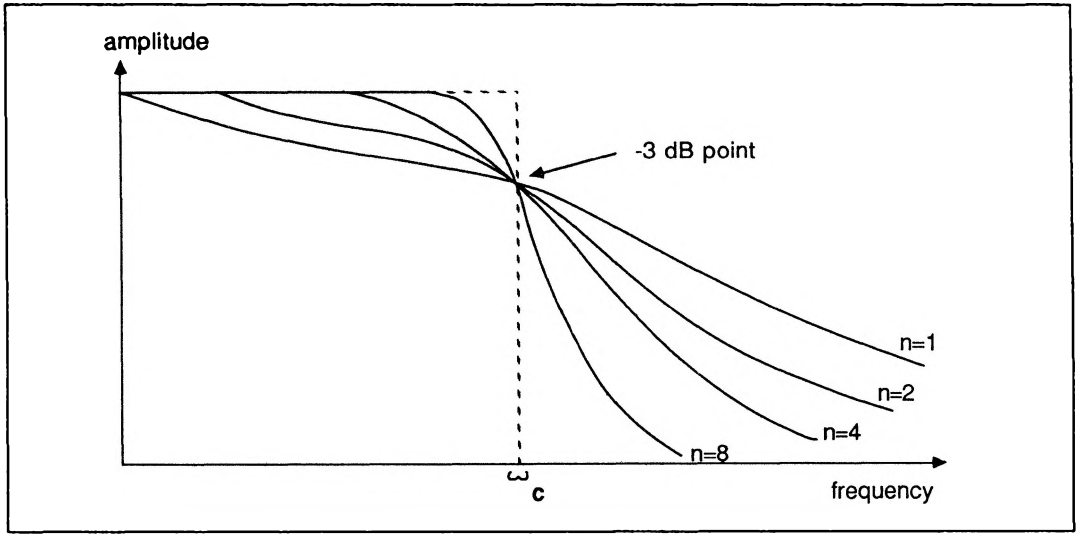


Figure 8.14 Frequency response of the Butterworth filter

2 **Chebyshev filters:** In these types of filters the peak magnitude of the approximation error is minimized over a prescribed band of frequencies and is also equiripple over the band. Chebyshev filters are specified by the magnitude-square function:

$$|H(s)|^2 = \frac{1}{1+\epsilon^2 C_N^2(\frac{s}{s_c})} \tag{19}$$

where $C_N(s)$ is a Chebyshev polynomial of order N. The parameter ϵ is used to specify a magnitude function with equal ripple in the pass band and monotonic decay in the stop band. Figure 8.15 shows the magnitude-square transfer function for the Chebyshev filter (type I) where the amplitude of the ripple is given by:

$$\delta = 1 - \frac{1}{\sqrt{1+\epsilon^2}} \tag{20}$$

The poles of the Chebyshev filter lie on an ellipse determined from the parameters ϵ, N and s_c. Chebyshev filters of type II on the other hand have monotonic behaviour in the pass band (maximally flat around ω_0) and exhibit equiripple behaviour in the stop band. For further details refer to references 1 & 2.

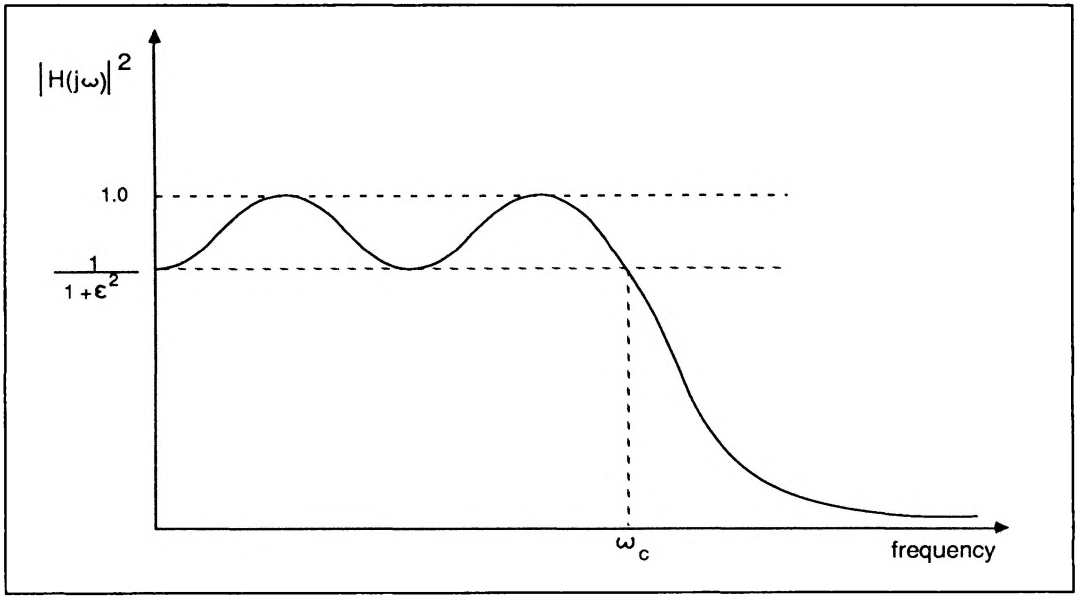


Figure 8.15 Frequency response of the Chebyshev filter (type I)

