ECSESS Robotics Club

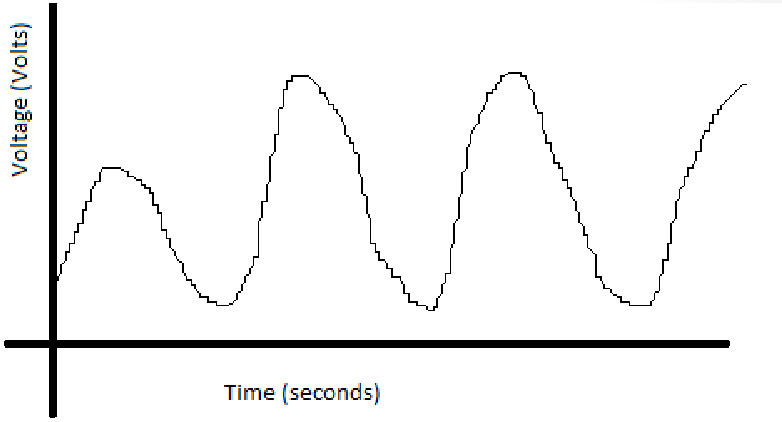
Fundamentals of Analog to Digital Converters (ADC)

Topics Covered

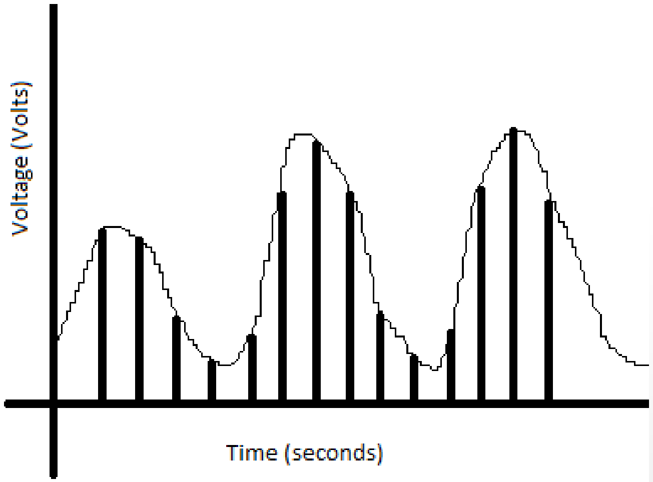
- ADC concepts
- Configuring and using the ADC
- Using the ADC to read a voltage
  - An aside about mults, divs, and float operations

Analog to Digital Converters

- Up to now we are able to read a voltage as either one or zero
- What if we want to read an analog voltage
  - i.e a voltage that is continuous
- The analog to digital converter is a special hardware peripheral on board the micro controller that lets us measure continuous voltages.
- We want to bracket the continuous voltage into discrete values (digital)
  - Each discrete voltage sample can then be associated with a number



