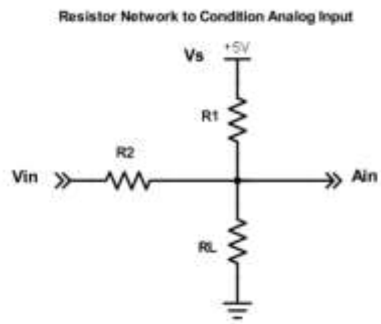
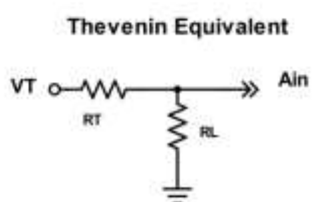
Analyzing a Resistor Network to Condition ADC Inputs for Negative and Out of Range Voltages for Microcontrollers such as the Arduino Uno

Microcontroller analog to digital converters (ADC) typically accept voltage inputs only in the range of zero to the positive supply voltage. Voltages outside of that range are likely to damage the device. Obviously, it is possible to condition the input voltage source with a preamplifier in order to keep the voltage applied to the ADC in the required range, but in some cases a simple resistor network and offset voltage may suffice. There are a number of contributions to the Arduino forums that indicate such a solution, but they oversimplify the result because they fail to account for the interaction of the input voltage with the offset voltage through the resistor network. Consider the following resistor network:



The positive supply voltage is applied to the ADC input pin of the microcontroller (*Ain*) through a voltage divider (*R1+RL*) which should offset, to some extent, negative voltage swings at *Vin*. But calculation of the relationship between the signal voltage at *Vin* and the *Ain* input voltage to the ADC is not as simple as computing the ratio from the voltage divider. There are many techniques for network or nodal analysis, but I prefer using Thevenin equivalent analysis since it is simple and straight forward.

Thevenin's theorem is that any linear circuit can be reduced to a single voltage source (*VT*) and a series resistor (*RT*) connected to a load. Thus in the circuit above can be reduced to the following:



VT is determined by removing the load resistor (*RL*) and calculating the voltage at the node between *R1* and *R2*. This is simply the voltage of *Vs* less the drop across *R1*:

$$VT = Vs - \frac{(Vs - Vin)R1}{R1 + R2}$$

RT is determined again with the load resistor (*RL*) removed and by shorting the voltage sources to ground then calculating the resistance from the node between *R1* and *R2* to ground. This would be the parallel resistance:

$$RT = \frac{R1R2}{R1 + R2}$$

The voltage (*Ain*) at the input pin of the ADC can be calculated from the Thevenin equivalent circuit as the voltage drop across the load resistor (*RL*):

$$Ain = VT(\frac{RL}{RL + RT})$$

Expressing *Ain* in terms of the elements in the original circuit:

$$Ain = \left[Vs - \frac{(Vs - Vin)R1}{R1 + R2}\right]\left[\frac{RL}{RL + \frac{R1R2}{R1 + R2}}\right]$$

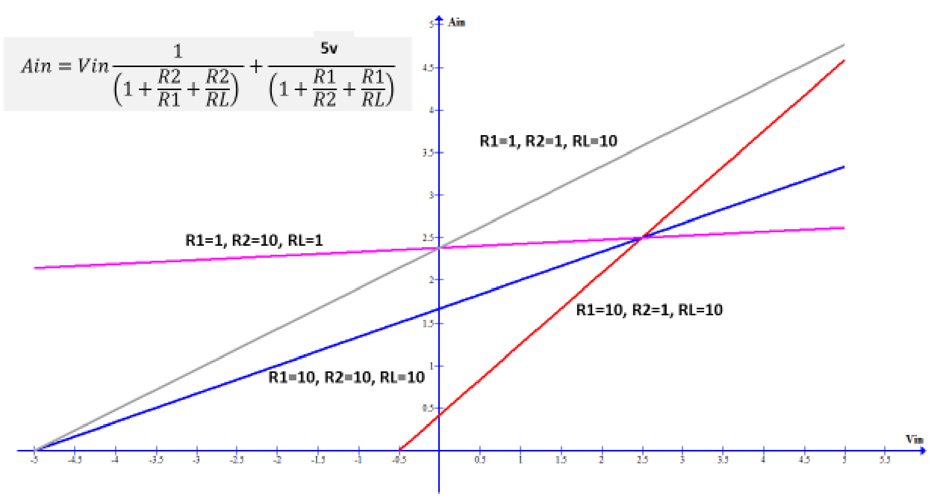
Combining terms and simplifying, the expression *Ain* is a linear function of *Vin* if *Vs* is held to a constant value. This is the well-known straight line equation of the form y=mx+b.

$$Ain = Vin\frac{1}{\left(1 + \frac{R2}{R1} + \frac{R2}{RL}\right)} + \frac{Vs}{\left(1 + \frac{R1}{R2} + \frac{R1}{RL}\right)}$$

Inspection of the equation suggests that high values of *R2* relative to *R1* and *RL* would decrease the slope and thus the sensitivity of the input, and push the y intercept (*Ain* when *Vin* = 0). Reversing the resistor ratios would do the just the opposite.

For the Arduino Uno, resistances should be greater than 270 ohms so as not to exceed the maximum current drain limit of 40 ma – 1K ohms is suggested. The input resistance *R2* should not exceed 10K ohms in order to insure full charging of the sample and hold capacitor in the ADC with measurement cycle. *Vs* can be conveniently set to 5 volts from the Vdd pin on the board.

The relationship of *Vin* to *Ain* is plotted below for various resistor values (K ohms).



Donald McAfee, PhD
McAfeeScientific.com

Intuitively, one might think that if *R1=RL, Vs* would be divided in half and the voltage at *Ain* would be 2.5 volts when *Vin* was zero. This only occurs when *R2* is relatively large. Furthermore, this has the effect of voltage dividing the input signal and reducing the slope or the sensitivity of the *Vin, Ain* relationship.

The table below provides the ranges of *Vin* with all combinations of 1K and 10K resistors that will keep Ain voltages between 0 and 5 vdc. The Arduino Uno uses a 10 bit ADC which provides 1024 bits of resolution. Thus it is possible to have an input voltage (*Vin*) range from -50 to +55 vdc with a sensitivity of 103 mv/bit, relative to the unconditioned input (*R1,RL* absent) of 0 to +5 vdc with a sensitivity of 5 mv/bit.

The Range of Vin with Ain Limited 0 to 5 volts

Vs=5v			**Vin if:**		**Sensitivity**
R1	**R2**	**RL**	**Ain=0v**	**Ain=5v**	**mv/bit**
10	**10**	**10**	-5	10	15
10	10	1	-5	55	59
10	1	10	-0.5	5.5	6
1	10	10	-50	10	59
10	1	1	-0.5	10	10
1	1	10	-5	5.5	10
1	10	1	-50	55	103
1	1	1	-5	10	15
1000	1	1000	0	5	5