3 **Elliptic filters:** These filters exhibit a magnitude response that is equiripple in both the pass band and the stop band. These filters are optimum in the sense that for a given order and for a given ripple specification the transition band is the shortest possible. Elliptic filters are specified by the magnitude-square transfer function:

$$|H(j\omega)|^2 = \frac{1}{1+\epsilon^2 C_N^2(\omega)} \tag{21}$$

Where $C_N(\omega)$ is a rational Chebyshev function involving elliptical functions. Figure 8.16 illustrates the magnitude-square response for an elliptic filter.

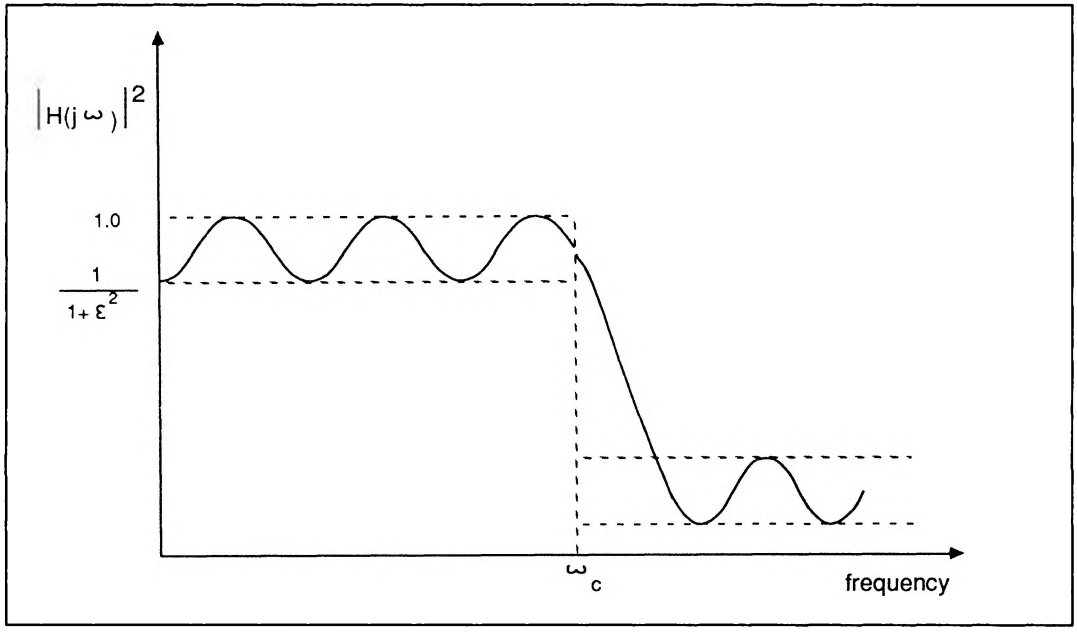


Figure 8.16 Frequency response for an elliptic filter

It is not possible to discuss all analogue filter types in this applications note as the main objective here is to summarize the basic design technique which allow transformation of analogue filters to digital realizations. Interested readers can refer to numerous books available on analogue filters.

Having decided the type and the specification of the analogue filter that satisfies the requirement, the next step in the indirect design method is to use one of the three following techniques to obtain the corresponding digital filter.

Impulse Invariant Transformation

One of the most common techniques for deriving a digital filter from a given analogue filter is the impulse-invariant transformation. As the name suggests this technique consists of using a sampled version of the impulse response of the analogue filter as the impulse response of the digital filter, i.e. the transformation does not change the impulse response of the analogue filter. Figure 8.17 illustrates the relationship between the analogue and the resulting digital responses of a typical low-pass filter obtained via the impulse-invariant method. The important point to note here is that sampling the analogue impulse response results in the frequency response of the resulting digital filter being periodic with a period equal to the sampling frequency f_s. This means that the digital filter will have a frequency response similar to a repetitive version of that of the analogue filter. If the frequency response of the analogue filter does not decay to near zero beyond $\frac{f_s}{2}$ then serious aliasing would occur and the digital filter response would be corrupted. This aliasing problem means that this design technique is not suitable for high pass filters. However for low-pass and band-pass filters the

problem can be avoided by choosing the sampling frequency high enough to ensure that the magnitude of the analogue filter response is negligible beyond $\frac{f_s}{2}$. (Note that the IMS A100 is capable of a sampling rate of 2.5MHz for 16-bit data and coefficients).

