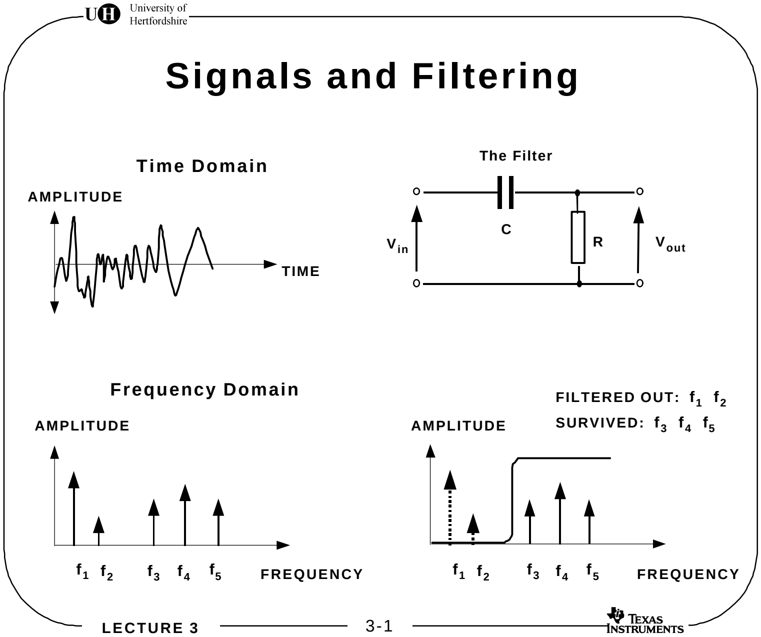
LECTURE 3

FILTERING

OBJECTIVES

The objectives of this lecture are to:

- Introduce signal filtering concepts
- Introduce filter performance criteria
- Introduce Finite Impulse Response (FIR) filters
- Introduce Infinite Impulse Response (IIR) filters
- Consider advantages of digital filters
- Consider advantages of using DSP in digital filter implementation
- Consider sources of noise in digital filters



- **Signals**

Real signals are comprised of a number of frequencies. Some signals may contain both high frequency and low frequency components. Depending on the application, some frequencies may be undesirable, such as a low frequency AC power supply hum or interference from some other source. Filters can be used to remove these undesirable frequency components. As an example, the signal shown on the above diagram has five components, marked in increasing frequency order, f_1 to f_5. A filter could be used to remove f_1 and f_2. The circuit shown at the top right acts as a filter that will remove most of the frequencies f_1 and f_2 so that only the higher frequencies f_3, f_4 and f_5 remain. This is shown in the frequency domain graph in the bottom right of the diagram.

Filters are applied in audio systems. The bass control on an audio preamplifier applies more or less gain at lower frequencies than at higher frequencies. The end result is that lower frequencies (bass) are emphasized or attenuated. Note that in this case, frequency components were not being removed, but simply shaped.

- **Filtering**

It is particularly easy to observe the effect of filtering in the frequency domain. For example, the high-pass filter shown on the diagram, attenuates (resists) the low-frequency components of the signal, while the high-frequency components of the signal are passed through without noticeable modification. This is the essence of filtering.

The main filter types are as follows:

Low-pass Filters (LPF) – These filters pass low frequencies and stop high frequencies.

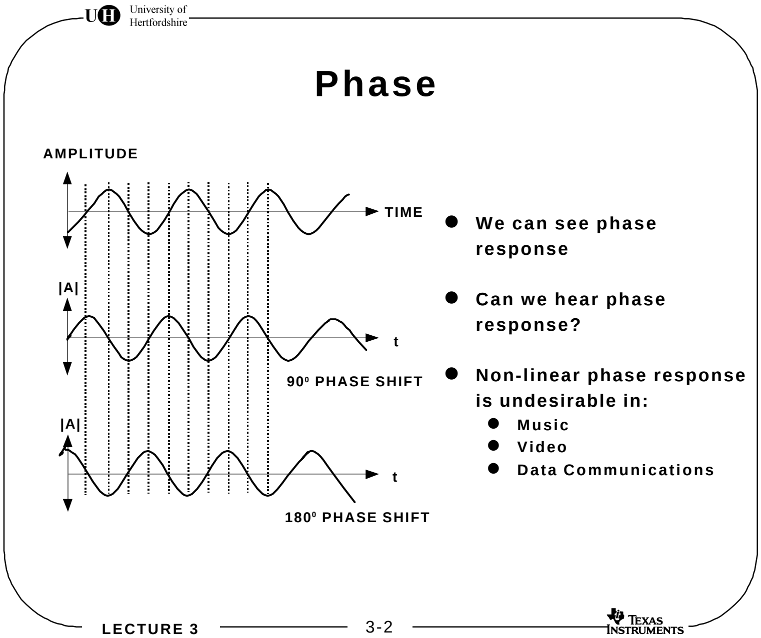
High-pass Filters (HPF) – These filters pass high frequencies and stop low frequencies.

Band pass Filters (BPF) – These filters pass a range of frequencies and stop frequencies below and above the set range.

Band-Stop Filters (BSF) – These filters pass all frequencies *except* the ones within a defined range.

All-Pass Filters (APF) – These filters pass all frequencies, *but* they modify the phase of the frequency components.

We shall examine analog high- and low-pass filters in some detail, but all filter types can be made both with analog electronics and as digital filters using DSPs.

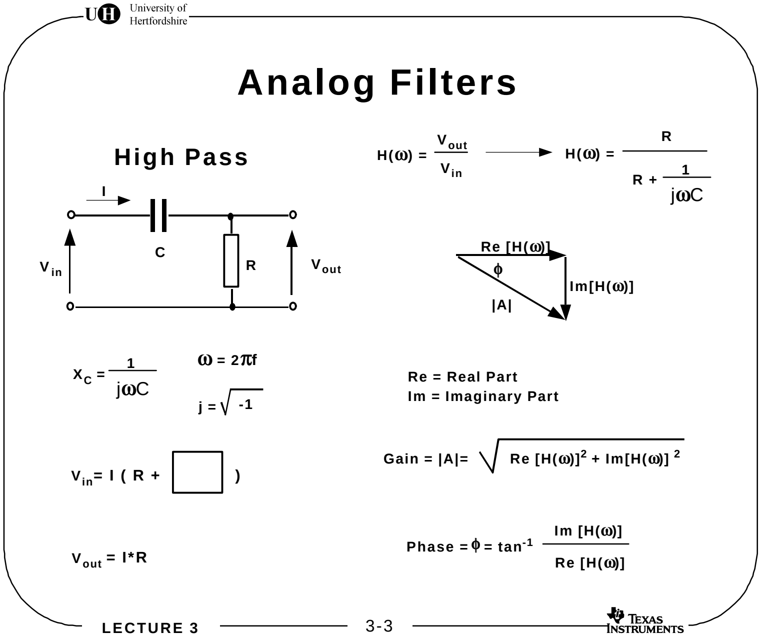


- **Phase**

Before we start examining filters, let us look at another property of signals that we shall frequently refer to–*phase*. The phase of a signal refers to its timing. Two signals of the same frequency can be in phase or out of phase, and when they are out of phase, one of the frequencies has been delayed. As the above design shows, a 90-degree phase-shifted sine wave has its peaks where the original waveform has zero amplitude value. A 180-degree phase shift represents a signal, which is completely *out of phase* with the original signal. If these two signals were added, the total would amount to no signal at all, because while one signal is positive, the other signal is negative with equal amplitude. If the phase shift were 360 degrees, the signal would be delayed by one period, and would once again be back in phase.

Electrical components are one source of phase shift in signals. For example, capacitors and inductors cause shifts in phase. The phase shift that they introduce can depend on the frequency of the input signal, as well as the other components around them.

