TI-RSLK MAX
Texas Instruments Robotics System Learning Kit
Module 15

Lecture: Data Acquisition Systems - Theory
Data Acquisition Systems

You will learn in this module

- Signals & Sampling
  - ADC, DAC
  - Range, resolution, precision
  - Successive approximation

- MSP432
  - Software driver
  - Spectrum Analyzer
  - Central Limit Theorem
Data Acquisition Systems

Application Problem
- Significance
- Safety
- Theory
- Environment

Transducer
- Physics
- Performance
- Noise

Analog Electronics
- Specifications
- Design
- Performance
- Noise

Computer Interface
- ADC and DAC
- Period/Frequency
- PWM

Microcomputer Based System

Software
- Sampling
- Formatting
- Calibration
- Digital filters
- Compression
- Curve fitting
- DSP
- Event detection
- Decision Making
- Control
- Open or closed
- Linear/nonlinear
- Human Interface
- Graphics
- Touch screen
- Audio In/Out

Error Analysis
- Theoretical
- Experimental
A Control System includes a Data Acquisition System

Real World Measurand → Transducer → Analog Interface Circuits

Actuator
- Mechanical
- Electrical
- Thermal
- Sound
- Light

Microcontroller
- ADC
- Timer
- DAC
- PWM
Sampling: conversion from analog to digital

Amplitude
- Range
- Resolution
- Precision

Time domain
- Sampling rate, $f_s$
  - 0 to $\frac{1}{2} f_s$
- Number of samples
  - Buffer size $N$
- Frequency resolution
  - $f_s/N$
DAC versus ADC

**DAC**
- Digital to Analog
- uC output
- Signal generation

**ADC**
- Analog to Digital
- uC input
- Measurements
MSP432 ADC14

- 14 bits
- 24 channels
- 1 Msps
Successive Approximation

8-bit Successive Approximation Game

• I pick a number from 0 to 255
• You can guess
• I will respond high or low (same)
• How many guesses will it take you?

What is your first guess?
Successive Approximation – How it works

8-bit Successive Approximation Game
• You asked, “what is bit 7?”
• You asked, “what is bit 6?”
...  
• You asked, “what is bit 0?”

Good information [https://e2e.ti.com/blogs_/b/msp430blog/archive/2016/05/10/how-to-leverage-the-flexibility-of-an-integrated-adc-in-an-mcu-for-your-design-to-outshine-your-competitor-part-1](https://e2e.ti.com/blogs_/b/msp430blog/archive/2016/05/10/how-to-leverage-the-flexibility-of-an-integrated-adc-in-an-mcu-for-your-design-to-outshine-your-competitor-part-1)

- Uses a DAC
- Uses a comparator
- 1 bit/clock
### ADC14 Software Initialization

#### ADC14→CTL0

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31-30</td>
<td>PDIV, SHSx, SHP, ISSH, DIVx, SSELx, CONsx, BUSY</td>
</tr>
<tr>
<td>15-12</td>
<td>SHT1x, SHT0x, MSC, ON, ENC, SC</td>
</tr>
</tbody>
</table>

- Clock (speed/power)
- Sample and hold (noise)
- Sequence or single channel
- Reference (range)
- Enable
- Start sample
### ADC14 Software Initialization

**ADC14->CTL1**

<table>
<thead>
<tr>
<th>31-28</th>
<th>27 – 24</th>
<th>22</th>
<th>21</th>
<th>20-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH3MAP – CH0MAP</td>
<td>BATmap</td>
<td>CStartAdr</td>
<td>RES</td>
<td>DF</td>
</tr>
<tr>
<td>15-6</td>
<td>5 – 4</td>
<td>3</td>
<td>2</td>
<td>1-0</td>
</tr>
</tbody>
</table>

- Address
- Resolution
- Format
- Power

**ADC14->IFGR0**

<table>
<thead>
<tr>
<th>31</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFG31</td>
<td>…</td>
<td>IFG5</td>
<td>IFG4</td>
<td>IFG3</td>
<td>IFG2</td>
<td>IFG1</td>
</tr>
</tbody>
</table>

- Conversion complete
## ADC14 Software Initialization

### ADC->MCTL[n]

<table>
<thead>
<tr>
<th>31-16</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11-8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WINCTH</td>
<td>WINC</td>
<td>DIF</td>
<td></td>
<td>VRSEL</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td></td>
<td></td>
<td>4 – 0</td>
</tr>
</tbody>
</table>

- Comparator
- Differential/single
- Reference
- Channel
ADC14 Software Conversion

1. Wait for BUSY to be zero
2. Start conversion
3. Wait for completion
4. Read result

```c
uint32_t ADC_In6(void){
    while((ADC14->CTL0&0x00010000){};
    ADC14->CTL0 |= 0x00000001;
    while(((ADC14->IFGR0&0x01) == 0){};
    return ADC14->MEM[0];
}
```
Periodic Interrupt and Mailbox

