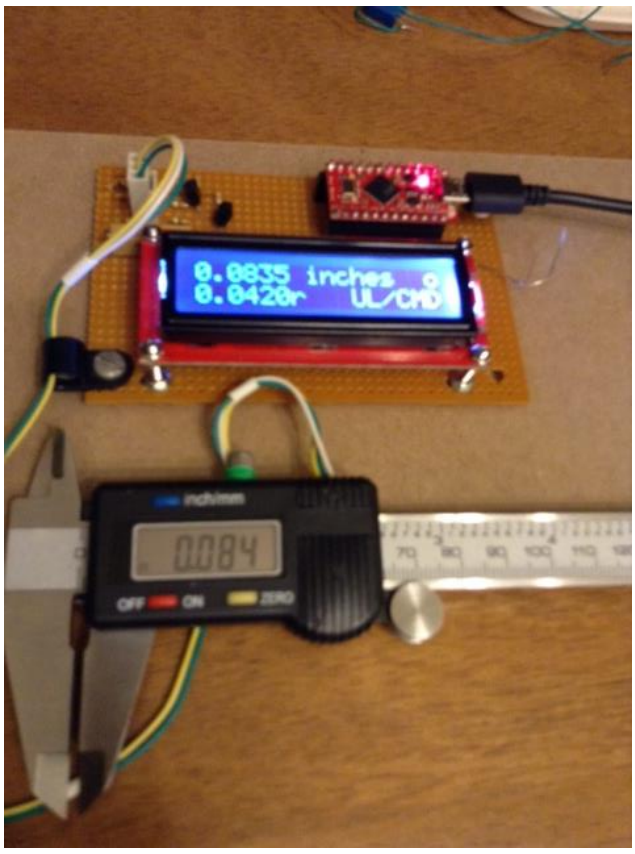
An Experimental Measurement System, version 2.3

By R. G. Sparber

Conclusion

Using a low cost Harbor Freight® digital caliper enhanced by software, I was able to create a measurement instrument accurate to ± 0.0005 inches over a 5.5 inch range. Further performance enhancements are also provided.

System Overview



I was curious to know if software could be used to improve the accuracy of a low cost digital caliper. Of equal importance was that I wanted a project that would justify learning about Arduinos®[2].

This experience showed me that digital caliper accuracy could be improved. I also learned that Arduinos are a very useful addition to my "bag of tricks".

What started out as an experiment has grown into a rather large project. The software has many features that might turn out to be useful.

[1] You are free to copy and distribute this document but not change it.
[2] Arduinos are a large family of development boards which mostly use AVR microprocessors. A set of libraries and a simple Integrated Development Environment are available for free to make writing code easier.

Software Overview

The default display shows decimal inches in the upper left hand corner. Below it is half the value rounded to the nearest half thou. If you use a lathe with in-feed dial in radius, then you can zero the jaws of the caliper at the target diameter and this will tell you how much radius you have left to go.

The upper right corner shows a small circle. This indicates that the caliper is running uncalibrated. The value on the caliper is displayed uncorrected. When a "*" is displayed, the numbers are an interpolation between gage blocks that have been measured.

The text in the bottom right corner tells the user that they are in autolock mode and can go into command (CMD) mode. By moving the caliper jaws more than 0.1" you "UnLock" (UL) it and will be able to get a new reading. By default, the system measures outside diameters. It does this by scanning about 8 times a second looking for the smallest value. This means that you can place the calipers on the OD approximately perpendicular to the diameter and sweep through a range of angles. The system will lock in the minimum.

If the user presses the inch/mm button while the UL/CMD is displayed, they are offered a choice of features. Opening the jaws to around 1" selects the calibration mode. This enables the user to measure a series of gage blocks and improve the accuracy of the system.

If they move to around 2", they enable the Go/No Go feature. This enables the user to measure two gage blocks which set limits. Subsequent measurements will cause the display to read "Below", "OK", or "Above". This feature has been designed but not coded yet.

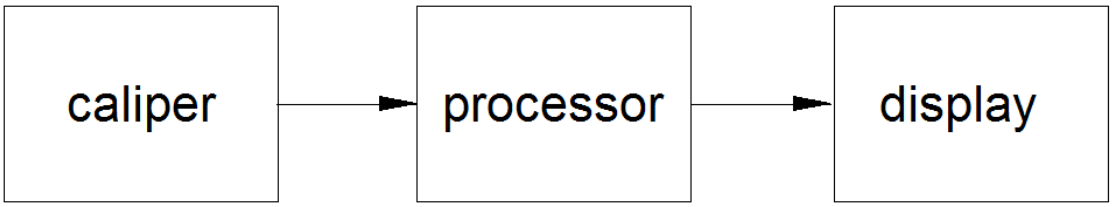
If the user moves to around 3", they enable the Averaging feature. The user can take up to 10 measurements of the same thing and the program will average the result and round to the nearest half thou. This further improves system accuracy. This feature has been designed but not coded yet.

If the user moves to around 4", they will change the display to show decimal inches and fractions of an inch up to thirty-seconds of an inch.

In this example, the fractional equivalent can be reduced to sixteenths.

