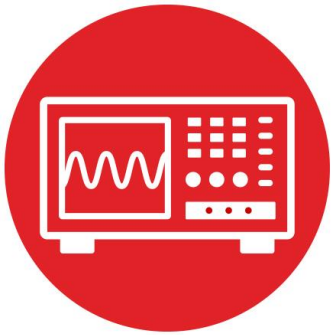

Module 15

Lab 15: Data Acquisition Systems

Lab: Data Acquisition Systems

15.0 Objectives

The purpose of this lab is to interface IR distance sensors that allow the robot to explore its world, see Figure 1.

1. You will use the ADC to input data into the microcontroller.
2. You will use periodic interrupts to sample the ADC at a regular rate.
3. You will study the noise generated in the data acquisition system.
4. You will evaluate a simple digital filter in an attempt to improve **signal to noise** ratio, which is defined as the signal amplitude divided by the noise amplitude.
5. You will evaluate the accuracy and resolution of the measurement.

Good to Know: You have created a sampling data acquisition system in Labs 6 and 10. I.e., the software sampled the line sensor 100 times/sec. However, in this lab you will study sampling in a more fundamental way. There are three tasks that most embedded systems perform: collecting data, making decisions, and affecting outputs. In this lab, we will focus on collecting data, but the robot challenge will require decision making and outputs.

15.1 Getting Started

15.1.1 Software Starter Projects

Look at these three projects:
ADCSWTrigger (busy-wait ADC interface, simple digital filter, periodic interrupt sampling),
Lab13_Timers (your solution to Lab 13),
Lab15_ADC(starter project for this lab)

15.1.2 Student Resources (in datasheet directory)

MSP432P4xx Technical Reference Manual, ADC14 (SLAU356)
MSP432P401R Datasheet, msp432p401m.pdf (SLAS826)
GP2Y0A21YK0F_IR_Distance_Sensor.pdf, datasheet
Pololu Romi Chassis User's Guide.pdf

15.1.3 Reading Materials

Volume 1 Sections 10.1, 10.4, and 10.5
Embedded Systems: Introduction to the MSP432 Microcontroller",
or
Volume 2 Section 8.4, and Chapter 10
Embedded Systems: Real-Time Interfacing to the MSP432 Microcontroller"

Good to Know: Analog to digital conversion is one of the most basic operations a microcontroller performs. ADC sampling requires us to make approximations in both the amplitude and time dimensions. Noise is usually the limiting factor on most data acquisition systems.

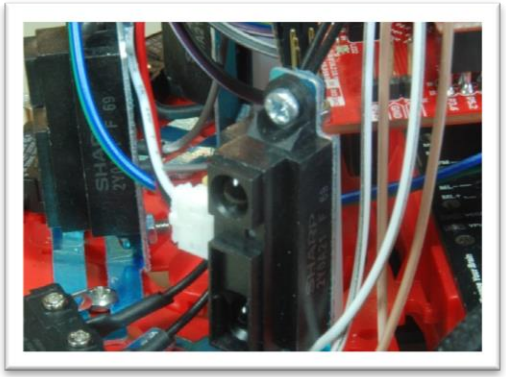

Figure 1. Three IR distance sensors positioned at the front of the robot.

15.1.4 Components needed for this lab

Quantity	Description	Manufacturer	Mfg P/N
1	MSP-EXP432P401R LaunchPad	TI	MSP-EXP432P401R
2	IR bracket pair with 4X bolt and 4X nut	Pololu	#2679
3	IR sensor	Pololu	#136
3	IR cable	Pololu	#1799
3	10 uF tantalum capacitors	AVX	TAP106K020SCS

Table 1 Parts needed for this lab.

Lab: Data Acquisition Systems

15.1.5 Lab equipment needed

Oscilloscope (one or two channels at least 10 kHz sampling)
Logic Analyzer (4 channels at least 10 kHz sampling
Optional: Spectrum Analyzer

15.2 System Design Requirements

The first goal is to attach three GP2Y0A21YK0F IR distance sensors to the robot, and then interface the outputs of the sensors to inputs of the ADC converter on your MSP432 Launchpad. You will then convert raw ADC signal to "distance" from the analog domain to the digital domain. Software plus calibration will allow the robot to measure distance to the wall. The range of the measurement will be 70 to 800 mm. The resolution will be about 1 mm at 200 mm.