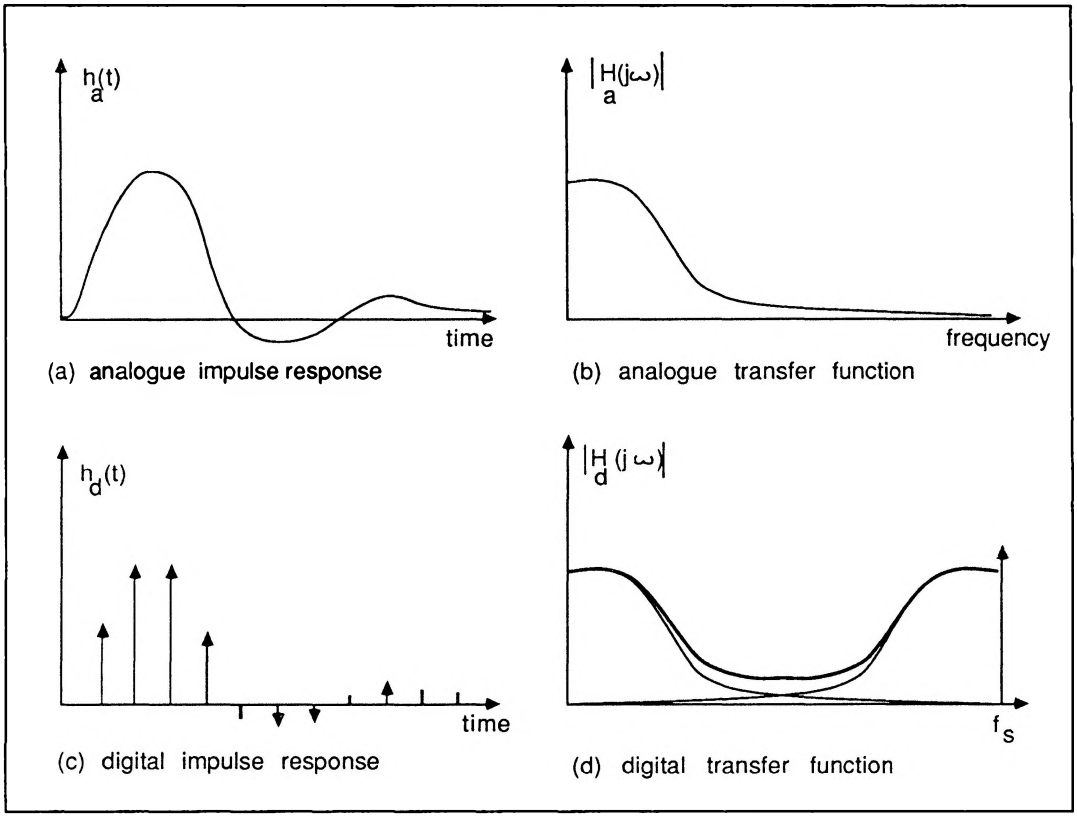


Figure 8.17 The impulsive invariance transformation relationship between analogue and digital impulse and frequency responses

To demonstrate how the impulse-invariant transformation is used to digitize an analogue filter, consider the simple case of an analogue filter with an impulse response $h_a(t) = Ae^{-\alpha t}$ i.e. a simple RC filter (the s-domain transfer function of this filter is $\frac{A}{s+\alpha}$). We start by sampling the impulse response of this analogue filter with a sampling interval T to obtain the corresponding impulse response for the digital filter, i.e.

$$h_a(kT) = Ae^{-\alpha kT} \tag{22}$$

The z-transform of equation (22) is

$$H_d(z) = \sum_{k=0}^{\infty} Ae^{-\alpha kT}\ z^{-k} \tag{23}$$

Noting that as equation (23) is a geometric series the result of the summation would be

$$H_d(z) = \frac{A}{1 - z^{-1}e^{-\alpha T}} \tag{24}$$

Equation (24) provides the z-domain transfer function of the resulting digital filter. To determine the filter coefficient (b_k's and a_m's), equation (24) can be compared with equation (8). For this simple example it can be seen that we have

$$a_1 = -e^{-\alpha T} \qquad and \qquad b_0 = A.$$

In this example, for the sake of clarity, the impulse responses were used to arrive at the z-domain transfer function. As analogue filters are often specified in the s-domain, it is more convenient to perform the impulse-invariant transformation directly from the s-domain to the z-domain. It should be obvious to the reader from the previous example that the required mapping is of the form

$$\frac{1}{s+\alpha} \Longrightarrow \frac{1}{1-e^{-kT}\,z^{-1}} \tag{25}$$

It can be shown that this is indeed a general mapping (reference 1), applicable to the impulse-invariant method for both real and complex s-plane poles.

