Voltage Measurement with A PIC Microcontroller

Ryan Popa 03/30/2012 Design Team 3

Abstract

The purpose of this application note is to explain how to measure a voltage using a PIC18F4520 microcontroller. This note will start by explaining how to correctly power and wire the microcontroller for sensing voltage. It will then explain the code needed to measure a voltage and how it works.

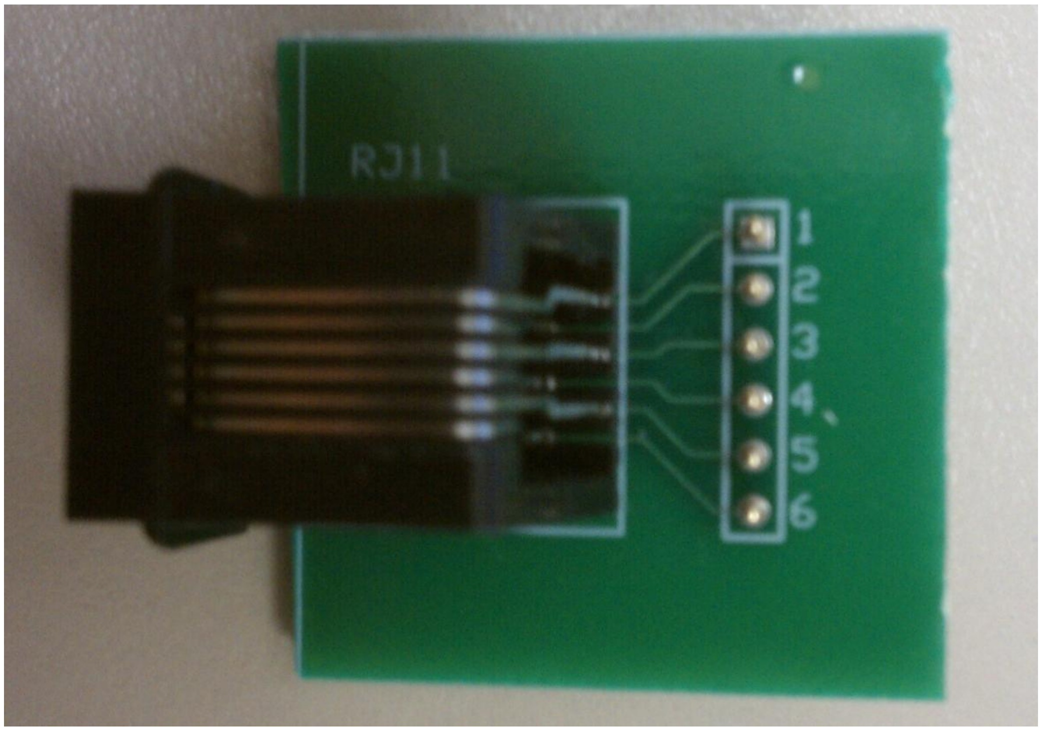
Keywords: PIC, voltage sensing, voltage measurement, PIC18F4520

Background

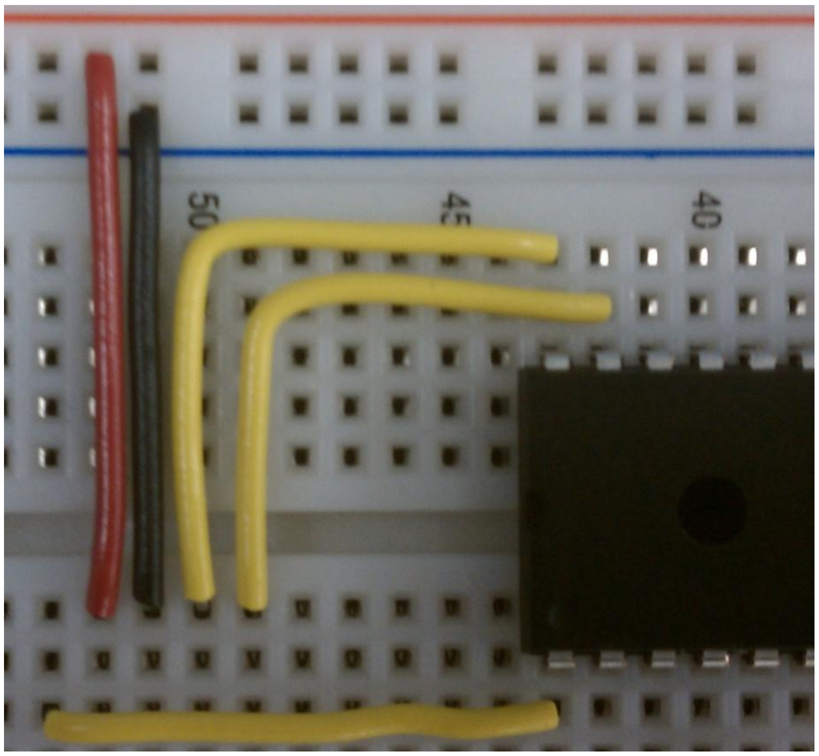
The PIC18F4520 is a 40pin microcontroller with numerous functions and applications. One of its many functions is analog to digital conversion. Through this the PIC is capable of reading an analog voltage and turning it into a digital readout that can be used in a program. The circuitry and code required to create a basic voltage sensor is minimal. This note assumes a basic knowledge of how to create a PIC program and load it from the computer to the microcontroller. The objective of this note is to show how to wire a RJ11 jack to a PIC to allow communication with a computer, how to wire the PIC to power, how to connect an external clock to a PIC and where to connect a voltage source to read its value. It will also provide a small program to read in a voltage and convert it into a digital form for use within the program.