The last function, OD/ID, controls what the system will measure. By default, it only measures Outside Diameter. The software can be set to measure only Outside Diameter, only Inside Diameter, or to measure both.

High Level Architecture



This measurement system consists of a $10 Harbor Freight® digital caliper, a $20 Arduino® - compatible Pro Micro processor, and a $15 Liquid Crystal Display. There is also about 18K of software hidden in that Arduino.

With this collection of hardware and software, it is able to measure any value between 0.0000" and 5.5000" to within ± 0.0005" after it was calibrated with the appropriate precision gage blocks.

The digital caliper, without modification, is advertised as being accurate to ± 0.001" with a resolution of ± 0.0005"[3].

The software is almost complete while the hardware would needs to be put in a protective enclosure before it could be used in a real shop environment.

[3] See http://www.harborfreight.com/6-inch-digital-caliper-47257.html for details. During the testing of this caliper with a set of spacer blocks accurate to ± 0.0001", I have seen errors as high as 0.0015" .

Contents

Background

The family of Arduino processors is impressive. They are really more of a system on a chip. It is sort of like a Lego® set for computers. You buy the hardware, snap the pieces together, load in lots of free software to make the hardware easy to drive, and write code to do what you want.

For the longest time I viewed this product line as an amazing solution to an unknown problem. It has taken me a long time to find an application that could justify the cost and time needed to get up to speed.

Many people have interfaced low cost digital calipers to various computers. A few have even done it with an Arduino. Part of the challenge is that Harbor Freight does not recognize the data port on this caliper and often changes it. So I had to first figure out how to read the data streaming from the caliper. That was documented in

http://rick.sparber.org/electronics/hf6.pdf

Connecting up the Pro Micro processor couldn't be easier. You take a USB cable, plug one end into the PC or Mac, and the other end into the little 0.7" x 1.3" board. Loading the correct drivers was a bit confusing but not hard. Loading the free software development environment was rather straightforward.

Then the fun began. I had a few ideas for features and knew that more would come to mind as I went along.

What does the system do?

I will defer to these two YouTube videos to answer this question:

For the basic operation, see

http://www.youtube.com/watch?v=GPhPrXT5kwY&feature=c4-overview&list=UUQowQlSfFxybveyBDVOHXxw

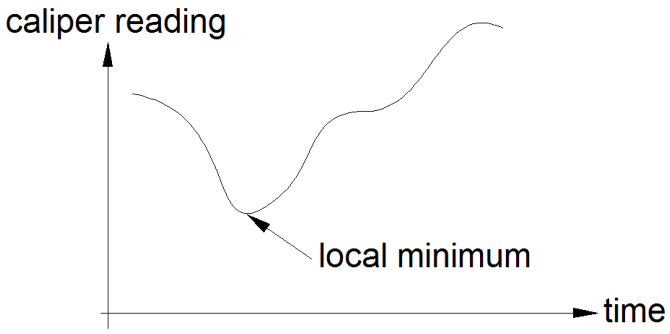
To see how calibration is done with gage blocks, see

http://www.youtube.com/watch?v=c3amC9C7Ww4&feature=c4-overview&list=UUQowQlSfFxybveyBDVOHXxw

High Level View of the Theory Behind the Software

There are a few layers of functionality that come together here.

Automatic Local Minimum Detection



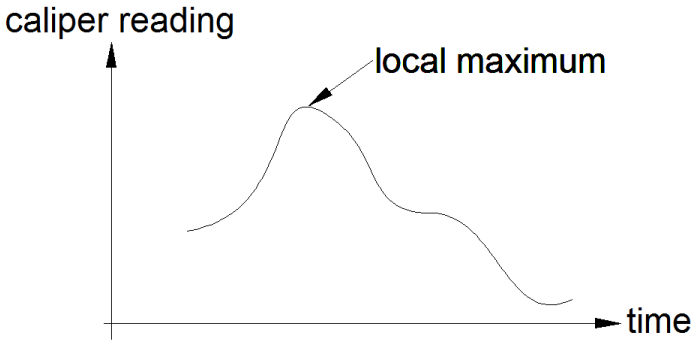
As the jaws close, the caliper reading shrinks in value. Then the jaws contact the workpiece and they can't close anymore. This is the point of local minimum. When the jaws start to open, the caliper reading rises. The program records the caliper reading 8 times per second looking for this local minimum. When it sees it, that value is recorded.

Not all local minima are preceded by a constantly decreasing set of data points. The user may start to close the jaws, hesitate, and then close them a bit more. The software is designed to handle this case.

Note that a person skilled in the use of a caliper will be able to get to this local minimum directly because they hold the caliper perpendicular to the workpiece.

A novice might have trouble finding that exact position. In this case, the program helps because as long as the measurements include the local minimum, there is no reason to hold at that point. So it is possible to sweep from less than 90 ° to above 90 ° and someplace in there the local minimum will be found.