Figure 2 shows the measured relationship between output of the IR sensor and the distance to the wall (use a block wood and a ruler). Notice the non-monotonic behavior of the sensor... For example, if the system records a sensor value of 2 V, it could mean 33 mm or 130 mm. During the robot challenge you will endeavor to keep the robot away from the wall, so you will assume the sensor distance is greater than 70 mm. Due to the nature of the robot challenges, we are not interested in distances beyond 800 mm.

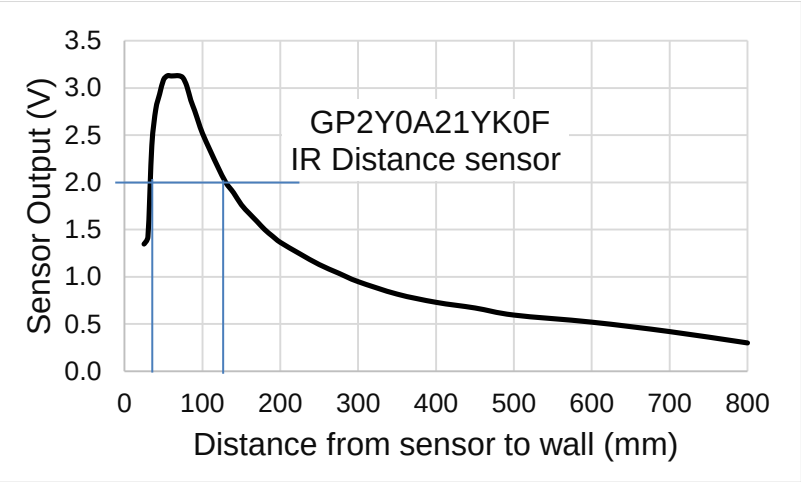

Figure 2. Typical sensor output as a function of distance.

You will develop a function that converts raw ADC samples into distance measured from the wall. Let **n** be a 14-bit sample from the ADC (0 to 16383), and **X** be the distance in mm from the sensor to the wall. The basic form of this nonlinear transfer relation is hyperbolic,

$$\mathbf{X} = A/(\mathbf{n} + B)$$

where A and B are calibration coefficients to be empirically determined.

The above equation defines distance from the sensor to the wall. Your system will however define all distances in mm from a common spot on the robot ($\mathbf{D_r}$, $\mathbf{D_c}$, $\mathbf{D_l}$), as illustrated in Figure 3. This common reference will allow the robot to calculate angle to the wall using geometry and two distance measurements. To handle this change of reference, we will introduce a third calibration coefficient. Define $\mathbf{n_r}$, $\mathbf{n_c}$, and $\mathbf{n_l}$ as the three ADC inputs. Let A_r, A_c, A_l, B_r, B_c, B_l, C_r, C_c, and C_l, be calibration coefficients.

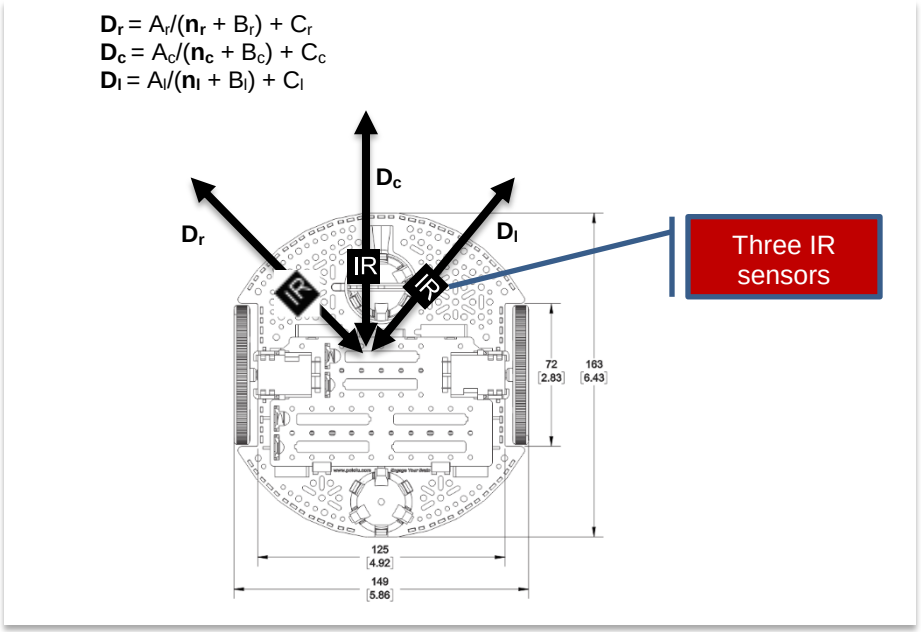

Figure 3. Define distance measured from a central point on the robot.

