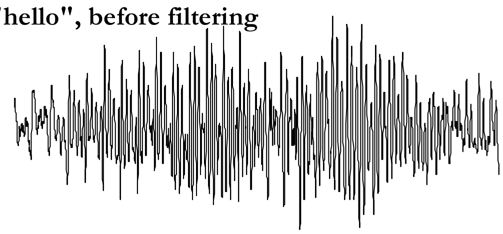
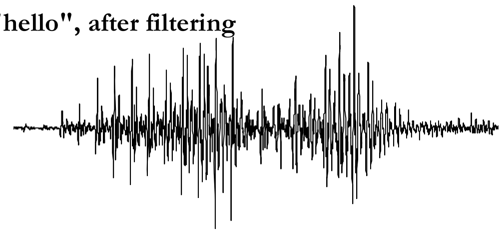
Lecture 10 objectives are to:

- Introduce basic principles involved in digital filtering,
- Define the Z Transform and use it to analyze filters,
- Develop digital filter implementations



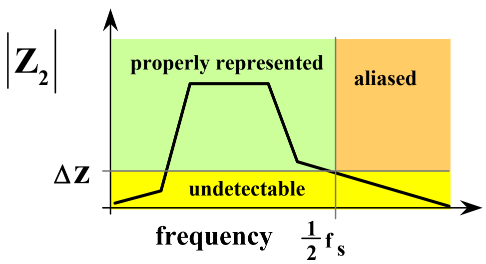


Basic Principles

$\mathbf{x_c(t)}$ is a continuous analog signal. $\mathbf{f_s}$ is the sample rate

$$\mathbf{x(n) = x_c(nT)} \qquad \text{with } -\infty < \mathbf{n} < +\infty.$$

There are two types of approximations associated with the sampling process.
finite precision of the ADC
finite sampling frequency.



To prevent aliasing there should be no measurable signal above 0.5$\mathbf{f_s}$.

A **causal** digital filter calculates
y(n) from **y(n-1), y(n-2),...** and **x(n), x(n-1), x(n-2),...**
not future data (e.g., **y(n+1)**, **x(n+1)** etc.)
A **linear** filter is constructed from a linear equation.
A **nonlinear** filter is constructed from a nonlinear equation.
One nonlinear filter is the median.
A **finite impulse response** filter (FIR) relates **y(n)** only in terms of **x(n), x(n-1), x(n-2),...**

$$\mathbf{y(n) = \frac{x(n) + x(n-3)}{2}}$$

An **infinite impulse response** filter (IIR) relates **y(n)** in terms of both **x(n), x(n-1),..., and y(n–1), y(n-2),...**

$$y(n) = (113 \bullet x(n) + 113 \bullet x(n-2) - 98 \bullet y(n-2))/128$$

The definition of the Z-Transform:

$$\boxed{\mathbf{X(z) = Z[x(n)] \equiv \sum_{n=-\infty}^{\infty} x(n)\, z^{-n}}}$$

Consider the Laplace Transform

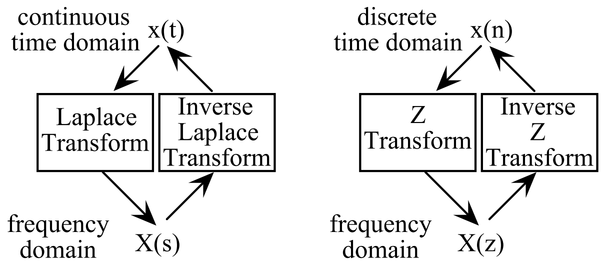


Fig 5.1 A transform is used to study a signal in the frequency domain.

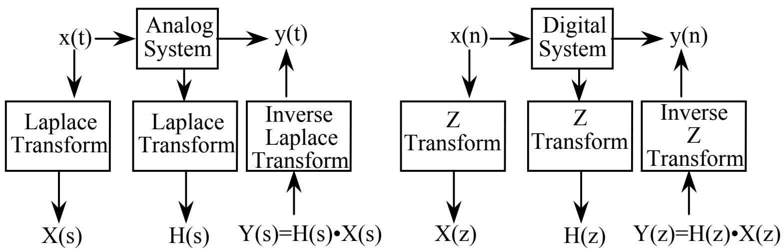


Figure 5.2. A transform can also be used to study a system in the frequency domain.