As a second example consider the two-pole analogue filter specified by:

$$H_a(s) = \frac{2}{(s+3)(s+1)}$$

expanding using partial fraction yields

$$H_a(s) = \frac{1}{s+1} - \frac{1}{s+3} \tag{26}$$

Using equation (25) the digital transfer function would be:

$$\begin{aligned} H_d(z) &= \frac{1}{1-e^{-T}\,z^{-1}} - \frac{1}{1-e^{-3T}\,z^{-1}} \\ &= \frac{(e^{-T}-e^{-3T})z^{-1}}{1-(e^{-T}+e^{-3T})z^{-1}+e^{-4T}\,z^{-2}} \end{aligned} \tag{27}$$

Again by comparing equation (27) with (8) we obtain the filter coefficients,

$$b_0 = 0 \qquad b_1 = e^{-T} - e^{-3T}$$

and

$$a_1 = -(e^{-T} + e^{-3T}) \qquad a_2 = e^{-4T}.$$

As described earlier the sampling period T is chosen to ensure negligible aliasing in the filter transfer function.

The bilinear z-transformation

Another indirect design method commonly used for recursive filters is the bilinear z-transformation. The major characteristic of this transformation is that it avoids the aliasing problem which was inherent in the impulse-invariant transformation. Given an analogue transfer function $H(s)$, let us rename the variable s to s_a to indicate the reference to the analogue world i.e. $H(s) = H(s_a)$. Now let us define a new variable s_d related to s_a by the following mapping:

$$s_a = j\frac{2}{T}\tan(\frac{s_d T}{2j}) \tag{28}$$

where T is the sampling period.

Since the analogue frequency variable ω_a is related to the s-plane variable by $s_a = j\omega_a$, we can also express the above mapping as:

$$\omega_a = \frac{2}{T}\tan(\frac{\omega_d T}{2}) \tag{29}$$

where w_d is defined as $s_d = j\omega_d$.

Starting from an analogue transfer function $H(j\omega_a)$, figure 8.18 illustrates the effect of this mapping on this transfer function. It can be seen from this diagram that the bilinear transformation compresses the entire analogue frequency range ($\omega_a = 0 \rightarrow \infty$) into a finite range equal to half the sampling frequency. This means that the spectral folding problem is completely eliminated and aliasing is therefore avoided. This compression of analogue frequency axis is usually referred to as frequency warping.

The price that is paid for this advantage is a distorted digital frequency scale resulting from this frequency warping. It can be seen from figure 8.18 that due to the non-linear mapping the specification of the resulting filter, such as the cut-off frequency, would be somewhat different from the starting analogue filter. This distortion can be taken into account in the course of digital filter design. For example the cut-off frequency of the original analogue filters are modified slightly so as after the mapping the resulting filter has the desired cut-off frequencies.