The maximum measurement distance for the sensor is 800 mm, so if the ADC value is less than a certain threshold, your function should return 800+C. The C prototypes for your functions are

```
int32_t LeftConvert(int32_t nl);      // returns left distance in mm
int32_t CenterConvert(int32_t nc);    // returns center distance in mm
int32_t RightConvert(int32_t nr);     // returns right distance in mm
```

The second goal of this lab is to use periodic interrupts to **sample** the ADC. Triggering the ADC periodically is defined as **sampling**, allowing the system to process the data in both the time and frequency domains. For example, collecting data periodically allows you to implement a digital filter, which passes some data of some frequencies while rejecting others. The purpose of the digital filter is to improve signal to noise ratio. You will study this low pass filter

$$y(n) = (x(n)+x(n-1)+\ldots+x(n-N-1)/N$$

for N = 1 to 512. x(n) is the current sample, x(n-1) is the previous sample, x(n-2) is two sample ago, … and y(n) is the current filter output. You will use the sampled data to study fundamental concepts like the range, resolution, precision, the Oversampling and Nyquist Theorem, aliasing, noise, probability mass function, signal to noise ratio, and the Central Limit Theorem.

An **impulse** digital sequence has one nonzero value and the rest of the points in the sequence are zero, …,0,0,0,1,0,0,0,… The **impulse response** of a filter is the output of the filter given the input is an impulse. If N=4, the impulse response of this filter is …,0,0,0,¼ ,¼,¼,¼,0,0,0,… This filter is called a **finite impulse response** (FIR) filter, because the impulse response has a finite number on nonzero outputs.

A **step** digital sequence has an infinite number of zeros, followed by an infinite number of non-zeros of the same value, …,0,0,0,1,1,1,… The **step response** of a filter is the output of the filter given the input is a step function. If N=4, the step response of this filter is …,0,0,0,0.25 ,0.5,0.75,1,1,1,… In other words if the distance to the wall were to change, this filter will cause a delay in the time the software sees this change in input. You will choose a filter than improves signal to noise ratio without causing too much delay in the step response.

The third goal is to evaluate the accuracy of the distance measurement. **Accuracy** is defined as the difference between truth and measured. Let x_t be the true distance from the robot reference point to the wall, as measured with a ruler. The instrument accuracy is the absolute error referenced to the National Institute of Standards and Technology (NIST) of the entire system including transducer, electronics, and software. Let x_m be the values as measured by the instrument. We define average accuracy of full scale in percent as

$$\frac{100}{n}\sum_{i=0}^{n}\frac{|x_{ti}-x_{mi}|}{x_{t\max}}$$

*The system **resolution** is the smallest input signal difference, Δx that can be detected by the entire system including transducer, electronics, and software. The resolution of the system is limited by noise processes in the transducer itself, noise processes in the electronics, and the number of bits in the ADC. For this lab, resolution will be limited by noise in the IR distance sensor.*

15.3 Experiment set-up

You will attach three IR distance sensors near the front of the robot. Recall that the sensor is confused for distances from 0 to 70 mm (Figure 2), it is preferable to recess the sensors away from the edge of the robot as much as possible. The robot in Figure 1 has the sensors recessed 25 mm from the point at which the bump sensors (used in Module 10) will activate. Therefore, this robot will report incorrect distances when 25 to 70 mm from a wall. The exact placement and angle will be readjusted as part of the robot challenge you attempt. For this lab, it is not critical exactly where the sensors are placed, see Figure 3.

The interface circuit in Figure 4 connects the sensor output directly to the ADC input. Any ADC input pins could have been used, but the issue with a complex design is assigning pins for special purpose like timer PWM, edge-triggered input, UART, SPI and ADC. The three pins shown in Figure 4 do not conflict with the other labs in these modules. Connect the IR sensor to +5V supply, which on the Pololu Motor Driver Power Distribution board labeled VREG. On the LaunchPad, +5V power is simply labeled 5V. The sensor is very noisy, and you will need a supply capacitor for each sensor. The figure shows 10 µF, but any capacitor from 4.7 to 47 µF would be ok.