The gain = $|\mathbf{H(s)}|$ at $\mathbf{s = j\ 2\pi f}$, for all frequencies, **f**.
The phase = angle($\mathbf{H(s)}$) at $\mathbf{s = j\ 2\pi f}$.

The gain and phase of a digital system is specified in its transform, **H(z) = Y(z)/X(z)**.

from DC to $\frac{1}{2}\ \mathbf{f_s}$

One can use the definition of the Z-Transform to prove that:

$$\mathbf{Z[x(n-m)] = z^{-m}\ Z[x(n)] = z^{-m}\ X(z)}$$

For example if **X(z)** is the Z-Transform of **x(n)**,

then $\mathbf{z^{-2} \bullet X(z)}$ is the Z-Transform of **x(n-2)**.

$$\mathbf{H(z) \equiv \frac{Y(z)}{X(z)}}$$

To find the response of the filter, let **z** be a complex number on the unit circle

$$\mathbf{z \equiv e^{j2\pi f/f_s}} \qquad \text{for } 0 \le \mathbf{f} < \frac{1}{2}\ \mathbf{f_s}$$

or

$$\mathbf{z = \cos(2\pi f/f_s) + j\ \sin(2\pi f/f_s)}$$

Let

$$\mathbf{H(f) = a + bj} \qquad \text{where } \mathbf{a} \text{ and } \mathbf{b} \text{ are real numbers}$$

The gain of the filter is the complex magnitude of **H(z)** as **f** varies from 0 to $\frac{1}{2}\ \mathbf{f_s}$.

$$\boxed{\mathbf{Gain \equiv |H(f)| = \sqrt{a^2 + b^2}}}$$

The phase response of the filter is the angle of **H(z)** as **f** varies from 0 to $\frac{1}{2}$ $\mathbf{f_s}$.

$$\textbf{Phase} \equiv \textbf{angle[H(f)]} = \tan^{-1}\frac{b}{a} \qquad (13)$$

5.3 MACQ

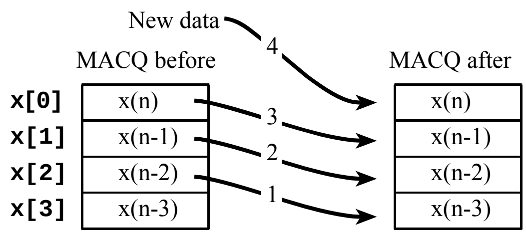


Figure 5.8. When data is put into a multiple access circular queue, the oldest data is lost

$$d(n) = \frac{x(n)+3x(n\text{-}1)\text{-}3x(n\text{-}2)\text{-}x(n\text{-}3)}{\Delta t}$$

```
short x[4]; // MACQ (mV)
short d;    // derivative(V/s)
void ADC3_Handler(void){
  ADC_ISC_R = ADC_ISC_IN3;      // acknowledge ADC sequence 3 completion
  x[3] = x[2];  // shift data
  x[2] = x[1];  // units of mV
  x[1] = x[0];
  x[0] = (375*(ADC_SSFIFO3_R&ADC_SSFIFO3_DATA_M))>>7; // in mV
  d = x[0]+3*x[1]-3*x[2]-x[3]; // in V/s
  Fifo_Put(d);  // pass to foreground
}
```

Program 5.3. Software implementation of first derivative using a multiple access circular queue.