Phase is an important property of the signal. Humans locate medium frequency sound by working out the phase difference between signals arriving at each ear. This is a property that is used in stereo hi-fi reproduction. In stereo recording, two microphones are used instead of one to capture the phase information in the sound field. When the recording is played back, two speakers are used to preserve the phase information in the sound field. The effect of this phase information is easy to examine by the use of a stereo amplifier that can be switched into mono (both microphone signals are added together) mode. In stereo mode, the sum of the two microphone signals occurs in space and the sound that is heard at the two ears is *slightly* different. The phase information between these two signals is used by the brain to locate the sound source. In mono mode, the summation is done in the amplifier and both speakers output exactly the same sound. The result is both ears hear *exactly* the same thing and the phase information between the signals is lost.



- **Analog Filters**

The simple capacitor resistor circuit on the above diagram is a high-pass filter. It will pass higher frequencies without much modification and stop (attenuate) lower frequencies. This can be predicted because a capacitor will not pass low frequencies currents, but will allow high frequencies currents to pass through.

The reactance Xe of a capacitor is frequency-dependent, and its mathematical representation contains the operator (j), which is used to represent the square root of -1. Let us take the reactance equation for a capacitor as granted and work with it. By calculating the input and output voltages in terms of the components in the circuit, we can arrive at an equation that represents the response of the circuit to input excitations, (V_{out}/V_{in}) This is commonly called the *transfer function*.

The first step is to work out an equation for the input voltage (V_{in}).

$$V_{in} = I*Z_{in}$$

where Z_{in} is the input impedance. In this case, the input impedance is the resistance of R added to the reactance of C, since they are in series. So:

1) $V_{in} = I*(R+ \qquad\qquad)$ complete the blank space in the equation

Next, the output voltage (V_{out}) is simply the voltage across R, so:

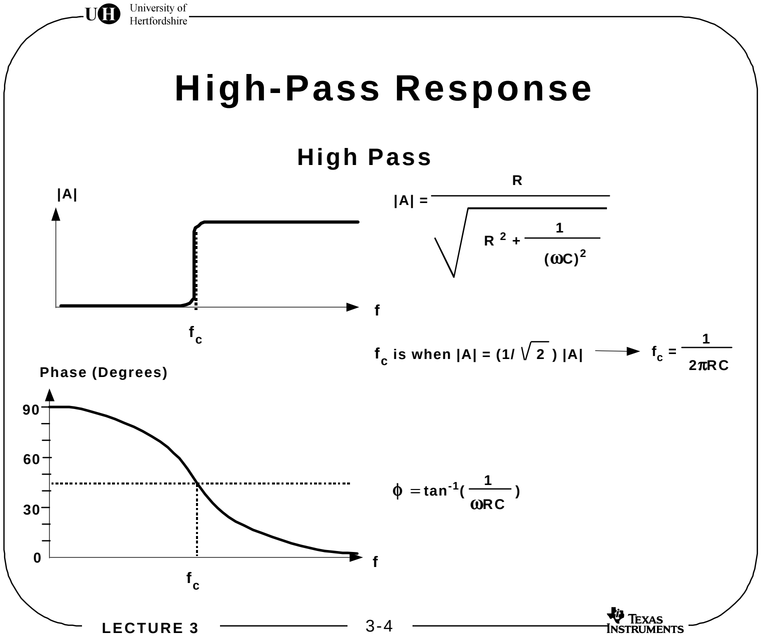
2) $V_{out} = I*R$

Given this, the transfer function (also called H(ω)) V_{out}/V_{in}, is obtained by dividing equation 2 by equation 1.

- **Gain and Phase Response**

H(ω) is a frequency-dependent complex function. It is frequency-dependent because it includes ω, and complex because it includes j. This means that the gain and phase will vary with frequency. Using trigonometry, separate equations can be worked out for the gain and phase of the circuit. The vector diagram shows geometrical representation of the transfer function, making it is easier to see how the gain and phase equations are calculated. The magnitude of the signal is the hypotenuse of the triangle, and can be worked out using the Pythagorean theorem. The phase of the circuit is shown as the angle (∅) on the phasor diagram, and is the inverse tangent of the opposite side length divided by the adjacent side length.

Since both gain and phase equations contain frequency-dependent components, it is natural to expect both to be frequency-dependent. This frequency dependence is used to our advantage in filtering.



- **Gain**

$$|A| = \frac{R}{\sqrt{R^2 + \frac{1}{(\omega * C)^2}}}$$

Let us just consider the gain equation at two extremes. When $\omega = 0$,

$$|A| = \frac{R}{\sqrt{R^2 + \frac{1}{(0 * C)^2}}} = \frac{R}{\sqrt{R^2 + \infty}} = \frac{R}{\infty} = 0 \qquad \left(\text{Remember } \frac{1}{0} = \infty, \text{ and } \frac{1}{\infty} = 0 \right)$$

This means that if the input is direct current (DC), the gain tends towards zero due to the high-value denominator, so the signal will be stopped.

At the other extreme, when ω tends towards infinity,

$$|A| = \frac{R}{\sqrt{R^2 + \frac{1}{(\infty * C)^2}}} = \frac{R}{\sqrt{R^2 + 0}} = \frac{R}{R} = 1$$

The gain equation approaches unity, so the signal will be passed. This gives us an indication about the behavior of gain with frequency. If the signal is stopped at low frequencies, and passes at high frequencies, it is a high-pass filter. Another important point on the gain versus frequency plot is the cut-off frequency. This is defined as the point where the gain falls to ($1/\sqrt{2} = 1/1.414 = 0.707 = 70.7\%$) of its original value, commonly called the 3dB point (the reason for this naming is explained later on in this lecture). This can be calculated from the gain equation.

$$\frac{1}{\sqrt{2}} = \frac{R}{\sqrt{R^2 + \frac{1}{(\omega * C)^2}}} \qquad \text{so} \qquad \sqrt{R^2 + \frac{1}{(\omega * C)^2}} = R\sqrt{2}$$

Squaring both sides gives $R^2 + \frac{1}{(\omega * C)^2} = 2R^2$.

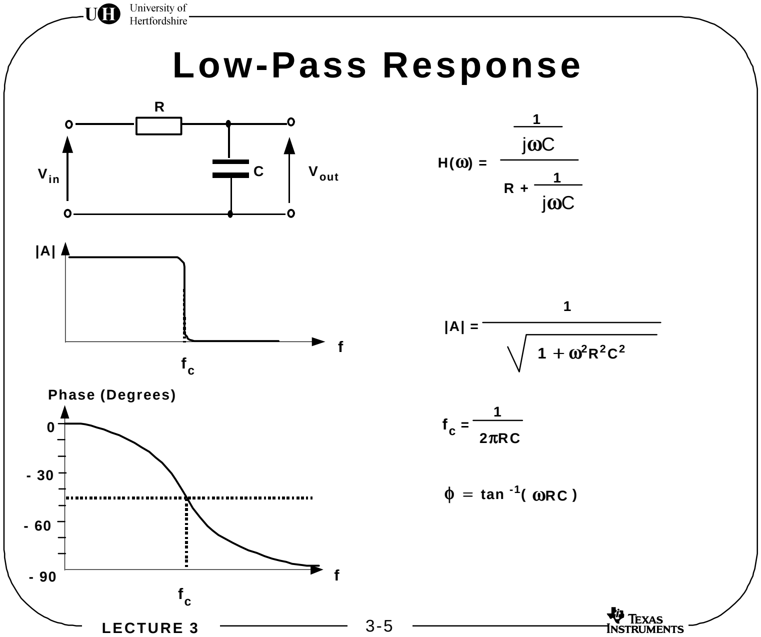
Taking R^2 to the other side gives $\frac{1}{(w * C)^2} = 2R^2 - R^2 = R^2$

Now take the square root of both sides $\frac{1}{(w * C)} = R$ and rearrange $\frac{1}{CR} = \omega \ = 2\pi f$

This gives the cut-off frequency $f_c = 1/2\pi RC$. The cut-off frequency identifies a turning point in the behavior of the filter, and marks the start of the pass band. In the case of a high-pass filter, any frequency above this point will be passed with little attenuation, and frequencies below this point will be attenuated.