Automatic Local Maximum Detection



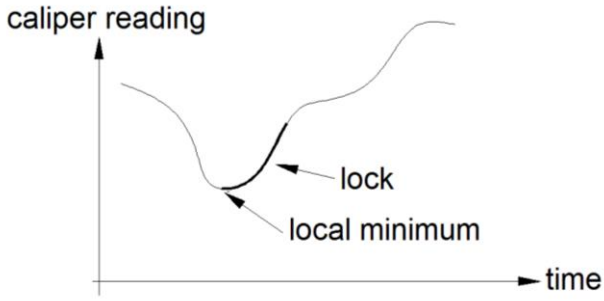
This is the twin of Automatic Local Minimum Detection. Rather than detecting the best estimate of outside diameter, we are dealing with inside diameter. The jaws open until they contact the walls of a bore. The maximum reading is recorded. Then the jaws close as we remove the caliper.

One of the functions available to the user permits them to measure only outside diameters (local minimum), only inside diameters (local maximum), or both.

One subtle limitation of measuring local minimum or maximum is dealing with measurement jitter. It is possible to set the jaws so the half thou digit jitters between 0 and 5. This jitter is enough to trick the software into thinking it has reached a local max or min. We do not want to ignore this digit because we are going for the best possible accuracy. The solution is to keep the jaws moving to touchdown at a rate of at least 0.0005" per 0.15 seconds which is 0.003" per second. This is not hard to do and does not crash the jaws into the surfaces being measured.

Automatic Lock

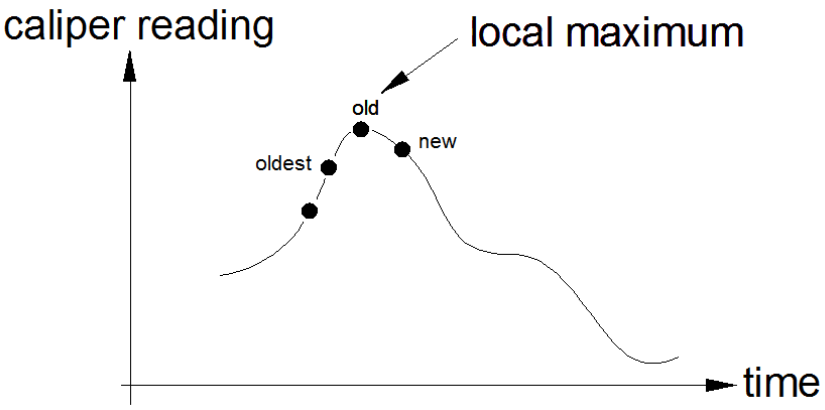
Better digital calipers have a hold button. Touch it at any time and it will freeze the current reading. My problem was that I often bumped the jaws of the caliper while I attempted to push the hold button. This led me to develop an automatic lock function. As long as the jaws are not moved more than 0.100", the last reading is held. This is true for both inside diameter and outside diameter measurements.



Here you see automatic lock enabled after a local minimum has been detected. Lock is disabled when the caliper readings have increased more than 0.100" from the local minimum reading. It would have also unlocked if the jaws have been closed by more than 0.100".

Automatic lock works the same way when a local maximum is detected. Lock is disabled when the caliper readings have changed by more than 0.100" from the local maximum reading.

Velocity Check



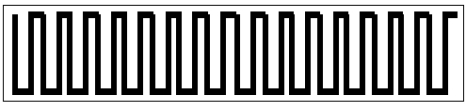
Each time a data point is recorded, the time is also saved. When the software sees a reversal in direction of the data, it knows the middle one is the local minimum or maximum. You see here a data point marked "new". It was just recorded. About 1/8th of a second before it the data point "old" was recorded. About 1/8th of a second before "old", "oldest" was recorded. This up and then down pattern of data identifies "old" as the local maximum.

The distance traveled from "oldest" to "old" is divided by the time it took to the nearest millisecond. The result is the velocity of the jaws as they approached touchdown. If the jaws were going faster than 0.100" per second, a warning is flashed on the display saying "Jaws hit at: " and the velocity. This warning has no effect on the recorded data. I see this feature as a possible training aid for novices.

Calibrated Measurements

Before we dive into calibration, let's look at what we are calibrating: a Harbor Freight 6" digital caliper. The spec sheet says it has a resolution of 0.0005". This means the smallest digit can be a 0 or a 5.

Accuracy is listed as ± 0.001". Without further information, I have to assume they mean absolute accuracy over the full range of the caliper. This is substantially better than previous versions[4].



These digital calipers depend on a strip of metal for their accuracy. This strip has been "precision" etched with a pattern. The pattern is typically defined using photo lithographic process. Knowing this gives us a hint at the type of error to expect. Rather than large, quick jumps in error, expect a slow and smooth change as the jaws are moved.