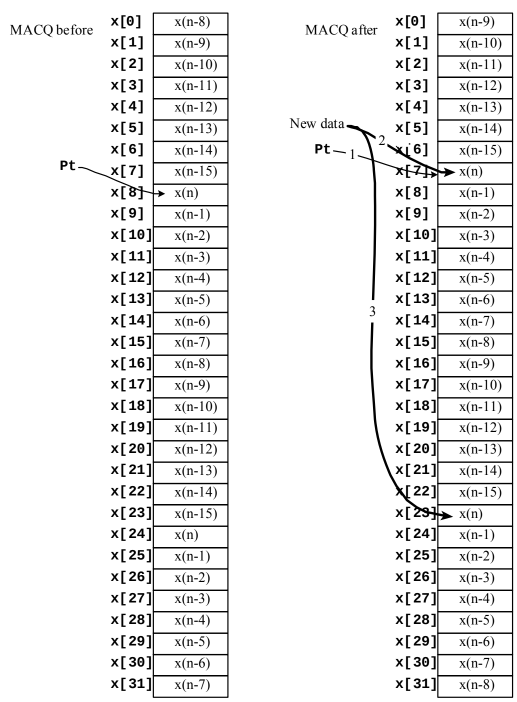


Figure 5.9. When data is put into a multiple access circular queue, the oldest data is lost.

```
unsigned short x[32];     // two copies
unsigned short *Pt;       // pointer to current
unsigned short Sum;       // sum of the last 16 samples
void LPF_Init(void){
  Pt = &x[0]; Sum = 0;
}
// calculate one filter output
// called at sampling rate
// Input: new ADC data
// Output: filter output, DAC data
unsigned short LPF_Calc(unsigned short newdata){
  Sum = Sum - *(Pt+16);     // subtract the one 16 samples ago
  if(Pt == &x[0]){
    Pt = &x[16];            // wrap
  } else{
    Pt--;                   // make room for data
  }
  *Pt = *(Pt+16) = newdata; // two copies of the new data
  return Sum/16;
}
```

Program 5.4. Digital low pass filter implemented by averaging the previous 16 samples (cutoff = $f_s/32$).

5.4. Using the Z-Transform to Derive Filter Response

Although this filter appears to be simple, we can use it to implement a low-Q 60 Hz notch.

$$y(n) = (x(n)+x(n-3))/2$$

Again we take the Z-Transform of both:

$$Y(z) = (X(z) + z^{-3}X(z))/2$$

Next we rewrite the equation in the form of **H(z)=Y(z)/X(z)**.

$$H(z) \equiv Y(z)/X(z) = ½ (1 + z^{-3})$$

We can to determine the gain and phase response of this filter.

$$H(f) = ½ (1 + e^{-j6\pi f/fs}) = ½ (1 + \cos(6\pi f/f_s) - j \sin(6\pi f/f_s))$$

$$\textbf{Gain} \equiv |H(f)| = ½ \text{ sqrt}((1 + \cos(6\pi f/f_s))^2 + \sin(6\pi f/f_s)^2))$$

$$\textbf{Phase} \equiv \text{angle}(H(f)) = \tan^{-1}(-\sin(6\pi f/f_s)/(1 + \cos(6\pi f/f_s))$$

```
short x[4]; // MACQ
void ADC3_Handler(void){ short y;
  ADC_ISC_R = ADC_ISC_IN3;      // acknowledge ADC sequence 3 completion
  x[3] = x[2];  // shift data
  x[2] = x[1];  // units, ADC sample 0 to 1023
  x[1] = x[0];
  x[0] = ADC_SSFIFO3_R&ADC_SSFIFO3_DATA_M; // 0 to 1024
  y = (x[0]+x[3])/2; // filter output
  Fifo_Put(y);       // pass to foreground
}
```

Program 5.5. If the sampling rate is 360 Hz, this filter rejects 60 Hz.

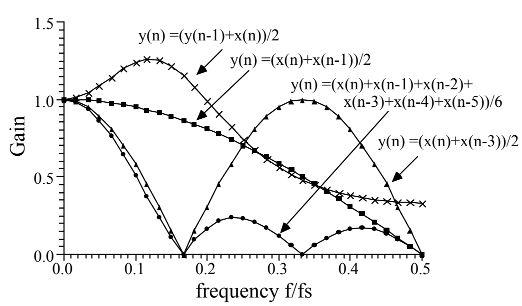


Figure 5.10. Gain versus frequency response for four simple digital filters.

5.5. IIR Filter design

There are two objectives for this example

show an example of a digital notch filter,
demonstrate the use of fixed-point math.