- **Phase**

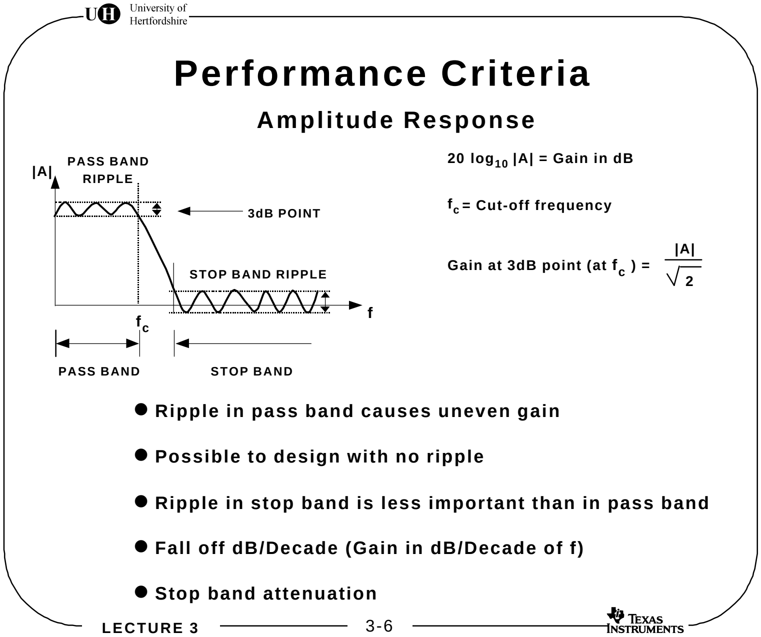
Phase response can be calculated from the phase equation. Phase response starts with a 90-degree lead at low frequencies, and falls to 45 degrees at the cut-off frequency. Beyond the cut-off and towards higher frequencies, phase shift continues to drop. In any realistic scenario, we are concerned with the phase response in the pass band. When the amplitude is small, the phase has little effect on the signal. In this particular case, the phase response may be adequate for some applications.



- **Low-pass Filter**

A simple low-pass filter consists of a resistor and a capacitor, as did the high-pass filter, but notice that the two components have been swapped. Now the capacitor will be shorting the high frequencies down to ground, leaving the lower frequencies. The low-pass filter response is very similar to the high-pass filter response that we have just examined. The only difference is that it is reversed in frequency. The equations are worked out the same way, but because the two components are swapped, R must be swapped with $1/j\omega C$. Gain response falls below unity beyond the cut-off frequency. The phase of the output signal lags behind the input by 45 degrees at cut-off, and this lag increases to 90 degrees at higher frequencies.

We have looked at two very simple filters. We know that the signal is attenuated at certain frequencies and the phase of the output signal changes with the frequency. How then, do we decide that the performance of the filter is adequate for our purposes? What are the criteria for comparing filter responses?



Let us now formally define filter terminology and establish some performance criteria for filters.

- **Gain (in dB) and Frequency (in Decades)**

We can express a greater range of numbers with less zeros using a logarithm. It is traditional and useful to use *decibel*s (dB) for expressing gain of filters. One reason for using decibels is that it follows more closely the way the human ear detects different volumes in sound. Strictly speaking, the decibel is a measure of power and is defined:

$10 \log V^2$,

where in this case, V^2 represents the power dissipated if V is placed across a 1 ohm resistor. Hence, the *deci* in decibel comes from the 10, and *bel* is named after the originator of the unit, Alexander Graham Bell, from his work with the telephone. Using the properties of the logarithm, one can rewrite the decibel in terms of the amplitude assuming a 1 ohm resistor:

$20 \log V$.

Thus, to convert a gain to decibels, take the log to the base 10 of the gain and multiply by 20.

So, if a system at a particular frequency has $V_{in} = 5V$, and $V_{out} = 1V$ then:

$$\frac{V_{out}}{V_{in}} = \frac{1}{5} = 0.2$$

To convert this to decibels, first take $\log_{10}(0.2) = -0.69897$

and multiply this by 20.

20*-0.69897 = -13.97dB

Note that $\log_{10}(1) = 0$, which is why magnitude graphs in dB usually have 0 at the top of the scale.

For frequency, *decades* are used to cover a greater range of frequencies in a meaningful way. A decade is the distance between a frequency and 10 times that frequency (e.g., 3.4kHz to 34kHz). Therefore, a 20dB/decade roll-off means that the filter increases its attenuation by 20 dB for every decade of frequency. In digital filters, linear frequency ranges are also used, since the frequency range is usually not as great.

- **Cut-Off Frequency**

Cut-off frequency is defined as the frequency where the gain of the filter falls to ($1/\sqrt{2}$ = 1/ 1.414 = 0.707) of its value in the pass band. It is also referred to as the -3dB point since ($20\log_{10}$ (0.707) = -3).

- **Pass Band, Stop Band and Transition Region**

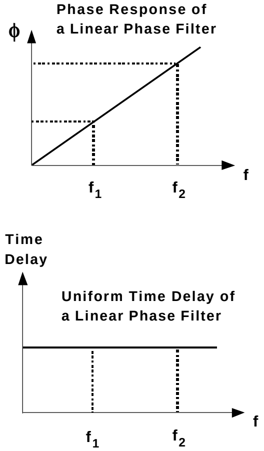
The pass band is the range of frequencies over which signals pass through the filter with virtually no attenuation. The stop band is chosen by the designer to be at or below a certain level of attenuation. The amount of attenuation varies from application to application, but there will always be a small amount of the stop band frequencies left in a realistic filter. The frequencies between the 3dB point and stop band are referred to as the *transition region*. The transition region is characterized by its fall-off rate that is usually expressed in dB/decade.

- **Ripples**

A *ripple* occurs when the gain is not even or level throughout the pass band or stop band, fluctuating as shown in the diagram. While the ripple is very common in digital filters, it is not found in simple RC analog filters. A filter can produce ripples in both the pass band and the stop band, but we are more concerned with pass-band ripple, since it causes some degree of noticeable gain or loss to the signal in which we are interested. It is possible to design ripple-free filters, but there is usually a trade-off between the amount of ripples in the pass band, the fall-off rate in the transition region, and the stop-band attenuation. Some ripples may be tolerable in the pass band depending on the application.

- **Summary of Filter Performance Criteria**
    - Pass-band ripples
    - Fall-off rate in transition region
    - Stop-band attenuation
    - Phase response

Phase Response



- Phase response represents time delay of different frequencies
- Linear phase response delays all frequencies by the same amount
- Time delays at f_1 and f_2 are equal
- Non-linear phase response
  - Delays all frequencies by different amounts
  - Causes distortion to original signal
  - Is audible in a music application
  - Is visible in a video application
- Linear phase is only important in pass band
- Some non-linearity can be tolerated

LECTURE 3 3-7

- **Phase Response**

Phase response of a filter is one of the important performance criteria that determine a filter's suitability for a particular application. Phase response represents the time delay introduced to all or part of the signal.