1. Sample ADC
2. Run digital filter
3. Save in global
4. Set semaphore

```c
void SysTick_Handler(void){
    uint32_t RawADC;
    P1OUT ^= 0x01;
    P1OUT ^= 0x01;
    RawADC = ADC_In6();
    ADCvalue = LPF_Calc(RawADC);
    ADCflag = 1;  // semaphore
    P1OUT ^= 0x01;
}
```

9us
Summary

Analog to Digital Converter
- Successive Approximation
- Range
- Resolution
- Precision

Software
- Initialization
- Mailbox

\[
\frac{100}{n} \sum_{i=0}^{n} \frac{|x_{ti} - x_{mi}|}{x_{tmax}}
\]
Module 15

Lecture: Data Acquisition Systems – Performance Measurements
Data Acquisition Systems

You will learn in this module

- Analog to Digital Converter
  - Sampling, Nyquist Theorem
  - Digital filtering

- Noise and statistics
  - Probability Mass Function
  - Spectrum Analyzer
  - Central Limit Theorem

- Data Acquisition Systems
  - Range, resolution, precision
  - Calibration
  - Accuracy
The **Nyquist Theorem** states that if the signal is sampled with a frequency of $f_s$, then the digital samples only contain frequency components from 0 to $\frac{1}{2} f_s$. 
Statistics

- Probability Mass Function (PMF)
- Average ($\mu = \text{mean}$)
- Standard deviation ($\sigma = \text{sigma}$)
- Range (max-min)
- Coefficient of variation, $\text{CV} = \sigma/\mu$
- Precision $\log_2(\mu/\sigma)$
- Resolution, $\Delta$

![Diagram showing PMF with measurements at $P$ and $P + \Delta P$.]
Transducer

Physical Signal \( \rightarrow \) Transducer \( \rightarrow \) Electrical Signal

GP2Y0A21YK0F
IR Distance sensor

- Nonmonotonic
- Hyperbolic
- Noisy
GP2Y0A21YK0F IR distance sensors are noisy

\[ \text{dB}_{FS} = 20 \log_{10}(V/3.3) \]

- Generation/recombination
- Periodic
- White
- Flicker, 1/f
- EM field pickup

90 mV

\[ \frac{3.3V}{90mV} = 37 \]

50 Hz analog LPF
16 Hz digital LPF
CLT states that as independent random variables are added, their sum tends toward a Normal distribution.

- Constant input
- Average of last N samples
- $f_s = 1000$ Hz
Analog Low Pass Filter to remove Aliasing

1) Select the cutoff frequency, $f_c$ (50 Hz)
2) Divide the two capacitors by $2\pi f_c$ (let $C_{1A}$, $C_{2A}$ be the new values)
   
   \[ C_{1A} = \frac{141.4\mu F}{2\pi f_c} = 0.45\mu F \]
   
   \[ C_{2A} = \frac{70.7\mu F}{2\pi f_c} = 0.225\mu F \]

3) Locate two standard value capacitors (with the 2/1 ratio)
   
   \[ C_{1B} = \frac{C_{1A}}{x} = 0.44\mu F \text{ (two 0.22 } \mu F \text{ in parallel)} \]
   
   \[ C_{2B} = \frac{C_{2A}}{x} = 0.22\mu F \]

4) Adjust the resistors to maintain the cutoff frequency
   
   \[ R = 10k\Omega \times x \]
   
   \[ (10k, f_c = 51 \text{ Hz}) \]

\[
\left| \frac{V_{out}}{V_{in}} \right| = \sqrt{\frac{1}{1 + \left(\frac{f}{f_c}\right)^4}}
\]

See https://www.ti.com/design-tools/signal-chain-design/webench-filters.html
Digital Filtering

\[ y(n) = (x(n) + x(n-1) + x(n-2) + x(n-3))/4 \]

**MACQ before**

|------|------|------|------|

**MACQ after**

|------|------|------|------|

\[ x[3] = x[2]; \]
\[ x[2] = x[1]; \]
\[ x[1] = x[0]; \]
\[ x[0] = \text{ADC}_\text{In6}(); \]
\[ y = (x[0] + x[1] + x[2] + x[3])/4; \]
Averaging Low Pass Filters

- Linear Filter
- Finite Impulse Response
- Low pass

\[
y(n) = \frac{x(n) + x(n-1) + \ldots + x(n-N-1)}{N}
\]

\( N = 64 \)
\( f_s = 2000 \text{ Hz} \)

\( f_c = 16 \text{ Hz} \)

Gain

Frequency (Hz)
Averaging Low Pass Filters

![Graph showing step response and formula for averaging low pass filters.

