Today

- Data acquisition
- Digital filters and signal processing
  - Filter examples and properties
  - FIR filters
  - Filter design
  - Implementation issues
  - DACs
  - PWM
Data Acquisition Systems

- Many embedded systems measure quantities from the environment and turn them into bits
  - These are data acquisition systems (DAS)
  - This is fundamental

- Sometimes data acquisition is the main idea
  - Digital thermometer
  - Digital camera
  - Volt meter
  - Radar gun

- Other times DAS is mixed with other functionality
  - Digital signal processing
  - Networking, storage
  - Feedback control
Big Picture
Why Care About DAS?

- July 1983: Air Canada 143, a Boeing 767, runs out of fuel in mid-air, lands on “abandoned” runway
- Poorly soldered fuel level sensor + mistakes that defeated backup systems
Accuracy

- *Instrument accuracy* is the absolute error of the entire system, including transducer, electronics, and software
- Let $x_{mi}$ be measured value and $x_{ti}$ be the true value

- Average accuracy:
  $$\frac{1}{n} \sum_{i=1}^{n} |x_{ti} - x_{mi}|$$

- Average accuracy of reading:
  $$\frac{100}{n} \sum_{i=1}^{n} \frac{|x_{ti} - x_{mi}|}{x_{ti}}$$

- Average accuracy of full scale:
  $$\frac{100}{n} \sum_{i=1}^{n} \frac{|x_{ti} - x_{mi}|}{x_{max}}$$
More Accuracy

- Maximum error: \( \max | x_{ti} - x_{mi} | \)

- Maximum error of reading: \( 100 \max \frac{| x_{ti} - x_{mi} |}{x_{ti}} \)

- Maximum error of full scale: \( 100 \max \frac{| x_{ti} - x_{mi} |}{x_{tmax}} \)
Resolution

- Instrument resolution is the smallest input signal difference that can be detected by the entire system
  - May be limited by noise in either transducer or electronics

- Spatial resolution of the transducer is the smallest distance between two independent measurements
  - Determined by size and mechanical properties of the transducer
Precision

- Precision is number of distinguishable alternatives, $n_x$, from which result is selected.
- Can be expressed in bits or decimal digits:
  - 1000 alternatives: 10 bits, 3 decimal digits
  - 2000 alternatives: 11 bits, 3.5 decimal digits
  - 4000 alternatives: 12 bits, 3.75 decimal digits
  - 10000 alternatives: >13 bits, 4 decimal digits
- Range is resolution times precision: $r_x = \Delta x \, n_x$
Reproducibility

- *Reproducibility* specifies whether the instrument has equal outputs given identical inputs over some time period
- Specified as full range or standard deviation of output results given a fixed input
- Reproducibility errors often come from transducer drift
ADC: How many bits?

- **Linear transducer case:**
  - ADC resolution must be $\geq$ problem resolution

- **Nonlinear transducer case:**
  - Let $x$ be the real-world signal with range $r_x$
  - Let $y$ be the transducer output with range $r_y$
  - Let the required precision of $x$ be $n_x$
  - Resolutions of $x$ and $y$ are $\Delta x$ and $\Delta y$
  - Transducer response described by $y = f(x)$
  - Required ADC precision $n_y$ (number of alternatives) is:
    - $\Delta x = r_x/n_x$
    - $\Delta y = \min \{ f(x + \Delta x) - f(x) \}$ for all $x$ in $r_x$
  - Bits is $\text{ceiling}(\log_2 n_y)$
ADC: How many bits?

- ADC must be able to measure a change in voltage of the smallest $\Delta y$
ADC: How many bits?

- ADC must be able to measure a change in voltage of the smallest $\Delta y$
DSP Big Picture

Diagram showing the process of analog to digital and digital to analog conversion, including an anti-alias filter and a reconstruction filter.
Signal Reconstruction

- Analog filter gets rid of unwanted high-frequency components in the output
Data Acquisition

◆ Signal: Time-varying measurable quantity whose variation normally conveys information
  ➢ Quantity often a voltage obtained from some transducer
  ➢ E.g. a microphone

◆ Analog signals have infinitely variable values at all times

◆ Digital signals are discrete in time and in value
  ➢ Often obtained by sampling analog signals
  ➢ Sampling produces sequence of numbers
    • E.g. { ... , x[-2], x[-1], x[0], x[1], x[2], ... }
  ➢ These are time domain signals
Sampling

