ADC, DAC & Control

Tajana Simunic Rosing
Department of Computer Science and Engineering
University of California, San Diego.
Hardware platform architecture
Sensors and Actuators

- **Sensors**
  - Capture physical stimulus
  - Convert it to electrical signal

- **Actuators**
  - Create physical stimulus
  - Given electrical signals
  - Examples:
    - Pneumatic systems, IR, thermal, motors, MEMS

- ADC and DAC are needed

- Control – stability, convergence etc.
Embedded System Hardware
Interfacing Sensors and Actuators

A/D converter
sample-and-hold

sensors
environment

information processing

D/A converter

actuators
display
“Real World” Sampled Data Systems Consist Of ADCs and DACs

ADC SAMPLED AND QUANTIZED WAVEFORM

DAC RECONSTRUCTED WAVEFORM

Real World Sampled Data Systems

ADC
DAC
DSP
Memory

Analog
Digital

Channel

Analog Amplitude

Digital Value

time

time
Sampling: Time Domain

- Many signals originate in continuous-time
  Talking on cell phone, or playing acoustic music
- By ideally sampling a continuous-time signal at isolated, equally-spaced points in time, we obtain a sequence of numbers

\[ f[n] = f(n T_s) \]

\[ n \in \{\ldots, -2, -1, 0, 1, 2, \ldots\} \]

\( T_s \) is the sampling period.

\[ f_{\text{sampled}}(t) = f(t) \sum_{n=-\infty}^{\infty} \delta(t - n T_s) \]

Sampled analog waveform
Signal Sampling

- **Sampling** converts a *continuous time* signal into a *discrete time* signal
  - It replicates spectrum of continuous-time signal at integer multiples of sampling frequency

- **Categories:**
  - Impulse (ideal) sampling
  - Natural Sampling
  - Sample and Hold operation
Impulse Sampling

\[ x_\delta(t) = \sum_{n = -\infty}^{\infty} \delta(t - nT_s) \]

\[ x_s(t) = x(t) \cdot x_\delta(t) \]

\[ X_\delta(f) = \frac{1}{T_s} \sum_{n = -\infty}^{\infty} \delta(f - nf_s) \]

\[ X_s(f) \]

|f|
Impulse Sampling

- Impulse train spaced at $T_s$ multiplies the signal $x(t)$ in time domain, creating
  - discrete time,
  - continuous amplitude signal $x_s(t)$

- Impulse train spaced at $f_s$ convolutes the signal $X(f)$ in frequency domain, creating
  - Repeating spectrum $X_s(f)$
  - spaced at $f_s$
The Aliasing Effect

\[ f_s > 2f_m \]

\[ f_s < 2f_m \]

Aliasing happens
Aliasing

Under sampling will result in aliasing that will create spectral overlap.
Ideal Sampling and Aliasing

- Sampled signal is discrete in time domain with spacing $T_s$
- Spectrum will repeat for every $f_s$ Hz
- Aliasing (spectral overlapping) if $f_s$ is too small ($f_s < 2f_m$)
- Sample at least at sampling rate $f_s = 2f_m$
- Generally oversampling is done $\Rightarrow f_s > 2f_m$
Nyquist theorem

- Analog input can be precisely reconstructed from its output, provided that sampling proceeds at $\geq$ double of the highest frequency found in the input [Nyquist 1928]

Does not capture effect of quantization: Quantization noise prevents precise reconstruction.
Natural Sampling

- Sampling pulse train has a finite width $\tau$

- Sampled spectrum will repeat itself with a ‘Sinc’ envelope

- More realistic modeling

- Distortion after recovery depends on $\tau/T_s$
Natural Sampling
Sample and Hold

$V_e$ is analog input signal
$V_e$ is digital sampled signal
Different Sampling Models

Figure 2.14 Amplitude and time coordinates of source data. (a) Original analog waveform. (b) Natural-sampled data. (c) Quantized samples. (d) Sample and hold.
Quantization

- Quantization is done to make the signal amplitude discrete.
Linear Quantization