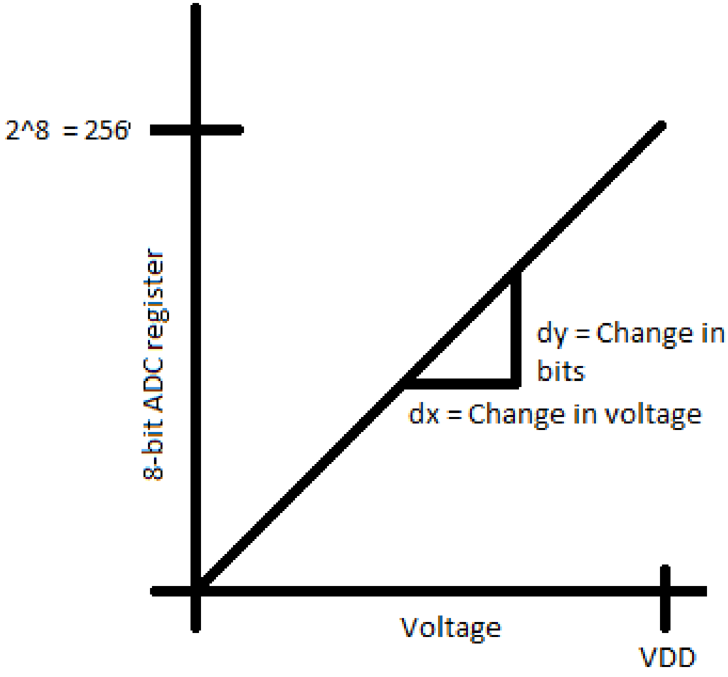
A continuous voltage signal



A discretized voltage signal

Analog to Digital Converters continued

- ADCs use a fixed voltage range to establish a reference to measure the analog voltage
- When we sample a voltage the ADC returns a bit value, a number between 0 and 2^8 (256)
  - We need to convert this to a real world voltage that has meaning
  - We can use a linear scale to map the bit value of the ADC to the actual voltage



Bit Value = (dy/dx)*Voltage

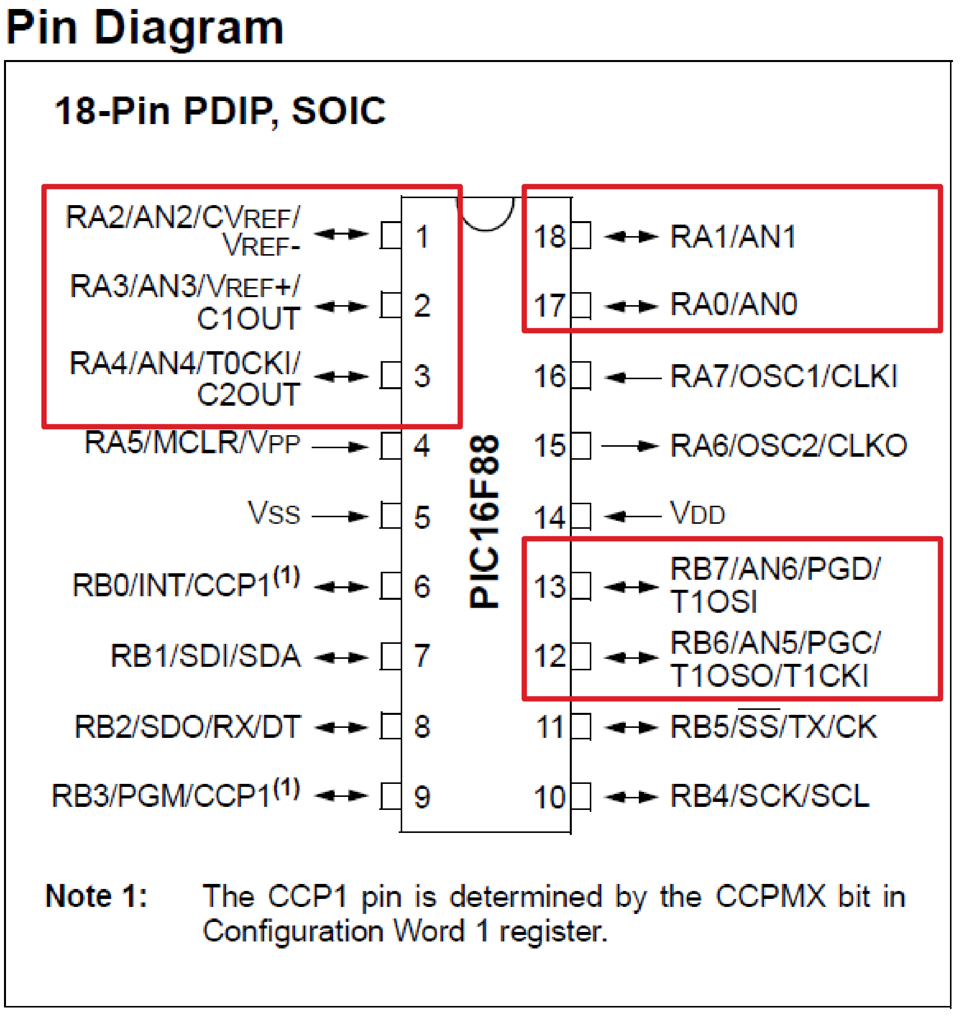
Voltage = (dx/dy)*Bit Value

Dx/dy = (5 – 0)/(256 – 0)