◆ Transducers
  - Transducer turns a physical quantity into a voltage
  - ADC turns voltage into an $n$-bit integer
  - Sampling is typically performed periodically
  - Sampling permits us to reconstruct signals from the world
    • E.g. sounds, seismic vibrations

◆ Key issue: aliasing
  - *Nyquist rate*: $0.5 \times$ sampling rate
  - Frequencies higher than the Nyquist rate get mapped to frequencies below the Nyquist rate
  - Aliasing cannot be undone by subsequent digital processing
Sampling Theorem

◆ Discovered by Claude Shannon in 1949:

A signal can be reconstructed from its samples without loss of information, if the original signal has no frequencies above 1/2 the sampling frequency

◆ This is a pretty amazing result
  - But note that it applies only to discrete time, not discrete values
Aliasing Details

- Let N be the sampling rate and F be a frequency found in the signal
  - Frequencies between 0 and 0.5*N are sampled properly
  - Frequencies >0.5*N are aliased
    - Frequencies between 0.5*N and N are mapped to (0.5*N)-F and have phase shifted 180°
    - Frequencies between N and 1.5*N are mapped to f-N with no phase shift
    - Pattern repeats indefinitely
- Aliasing may or may not occur when N == F*2*X where X is a positive integer
No Aliasing

125 Hz sin wave sampled at 1000 Hz.

250 Hz sin wave sampled at 1000 Hz.
1 kHz Signal, No Aliasing

Sampled in phase at 2667 Hz

Sampled out of phase at 2667 Hz
Aliasing

500 Hz sin wave sampled at 1000 Hz.

533 Hz sin wave sampled at 1000 Hz.
Avoiding Aliasing

1. Increase sampling rate
   - Not a general-purpose solution
     - White noise is not band-limited
     - Faster sampling requires:
       - Faster ADC
       - Faster CPU
       - More power
       - More RAM for buffering

2. Filter out undesirable frequencies before sampling using analog filter(s)
   - This is what is done in practice
   - Analog filters are imperfect and require tradeoffs
Signal Processing Pragmatics
Aliasing in Space

- Spatial sampling incurs aliasing problems also
- Example: CCD in digital camera samples an image in a grid pattern
  - Real world is not band-limited
  - Can mitigate aliasing by increasing sampling rate
Point vs. Supersampling

Point sampling  4x4 Supersampling

Point sampling  4x4 Supersampling
Digital Signal Processing

◆ Basic idea
  - Digital signals can be manipulated losslessly
  - SW control gives great flexibility

◆ DSP examples
  - Amplification or attenuation
  - Filtering – leaving out some unwanted part of the signal
  - Rectification – making waveform purely positive
  - Modulation – multiplying signal by another signal
    - E.g. a high-frequency sine wave
Assumptions

1. Signal sampled at fixed and known rate $f_s$
   - I.e., ADC driven by timer interrupts

2. Aliasing has not occurred
   - I.e., signal has no significant frequency components greater than $0.5*f_s$
   - These have to be removed before ADC using an analog filter
   - Non-significant signals have amplitude smaller than the ADC resolution
Filter Terms for CS People

- Low pass – lets low frequency signals through, suppresses high frequency
- High pass – lets high frequency signals through, suppresses low frequency
- Passband – range of frequencies passed by a filter
- Stopband – range of frequencies blocked
- Transition band – in between these
Simple Digital Filters

- \( y(n) = 0.5 \times (x(n) + x(n-1)) \)
  - Why not use \( x(n+1) \)?
- \( y(n) = \frac{1.0}{6} \times (x(n) + x(n-1) + x(n-2) + \ldots + x(n-5)) \)
- \( y(n) = 0.5 \times (x(n) + x(n-3)) \)
- \( y(n) = 0.5 \times (y(n-1) + x(n)) \)
  - What makes this one different?
- \( y(n) = \text{median} \left[ x(n) + x(n-1) + x(n-2) \right] \)
Gain vs. Frequency

\[ y(n) = \frac{y(n-1) + x(n)}{2} \]
\[ y(n) = \frac{x(n) + x(n-1)}{2} \]
\[ y(n) = \frac{x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4) + x(n-5)}{6} \]
\[ y(n) = \frac{x(n) + x(n-3)}{2} \]
Useful Signals

◆ Step:
  ➢ ..., 0, 0, 0, 1, 1, 1, ...