$L$ levels

$(L-1)q = 2V_p = V_{pp}$

For large $L$

$Lq \approx V_{pp}$

Figure 2.15  Quantization levels.
Quantization Noise

quantization noise = approx - real signal
Linear Pulse Code Modulation

\[ V_{\text{max}} = 7.5V \]

<table>
<thead>
<tr>
<th>Analog Input (V)</th>
<th>Binary Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0V</td>
<td>1110</td>
</tr>
<tr>
<td>6.5V</td>
<td>1101</td>
</tr>
<tr>
<td>6.0V</td>
<td>1100</td>
</tr>
<tr>
<td>5.5V</td>
<td>1011</td>
</tr>
<tr>
<td>5.0V</td>
<td>1010</td>
</tr>
<tr>
<td>4.5V</td>
<td>1001</td>
</tr>
<tr>
<td>4.0V</td>
<td>1000</td>
</tr>
<tr>
<td>3.5V</td>
<td>0111</td>
</tr>
<tr>
<td>3.0V</td>
<td>0110</td>
</tr>
<tr>
<td>2.5V</td>
<td>0101</td>
</tr>
<tr>
<td>2.0V</td>
<td>0100</td>
</tr>
<tr>
<td>1.5V</td>
<td>0011</td>
</tr>
<tr>
<td>1.0V</td>
<td>0010</td>
</tr>
<tr>
<td>0.5V</td>
<td>0001</td>
</tr>
<tr>
<td>0V</td>
<td>0000</td>
</tr>
</tbody>
</table>

Proportionality

Analog to Digital

Defined by a sampling rate & bit depth \( L \) (total number of values that can be represented)
Non-Uniform Quantization

- In speech signals, very low speech volumes predominates
  - Voltage exceeds the RMS value only 15% of the time
- These low level signals are under represented with uniform quantization
  - Same noise power but low signal power
- The answer is non uniform quantization
Figure 2.18  Uniform and nonuniform quantization of signals.
Non-uniform Quantization

Compress the signal first
Then perform linear quantization
→ Result in nonlinear quantization

Figure 2.19 (a) Nonuniform quantizer characteristic. (b) Compression characteristic. (c) Uniform quantizer characteristic.
μ-law and A-law

Widely used compression algorithms

Figure 2.20  Compression characteristics. (a) μ-law characteristic. (b) A-law characteristic.
PCM: $\mu$ & A-law

- Non-linear PCM encodings are used where quantization levels vary as a function of amplitude
- **A-law algorithm** has less proportional distortion for small signals
- **$\mu$-law algorithm** has a slightly larger dynamic range
Embedded System Hardware
Interfacing Sensors and Actuators

A/D converter
sample-and-hold

sensors

evironment

information processing

display

D/A converter

actuators
Digital to Analog Converters

- Ideal sampling would allow us to reconstruct the signal perfectly with a sequence of impulses
  - But there is no ideal sampling, so…

- We use zero-order hold circuit to create an analog output
  - Holding each sample value for one sample interval causes multiple harmonics above the Nyquist frequency
  - A low pass reconstruction filter is needed to recreate the signal

Binary-weighted DAC
DAC: key parameters

- **Resolution**
  - # of possible output levels (n bit DAC gives $2^n$ levels)

- **Maximum sampling rate**
  - Defined by Nyquist theorem

- **Monotonicity**
  - The ability of a DAC's analog output to move only in the direction that the digital input moves; e.g. if the input increases, the output doesn't dip too early

- **Total harmonic distortion and noise (THD+N)**
  - A measurement of the distortion and noise introduced to the signal by the DAC.
  - A percentage of the total power of unwanted harmonic distortion and noise that accompany the desired signal.