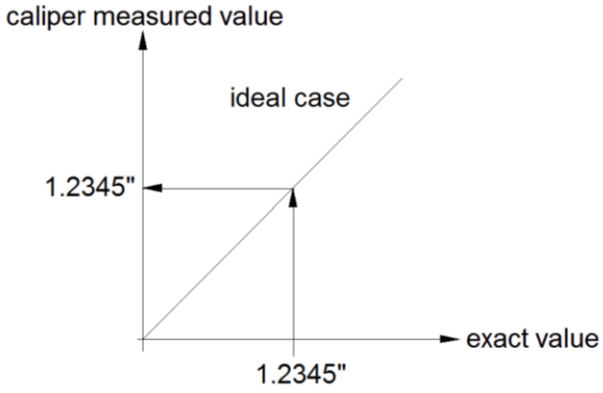
Spacer block value, inches	Caliper error, inches
0.0500 through 0.070	-0.001
0.0800	-0.0005
0.0900 through 0.150	-0.001
0.1600 through 0.240	-0.0005
0.2500 through 0.270	-0.001
0.2800 through 0.300	-0.0005
0.310 through 0.350	0
0.360 through 0.390	0.0005
0.400	0

For example, here is the error behavior of my caliper over the range of 0.050" to 0.400" in steps of 0.010". My spacer blocks have a tolerance of ± 0.0001".

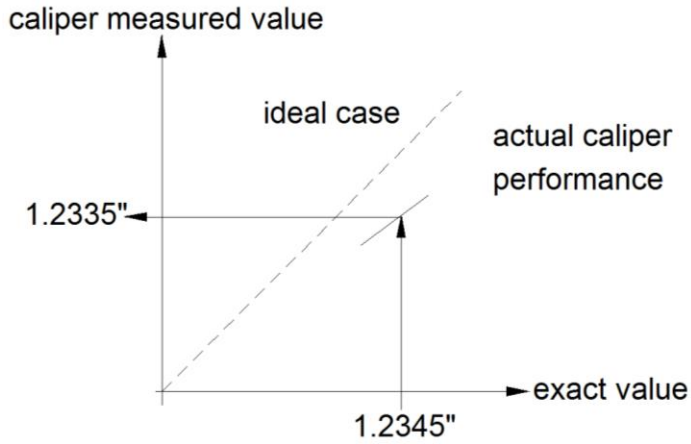
When the caliper is zeroed with the jaws closed, I get zero caliper error. This error is at -1 thou at 0.0500" which is the thinnest spacer block I own. The error stays constant for many tens of thou at a time. The worst case variation is +0.0005" -0.001". So at least from 0.0500" to 0.400", this caliper is in spec.

[4] See http://rick.sparber.org/Articles/PAC/PAC3.pdf

Knowing this behavior makes it possible to develop a means of reducing absolute error.



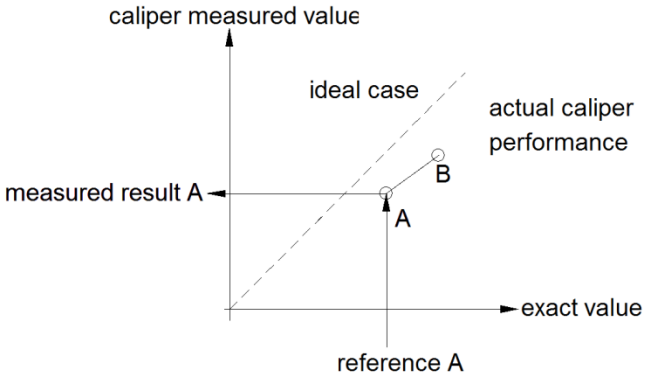
Consider the idea case. Say I measure a precision gage block that is 1.2345" thick. I see this value on the slider's display. The exact value equals the caliper measured value. I am ignoring limitations in the display.



Now look at how caliper error changes the story. We again measure 1.2345" but the display reads 0.001" low.

You could note this error and move on to measuring the workpiece. Say that reading came out to 1.2335" too. This would tell you to add 0.001" to yield the correct exact value of 1.2345".

But this calibration strategy is rather restrictive. What if the caliper reads 1.4005"? I didn't define a calibration point for that value. The answer is ***interpolation***.

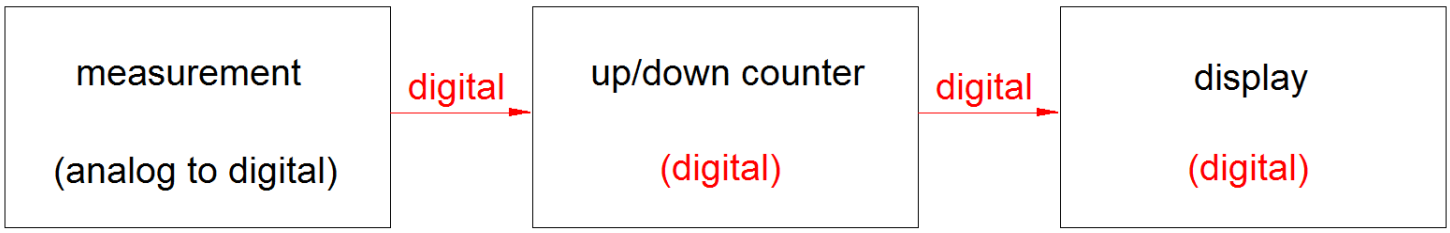


Say I measure a known reference called "A" and note the measured result. Repeat for reference B. This defines a range of values between A and B. If I measured an unknown that really was equal to reference A or B, I could apply exactly the correct compensation to the measured result. Between A and B, I can estimate the correct correction factor. The closer A and B are to each other, the better the estimation of the actual correction factor.