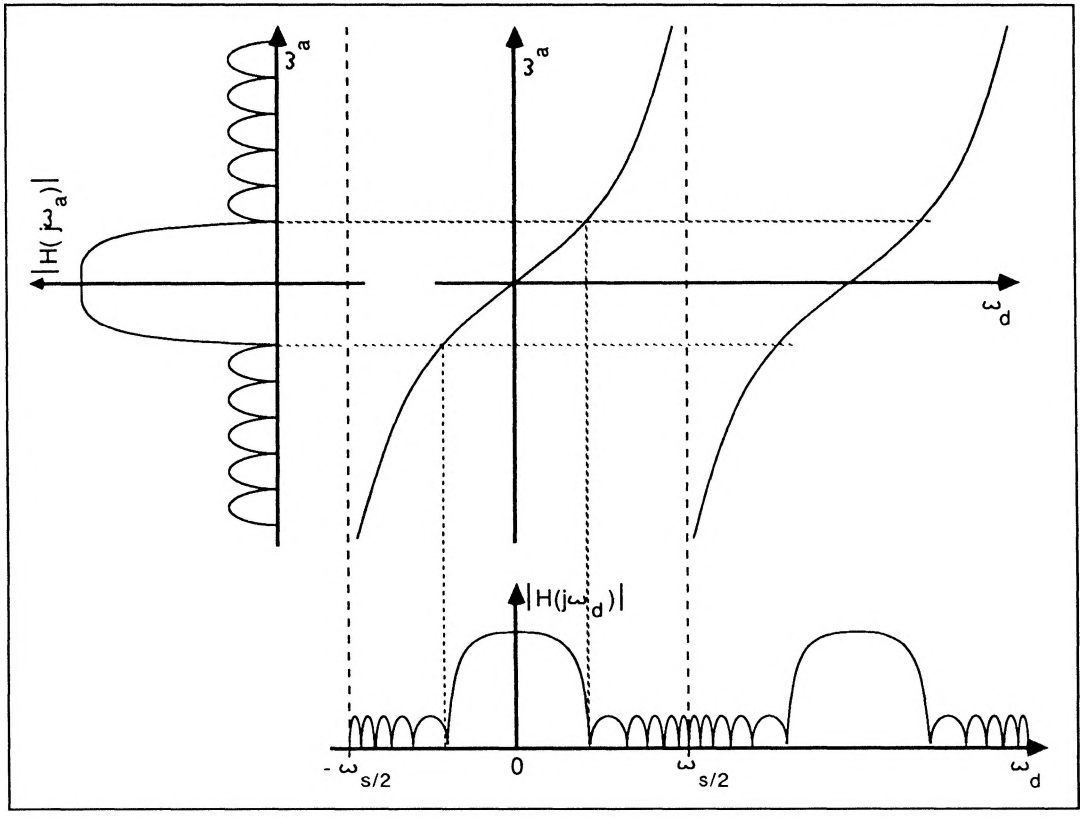


Figure 8.18 Graphical illustration of the bilinear z-transform

Returning to the transformation equation (28), we can rewrite it as:

$$s_a = \frac{2}{T}\left(\frac{1 - e^{-s_d T}}{1 + e^{-s_d T}}\right) \tag{30}$$

and remembering that $z^{-1} = e^{-s_d T}$ we can write

$$s_a = \frac{2}{T}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right) \tag{31}$$

Equation (31) provides the means for bilinear transformation directly from the s-domain to the z-domain suitable for digital filter implementation. To illustrate how the bilinear transformation technique is used consider the following example:

Filter specification

Low pass: 0 → 10kHz pass band
Sampling rate: 100kHz
Transition band: 10kHz to 20kHz
Stop-band attenuation: $-10dB$ (starting at 20kHz)
Filter must be monotonic in pass and stop band.

Design

The monotonicity requirement indicates a Butterworth filter (see previous sections).We have:

digital filter cut-off frequency=$\omega_{cd} = 2\pi \times 10000$
start of digital filter stop band=$\omega_{sd} = 2\pi \times 20000$.

Since the sampling rate is 100kHz, the sampling period would be

$$T = 10^{-5}$$

therefore

$$\omega_{cd}T = 0.2\pi \qquad and \qquad \omega_{sd}T = 0.4\pi$$

Using equation (29) we can calculate the corresponding analogue filter frequencies i.e.

analogue filter cut-off frequency=$\omega_{ca} = \frac{2}{T}\tan(0.1\pi) = 0.6498 \times 10^5$

Start of analogue filter stop band=$\omega_{sa} = \frac{2}{T}tan(0.2\pi) = 1.4531 \times 10^5$.

The required order of the Butterworth filter can be determined by using equation(18) and ensuring at least $10dBs$ attenuation at $\omega = \omega_{sa} = 1.4531 \times 10^5$ i.e.

$$10\log[1 + (\frac{1.4531 \times 10^5}{0.6498 \times 10^5})^{2n}] = 10$$

or

$$1 + (\frac{1.4531 \times 10^5}{0.6498 \times 10^5})^{2n} = 10$$

This gives $n = 1.367$, therefore we choose $n = 2$.

A second order butterworth filter with a cut-off at $\omega_{ca} = 0.650 \times 10^5$ has two equally-spaced poles on a circle of radius ω_{ca} (reference 1) given by

$$s_1,\ s_2 = -0.6498 \times 10^5(0.7071 \pm 0.7071j) = -0.4595(1.0 \pm j) \times 10^5$$

and the transfer function is given by:

$$H(s) = \frac{s_1 s_2}{(s - s_1)(s - s_2)} = \frac{4.223 \times 10^9}{s^2 + 0.919 \times 10^5 + 4.223 \times 10^9}$$

Now we apply the bilinear-z transformation by substituting for s in the above transfer function from equation (31). This gives the following digital filter transfer function:

$$H(z) = \frac{0.0675 + 0.1349z^{-1} + 0.0675z^{-2}}{1 - 1.1430z^{-1} + 0.4128z^{-2}} \tag{32}$$

The digital filter coefficients can be obtained by comparing equation (32) with (8) giving:

$$\begin{aligned} b_0 &= 0.0675 \quad - - - \\ b_1 &= 0.1349 \quad a_1 = -1.1430 \\ b_2 &= 0.0675 \quad a_2 = 0.4128 \end{aligned}$$

These coefficient values are then expressed in binary with the number of bits governed by the required accuracy. The factors affecting the necessary accuracy are discussed in section 5 of this application note.

Matched z-transform

This transformation is a direct mapping from the poles and zeroes in the s-plane to the poles and zeroes in the z-plane.