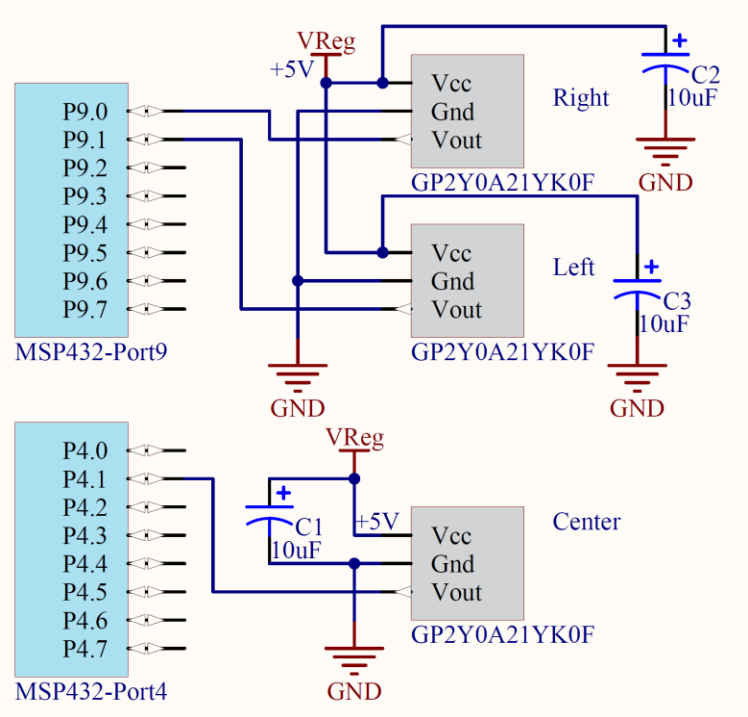

Figure 4. One possible interface circuit for the IR sensors.

Even though the sensor is powered with 5 V, the output will only range from 0 to 3.1 V, see Figure 2. Therefore, it is safe to connect this signal to the MSP432 powered at 3.3 V.

Note: You must use the 0 to 3.3V ADC range and not use the precision internal voltage reference at 2.5 V.

As an option, you could build three analog low pass filters (LPF) and place them between the sensor and ADC. If the sampling frequency is 1000 Hz, then the cutoff for the analog LPF should be about 100 Hz. For this you will need op amps to design a LPF filter and will need a breadboard, the components have not been included in the lab.

15.4 System Development Plan

15.4.1 Explore the ADC, discover the Central Limit Theorem

In this section we will explore how the ADC works using TI's Launchpad. You will using a power supply. Connect a voltage **of about 3 V (any value from 2.5 to 3.1V)** to P4.7. P4.7 is ADC channel 6. Build, debug and run the project **ADCSWTrigger**. This project samples P4.7 at 1000 Hz using SysTick interrupts, implements a simple averaging digital filter, and performs statistical analysis on the collected data. In particular, it calculates a PMF (probability distribution of noise), mean (μ), range (max-min), variance (σ^2) and standard deviation (σ). If you run a terminal emulator like PuTTY or TExaSdisplay, you can see the output of this statistical analysis.

Note: Alternatively, if you have interfaced the LCD to your robot, then outputting these parameters to the LCD will simplify testing.

Place a voltmeter on the P4.7 input and observe the true voltage. Comparing the true voltage with the ADC digital output allows you to see how the ADC operates. Assuming there were no noise and the input were constant, you would expect all the ADC samples to be equal. The fact that the samples are not the same is the result of **noise**. Without noise, the variance and standard deviations would be zero. The **coefficient of variation** (CV) is the standard deviation divided by the mean (μ),

$$CV = \sigma/\mu$$

1/CV is a simple estimate of the signal to noise ratio (SNR). With the input voltage around 3 V, we will approximate the precision in bits of the system as

$$\mathbf{\log_2(\mu/\sigma)}$$

This project also implements a simple digital filter. This filter calculates the average of the last N samples.

$$y(n) = (x(n)+x(n-1)+\ldots+x(n-N-1)/N$$

Averaging is a simple method to improve signal to noise ratio. In this project the value N is defined in the variable Size, varying from N = 1 to 512. By pressing and releasing either of the LaunchPad switches, the software cycles through these ten values of N. With the input voltage at about 3V, collect 1/CV (SNR) and $\log_2(\mu/\sigma)$ data as a function of N = 1, 2, 4, …, and 512. What do you observe?

Lab: Data Acquisition Systems

Next review the Central Limit Theorem (CLT) from your Probability and Statistics class. Look up the assumptions to see if the CLT applies to these measurements. The CLT states that as independent random variables are added, their sum tends toward a Normal or Gaussian distribution. In this experiment you should find, as N is increased the PMF goes from having multiple peaks to having just one peak (see lecture slides). This behavior will be even more pronounced when studying the noise from the IR distance sensor.