- **Linear and Non-Linear Phase Response**

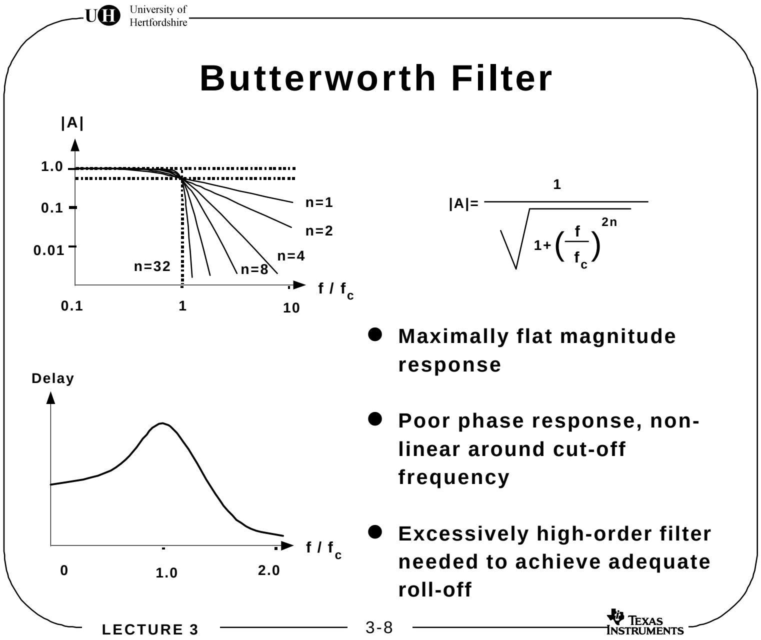
A filter with a linear phase response delays all frequencies by the same amount. To demonstrate this concept, lets take two frequencies (f_1 and f_2), that are both delayed by the same amount of time (0.2mS).

If f_1 = 100Hz, then the period of the signal is 1/100 = 10mS. If the signal is delayed by 0.2ms, then 0.2mS/10mS = 0.02, or 2%.

If f_2 = 1kHz, then the period of the signal is 1/1000 = 1mS. In this case, the signal would be delayed by 0.2mS/1mS = 0.2, or 20%.

So, with these two signals, they are delayed by the same amount, but they will have different phase shifts. This is shown on the diagram above. The output signal is not distorted, but delayed by a certain amount. Because a real signal contains a large number of frequencies, if each frequency is delayed by a different amount, the output signal will be distorted. Some applications cannot tolerate a non-linear phase response (e.g., modems). Other applications, such as stereo music, use complex phase relationship filters and delays to enhance the stereo effect.

A linear phase response is really only important in the pass bands, since the passed signals are the ones in which we are interested. There is usually a trade-off between a linear phase response in the pass band and other filter performance criteria, such as steepness of roll-off and stop-band attenuation.



- **Practical Analog Filters**

There are a number of practical analog filter designs with different performances for gain and phase. We shall closely examine the "Butterworth" filter

- **Butterworth Filter**

This filter is commonly referred to as a *maximally flat* filter due to its amplitude response in the pass band. The pass band in a Butterworth filter is virtually ripple-free. However, there are two problems:

1. A non-linear phase response in the pass band rules out its use in applications that require a linear phase response. On the above diagram, the delay plot versus frequency shows the non-linearity of the phase response on a normalized frequency scale. The delay is worst at $f/f_c = 1$, which is the 3dB point.

2. It has a slow roll-off in the transition region. In order to achieve adequate roll-off, a number of stages need to be cascaded. The order of a filter refers to the number of cascaded stages. Our amplitude response plots show various responses that are achievable for different orders of Butterworth filters. The greater the number of stages, the worse the phase response and the greater the attenuation in the stop band.

University of Hertfordshire

Filter Types

Chebyshev

- Steeper roll-off than Butterworth
- More ripple in pass band
- Poor phase response

Bessel

- Maximally flat phase response
- Less steep roll-off

Filter design software packages allow us to:

- Experiment with many designs
- Evaluate suitability of gain and phase responses

LECTURE 3 3-9 TEXAS INSTRUMENTS

- **Chebyshev Filter**

There are a number of other commonly-used filter types. The Chebyshev design has a steeper roll-off than the Butterworth design. However, it has more ripple in the pass band and poor phase response. Since the steepness of the roll-off is considerably better than Butterworth, the ripple in the pass band may be tolerated in some applications.

- **Bessel Filters**

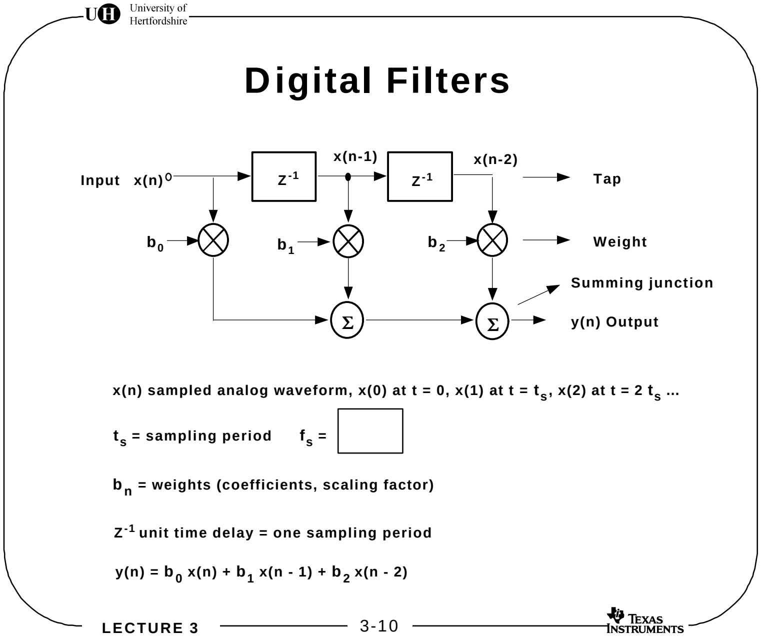
Bessel filters are designed using Bessel functions. They have a better phase response than either Butterworth or Chebychev. However, their roll-off is much less steep.

It is clear from the introduction and comparison of these filters that there is a trade-off between steeper roll-off and flat phase response. In most practical cases, the Butterworth filter seems to be a good compromise.

- **Filter Design Packages**

Designing filters is a calculation-intensive, formula-based task, which is perfectly suitable for computerization. Filter design software packages offer this functionality and increase the efficiency and effectiveness of designers. A filter may easily be designed by using one of these packages, simply by specifying a number of its features, such as type of filter, cut-off , pass-band ripple, required roll-off rate, etc.

The software package usually computes all of the component values, the amplitude, and the phase response of the filter. The designer can then assess the suitability of the design and change some of the specifications without actually building the filter. This is one of the major advantages of using such a software package.



- **Digital Filters**

We will now examine digital filters, starting with a simple example – a moving average filter (see above flow diagram). Let us first identify the major components of a digital filter.

- **The Input x(n)**

The input of a digital filter is a series of discrete samples obtained by sampling the input waveform. The sampling rate must meet the Nyquist criteria that we covered in our sampling lecture (highest frequency of input signal </ = 2 x sampling frequency). The term x(n) means the input at a time (n).

- **Z^{-1}**

Z^{-1} represents a time delay that is equal to the sampling period. This is also called a *unit delay*. Therefore, each z box delays the samples for one sampling period. In the diagram, this is shown by the input going into the delay box as x(n) and coming out as x(n-1). We see this because x(n) means the input at a time (n), and x(n-1) means the input at time (n-1). What actually happens is that x(n-1) is the previous input that has been saved in the memory of the DSP.

- **Filter Taps and Weights**

The output of each delay box is called a *tap*. Taps are usually fed into scalers which scale the value of the delayed sample to the required value by multiplying the input (or delayed input) by a coefficient. In the diagram, these are marked as b_0, b_1 and b_2. The scaling factor is called the *weight*. In mathematical terms, the weight is multiplied by the delayed input, so the output of the first tap is b_0*x(n). The next tap output will be b_1*x(n-1), and the output of the last tap is b_2*x(n-2).

- **Summing Junctions**

The output of the weights are fed into summing junctions, which add the weighted, delayed, forward-fed forward outputs from taps. So in this example, the output of the first summing junction is b_0*x(n) + b_1*x(n-1). At the next summing junction, this is added to the output of the final tap, giving b_0*x(n) + b_1*x(n-1) + b_2*x(n-2), which is the output.

- **The Output y(n)**

The output of a digital filter is a combination of a number of delayed and weighted samples, and is usually called y(n).