In general the two previous method i.e. the impulse invariant and the bilinear transformations are preferred to the matched z-transform as there are many cases where the matched z-transformation is not applicable. For this reason this technique is not detailed here. It would be sufficient to point out that the mapping is defined by the replacement relationship:

$$s + k = 1 - z^{-1}e^{-kT}. \tag{33}$$

The direct design techniques for IIR filters

The IIR design techniques described so far were based on transforming a known analogue transfer function into the required digital filter transfer function. It is however possible to design digital IIR filters directly without reference to an analogue filter. Direct design methods fall into two categories namely direct closed form designs and optimisation techniques.

The direct closed form design techniques begin with the desired response of the filter from which one can often decide where to place poles and zeroes to approximate this response. These techniques are not very common and as such will not be discussed here.

The second classes of direct IIR filter design techniques are based on computer optimisation. In these approaches the set of design equations cannot be solved explicitly, instead mathematical optimization techniques are employed to determine the filter coefficients that minimize some error criterion, subject to a set of design equations. The algorithms involved in these optimisation techniques are of an iterative nature and are terminated when the error reaches a minimum or the number of iterations exceeds a specified limit.

Among the most commonly used optimisation technique is one which minimizes the pass-band ripples in filters exhibiting a given stop-band attenuation. This technique is sometimes referred to as the minimax method and the optimization algorithm involved has been developed by Fletcher and Powell (reference 7). The Fletcher-Powell optimization algorithm generates the filter coefficients by using a convergent descent method.

The spectral flatness approach is another optimisation technique and is based on the fact that multiplying the desired frequency response by its inverse should result in unity throughout the frequency spectrum (i.e. a flat spectral line). Any deviations from the ideal response would result in ripples in this flat spectral line. Optimisation techniques have been developed which attempt to minimize these ripples (reference 8). The difficulty with this technique is the modeling of the desired frequency response.

Mean-square-error optimization techniques have also been developed for IIR filter design. One such technique has been described by Steiglitz (reference 9) which involves minimizing the square of the difference between actual filter behaviour and the desired performance. This algorithm searches an error *.vs.* design-parameter curve for a local minimum.

The details of the above optimisation techniques are beyond the objectives of this application note. However the references given should prove adequate for interested readers.

8.5 Finite word-length considerations and problems

In implementing digital filters both the input samples and the filter coefficients have to be quantised and expressed in a limited number of bits. In the IMS A100 chip both the coefficients and data samples can be quantized up to 16-bits of accuracy, although smaller word-lengths can be used if desired.

The problems of finite word length in digital filters apply to both FIR and IIR filters but their implications are much more severe for the IIR filters, due to their inherent feedback nature. In the fixed-point implementations of digital filters it is usual to normalize the numbers so as to make their absolute values less than one i.e. in the form of

$$d_{N-1}.d_{N-2}d_{N-3} - - - - - - - - - d_5d_4d_3d_2d_1d_0$$

where d_n represents the nth bit in the word and (.) indicates the binary point. Using this format (and two's complement notation) the number

$$0.1111\ldots1111$$

would represent a vlaue very nearly equal to +1, while the number

$$1.0000\ldots0000$$

would represent a value equal to -1.0.

If purely integer numbers were to be used the process of truncation or rounding after multiplications would become meaningless. However using the above fractional-number representation, where the numbers are

normalized to be less than one, the problems would not arise as the product of two numbers which are less than one would also be less than one.

In general there are three sources of error arising in the implementation of digital filters these are:

(i) Finite precision of the filter coefficients

(ii) Limited word length of the input data

(iii) Round-off and truncation errors in the multiplication and addition operations.

The finite precision in the representation of the filter coefficients will obviously cause the frequency response of the filter to depart to some extent from that desired for both FIR and IIR filters.

Furthermore in the case of the recursive IIR implementation, because of the existence of feedback paths, this finite precision may cause instabilities in the filter behaviour. This happens because the inaccuracies may move the z-plane poles outside the unit circle hence causing instabilities. The chances of this happening depends on how close the poles are to the unit circle in the first place. If multiplication and addition operations are followed by truncation and rounding (in order to contain word growth) further difficulties may arise. These problems may manifest themselves in undesirable oscillations in the form of 'limit cycle' or 'overflow' oscillations (discussed later). It is therefore absolutely essential for the filter behaviour to be simulated using the precision and roundings involved in the intended implementation. This is particularly relevant to recursive IIR filter where a risk of instability exists.

One of the consequences of rounding and quantisation in the digital recursive(IIR) filters is the limit-cycle phenomenon, which takes the form of a stable periodic non-zero output for zero or constant input. The limit cycle behaviour of a digital filter in general is complex and difficult to analyse. However for simple first order filters, it is possible to illustrate the effect by way of an example. Consider the first order recursive filter with the following equation:

$$y(n) = 0.09x(n) + 0.91y(n-1)$$

Assume that each output y(n) gets rounded to the nearest integer, also assume that the input is constant at 100 and the previous output is 90.