Voltage = (5/256) *Bit Value

ADC on the PIC16F88

- The pins marked AN(X), are known as analog pins
- One of these pins can be connected to the ADC at a time
- Note, both PORTA and PORTB have analog points
  - Also, the AN number doesn’t not necessarily match the PORT number



Note 1: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

ADC Library Functions

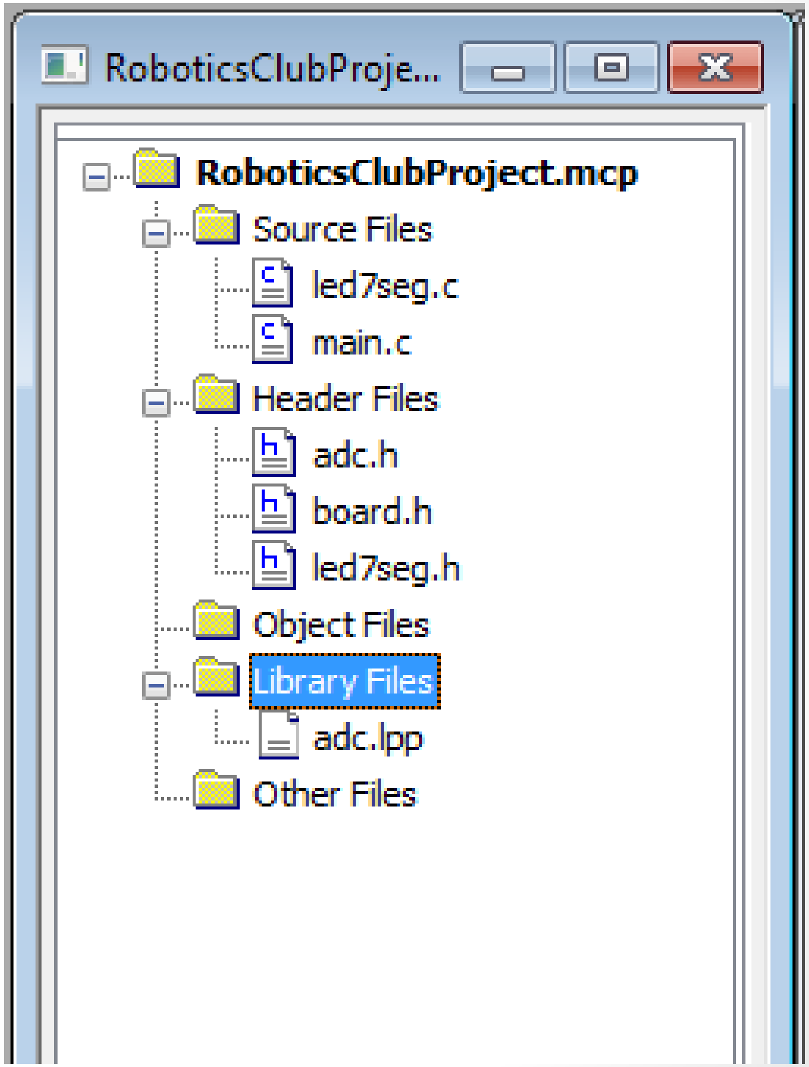
- There are two functions available to you in the ADC library
- The initialization function configures an analog port for use with the ADC
- The other function reads the ADC and returns to you an 8bit value

```
/***@brief An initalization function for the ADC
*@param[in] channel The analog channel to use with the adc
*@retval None
*@note Adjustes the ANSEL and TRIS settings for the
corresponding channel*/
void adcInit(unsigned char channel);

/***@brief A function the reads the ADC value
*@retval The bit value of the adc
*@note This function uses a busy wait for conversion to
complete*/
unsigned int adcRead(void);
```