15.4.2 Study the digital filters (optional)

If you wish to learn more about the simple averaging filter, open the spreadsheet **FIR_Digital_LowPassFilter.xls** (you should have this in your folder when you opened up the zip file). You can change the sampling rate (fs) and filter size (N) and visualize the frequency and step responses. The two parameters you can adjust are highlighted in yellow. If the sampling rate is 2000 Hz, and the size N is 64, then the filter has a cutoff frequency (fc) of 16 Hz, see Figure 5.

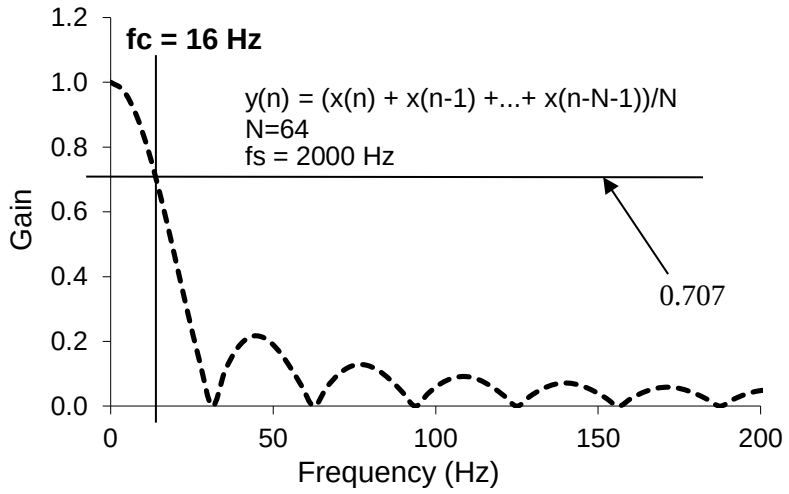

Figure 5. Frequency response of averaging filter with N=64.

15.4.3 Low-level ADC software driver

The first software step is to write two functions that sample the analog signal from the center IR distance sensor. The prototypes for these functions are

void ADC0_InitSWTriggerCh12(void); // initialize P4.1, channel A12
uint32_t ADC_In12(void); // sample P4.1, channel A12

Basically you will convert the initialization and sample function for channel 6 (P4.7) to channel 12 (P4.1), leaving all the design decisions the same, configured for the following:

- Single channel
- Software start
- Busy wait synchronization
- 14-bit, unsigned binary
- 3.3V V(R+), analog input range is 0 to 3.3V
- ADC14MEM0 address

Note: You will not be able to complete this lab without reading the MSP432 data sheet. Look at the chapter on ADC14, and go line by line through the existing ADC0_InitSWTriggerCh6 and ADC_In6 functions. These two functions work, but you need to understand each line, by looking up each of the registers it accesses. Once you understand each line, you will be able to convert it from sampling channel 6 to sampling channel 12.

To test these functions you can run Program15_1. The ISR samples channel 12, runs an averaging digital low pass filter, and passes the data through a mailbox (variable and semaphore) to another thread.

```
void Program15_1_ISR(void){  // runs at 2000 Hz
  uint32_t RawADC;
  P1OUT ^= 0x01;                    // profile
  P1OUT ^= 0x01;                    // profile
  RawADC = ADC_In12();  // sample P4.1/channel 12
  ADCvalue = LPF_Calc(RawADC);
  ADCflag = 1;                      // semaphore
  P1OUT ^= 0x01;                    // profile
}
```

Lab: Data Acquisition Systems

15.4.4 Signal to Noise Ratio of IR distance sensor

To analyze sensor noise, you can use **Program15_1**, which is similar to the project **ADCSWTrigger**, replacing all the channel 6 accesses to channel 12. Run **Program15_1** and determine the SNR for various sizes of the averaging filter, from **N = 32 to 512.**

Low pass FIR $y(n) = (x(n)+x(n-1) +\ldots+x(n-N-1)/N$

While deciding the size of the averaging, also consider the step response. In other words, let the input go from 0 to 1, and calculate the output of filter as a function of time, assuming a sampling rate of 2000 Hz. Calculate the time constant of the filter, defined as the time it takes the output to achieve 63% ($1-e^{-1}$) of the new value given a change in input. The time constant of the FIR filter is 20 ms, see Figure 6. Select a value of N for your robot.