The following table shows the resulting rounded output sequence for each iteration.The last column shows the perfect output (without rounding) for comparison.

n	x(n)	y(n)	rounded y(n)	perfect y(n)
0	100	–	90	–
1	100	90.9	91	90.9
2	100	91.81	92	91.72
3	100	92.72	93	92.46
4	100	93.63	94	93.14
5	100	94.54	95	93.76
6	100	95.45	95	94.32
7	100	95.45	95	94.83
8	100	95.45	95	95.30
9	100	95.45	95	95.72
10	100	95.45	95	96.11
..	...	.....	..	.....
..	...	.....	..	.....
..	...	.....	..	.....
..	100	95.45	95	100.0

It is observed that the output sticks at a value of 95. However if the same filter is implemented with very high precision and no rounding the filter output would closely approach 100 (last column in the table).

If we approach the limit from the opposite side by starting with a value of $y(n)$ of say 110, the output would arrive at a limit of 105. You can see from this example that the system has a dead zone of ±5 units around the ideal output of 100.

In fact it can mathematically be shown that for a first-order recursive filter of the form

$$y(n) = bx(n) + ay(n-1)$$

The dead zone is given by

$$|dead\ zone| \leq \frac{\frac{q}{2}}{1-|a|} \tag{34}$$

where q is the quantisation step. In the above example a quantised step of 1 was used and equation (34) gives a dead zone of ±5 too.

For second-order systems similar results to (34) have been derived in the literature (reference 1).

Overflow oscillation is another problem associated with digital recursive filters. In the IMS A100 chip the full internal precision ensures that no overflow occurs in the multiply-accumulator array. The only source of possible overflow is the external addition which is performed in combining the feedback terms with the input samples (see section 4.4). A simple but effective way to eliminate these oscillations is to perform this addition in a saturating manner (similar to analogue adders). This operation can easily be taken care of by the controlling host processor.

In the IMS A100 device the data and coefficients can be expressed to a precision as high as 16 bits. The 32 multiplications and additions are carried out to 36-bit precision. This ensures that no overflow occurs in the multiply-accumulation array (unless all the coefficients and 32 consecutive data items have values equal to the most negative 16-bit number i.e. 1000000000000000 in binary, which is of course highly unlikely). The selector at the output of the multiply and accumulate array allows the rounding and selection of 24 bits out of this 36 bits. The combination of full internal accuracy, the selector functionality and the fact that the IMS A100 devices can easily be cascaded allows high quality FIR filters to be readily implemented. As described earlier the device can also be used to implement efficient IIR filters only in direct forms. It is well known that for high order filters direct implementations of IIR filters are more prone to instabilities compared to cascade or parallel arrangements. However the full internal precision of the IMS A100 combined with comprehensive filter simulations should minimize these instabilities. It should however be emphasized that it is possible to implement a high order high precision IIR filter in the cascade form on the IMS A100 at the expense of processing speed. In this case the IMS A100 should be used to implement low order (2nd or 4th order) sections of a cascade arrangement in turn by reloading suitable coefficients. The functionality of the whole filter is obtained by recirculating the first output batch through the chip with its coefficients modified to implement the 2nd section in the cascade array and so on.

For the IIR filter implementations figure 8.11c & 8.11b can be considered as possible system configurations.

8.6 Adaptive filters

So far we have discussed digital filters with fixed characteristics. Fixed filters are used in many practical situations to combat noise or interfering signals (e.g. a matched filter) or to select a desired frequency band (e.g. a band-pass filter). In digital signal processing the parameters of such fixed filters are determined once and remain unchanged during processing. Adaptive filters on the other hand automatically adjust their own parameters and seek to optimize their performance according to a specific criterion. The adaptive nature of such filters makes them particularly suitable for situations where signal properties are unknown or variable with time.

Figure 8.19 illustrates the basic structure of an adaptive filter. The input signal $x(t)$ is filtered or weighted in a programmable filter to yield an output $y(t)$. The filter output $y(t)$ is then compared with a reference (sometimes called a training signal) waveform to yield an error signal $e(t)$. This error is then used to update the filter coefficients in such a way that the error is progressively minimized. Several algorithms for updating the filter coefficients have been developed and can be found in references 10, 11 & 12.