Alternately, if the correction factor was the same for A and B, then all estimates would be exact. For example, if I must subtract 0.001" from measured result A and B, then any measured result between A and B ***probably*** needs to have 0.001" knocked off of it. There are no guarantees here. Looking back at the table on page 11. Note that with a spacer block of 0.0800" the error was half a thou smaller than the rest of the readings in its neighborhood.

Absolutely critical to being able to make this correction are two assumptions. The first is knowing that readings are repeatable. It is impossible to compensate for an error that changes randomly over time. The second assumption is that error changes in a straight line between two calibration points. This last assumption can be turned around to say that you must pick calibration points that are close enough together that the error changes linearly between them. Only through testing can we be sure we got this one right.

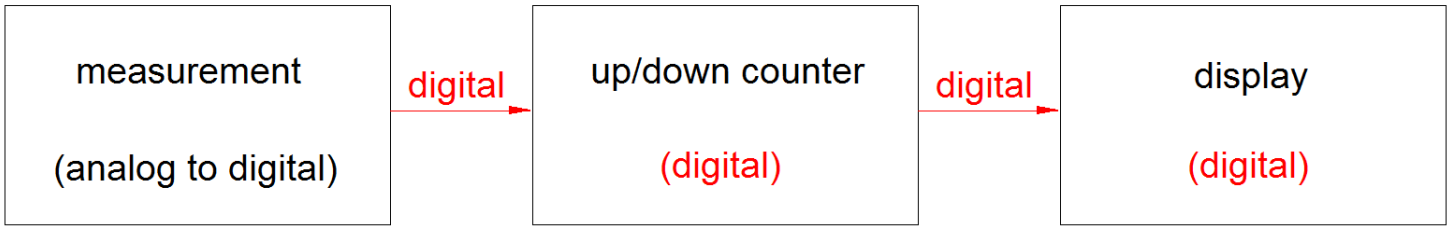
Let's now take a close look at error sources.



The caliper has three main functional parts. First it has a circuit that senses the strip of metal glued to the body of the caliper. It determines if the slider has moved some minimum distance and in what direction. The answer can only be one of three values: no motion, positive movement by this minimum distance, and negative movement by this minimum distance.

No matter how fast the slider is moved, this circuit can keep up so never has to report a movement more than this minimal distance at a time. However, the measurement function most likely does have "jitter" which means that with the jaws not moving, it would report a constant stream of a single positive followed by a single negative movement.

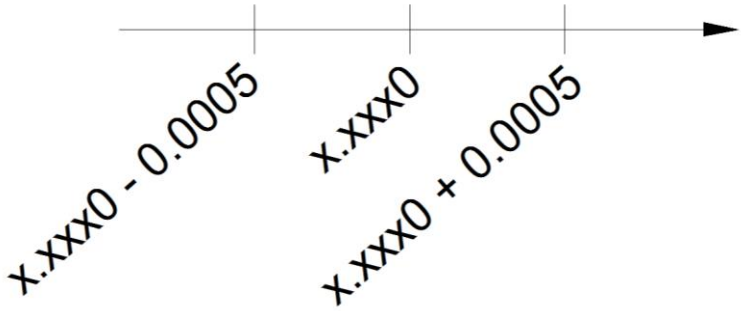
This result is fed into a counter that will increment one count if the movement was positive and decrement one count if negative. The output of this counter goes to the display.



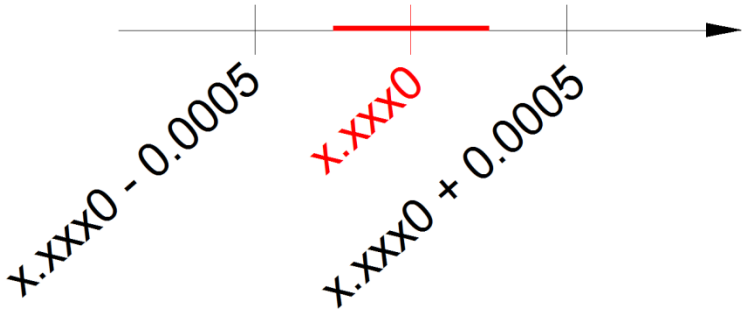
There are only two error sources here. The first is the measurement function which includes the error associated with the strip of metal glued to the body of the caliper and the sensor circuit built into the slider. I can further divide the measurement error into two parts. We have absolute accuracy and repeatability. Due to the way the Experimental Measurement System works, the absolute accuracy of the caliper does not matter.

What matters a lot is repeatability. ***There is no specification on just the repeatability of the measurement function so, for now, I will define it as ±m inches.*** Later on I will attempt to put a number to this unknown.