- **Dynamic range**
  - A measurement of the difference between the largest and smallest signals the DAC can reproduce expressed in decibels.
Basics of Control

Tajana Simunic Rosing
Department of Computer Science and Engineering
University of California, San Diego.
Adding Control

A/D converter
sample-and-hold

sensors

information
processing

environment

D/A converter
display

actors
Control System

- **Objective**: output tracks a reference even in the presence of measurement noise, model error and disturbances

- **Metrics**
  - Stability - Output remains bounded
  - Performance - How well an output tracks the reference
  - Disturbance rejection – Tolerate outside error sources
  - Robustness - Ability to tolerate modeling error of the plant

- Software gives commands to meet a setpoint, the system responds; E.g: Thermostat, Aircraft altitude control
Performance

- **Rise time**
  - Time it takes from 10% to 90%

- **Peak time**

- **Overshoot**
  - Percentage by which Peak exceed final value

- **Settling time**
  - Time it takes to reach 1% of final value
Open-Loop Control Systems

- **Plant** - Physical system to be controlled
  - Car, plane, disk, heater,…
- **Actuator** - Device to control the plant
  - Throttle, wing flap, disk motor,…
- **Controller** - Designed product to control the plant
- **Output** - Aspect of the physical system we are interested in
  - Speed, disk location, temperature
- **Reference or Setpoint** - Value we want to see at output
  - Desired speed, desired location, desired temperature
- **Disturbance** - Uncontrollable input to the plant imposed by environment
  - Wind, bumping the disk drive, door opening
Closed Loop Control Systems

- Sensor
  - Measure the output
- Error detector
- Feedback control systems
- Minimize tracking error

![Control System Diagram]

**Car model**
\[ v_{t+1} = 0.7v_t + 0.5u_t - w_t \]

**Control law**
\[ u_t = F^*(r_t - v_t) \]

**System model**
\[ v_{t+1} = (0.7-0.5P)v_t + 0.5P r_t - w_t \]
Controller Design: On-off control

- If the system is below a set point, then control is turned-on, else it is turned-off.
- The difference between on and off is deliberately small, known as the hysteresis $H$, to prevent noise from switching control rapidly and unnecessarily when near the set-point.
- Used by almost all domestic thermostats.

Figure source: http://newton.ex.ac.uk/teaching/CDHW/Feedback/ControlTypes.html#OnOffCtl
Proportional control

- Good alternative to on-off control
- Signal becomes proportional to the error
  - $P (\text{setpoint} - \text{output})$
  - Example, car speed for cruise control
- Need to find out value of constant $P$
  - Tuning the controller is hard
  - What happens if $P$ is too low/high?
- Typically a proportional controller decreases response time so it quickly gets to the setpoint but it increases overshoot
Controller design - P

- Make a change to the output that is proportional to the current error
  - $P_{out}$: Proportional term of output
  - $K_p$: Proportional gain
  - $e$: Error = Setpoint – Current Value

- P controller often has a permanent offset from setpoint
  - retains error that depends on $K_p$ & the process gain as it needs non-zero error to drive it

- System can become unstable when $K_p$ is too high

$P_{out} = K_p e(t)$

Source: wikipedia
P-only Control

For an open loop overdamped process as $K_p$ is increased the process dynamics goes through the following sequence of behavior:

- overdamped
- critically damped
- oscillatory
- ringing
- sustained oscillations
- unstable oscillations
Dynamic Changes as $K_p$ is Increased
Adding derivative control

- To reduce overshoot/ripple, take into account how fast are you approaching the setpoint
  - If very fast, overshoot may be forthcoming: reduce the signal recommended by the proportional controller
  - If very slow, may never get to setpoint: increase the signal

- Proportional-Derivative controllers are slower than proportional, but have less oscillation, smaller overshoot/ripple
Controller Design: Derivative

- Reduces the magnitude of the overshoot produced by the integral component and improves the combined controller-process stability
- Proportional to the derivative of the error
- Large $K_d$ decreases the overshoot but amplifies the noise in the signal

\[ D_{out} = K_d \frac{de}{dt} \]
Integral control

- There may still be error in the PD controller
  - For example, the output is close to the setpoint
    - Proportional is very small and so is the error, discretization of signal will provide no change in the proportional controller
    - Derivative controller will not change signal, unless there is a change in output

- Take the sum of the errors over time, even if they’re small, they’ll eventually add up
Controller Design: Integral

- Proportional to both the magnitude of the error and the duration of the error

- Large $K_i$ eliminates steady state errors faster but can cause overshoot

\[ I_{out} = K_i \int_0^t e(\tau) \, d\tau \]