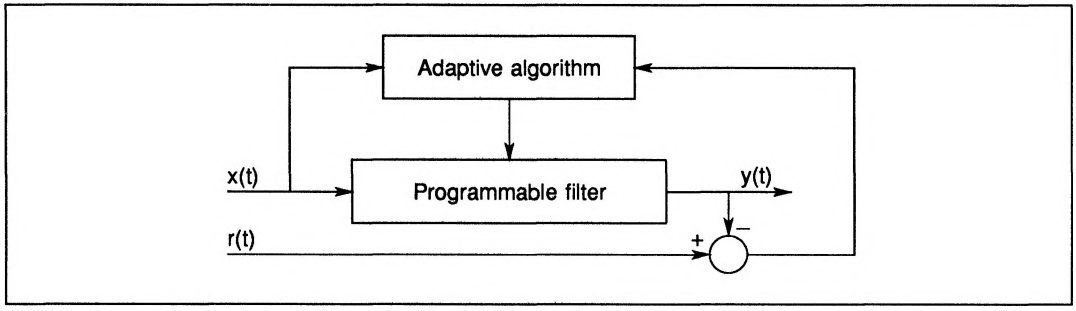


Figure 8.19 Basic structure of an adaptive filter

One example of adaptive filtering is echo cancellation in telephony. Echoes are the result of impedance mismatches in the communication circuits. The hybrid couplers which are used at the interface between two-wire and four-wire circuits are a major source of echoes. Figure 8.20 shows how an adaptive filter arrangement can be used to cancel these echoes at the hybrid interface. Notice that in this case the training signal contains the echo, while the input to the adaptive filter is the signal arriving at the hybrid. Effectively the filter adaptively models the echo path and produces a synthetic antiphase echo return which cancels the echo in the 4-wire path returning from the hybrid.

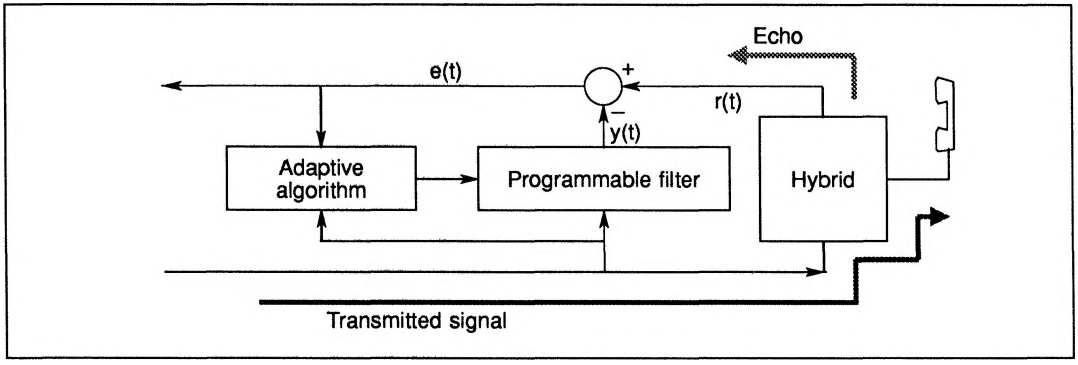


Figure 8.20 Application of adaptive filtering techniques to echo cancellation

Adaptive filters have application in low-bit rate speech coding based on linear prediction where the filter coefficients, after adaption, are transmitted instead of the speech signal itself.

The programmability of the IMS A100 can be exploited in the implementation of adaptive filters as well as fixed filters discussed earlier.

8.7 References

1 *Theory and application of digital signal processing*, Rabiner L.R., and Gold B., Prentice-Hall Inc, Englewood Cliffs, NJ, 1975.

2 *Digital signal processing*, Oppenheim A.V., and Schafer R.W., Prentice-Hall Inc, Englewood Cliffs, NJ, 1975.

3 *A computer program for designing optimum FIR linear-phase digital filters*, McClellan J.H., Parks T.W., and Rabiner L.R., IEEE Transactions on Audio and Electroacoustics, Vol. AU-21, No.8 , December 1973.

4 *Digital signal processing*, Peled A., and Liu B., John Wiley and Sons Inc., New York, 1976.

5 *Practical design rules for optimum finite impulse response low-pass digital filters*, Rabiner L.R., Bell Systems Technical Journal, Vol. 52, No.6, July-August 1973.

6 *Approximate design relationships for low-pass FIR digital filters*, Rabiner L.R., IEEE Transactions on Audio and Electroacoustics, Vol. AU-21, No.5, October 1973.

7 *A rapidly convergent descent method for minimization*, Fletcher R., and Powell M., Computer Journal 6 (No.2) 1963, pp 163-168.

8 *Time series analysis: Forcasting and control*, Box G., and Jenkins G., Holden Day, San Francisco, CA, 1975

9 *Computer aided design of recursive digital filters*, Steiglitz K., IEEE Transactions on Audio and Electroacoustics 18 (No.2), June 1970, pp123-129.

10 *Adaptive noise cancelling: principle and applications*, Widrow B. *et al*, Proc. IEEE, 1975, Vol.63, No.12, pp 1692-1716.

11 *Design and application of adaptive filters*, Grant P.M., Cowan C.F.N., Electronics & Power, February 1985.