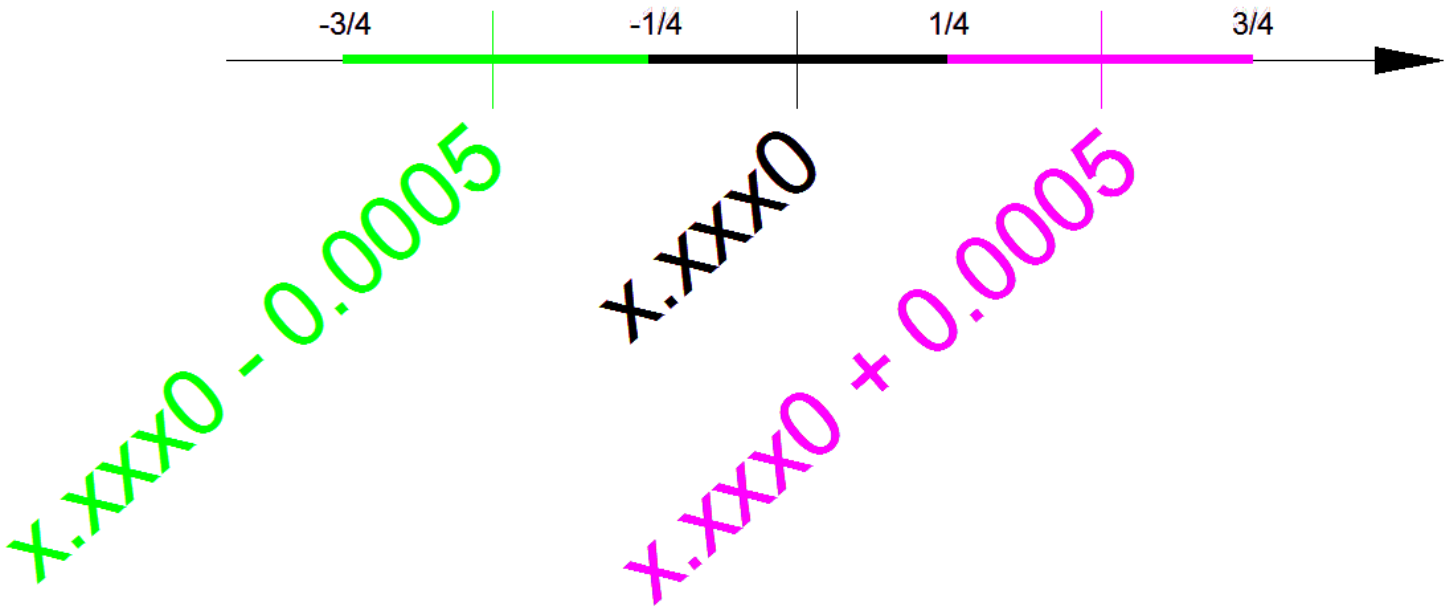
The second error source is due to the limited number of digits in the display.



If you look at the digital caliper's display, you will see that the smallest digit is limited to 0 or half a thou.

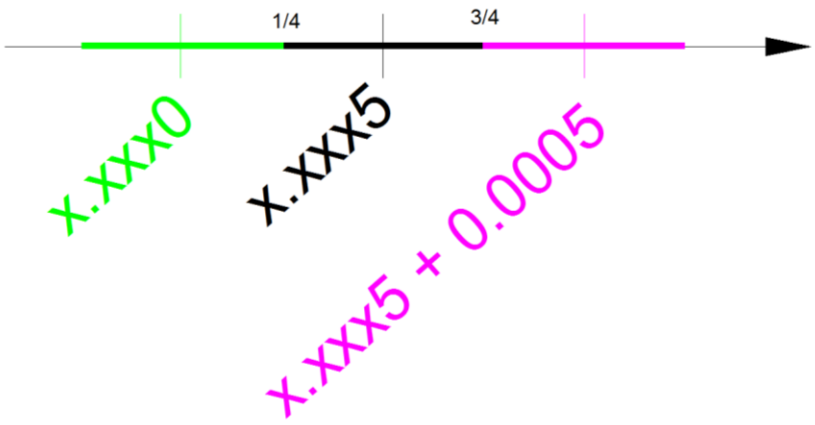


In the ideal case, my x.xxx0" reading represents actual measurements ¼ thou above x.xxx0" to ¼ thou below x.xxx0" as shown in red.

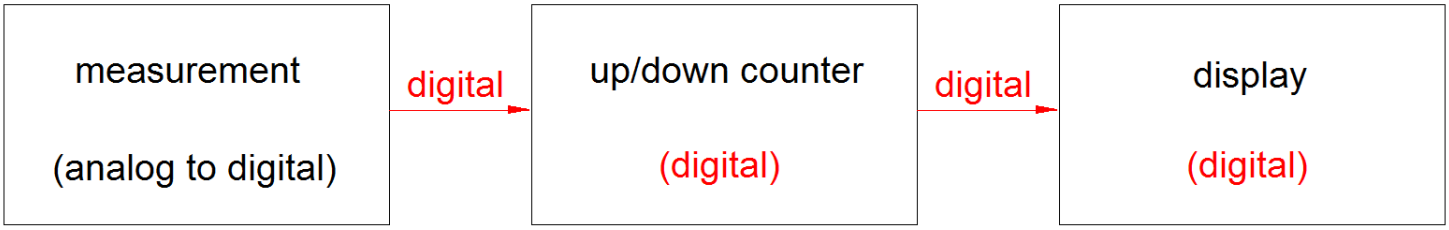


Looking at the full range, we can see that in the ideal case, each of the displayed smallest digits has uncertainty.