Configuring your ADC with our library

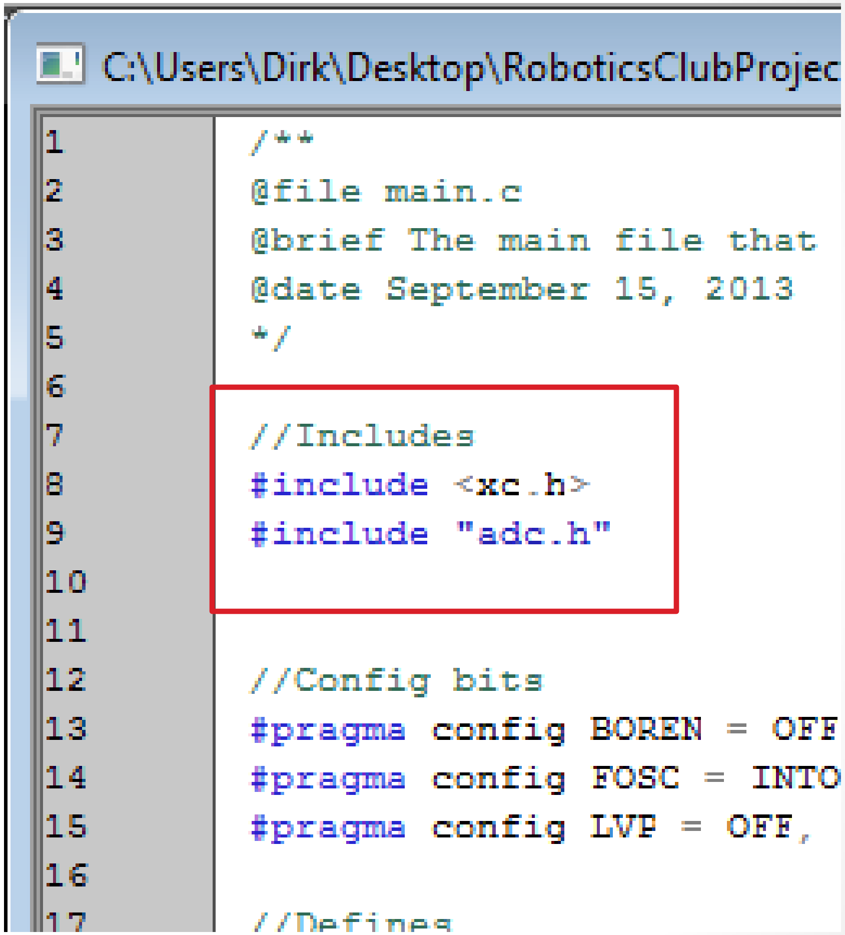
- We are providing you with an ADC library that sets up the ADC peripheral.
- You will have to include the header file(.h) and library (.lpp) in your MPLab Project
    - Right mouse click on the folder and select add files
    - Add the files unzipped from the website
- Once included in your project, you can use the functions listed in the “adc.h”
    - All the functions are fully documented in the .h file, look there for more information



```
RoboticsClubProje...
RoboticsClubProject.mcp
    Source Files
        led7seg.c
        main.c
    Header Files
        adc.h
        board.h
        led7seg.h
    Object Files
    Library Files
        adc.lpp
    Other Files
```

Using the ADC to measure a voltage

- Once the library and header file have been added to your project, you will need to include the adc.h header file in your main.c code
  - This tells the compiler that you want to use functions from this library
  - The files should be in the same directory as your main.c file
  - When you include the file you use “ “ to indicate that the file is in the local directory



C:\Users\Dirk\Desktop\RoboticsClubProjec

```
/**
@file main.c
@brief The main file that
@date September 15, 2013
*/

//Includes
#include <xc.h>
#include "adc.h"


//Config bits
#pragma config BOREN = OFF
#pragma config FOSC = INTO
#pragma config LVP = OFF,

//Defines
```

Multiplying, Dividing, and Floats

- Since this is a simple micro controller with limited memory and processing power, it does not have a dedicated floating point unit.
  - This means doing floating point arithmetic is very expensive. It is better to work with integers
- Multiplying and dividing a number on a micro controller can be a very expensive operation. It can take many cycles to multiply two numbers and even more to divide two numbers.
- It is best to avoid divides all together
  - This can be done by doing offline computations by hand and replacing a divide operation with a multiply

Resources

- Datasheets
  - PIC16F88: http://ww1.microchip.com/downloads/en/DeviceDoc/30487D.pdf
    - For the ADC see the subsection about ADCs in the data sheet
- Further Reading
  - Sparkfun Tutorial: https://learn.sparkfun.com/tutorials/analog-to-digital-conversion