60 Hz noise is a significant problem in most data acquisition systems. The 60 Hz noise reduction can be accomplished:

1) Reducing the noise source, e.g., shut off large motors;
2) Shielding the transducer, cables, and instrument;
3) Implement a 60 Hz analog notch filter;
4) Implement a 60 Hz digital notch filter.

analog condition	digital condition	consequence
zero near $s=j2\pi f$ line	zero near $z=e^{j2\pi f/fs}$	low gain near the zero
pole near $s=j2\pi f$ line	pole near $z=e^{j2\pi f/fs}$	high gain near the pole
zeros in conjugate pairs	zeros in conjugate pairs	the output y(t) is real
poles in conjugate pairs	poles in conjugate pairs	the output y(t) is real
poles in left half plane	poles inside unit circle	stable system
poles in right half plane	poles outside unit circle	unstable system
pole near a zero	pole near a zero	high Q response

Table Analogies between the analog and digital filters.

It is the 60 Hz digital notch filter that will be implemented in this example. The signal is sampled at $\mathbf{f_s}$=480 Hz. We wish to place the zeros (gain=0) at 60 Hz, thus

$$\theta = \pm 2\pi \bullet \frac{60}{f_s} = \pm \pi/4$$

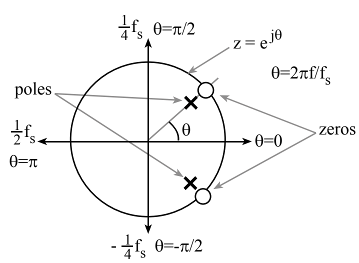


Figure 5.13. Pole-zero plot of a 60 Hz digital notch filter.

The zeros are located on the unit circle at 60 Hz

$$z_1 = \cos(\theta) + j \sin(\theta) \qquad z_2 = \cos(\theta) - j \sin(\theta)$$

To implement a flat pass band away from 60 Hz the poles are placed next to the zeros, just inside the unit circle. Let α define the closeness of the poles where $0 < \alpha < 1$.

$$p_1 = \alpha\, z_1 \qquad p_2 = \alpha\, z_2$$

for $\alpha = 0.75$

The transfer function is

$$H(z) = \prod_{i=1}^{k} \frac{(z - z_i)}{(z - p_i)} = \frac{(z - z_1)(z - z_2)}{(z - p_1)(z - p_2)}$$

which can be put in standard form (i.e., with terms 1, z^{-1}, z^{-2} ...)

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

$$y(n) = x(n) + x(n\text{-}2) \ \ \text{-}(49 * y(n\text{-}2))/64$$

$$H(z) = \frac{1 + z^{-2}}{1 + \frac{49}{64} z^{-2}}$$

At z=1 this reduces to

$$\text{DC Gain} = \frac{2}{1 + \frac{49}{64}} = \frac{128}{64 + 49} = \frac{128}{113}$$

$$y(n) = (113 \bullet x(n) + 113 \bullet x(n\text{-}2) \ - \ 98 \bullet y(n\text{-}2))/128$$

```
long x[3]; // MACQ for the ADC input data
long y[3]; // MACQ for the digital filter output
void ADC3_Handler(void){
  ADC_ISC_R = ADC_ISC_IN3;   // acknowledge ADC sequence 3 completion
  x[2] = x[1]; x[1] = x[0];  // shift data
  y[2] = y[1]; y[1] = y[0];
  x[0] = ADC_SSFIFO3_R&ADC_SSFIFO3_DATA_M; // 0 to 1024
  y[0] = (113*(x[0]+x[2])-98*y[2])/128; // filter output
  Fifo_Put((short)y[0]);       // pass to foreground
}
```

Program 5.7. If the sampling rate is 240 Hz, this filter rejects 60 Hz.

The "Q" of a digital notch filter is defined to be

$$Q \equiv \frac{f_c}{\Delta f}$$

where f_c is the center or notch frequency, and Δf frequency range where is gain is below 0.707 of the DC gain.

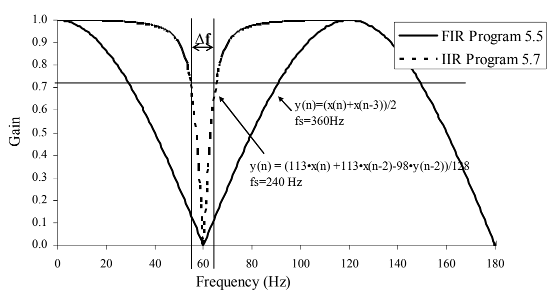


Figure 5.14. Gain versus frequency response of two 60 Hz digital notch filters.

Show the two spreadsheets DigitalNotchFilter.xls (**DigitalFilterDesign.xls**)