Connecting the PIC

Before any code can be written the PIC must first be connected and powered on. Minimal wiring for PIC operation is connecting it to power and wiring a communication port to be connected to a computer. The PIC18F4520 uses a 5V DC power source and communicates with a computer through a RJ11 jack like the one shown below, the pin layout of the jack below will be used as a reference for wiring the PIC in the rest of this note.

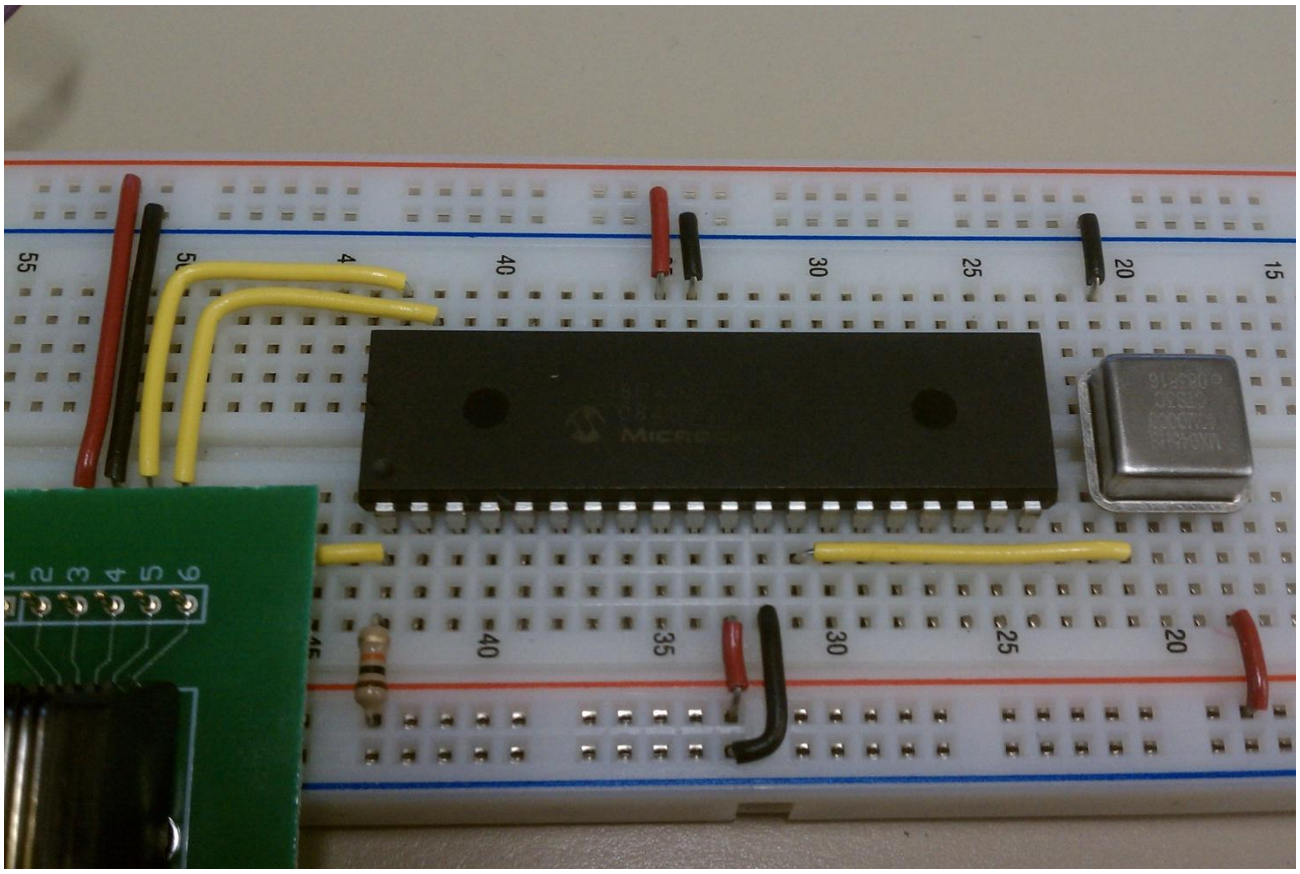


Connect pins 1, 4 and 5 on the RJ11 jack to pins 1, 40 and 39 on the PIC respectively. Pin 2 on the jack should be wired to VDD and pin 3 should be wired to ground. A properly wired setup is shown below where the RJ11 jack is connected on the bottom with pins 1 – 6 wired from left to right. Pin 6 on the RJ11 jack is not used.



The PIC microcontroller will also need to be wired to power in order to work properly. Pins 11 and 32 should be wired to VDD and pins 12 and 31 should be wired to ground. The last thing the controller needs to work properly is a clock. The PIC18F4520 does not have an internal clock so a 40M000 clock like the one below will be connected to the microcontroller.

The bottom left corner of the clock has a little notch cut out of it to indicate its orientation. The bottom right pin of the clock needs to be wired to ground and the top left pin needs to be wired to VDD. The top right pin of the clock should be wired to pin 13 on the microcontroller. Lastly a 10KΩ resistor needs to be wired from pin 1 on the PIC to VDD. This resistor is acting as a pull-up resistor and keeps the microcontroller out of the reset state. A fully wired RJ11, PIC microcontroller and 40M000 clock is shown below.



Writing the Code

Now that the microcontroller is powered and hooked up the code needs to be loaded to measure the voltage. The analog voltage will be connected to pin 2 of the PIC, and use VDD as a reference value. The digital value of the voltage will be read as an integer and the equivalent analog voltage can be calculated as a fraction of VDD. The voltage can be calculated as

*(x/1024) * VDD.*