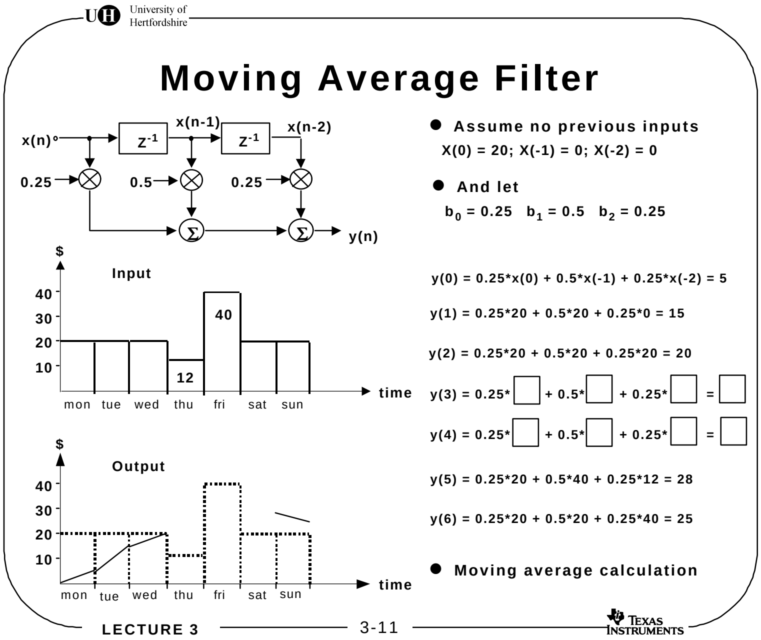
- **The Operation of Digital Filters**

In summary, the output is y(n) and the present sample is x(n). The previous samples would then be:

x(n-1) = one unit time delay
x(n-2) = two unit time delay

When x(n) arrives at the input, the taps are feeding the delayed samples to weights b_1 and b_2. Therefore sampling at any sampling instant, the value of the output can be calculated using the weighted sum of the current sample and two previous samples as follows:

$$y(n) = b_0*x(n) + b_1*x(n-1) + b_2*x(n-2)$$



- **The Practical Operation of the Filter**

Let us now observe the operation of the filter on our sample data by adding some numbers. Let's start with the coefficients.

Let $b_0 = 0.25$ $b_1 = 0.5$ and $b_2 = 0.25$.

These weights were selected to give a good averaging performance on our share prices. Later on in this lecture, we shall consider how filter weights are calculated from performance requirements and specifications by using a software package.

Now let's define some inputs. The samples chosen represent the value of a stock during the course of a week, the sample rate being one per day. For our purpose, time starts on Monday, so the sample on Monday is the value for x(0), Tuesday is x(1), and so on.

Day	Time Period	x(n)	Price $
Monday	0	x(0)	20
Tuesday	1	x(1)	20
Wednesday	2	x(2)	20
Thursday	3	x(3)	12
Friday	4	x(4)	40
Saturday	5	x(5)	20

These values are shown on the graph on the slide. Note that major variations took place on Thursday and Friday.

Let us now assume no previous inputs (meaning that x(-1) = 0, x(-2) = 0, etc.) and calculate the output of the filter. We know that:

$$y(n) = b_0*x(n) + b_1*x(n-1) + b_2*x(n-2)$$

So entering in the values of the coefficients gives:

$$y(n) = 0.25*x(n) + 0.5*x(n-1) + 0.25*x(n-2)$$

On Monday, the time period is 0, so we can work out y(0) as follows:

$$y(0) = 0.25*x(0) + 0.5*x(-1) + 0.25*x(-2)$$

$$y(0) = 0.25*20 + 0.5*0 + 0.2*5.0 = \mathbf{\$5}$$

Therefore, the output of the filter on Monday is $5. For Monday, the only input that has an effect on the output is Monday's input. For Tuesday, there is now one more input to consider. This is delayed input sample from Monday, now unit-delayed and weighted by 0.5. On Tuesday, the time period is 1, so:

$$y(1) = 0.25*x(1) + 0.5*x(0) + 0.25*x(-1) = 0.25*20 + 0.5*20 + 0.25*0 = \mathbf{\$15}$$

For Wednesday, all three inputs need to be considered. On Wednesday, the time period is 2, so:

$$y(2) = 0.25*x(2) + 0.5*x(1) + 0.25*x(0) = 0.25*20 + 0.5*20 + 0.25*20 = \mathbf{\$20}$$

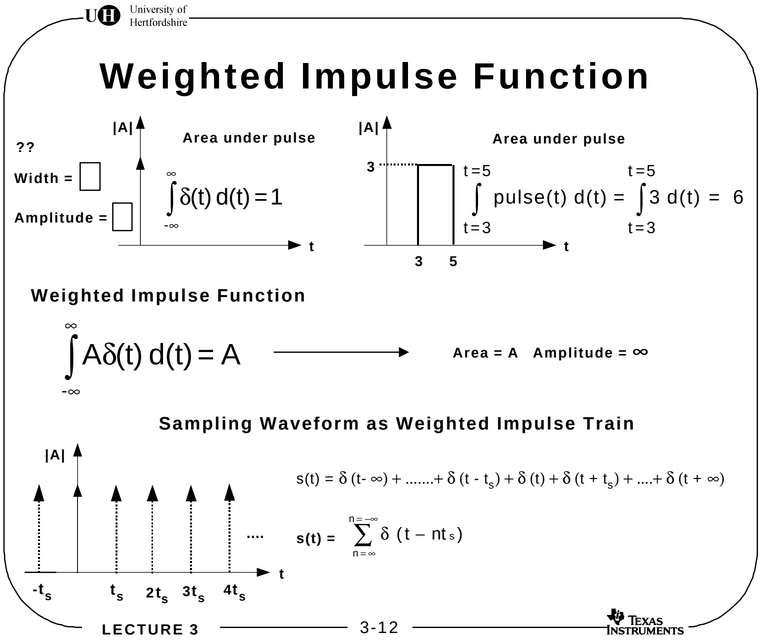
Now, work out the values for Thursday and Friday and fill in the blanks on the slide. Then, complete the missing section of the bottom graph.

y(3) = = **$**

y(4) = = **$**

You may be asking what this has to do with filters. Well, if you look at the completed graph, you will see that the system has filtered out some of the fast movement on Thursday and Friday to give a smoother graph. The filter is actually performing a moving average calculation. While it may not be much of a filter, it demonstrates the basic principles involved in digital filters by doing a number of calculations with past and present inputs to generate an output. In complex filters, other information can also be included in the calculation, but the basic principals remain the same.

These calculations also demonstrate the importance of MAC instruction in DSPs. Filter outputs consist of a series of multiplications and successive additions (called accumulate) operations, and the MAC instruction is designed to perform these as fast as possible.



- **Tools**

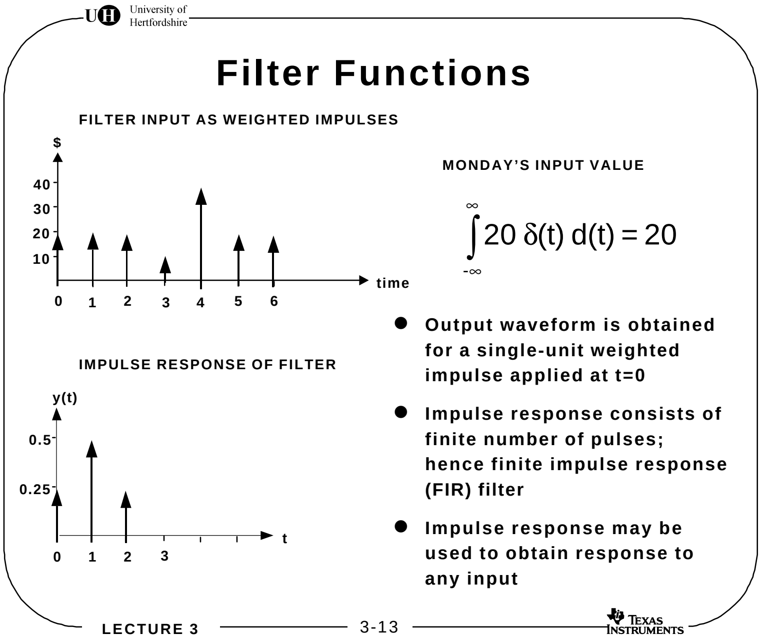
Before we consider more complex digital filters, let us first learn about some mathematical tools used in digital filtering. This will solidify our understanding of digital filters and provide a foundation for future learning of more complex subjects.