For example, say the caliper displayed 1.2345" and the only error was due to this limitation in the display. Then the actual measured value could be anywhere between 1.23425" and 1.23475". If the actual measured value was less than 1.23425", then ideally it would jump to the next lowest half thou value and display 1.2340". Similarly, if the value was slightly above 1.23475", the display would read 1.2350".



It is easy to adjust the caliper jaws to make the half thou digit flicker between half thou steps. You might, for example, see it flicker between x.xxx5 and x.xxx0 ***or*** between x.xxx5 and x.xxx5 + 0.0005. ***You would not see flicker across two boundaries like from x.xx0 to x.xx1.***



Recall that the measurement function has jitter which will drive the up/down counter to move up and down one step. This flicker means that the up/down counter is sitting right on a transition point of the display.

I am aware of only one way to reduce this flicker: filter the up/down counter output so changes are slowed down. A little filtering can be good but too much filtering will cause the system to be sluggish. You move the jaws and it takes a few seconds to see the results. Not acceptable to me.

You cannot solve this problem by rounding off the number of digits displays. For example, say the full display was 0.9995" and you were right on the boundary with 1.0000". Then flicker in the half thou position would cause the entire number to dance.

Note that there is no error associated with the position of these boundaries. They are in the digital circuits so are set by design. I therefore think it is reasonable to assume that the ideal transition points presented here are also the actual ones.

All of this discussion boils down to two errors that effect the overall accuracy of the Experimental Measurement System. The measurement circuit has a repeatability error of ± m inches and the display has an error of ± 0.00025 inches.

Complicating matters here is the fact that repeatability is also a function of the user. A person may slam the jaws together one time and do it gently the next time. The Velocity Check discussed on page 10 attempts to address this problem.

The issue of consistency between users is not a problem here. A given user first calibrates the Experimental Measurement System with gage blocks. Then they can do their measurements. As long as they are consistent, any user error due to user behavior is canceled.

The bottom line is that we do have a repeatability error that is ***at least*** ± 0.00025 inches. A search of the web turns up a few vendors claiming a repeatability of ± 0.0005 inches but with so much of this error dependent on the user, it is hard to accept this claim as the full picture.

Very limited testing of my particular caliper being used by me shows a worst case error of ± 0.0005 inches. See the appendix for details. I believe that the only way to estimate the repeatability for a given user and their caliper is to measure it.

Bottom Line: The Experimental Measurement System can cancel error that does not change. It cannot reduce error that randomly changes like that associated with repeatability.

It is time to look at some real performance data. I will give the results here. The appendix contains the raw data to back it up.

Between 0.1000" and 0.1100", my caliper reads 0.001" low. Using a 0.1000 ± 0.0001" spacer block, the caliper read 0.0990" for ten consecutive readings. Then I measured a 0.1100 ± 0.0001" spacer block and saw 0.1090" for ten consecutive readings.

I could see that the caliper was giving repeatable results. And if I added 0.001" to all caliper reads between 0.1000" and 0.1100", I might be able to compensate for the error.

Using spacer blocks from 0.1000" to 0.1090" in steps of 0.001", I took a lot of data. Each measurement was performed ten times.

The results can be presented in two ways. It is common metrology practice to take a "large" number of readings, throw away the highest and lowest values, and average the rest. When I did this, I found an error of +0 and - 0.0002".

A more severe way to evaluate the data is to look at all value and define the tolerance from them. This is called absolute worst case. It showed an error was ± 0.0005".

So in this test, I started out with a constant error of -0.001". The procedure was able to reduce the error to +0 and - 0.0002". This is an error reduction by at least a factor of 5.

If you prefer to look at absolute worst case error, then we went from -0.001 ± 0" to 0 ± 0.0005". This is a worst case error reduction of 50%.

High Level View of the Software

The code has been heavily commented plus contains a lot of diagnostic prints to the COM port so you can follow how it works. The software as a .txt file can be found at https://rick.sparber.org/SCC_1_6.txt .