\( y(n) = \frac{x(n) + x(n-1) + \ldots + x(n-N)}{N} \)

- \( t = 20\text{ms} \)
- \( N = 64 \)
- \( f_s = 2000 \text{ Hz} \)
Distance to wall

Distance to wall formulas:

\[ D_r = \frac{A_r}{(n_r + B_r)} + C_r \]
\[ D_c = \frac{A_c}{(n_c + B_c)} + C_c \]
\[ D_l = \frac{A_l}{(n_l + B_l)} + C_l \]
Calibration

- Distance, X, from the sensor to wall, 80 to 400mm
- ADC value, n
- Linear fit $1/X$ versus n
- Solve for $X = A/(n+B)$
- Add distance to central point, $D = A/(n+B)+C$

$\frac{1}{X} = 8.6067 \times 10^{-7}n - 7.5230 \times 10^{-4}$

$R^2 = 0.998$
Summary

Analog to Digital Converter
- Noise

Sampling
- Nyquist Theorem, Aliasing
- Central Limit Theorem

Filters
- Analog LPF
- Digital LPF

Data Acquisition Systems
- Calibration
- Accuracy
Module 15

Lecture: Data Acquisition Systems – Sound generation
Data Acquisition Systems

You will learn in this module

- Signals & Sampling
  - PWM, DAC
  - Range, resolution, precision

- Sound
  - Transducer
  - Analog Circuit
  - Sampling
  - Filtering

![Diagram of PWM and sound wave](image)
Speaker generates sound

Electromagnet
Permanent magnet
Diaphragm
Speaker voltages

Compression wave in both time and space

Pitch = 1/T
Pulse width modulation
- 20 kHz fixed-period digital output
- Variable duty cycle
- Analog low pass filter removes 20 kHz
- Change duty cycle every 50 µs
- Shape of sound encoded as sequence of duty cycles
Interface between microcontroller and speaker

LPF to reject PWM frequency  
HPF to reject DC from PWM DAC  
Provide power to speaker  
TPA731 adds DC offset of 2.5V  
Gain of 4

LPF cutoff = $1/(2\pi CR) = 723\text{Hz}$
HPF cutoff = $1/(2\pi CR) = 159\text{Hz}$
C6 creates DC offset of 2.5V
Gain = $2*R9/R8 = 4$
f_{PWM} = \frac{48 \text{ MHz}}{2424} = 19.8 \text{ kHz}

f_{sound} = \frac{48 \text{ MHz}}{(2424 \times 45)} = 440 \text{ Hz}

\text{Dutycycle} = \frac{\text{High}}{\text{High} + \text{Low}} = \frac{\text{High}}{\text{Period}}

#define Period 2424
const uint16_t wave440[45] = {
    1212, 1339, 1463, 1583, 1695, 1798, 1890, 1968, 2032, 2079, 2110,
    2123, 2119, 2097, 2058, 2002, 1931, 1846, 1748, 1640, 1524, 1402,
    1276, 1148, 1022, 900, 784, 676, 578, 493, 422, 366,
    327, 305, 301, 314, 345, 392, 456, 534, 626, 729,
    841, 961, 1085
};
Software to generate PWM outputs

```c
uint32_t startTime;
void SysTick_Wait2(uint32_t delay){
    volatile uint32_t elapsedTime;
    do{
        elapsedTime = (startTime-SysTick->VAL)&0x00FFFFFF;
    } while(elapsedTime <= delay);
    startTime = SysTick->VAL;
}

while(1){
    High = wave440[i];
    Low = Period-High;
    SysTick_Wait2(Low);
    P3->OUT |= 0x40;   // P3.6 high
    SysTick_Wait2(High);
    P4->OUT &= ~0x40;  // P3.6 low
    i = (i+1)%45;
}

Dutycycle = \frac{High}{High+Low} = \frac{High}{Period}
```
**Summary**

**DAC Precision**
- Number of different duty cycles
- \( \frac{48\text{MHz}}{20\text{kHz}} = 2400 \) alternative \( \approx 11 \text{ bits} \)

**DAC Range**
- 0 to 3.3V

**DAC Resolution**
- \( \frac{3.3\text{V}}{2400} \)
- Limited by noise and LPF cutoff
- Use spectrum analyzer to measure SNR

**DAC Speed**
- Set by PWM period
- New duty cycle every 50 \( \mu \text{s} \)

**Sound**
- Loudness set by power to speaker
- Pitch set by size of duty cycle array
- Voice set by shape duty cycles in array
- Duration
- Envelope (time varying amplitude)

\[ f_{PWM} = \frac{48 \text{ MHz}}{2424} = 19.8 \text{ kHz} \]
\[ f_{sound} = \frac{48 \text{ MHz}}{(2424*45)} = 440 \text{ Hz} \]
Module 15