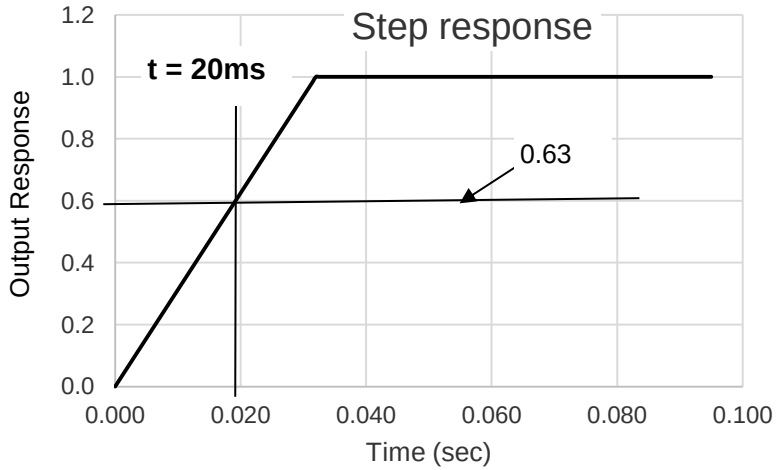

Figure 6. Step response of averaging filter with N= 64.

Note: Since the time constant of the motor is about 100ms, we wish to have the time constant of the filter to be smaller than 100ms.

If you have access to a spectrum analyzer, it is very interesting to observe the noise in the frequency domain. Figure 7 illustrates there is significant noise throughout the frequencies, with peaks at multiples of 1 kHz. The output of a spectrum analyzer is given in decibels full scale. The spectrum analyzer in Figure 7 has a full scale of 5V, so

$$dB_{FS} = 20 \log_{10}(V/5)$$

The noise at 1 kHz, -35 dB, is the equivalent of about 90 mV. On the MSP432, a 90 mV noise will reduce the precision of the system from 14 bits possible with the ADC to $\log_2(3.3V/0.09V)$, which is about 5 bits. The purpose of the digital filter is to remove some of the noise, and improve precision.

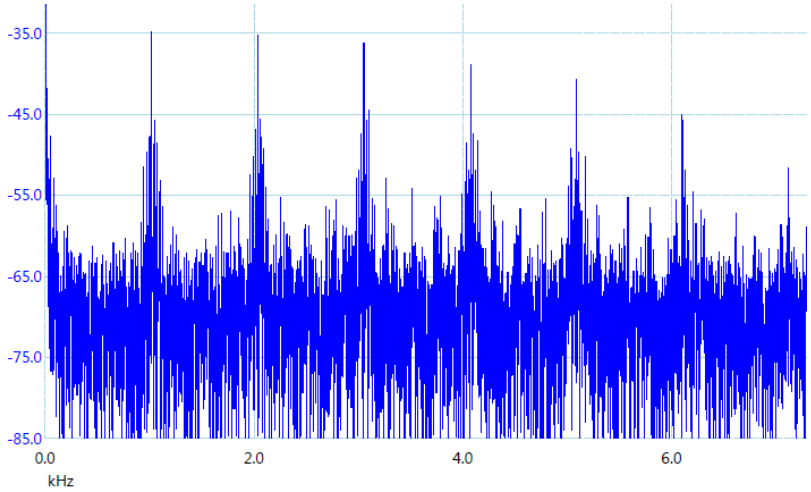

Figure 7. Frequency spectrum of the output of the distance sensor (without analog filter).

Lab: Data Acquisition Systems

15.4.5 Three-channel ADC software driver

The next software step is to write two functions that sample all three analog signals from the sensors. You will need to make three copies of the digital low pass filter, so all channels are filtered. The prototypes for your functions are

void ADC0_InitSWTriggerCh17_12_16 (void);
void ADC_In17_12_16(uint32_t *ch17, uint32_t *ch12, uint32_t *ch16);
Basically you will convert the initialization and function for channels 6 and 7 to channel 12 (P4.1), leaving most the design decisions the same

- Three channel sampling
- Software start
- Busy wait synchronization
- 14-bit, unsigned binary
- 3.3V V(R+), analog input range is 0 to 3.3V
- ADC14MEM2, ADC14MEM3, ADC14MEM4 addresses

To test these functions you can run Program15_2. The ISR samples the three channels, runs three averaging digital low pass filters, and passes the data through a mailbox (variables and semaphore) to another thread.