- **Impulse Function**

An *impulse* is defined as an idealized rectangular pulse of area 1.0, zero width, and infinite amplitude. It is typically expressed by an integral as shown on the above diagram. This is a general formula that allows us to calculate the area under any pulse.

- **Weighted Impulse Function**

Consider the pulse with an amplitude of 3 and a width of 2, as shown on the slide. Using the same integral to calculate the area under it, we find that it equals 6. A weighted impulse function is similar to this. It has an area of *A* and amplitude of infinity. It is represented by the integral as shown on the diagram. Obviously this is impossible in the real world, but the weighted impulse function is extensively used in digital signal processing to help explain DSP techniques. For example, an analog waveform can be represented as a multiplication of the analog signal with a periodic weighted impulse function whose frequency is equal to the sampling frequency.



- **Filter Input as Weighted Impulses**

Let us now use weighted impulses to represent our input signal (the stock values). Time (0) represents Monday. The stock value on Monday was $20. The weighted impulse function should evaluate to this value. The slide shows the integral for the weighted impulse function. In the same way, we can evaluate the weighted impulses to represent each input sample as shown on the first graph.

- **Impulse Response of the Filter**

Using the moving average filter again, if an input is applied at time t=0, with a value of 1, followed by all other inputs being 0, then the output at time t=0 will be 0.25 as the 1 goes down the first tap. Then, at time t=1, the 1 will go down the second tap, giving 0.5, and finally at time t=2, it will give an output of 0.25, as the 1 goes down the third tap.

This is shown on the bottom graph. This represents the impulse response of the filter. Some useful information can be extracted from this. First, we now have a list of coefficients for the filter, so if filter coefficients of a digital filter are unknown, then sending an impulse into the filter will reveal the coefficients for a FIR filter. Also, from this we can see that the input eventually drops out of the right side of the filter, at which point all future outputs are zero. This means that the length of the response to an impulse is finite. There are two main types of filters – those that produce a finite impulse response (FIR), and those that (ideally) produce an infinite impulse response (IIR).

We can use the weighted impulse response theory to help us with impulse response of our digital filter. The impulse response of a filter is defined as the waveform obtained for a single unity weighted impulse applied at time zero. Using this definition and the filter output equation, we can compute the impulse response.

$$y(n) = 0.25*x(n) + 0.5*x(n-1) + 0.25*x(n-2)$$

The impulse response is very important. Knowledge of a linear filter's impulse response allows its output to be determined due to any input by using a process called ***convolution*** in the time domain. Convolution is beyond the scope of the current discussion, however, in the frequency domain, convolutions become multiplications. The Fourier and inverse Fourier transforms (discussed in the next chapter) are used to move between the time and frequency domains. To compute a filter's output in the frequency domain, the Fourier transform of the impulse response is taken resulting in the filter's *transfer function* or *frequency response.* This transfer function is multiplied by the Fourier transform of the input to obtain the frequency domain representation of the output. Finally, the output is inverse Fourier transformed to obtain the time domain output. Fourier techniques lend themselves well to DSP because a computationally efficient algorithm, called the *Fast Fourier Transform* (FFT) exists. It is faster to perform convolutions using the FFT than by direct techniques.

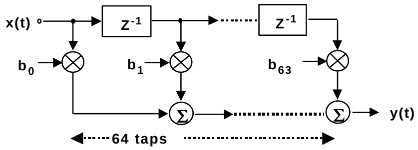
As a final note, it should be stressed that the impulse response is not a digital only concept and is applied as well to the analog world. When one strikes a bell with a hammer the sound that is heard is an approximation of the mechanical impulse response of the bell. The hammer blow models an impulse. The bell's ring is the system output.

- **Finite Impulse Response (FIR) filter**

The type of filter just discussed is classified as a *Finite Impulse Response* or FIR filter. It is called this because its response to a single impulse is finite. After a defined period of time (determined by the number of taps) following the impulse the output of the filter will be zero. This type of filter is also referred to as a *transversal*, *feedforward*, *all-zero,* or *moving average* (MA) filter. There is another type of filter called the *Infinite Impulse Response* (IIR) filter which will be examined after the next FIR filter example.

FIR Filters

- An FIR Filter with a steeper roll-off:



- A more realistic filter designed using a software filter design package
- Specifications:
  - Cut-Off Frequency = 975 Hz
  - Stop Band Attenuation > 80dB
  - Sharp Roll-Off
- Filter with 64 taps
- 64 different gain values
- This filter is used in our demonstration

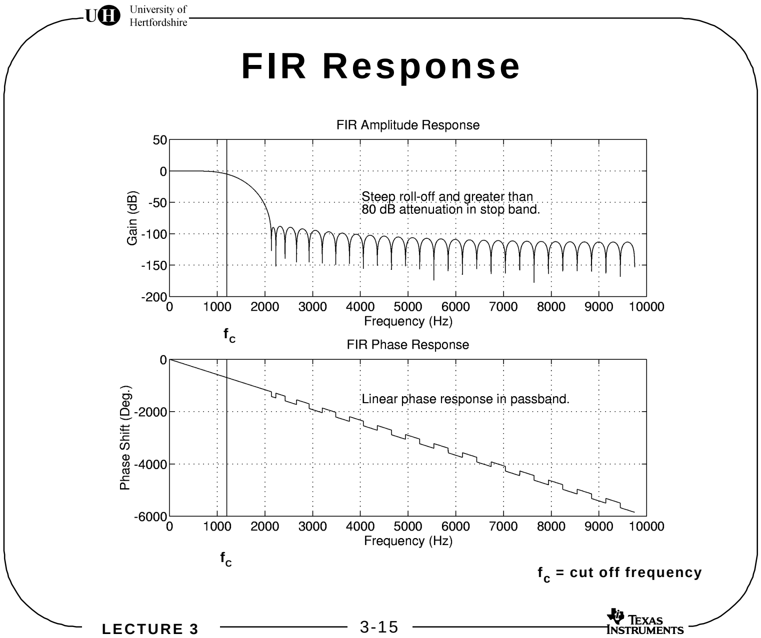
LECTURE 3 3-14

- **Longer FIR Filters**

Let us now consider a longer FIR filter – one that uses 64 taps instead of three. It requires 64 filter coefficients and the 63 delay units so the filter can operate on the current input plus the previous 63 samples. It is a low pass filter with a 1200 Hz cut-off frequency and a very steep roll-off in its transition region. Its attenuation is 3 dB at 1200 Hz and greater than 80 dB at frequencies above 1950 Hz. Note that as the number of taps increases so does the computational complexity of the filter. In general, longer filters need more computational clock cycles than shorter filters. One advantage of FIR filter over the IIR filter discussed in the next section is that a FIR filter is *always* stable.

You will come across its structure and specifications in our demonstration. Our tone generator can generate a number of tones. We are using this filter to filter out some high-pitch tones. When you listen to the tone generator with and without the filter, you will notice the difference.

Are you thinking about how the 64 different tap weights were calculated? We have used a software design package which automatically calculates the weights and more. We will see later in this lecture that the process of calculating these coefficients is not at all difficult.