Source: wikipedia
Controller Design: PID

- Combine Proportional, Integral, and Derivative control to change Manipulated Variable (MV)
  - Use P to control the amount of disturbance (error)
  - Use D to control the speed of reduction in error
  - Use I to ensure steady state convergence and convergence rate
- Does not guarantee optimality or stability, is not adaptive

\[
MV(t) = P_{out} + I_{out} + D_{out}
\]

\[
u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau) \, d\tau + K_d \frac{de}{dt}
\]

Source: wikipedia
Controller Design: PID

Effects of *increasing* parameters independently

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Rise time</th>
<th>Overshoot</th>
<th>Settling time</th>
<th>Steady-state error</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_p$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Small change</td>
<td>Decrease</td>
<td>Degrade</td>
</tr>
<tr>
<td>$K_i$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Increase</td>
<td>Eliminate</td>
<td>Degrade</td>
</tr>
<tr>
<td>$K_d$</td>
<td>Minor change</td>
<td>Decrease</td>
<td>Decrease</td>
<td>No effect in theory</td>
<td>Improve if small</td>
</tr>
</tbody>
</table>

Source: wikipedia
PID Controller Tuning

Model-based
1. Direct Synthesis
2. Internal Model Control (IMC)
3. Controller tuning relations
4. Frequency response techniques
5. Computer simulation

All these methods relay on off-line model design

On-Line Tuning
1. Continuous Cycling
2. Relay Auto-Tuning
3. Step Test Method
   - When in steady state, apply a small test step & set controller reaction settings by process reaction curve

Normally done after initial settings are created with model-based methods
Ziegler and Nichols (1942) introduced the continuous cycling method for controller tuning that is based on the following procedure:

- **Step 1.** After the process has reached steady state approximately, eliminate the integral and derivative control actions by setting

  \[ K_d = K_i = \text{zero} \]
Ziegler and Nichols steps 2 & 3

- **Step 2.** Set $K_p$ equal to a small value (e.g., 0.5) and place the controller in the automatic mode.

- **Step 3.** Gradually increase $K_p$ in small increments until continuous cycling occurs. The term *continuous cycling* refers to a sustained oscillation with a constant amplitude.
  - **Ultimate gain, $K_u$** - The value of $K_p$ that produces continuous cycling for proportional-only control
  - **Ultimate period, $T_u$** - The period of the corresponding sustained oscillation
Ziegler and Nichols: Steps 4 & 5

- **Step 4:** Use \( K_u \) and \( T_u \) to set the gains \( K_p, K_i \) & \( K_d \), with table below:

<table>
<thead>
<tr>
<th>Control Type</th>
<th>( K_p )</th>
<th>( K_i )</th>
<th>( K_d )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>P</strong></td>
<td>( 0.5K_u )</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>PI</strong></td>
<td>( 0.45K_u )</td>
<td>( 1.2\frac{K_p}{T_u} )</td>
<td>-</td>
</tr>
<tr>
<td><strong>PD</strong></td>
<td>( 0.8K_u )</td>
<td>-</td>
<td>( \frac{K_pT_u}{8} )</td>
</tr>
<tr>
<td><strong>classic PID</strong></td>
<td>( 0.60K_u )</td>
<td>( 2\frac{K_p}{T_u} )</td>
<td>( \frac{K_pT_u}{8} )</td>
</tr>
</tbody>
</table>

- **Step 5.** Fine-tune by introducing a small set-point change and observing the closed-loop response.
Relay Auto-Tuning

- Developed by Åström and Hägglund (1984)
- A simple & easily automated experimental test to get $K_u$ & $T_u$:
  - The feedback controller is temporarily replaced by an on-off controller or relay with amplitude $d$
  - After the control loop is closed, the controlled variable exhibits a sustained oscillation of period $T_u = P$ & $K_u = \frac{4d}{\pi a}$
Sources

- Real-time DSP Lab, Prof. Brian Evans, UTA
- Ryerson Communications Lab, X. Fernando
- Daniel Mosse & David Willson
- Charles Williams; http://newton.ex.ac.uk/teaching/CDHW/Feedback/