Note: Again, you must read the MSP432 data sheet when looking at the existing ADC0_InitSWTriggerCh67 and ADC_In67 functions. These two functions work, but you need to understand each line, by looking up each of the registers it accesses. Once you understand each line, you will be able to convert it from sampling channels 6 and 7 to sampling channels 17, 12, and 16.

```
volatile uint32_t nr,nc,nl;
void Program15_2_ISR(void){  // runs at 2000 Hz
  uint32_t raw17,raw12,raw16;
  P1OUT ^= 0x01;          // profile
  P1OUT ^= 0x01;          // profile
  ADC_In17_12_16(&raw17,&raw12,&raw16);  // sample
  nr = LPF_Calc(raw17);  // right is channel 17 P9.0
  nc = LPF_Calc2(raw12); // center is channel 12, P4.1
  nl = LPF_Calc3(raw16); // left is channel 16, P9.1
  ADCflag = 1;           // semaphore
  P1OUT ^= 0x01;          // profile
}
int Program15_2(void){ // example program 15.2
  uint32_t raw17,raw12,raw16; int32_t n; uint32_t s;
  Clock_Init48MHz();
  ADCflag = 0; s = 256; // replace with your choice
  ADC0_InitSWTriggerCh17_12_16();   // initialize
  ADC_In17_12_16(&raw17,&raw12,&raw16);  // sample
  LPF_Init(raw17,s);      // P9.0/channel 17
  LPF_Init2(raw12,s);     // P4.1/channel 12
  LPF_Init3(raw16,s);     // P9.1/channel 16
  UART0_Init();           // initialize UART0
  LaunchPad_Init();
  TimerA1_Init(&Program15_2_ISR,250);  // 2000 Hz
  UART0_OutString("Program 15.2 \n");
  EnableInterrupts();
  while(1){
   for(n=0; n<2000; n++){
     while(ADCflag == 0){};
     ADCflag = 0; // show every 2000th point
   }
   UART0_OutUDec5(nl);UART0_OutUDec5(LeftConvert(nl));
   UART0_OutUDec5(nc);UART0_OutUDec5(CenterConvert(nc));
   UART0_OutUDec5(nr);UART0_OutUDec5(RightConvert(nr));
   UART0_OutChar('\n'); // once a second
  }
}
```

Lab: Data Acquisition Systems

Connect a scope to P1.0 and measure the time between interrupts and the time within the ISR. Figure 8 shows this system (Size=256) executes 25 µs each time in the ISR. At 2 kHz, the ADC sampling and digital filtering consume 25/500 =0.05 (5%) of the processor time.

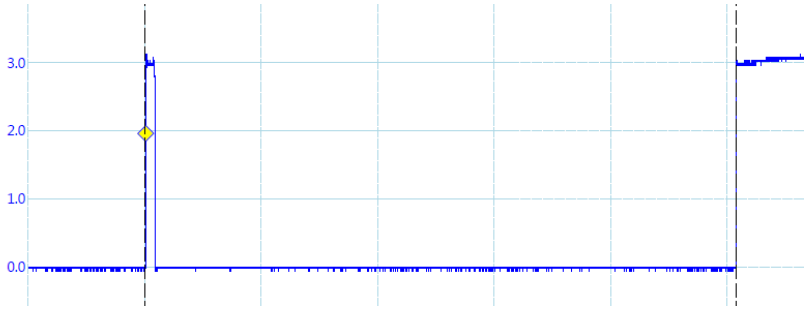

Figure 8. Thread profile. Scope attached to P1.0. The time scale is 5 µs per division.

15.4.6 Calibration and accuracy

Implement the three conversion functions, and use **Program15_2** first to calibrate and then to test these functions. Define exactly where on the robot you wish to set the reference point, see Figure 3. Use a ruler to measure the true distances ($\mathbf{D_r}$, $\mathbf{D_c}$, $\mathbf{D_l}$), and use **Program 15_2** to display the raw ADC values ($\mathbf{n_r}$, $\mathbf{n_c}$, and $\mathbf{n_l}$). Collect about 10 points for each sensor and fit the data to find the calibration coefficients for each sensor. Limit your calibration for distances between than 70 mm and 800 mm from the sensor.

Enter the coefficients into your software, and repeat the process collecting another set of 10 measurements for each sensor. Calculate the average accuracy of full scale in percent for each sensor.