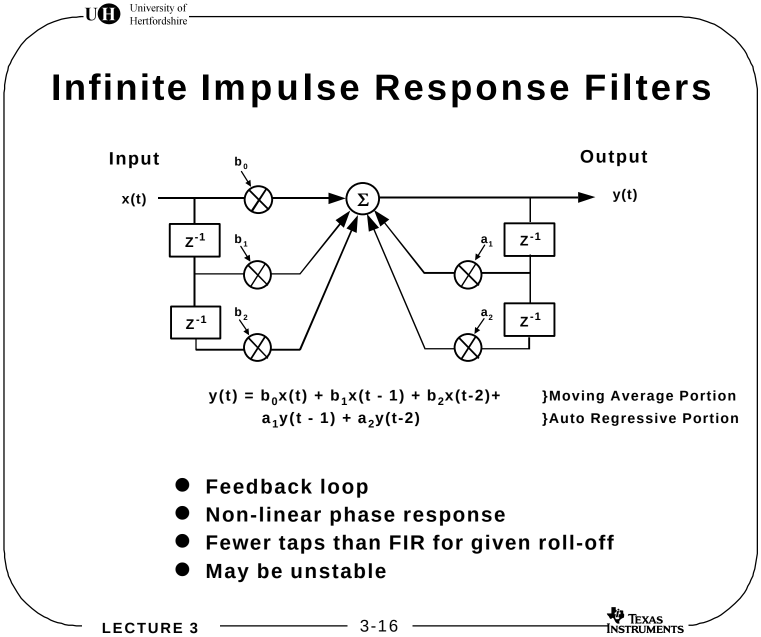


- **Amplitude Response**

The top graph shows the amplitude response of the filter. It is a low-pass filter because it passes frequencies from 0 to the cut-off frequency and attenuates the high frequencies. The stop-band attenuation is greater than 80dB. The transition region is very steep due to the high filter order. Implementing high-order (and hence high roll-off rate) filters is generally easier using DSP rather than analog processing, especially at lower frequencies.

- **Phase Response**

The bottom graph on the diagram shows the phase response of the filter. This phase response is linear in the pass-band. In the stop-band, the 180 degree phase reversals at the amplitude response nulls makes the phase response in this region piecewise linear.



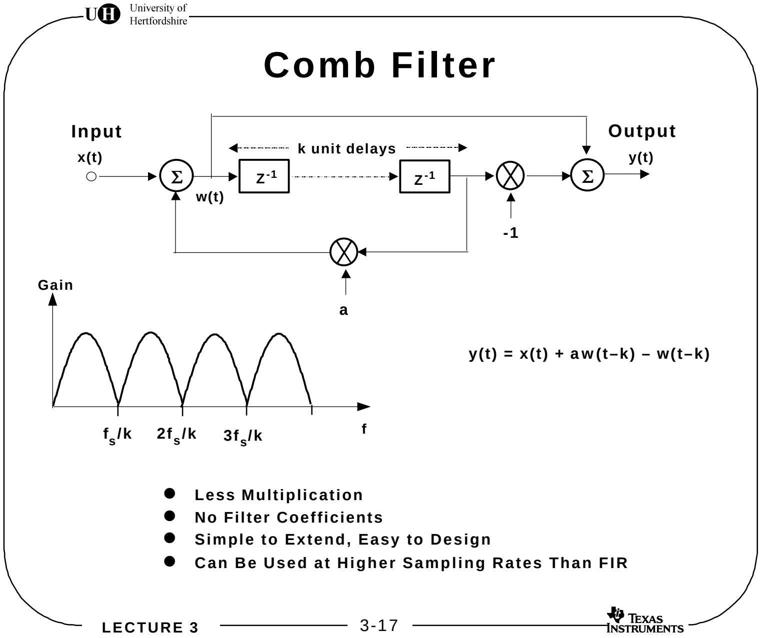
- **Infinite Impulse Response (IIR) Filters**

With a simple addition, the FIR filter can be transformed into an IIR filter. Refer to the slide above. In addition to the b coefficients from the FIR filter, a set of coefficients and unit delays are added to feedback the filter's output. These added coefficients are the a coefficients. The result is an IIR or *Autoregressive-Moving Average* (ARMA) filter. One can look at the FIR filter configurations discussed previously as ARMA filters with the a coefficients set to zero. It is also possible to have an IIR filter with all the b coefficients set to zero. This type of filter is referred to as an *all-pole*, *feedback*, or *autoregressive* (AR) filter.

The impulse response of an IIR filter has infinite length, hence the name infinite impulse response. The feedback loop makes it possible for an IIR filter to be unstable. It is possible to check for this instability during the design process, but sometimes a filter that is stable on paper may become unstable in practice due to roundoff and truncation in the DSP hardware. It is important to examine stability issues closely when working with IIR filters. In some cases, the instability conditions of an IIR filter can be used to advantage in designing oscillators.

- **Comparison Between FIR and IIR Filters**

Some comparisons are warranted between FIR and IIR filters. It is easy to design a FIR filter that has linear phase response in the pass-band; all that is required is that the impulse response be symmetric. By definition, a stable IIR filter *cannot* have linear phase. FIR filters are always stable while IIR filters *can* be unstable. FIR filters generally have more elements than an IIR filter for a given frequency response specification assuming that linear phase is unimportant.



- **Comb filters**

These are special type filters. They have a comb-like frequency response plot, hence the name. The transparency shows a typical comb filter and its equation.

Comb filters may have no coefficients. This means that they typically need much fewer multiplications. Basically the filter consists of unit delays and adders. This makes the design process very easy. Implementation is simpler as well. This feature also makes them ideal candidates for silicon implementation. Comb filters are commonly found on the output stage of sigma-delta ADCs chips.

Comb filters are also very useful in audio applications. Our demonstration for this lecture uses a comb filter to filter out a single tone.

University of Hertfordshire

DSP and Digital Filters

Advantages of Digital Filters

- Programmable
- It is possible to implement adaptive filters that change coefficients under certain conditions

Why use DSP for digital filter implementation?

REMEMBER: A = B*C + D

$$y(n) = a_0\,x(n) + a_1\,x(n-1) + a_2\,x(n-2)$$

LECTURE 3 3-18 TEXAS INSTRUMENTS

- **Advantages of Digital Filters**

One of the major advantages of digital filters is that they are programmable. To change the cut-off frequency, the roll-off rate, or the phase response, all one must do is change a few coefficients. We can quite easily make major changes, such as converting a low-pass filter into a high-pass filter.

The idea of changing filter characteristics by changing a few coefficients opens up even wider possibilities. It is possible to design adaptive filters that adapt themselves to changing conditions. For adaptive filters, a mechanism must be designed to change the coefficients of the filter in accordance with changing conditions. Such filters are very useful in modems. Since the properties of telephone lines change continuously, adaptive filters offer the ideal solution for these environments.

- **Why Use DSP for Digital Filter Implementation?**

DSPs are very efficient in performing successive multiply and add (MAC) operations. Most DSPs can perform a single-cycle multiply and add operation. Digital filters require delays and fast multiply and add operations. DSPs can provide both, making them the ideal medium for the implementation of digital filters. The increasing processing power of DSPs has made possible the implementation of complex digital filter structures, such as adaptive filters.

University of Hertfordshire

Performance Issues

Noise in Digital Filters

Signal Quantization

- Noise introduced is proportional to the number of bits that the conversion uses

Coefficient Quantization

- Coefficients determine the behavior of filters
- More significant in IIR

Truncation

- 0.64 x 0.73 = 0.4672 which truncates to 0.46
- Double-width product registers and accumulators help reduce truncation errors

Internal Overflow

0010 + 1111 = 10001 → SATURATE 1111

→ OVERFLOW

Dynamic-Range Constraints

16 bit → $20 \log_{10} (2^{16}) = 96\text{dB}$

32 bit → $20 \log_{10} (2^{32}) = 192\text{dB}$

LECTURE 3 3-19 TEXAS INSTRUMENTS

- **Noise in Digital Filters**

There are five major sources of error in digital filters. In IIR filters, these errors are exaggerated due to the feedback. FIR filters are feed-forward circuits, so errors appear only once.

