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

"hello", before filtering

"hello", after filtering

Basic Principles

\( x_c(t) \) is a continuous analog signal. \( f_s \) is the sample rate

\[ x(n) = x_c(nT) \]

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 \( 0.5f_s \).

A** causal** digital filter calculates
\( y(n) \) from \( x(n-1), y(n-2), \ldots \) and \( x(n), x(n-1), x(n-2), \ldots \)
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 \( x(n), x(n-1), x(n-2), \ldots \)

\[ y(n) = \frac{x(n) + x(n-3)}{2} \]

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

\[ y(n) = \frac{(113\cdot x(n) + 113\cdot x(n-2) - 98\cdot y(n-2))}{128} \]

The definition of the Z-Transform:

\[ X(z) = Z[x(n)] = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \]
Consider the Laplace Transform

\[ x(t) \quad \text{Laplace Transform} \quad X(s) \]

\[ x(n) \quad \text{discrete time domain} \quad X(z) \]

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

---

\[ \text{Laplace Transform} \quad X(s) \]

\[ \text{Inverse Laplace Transform} \quad x(t) \quad \text{Analog System} \quad y(t) \]

\[ \text{Z Transform} \quad X(z) \quad \text{Digital System} \quad y(n) \]

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

The gain = \(|H(s)|\) at \(s = j2\pi f\), for all frequencies, \(f\).
The phase = angle\((H(s))\) at \(s = j2\pi f\).

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

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

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

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

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

### 5.3 MACQ

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

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

```c
short x[4]; // MACQ (mV)
short d;    // derivative (V/s)
void ADC3_Handler(void){
    // 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]){            // wrap
        Pt = &x[16];
    } 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.
\[ y(n) = \frac{x(n) + x(n-3)}{2} \]
Again we take the Z-Transform of both:
\[ Y(z) = \frac{X(z) + z^{-3}X(z)}{2} \]
Next we rewrite the equation in the form of \[ H(z) = \frac{Y(z)}{X(z)} \].
\[ H(z) = \frac{1}{2} \left( 1 + z^{-3} \right) \]
We can to determine the gain and phase response of this filter.
\[ G = \sqrt{\left( \frac{1 + \cos(6\pi f_s/f_s)}{1 + \cos(6\pi f_s/f_s)} \right)^2 + \sin^2(6\pi f_s/f_s)} \]
\[ \Phi = \tan^{-1}\left( \frac{-\sin(6\pi f_s/f_s)}{1 + \cos(6\pi f_s/f_s)} \right) \]
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.

by Jonathan W. Valvano
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.

<table>
<thead>
<tr>
<th>analog condition</th>
<th>digital condition</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>zero near s=jω line</td>
<td>zero near z=e^(jω)</td>
<td>low gain near the zero</td>
</tr>
<tr>
<td>pole near s=jω line</td>
<td>pole near z=e^(-jω)</td>
<td>high gain near the pole</td>
</tr>
<tr>
<td>zeros in conjugate pairs</td>
<td>zeros in conjugate pairs</td>
<td>output y(t) is real</td>
</tr>
<tr>
<td>poles in conjugate pairs</td>
<td>poles in conjugate pairs</td>
<td>output y(t) is real</td>
</tr>
<tr>
<td>poles in left half plane</td>
<td>poles inside unit circle</td>
<td>stable system</td>
</tr>
<tr>
<td>poles in right half plane</td>
<td>poles outside unit circle</td>
<td>unstable system</td>
</tr>
<tr>
<td>pole near a zero</td>
<td>pole near a zero</td>
<td>high Q response</td>
</tr>
</tbody>
</table>

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 f_s=480 Hz. We wish to place the zeros (gain=0) at 60 Hz, thus

\[ \theta = \pm 2\pi \times \frac{60}{f_s} = \pm \pi/4 \]

The zeros are located on the unit circle at 60 Hz
\[ z_1 = \cos(\theta) + j \sin(\theta) \]
\[ z_2 = \cos(\theta) - j \sin(\theta) \]

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 < α < 1.

\[ p_1 = \alpha z_1 \]
\[ p_2 = \alpha z_2 \]

for \( \alpha = 0.75 \)

The transfer function is

by Jonathan W. Valvano
\[ H(z) = \prod_{i=1}^{k} \frac{(z - z_i)}{(z - p_i)} = \frac{(z - z_1)(z - z_2)}{(z - p_1)(z - p_2)} \]

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

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

\[ y(n) = x(n) + x(n-2) - (49*y(n-2))/64 \]

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

At \( z=1 \) this reduces to

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

\[ y(n) = (113\times x(n) + 113\times x(n-2) - 98\times y(n-2))/128 \]

```c
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\times x[0]+113\times x[2]-98\times 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

\[ Q = \frac{f_C}{\Delta f} \]

where \( f_C \) is the center or notch frequency, and \( \Delta f \) 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.](DigitalFilterDesign.xls)

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