◆ Impulse:
  ➢ ..., 0, 0, 0, 1, 0, 0, ...
Step Response

![Step Response Graph]

- **Step Input**
- **FIR**
- **IIR**
- **Median**

**Response** vs. **Sample Number, n**

- **Response** scale: 0 to 1
- **Sample Number, n** scale: 0 to 5
FIR Filters

- Finite impulse response
  - Filter “remembers” the arrival of an impulse for a finite time
- Designing the coefficients can be hard
- Moving average filter is a simple example of FIR
Moving Average Example

a. Original signal

b. 11 point moving average

c. 51 point moving average
FIR in C

SAMPLE fir_basic (SAMPLE input, int ntaps,
    const SAMPLE coeff[],
    SAMPLE z[])
{
    z[0] = input;
    SAMPLE accum = 0;
    for (int ii = 0; ii < ntaps; ii++) {
        accum += coeff[ii] * z[ii];
    }
    for (ii = ntaps - 2; ii >= 0; ii--) {
        z[ii + 1] = z[ii];
    }
    return accum;
}
Implementation Issues

- Usually done with fixed-point
- How to deal with overflow?
- A few optimizations
  - Put coefficients in registers
  - Put sample buffer in registers
  - Block filter
    - Put both samples and coefficients in registers
    - Unroll loops
  - Hardware-supported circular buffers

- Creating very fast FIR implementations is important
Filter Design

- Where do coefficients come from for the moving average filter?
- In general:
  1. Design filter by hand
  2. Use a filter design tool
- Few filters designed by hand in practice
- Filters design requires tradeoffs between
  1. Filter order
  2. Transition width
  3. Peak ripple amplitude
- Tradeoffs are inherent
Filter Design in Matlab

- Matlab has excellent filter design support
  - `C = firpm (N, F, A)`
  - `N = length of filter - 1`
  - `F = vector of frequency bands normalized to Nyquist`
  - `A = vector of desired amplitudes`

- `firpm` uses minimax – it minimizes the maximum deviation from the desired amplitude
Filter Design Examples

\[
f = [0.0 \ 0.3 \ 0.4 \ 0.6 \ 0.7 \ 1.0];
\]
\[
a = [0 \ 0 \ 1 \ 1 \ 0 \ 0];
\]
\[
\text{fil1} = \text{firpm}(10, f, a);
\]
\[
\text{fil2} = \text{firpm}(17, f, a);
\]
\[
\text{fil3} = \text{firpm}(30, f, a);
\]
\[
\text{fil4} = \text{firpm}(100, f, a);
\]

\[
\text{fil2} =
\begin{array}{cccccccc}
-0.0278 & -0.0395 & -0.0019 & -0.0595 & 0.0928 & 0.1250 & -0.1667 & -0.1985 \\
0.2154 & 0.2154 & -0.1985 & -0.1667 & 0.1250 & 0.0928 & -0.0595 & -0.001 \\
-0.0395 & -0.0278
\end{array}
\]
Testing an FIR Filter

- **Impulse test**
  - Feed the filter an impulse
  - Output should be the coefficients

- **Step test**
  - Feed the filter a test
  - Output should stabilize to the sum of the coefficients

- **Sine test**
  - Feed the filter a sine wave
  - Output should have the expected amplitude
Digital to Analog Converters

- Opposite of an ADC
- Available on-chip and as separate modules
  - Also not too hard to build one yourself
- DAC properties:
  - Precision: Number of distinguishable alternatives
    - E.g. 4092 for a 12-bit DAC
  - Range: Difference between minimum and maximum output (voltage or current)
  - Speed: Settling time, maximum output rate
- LPC2129 has no built-in DACs
Pulse Width Modulation

- PWM answers the question: How can we generate analog waveforms using a single-bit output?
  - Can be more efficient than DAC

![Diagram showing Pulse Width Modulation waveforms](image)

![Graphs showing Source Signal Sine Wave of 1 Hz and Pulse Width Modulated Sine Wave](image)
PWM

- **Approximating a DAC:**
  - Set PWM period to be much lower than DAC period
  - Adjust duty cycle every DAC period

- **Important application of PWM is in motor control**
  - No explicit filter necessary – inertia makes the motor its own low-pass filter

- **PWM is used in some audio equipment**
Summary

- Filters and other DSP account for a sizable percentage of embedded system activity
- Filters involve unavoidable tradeoffs between
  - Filter order
  - Transition width
  - Peak ripple amplitude
- In practice filter design tools are used
- We skipped all the theory!
  - Lots of ECE classes on this