- **Signal Quantization**

ADCs introduce quantization errors to the original signal. DACs do not eliminate this error in their reverse conversion process. If the resolution of a DAC is smaller than the DSP, there will be a small error. For example, a 16-bit DSP using a 10-bit DAC cannot produce an output signal greater than 10 bits, but this type of system can reduce the truncation error compared with a 10-bit DSP and a 10-bit DAC (details to follow).

- **Coefficient Quantization**

Coefficients of filters are calculated as analog values. Within DSPs, 16- or 32-bit representations are used. Such conversion inevitably introduces some errors. The 32-bit representations are clearly more accurate than 16-bit, but even with 16-bit representation, errors are quite small. However, coefficient quantization errors must be taken into account, particularly in design and implementation of IIR filters, because the effect of feedback increases the effect of such errors, and may even cause instability. Normally, this is only a problem in fixed-point DSPs, since floating-point DSPs with 32-bit accuracy apply 32 bits to all numbers. On a fixed-point DSP, a small number will be preceded by a large number of zeros, which will reduce the accuracy of the coefficient.

- **Truncation**

When two 16-bit numbers are multiplied, the result is 32 bits wide. This can be demonstrated with decimal multiplication (0.64 x 0.73 = 0.4672). Here, two numbers, each of two decimal places, produce a result with four decimal places. For this reason, most fixed-point DSPs use a product register and accumulator, which is double the width of all other registers. So, during multiplication and addition, 32-bit precision is maintained. The problem comes when this result must be stored in memory. It is possible to store all 32 bits, but this increases costs and computation time. Usually the most-significant 16 bits are used, and the least-significant 16 bits are truncated. The error due to this truncation is only in the 16th bit, and is less than 0.001%. By having 32-bit accuracy, this truncation is only done once for each output. If 16 bits were used, the truncation would be necessary for each tap, so in the previous example, it would happen 161 times, resulting in a much larger error.

- **Internal Overflow**

When you add two 4-bit binary numbers as shown on the diagram, the result can be 5 bits. In a 16-bit DSP, it could overflow to 17 bits. A 4-bit processor would have to discard the 5th bit. This is called overflow. In the same way, underflow is also possible as a result of adding two negative numbers. Both can cause errors in digital filters. DSPs allow designers to switch to *saturation mode*. If the result of an operation is greater than the largest possible positive value, the DSP's accumulator will saturate to the largest positive number it can store. Similarly, if the result is greater than the largest possible negative value, the output will saturate to the largest negative number it can store. Such situations should be avoided in digital filter design. This is one of the reasons why fixed-point DSPs are more difficult to program.

- **Dynamic Range Constraints**

The dynamic range of a device is directly proportional to the word width of DSPs. On a 16-bit device, the number of different values is 2^{16}. From this, the dynamic range for a 16-bit device can be calculated by converting 2^{16} to decibels, which is $20\log_{10}(2^{16})$ = 96dB. During arithmetical calculations this is extended to 192dB, since the product registers and accumulator have 32 bits. Such dynamic range is sufficient for most digital filter applications.

University of Hertfordshire

Digital Filter Design

- Automates design task by software
- Design software requires information such as:
  - Pass Band, Stop Band, Transition Region
  - Ripple in Pass Band
  - Required Roll-Off
- Design Software Generates:
  - Number of Taps
  - Coefficients Required
  - DSP Specific Assembly Code
  - Response Plots
    - Gain
    - Phase
    - Impulse
- Enables evaluation of design *before* implementation
- A low cost evaluation board such as DSK can be used for actual testing

LECTURE 3 3-20 Texas Instruments

- **Automation of Design**

Digital filters are now designed almost exclusively by appropriate design software. There are a number of commercially available digital filter design packages of varying complexity and functionality. Most packages are quite intuitive.

- **Design Process**

Design software requires the specification of the filter from the designer. The designer should provide information such as the pass band, stop band, transition region, and the required roll-off rate in the transition region. Then, the software will do all of the computation necessary to calculate the number of taps and coefficients required, and will even produce DSP-specific assembly code for the implementation of the filter. Quite commonly, most packages ask the user whether the number of taps required to implement the filter is acceptable. As the number of taps increases, the requirements on the processor resources increase as well. The designer may wish to trade off some of the performance for a less complex filter before the design is done.

On completion of the design, all necessary gain, phase, and impulse response plots are produced. This is one of the primary advantages of using a software design package. It allows the designers to evaluate the performance of the filter before the actual implementation.

University of Hertfordshire

Summary

- Filters are used for frequency selection
- Low and high pass analog filters
- Performance
  - Pass Band Ripple, Roll-Off and Phase Response
- Digital finite impulse response (FIR) filters
- Digital infinite impulse response (IIR) filters
- Advantages of Digital Filters
  - Programmable
  - Adaptive Filters
- DSP makes digital filter implementation easier

LECTURE 3 3-21 TEXAS INSTRUMENTS

- **Filters**

Filters are used in selecting certain frequencies in waveforms. A waveform usually contains a group of desirable frequencies and a group that we would like to eliminate.

- **Low and High-pass Analog Filters**

Analog filters use resistors, capacitors and inductors, but we have only discussed filters with resistors and capacitors, which are the most common. Filters are usually classified according to the group of frequencies they pass or attenuate. Hence, the names high-pass, band-pass, band-stop, and all-pass.

- **Performance**

Filter performance is commonly defined in terms of pass-band ripple, roll-off rate in transition region, and phase response. Non-linear phase response and high ripples in the pass band are often undesirable.

- **FIR Filters**

Digital filters operate on a basis of samples rather than continuous signals. The fundamental concepts of digital filter operation are well represented in FIR filters. Such filters are feed-forward circuits with a number of unit-delay elements. The filter impulse response consists of a finite of number of weighted impulses.

- **Advantages of Digital Filters**

The primary advantage of digital filters is that they are programmable (by changing their coefficients) and repeatable. This makes them easy to configure, so with DSP implementation, a new filter can be realized simply by changing a program. This is in contrast with analog filters, where even a minor change may require a soldering iron.

Adaptive filters *adapt* to external conditions. Through a suitable mechanism, external conditions change the filter coefficients, and therefore create a more desirable filter for the new conditions. Such filters are used extensively in data communications, where conditions in a transmission medium change constantly.

- **DSPs for Digital Filters**

DSPs are designed specifically for applications such as digital filters. Since DSPs are highly optimized for performing single-cycle multiply and add operations, they provide a very suitable engine for digital filter implementation.

REFERENCES

Ahmed, H. and Spreadbury, P. J. [1978]. *Electronics for Engineers*, Cambridge University Press, Cambridge, UK

Bateman, A. and Yates, W. [1988]. *Digital Signal Processing Design*, Pitman Publishing, London, UK

Chu, S. and Burrus, C. S. [Nov 1984]. "Multirate Filter Design Using Comb Filters," *IEEE Transactions on Circuits and Systems*, Vol CAS-31, No 11, pp. 913-924

Cowan, C. F. N. and Grant, P. M. [1985]. *Adaptive Filters*, Prentice-Hall, Englewood Cliffs, NJ

Hamming, R. W. [1989]. *Digital Filters*, Prentice-Hall, Englewood Cliffs, NJ

Haykin, Simon. [1991]. *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs, NJ

Jackson, L. B. [1989]. *Digital Filters and Signal Processing*, Second Edition, Kluwer Academic Publishers, Norwell, MA

Jury, E. I. [1964]. *Theory and Application of the Z-Transform Method*, John Wiley, New York

Oppenheim, A. V. and Schafer, R. W. [1975 and 1988]. *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ

Papamichalis, Panos (ed.) [1991]. *Digital Signal Processing with the TMS320 Family, Volume 3*, Prentice-Hall, Englewood Cliffs, NJ

Parks, T. W. and Burrus, C. S. [1987]. *Digital Filter Design*, Wiley and Sons, New York

Treichler, J. R.; Johnson, C. R. and Larimore, M. G. [1987]. *Theory and Design of Adaptive Filters*, Wiley and Sons, New York

Zaks, R. [1981]. *From Chips to Systems*, Sybex, California.