15.4.7 Discovering the Nyquist Theorem

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 ½ f_s. Conversely, if the analog signal does contain frequency components larger than ½ f_s, then there will be an aliasing error during the sampling process. Aliasing is when the digital signal appears to have a different frequency than the original analog signal.

Although the ADC is sampled at 2000 Hz, since Program15_2 outputs once a second, the data observed on the terminal program can be considered to have been sampled at 1 Hz.

1) Observing one sensor output, oscillate the wall (block of wood) 100 to 200 mm from the sensor with a period of 4 to 10 seconds. Notice data tracks the signal in such a manner that you could reconstruct both the frequency and amplitude of the oscillations.

2) Very carefully attempt to oscillate the wall at a constant amplitude but with a frequency of 0.5 Hz (period of 2 sec). At this frequency the sampled data will oscillate 100,200,100,200,100… This is the Nyquist frequency (½ f_s) at which the system transitions from operational region (able to recover amplitude and frequency) to a region at which the digital data cannot be used to recover the amplitude and frequency of the oscillations.

3) Finally, oscillate the wall at a constant amplitude but with a frequency much faster than 0.5 Hz. Data existing at frequencies above ½ f_s will be aliased (frequency folded into 0 to 0.5Hz), and the digital data cannot be used to recover the amplitude and frequency of the oscillations. The problem with aliasing is that high frequency noise will appear as low frequency signals. Therefore we must remove high amplitude signals at or above ½ $\mathbf{f_s}$ using an analog low pass filter.

Lab: Data Acquisition Systems

15.5 Troubleshooting

IR distance sensors don't work:

- Check the wiring as described in Figure 4,
- Using a voltmeter observe power (+5V), ground, and signal directly on the sensor

ADC doesn't work:

- Run the **ADCSWTrigger**. This project should properly sample channel 6
- Review the data sheet and double check each register accessed by your software

ISR takes more than 25us to execute:

- Notice that this implementation of the filter requires about 9us for one channel and 25 us for three channels. This is because each filter calculation requires one subtraction, one addition and one division.
- Make sure there are no loops in the ISR (except for the busy-wait in the ADC sampling).

Statistical calculations are incorrect:

- Look at the collected data in the array Data[N]. Bad statistics may be a result of bad data.
- You can reduce the statistical array to N=8, and perform the statistical calculated by hand to check the software.
- 100*Sum2 can overflow if the data is very noisy.

15.6 Things to think about

In this section, we list thought questions to consider after completing this lab. These questions are meant to test your understanding of the concepts in this lab.

- What is the mathematic relationship between ADC input voltage and digital output number?
- What is the limiting factor in this system that restricts resolution and accuracy of the distance measurement?
- Why did the function **ADC_In17_12_16** use call by reference parameter passing?
- What happens as the size of the averaging filter is doubled?
- Why are interrupts required in this lab? i.e., what do interrupts enable us to do?
- How is the mailbox used in the lab? What does ADCflag=0 mean? What does ADCflag=1 mean?
- What would it mean if the ADCflag were already 1 at the time the ISR is trying to set it to 1 again?
-

15.7 Additional challenges

In this section, we list additional activities you could do to further explore the concepts of this module. You could extend the system or propose something completely different. For example,

- If you add three analog filters between the sensor output and ADC input (fc=100 Hz) you can greatly reduce the noise and could also reduce the sampling frequency and size of the digital filter.
- A median filter is an alternate digital filter than could be added to improve signal to noise ratio.
- If you performed Lab 11 (LCD), then you could output data to the LCD, making it easier to debug, calibrate, and test.

Lab: Data Acquisition Systems

15.8 Which modules are next?

This was our first of many uses of interrupts in this course. The following modules will build on this module:

Module 16) Interface tachometers to the microcontroller and use input capture to measure wheel velocity.
Module 17) Combine modules 12, 13 and 16 to develop closed loop motor controllers. In this module you will be able to spin the motors at a constant speed.

15.9 Things you should have learned

In this section, we review the important concepts you should have learned in this module how to:

- Use periodic interrupts to implement sampling
- Use the ADC to convert from analog to digital domain
- Noise is difficult and important problem to solve; with a constant voltage connected the LaunchPad, noise will limit the 14-bit ADC to 10 or 11 bits; noise is often the limiting factor for resolution and not the number of bits in the ADC
- Software can efficiently and effectively implement filtering
- Software can effectively handle a nonlinear (hyperbolic) transducer
- Accuracy depends on two processes: resolution and calibration. Resolution depends on noise, and calibration depends on the stability of the transducer.

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2018, Texas Instruments Incorporated