Lecture: Data Acquisition Systems – Sound recording
You will learn in this module

- Signals & Sampling
  - ADC, DAC
  - Range, resolution, precision
  - Successive approximation

- Sound
  - Transducer
  - Analog Circuit
  - Sampling
  - Filtering
  - Pitch Recognition
Transducer

Primary sensing element

Secondary conversion element

Diaphragm
Caspule
Connector

PCB
JFET

Signal/bias
Gnd

2kΩ
R1

Vcc

Signal/bias
Gnd

Electret
Backplate

JFET

Output

C1

Thickness
Diameter
Interface between microphone and microcontroller

- Provide power to microphone
- HPF to reject DC
- Add DC offset of 1.65V
- Gain of 100
- LPF to prevent aliasing

Gain = $1 + \frac{R6}{R5} = 201$

HPF cutoff = $\frac{1}{2\pi C1(R2||R3)} = 14\text{Hz}$

LPF cutoff = $\frac{1}{2\pi C3R6} = 3600\text{Hz}$
Sampling: adaptive noise rejection

**Noise**
- Comes from microphone
- Very large
- Random
- Not correlated to itself

**Signal**
- Comes from sound pressure
- Highly correlated to itself

Cross correlation to see if signals are the same shape

Assumptions
- Sampled at $f_s$
- Data has zero average (just shape)

Measured data
- $x(n)$ is current sample
- $x(n-1)$ is sample $\Delta t$ ago
- $x(n-2)$ is sample $2\Delta t$ ago

Prerecorded data template
- $y(n)$ is current sample
- $y(n-1)$ is sample $\Delta t$ ago
- $y(n-2)$ is sample $2\Delta t$ ago

Use cross correlation to see if same shape
- $R_{xy} = +$large means same shape
- $R_{xy} = 0$ means not same shape
- $R_{xy} = -$large means same, but inverted shape

\[
R_{xy}(m) = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} x(n) * y(n - m)
\]

\[
f_s = \frac{1}{\Delta t}
\]
Autocorrelation

\[ R_{xx}(m) = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} x(n) * x(n - m) \]

\[ R_{xx2} \approx \frac{127}{128} R_{xx2} + \frac{1}{128} \ x(n) * x(n - 2) \]

\[ f_s = \frac{1}{\Delta t} \]

x(n) is current sample
x(n-1) is sample \( \Delta t \) ago
x(n-2) is sample \( 2\Delta t \) ago
x(n-m) is sample \( m\Delta t \) ago

ADC Input

Sample Number

Signal

x(n)

x(n-m)

shifted by m

Noise
Noise reject filter

Autocorrelation

- $R_{xx2} = M-1$ if signal correlated to itself
- $R_{xx2} = 0$ if signal uncorrelated to itself

```c
int32_t t0,t1,t2; // last 3 inputs/32
int32_t Rxx2;     // autocorrelation factor
#define K 128       // how fast it responds
#define M 128

int32_t NoiseReject(int32_t x){
    t2 = t1;
    t1 = t0;
    t0 = x/32;
    Rxx2 = ((K-1)*Rxx2 + t0*t2)/K;
    if(Rxx2 < -M) Rxx2 = -M;
    if(Rxx2 > M) Rxx2 = M;
    return (Rxx2*x)/M;
}
```
Results of noise suppression

![Graph showing the results of noise suppression with and without a filter. The graph plots ADC Input against Sample Number, with two lines indicating the noise levels before and after suppression. The y-axis ranges from -400 to 400, and the x-axis ranges from 1 to 49.]
volatile int32_t Raw, Sound;
#define DC 8192

void Program15_2_ISR(void) {
    // runs at fs=10kHz
    Raw = ADC_In23() - DC; // sample P8.2
    Sound = NoiseReject(Raw); // improves SNR
    ADCflag = 1; // semaphore
    return from interrupt
}
Use sound as command input to robot

Initial training
- A small number of sine-wave sounds
- Train by recording example of each
- Examples are the templates $y(n)$
  - Fixed sampling rate
  - Variable size to capture complete waves

Use cross correlation to distinguish
- Calculate $R_{xy}$ for each template
  - Use finite size buffers
  - Calculate max $R_{xy}$ for various $m$
- Pitch recognition
  - Best match is largest $R_{xy}$
  - Above a threshold

Frequency Key Shifting (FSK)
Summary

Sound recording
- Transducer
- Gain
- Analog filter

Digital processing
- Noise rejection using autocorrelation
  - Noise does not correlate with itself
  - Signal does correlate with itself
- Pitch recognition (FSK)
  - Training session
  - Cross correlation to find best match
ti.com/rslk
IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2019, Texas Instruments Incorporated