As an example: using a VDD of 5V, the value returned from pin 2 of the PIC is 884. This means the analog voltage at pin 2 is

*(884/1024) * 5V =* ***4.32V***

The annotated code to do this is shown below.

```
#include <p18cxxx.h>
#include <usart.h>
#include <ADC.h>
#pragma config LVP=OFF
#pragma config WDT=OFF
void rx_handler (void); //Declare the ISR function
int adc_result; //The variable to store the ADC value in

void main()
{
       RCONbits.IPEN = 1; /* Enable interrupt priority */
       IPR1bits.RCIP = 1; /* Make receive interrupt high priority */
       INTCONbits.GIEH = 1; /* Enable all high priority interrupts */
       OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD,
       ADC_CH0 & ADC_INT_OFF, 0); //open adc port for reading
       ADCON1 =0x00; //set VREF+ to VDD and VREF- to GND (VSS)

       while(1)
       {
              ConvertADC(); //perform ADC conversion
              while(BusyADC()); //wait for result
              adc_result = ReadADC(); //get ADC result

              ////////////////////////////////////////////////////////////////////////
              // adc_result is the digital representation of the voltage at pin2
              // it is a number out of 1024 that specifies a fraction of VDD
              ////////////////////////////////////////////////////////////////////////

       }
}
```

Conclusion

This application note showed how to wire a PIC18F4520 microcontroller for basic voltage sensing and provided some code to read a voltage from pin 2 of the microcontroller. Also, it explained how to calculate an analog value from the digital result of the ADC measurement. This technique can be combined with any other amount of code to make a multifunction PIC program capable of reacting to changes in an analog voltage input.

References

PIC18F4520 Data sheet: http://plc.mechatronika.hu/piclei/18F24-25-44-45_20.pdf

Information about ADC: http://extremeelectronics.co.in/microchip-pic-tutorials/using-analog-to-digital-converter-%E2%80%93-pic-microcontroller-tutorial/

Capstone lab about PIC: http://www.egr.msu.edu/classes/ece480/capstone/ForMiniprojects/Lab3.pdf