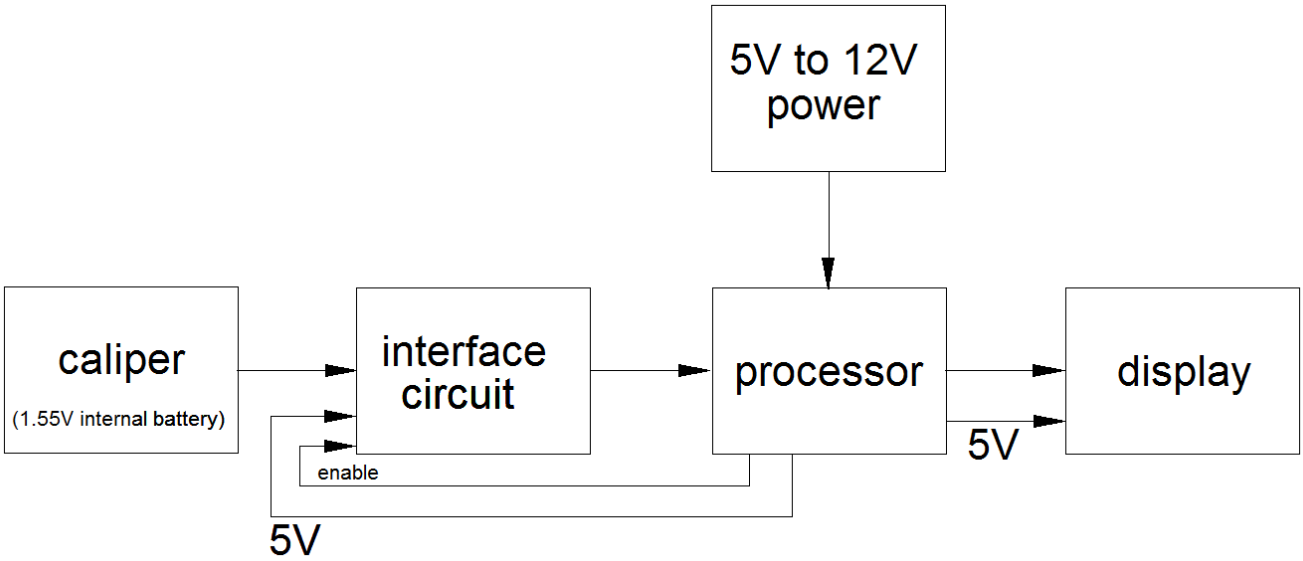
Here is the highest level code with a few minor functions removed. No interrupts are used.

```
void loop()    //loop runs continuously
{
  waitForCloseToStart();//caliper jaws have to close by 100 thou before we start to look for local minimum; also looks for command requests.

 localMinRead(); //prompt user, monitors caliper for local minimum and detect a possible push of inch/mm which means command will be executed. When done, we will return to this point in the loop; output is oldResult

 interpolation(); //improve accuracy of caliper

  displayAnswer(); //can be decimal only or decimal and fraction

 waitForOpenToStart();//caliper jaws have to open by 100 thou  before we start to look for local maximum; also looks for command requests.

 localMaxRead(); //monitor caliper for local maximum; then signal we are ready for new local minimum; also looking for command to run.
  //after command runs, it will return here.

 interpolation(); //improve accuracy of caliper

  displayAnswer(); //can be decimal only or decimal and fraction
}
```

The Hardware

The major elements of the hardware are:

- Harbor Freight 6" digital caliper: http://www.harborfreight.com/6-inch-digital-caliper-47257.html The list price $30 but on sale now for $10.
- Pro Micro 5V/16 MHz processor from SparkFun: https://www.sparkfun.com/products/11098 The price is $20.
- Basic 16 x 2 Character LCD 5V also from SparkFun: https://www.sparkfun.com/products/791 The price is $15.



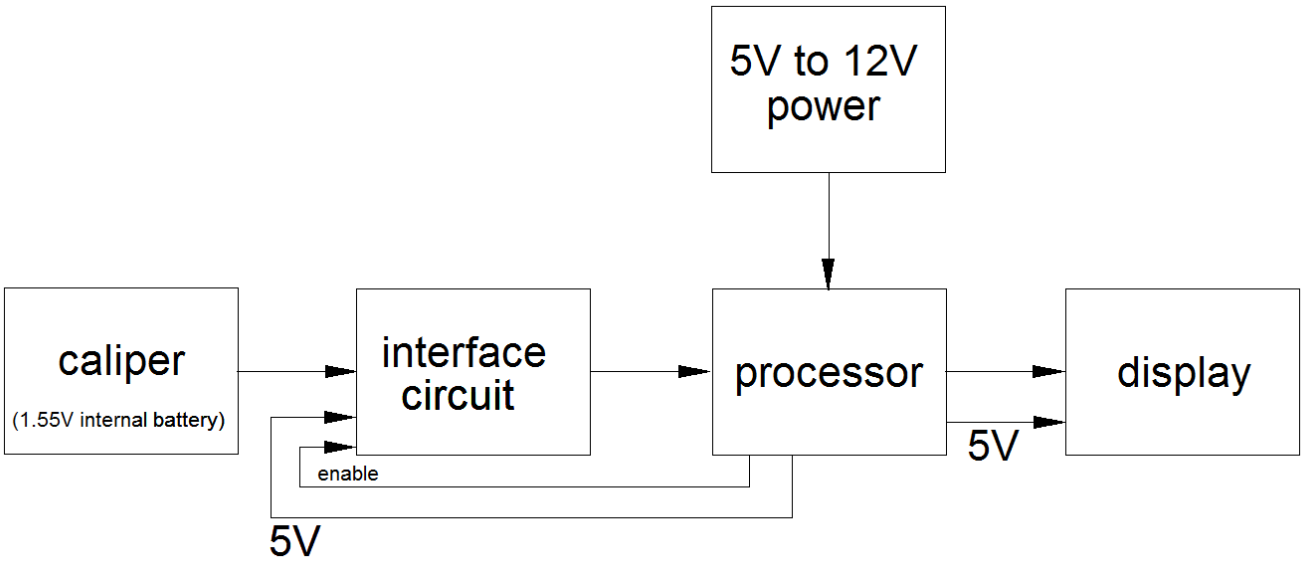
There is also an interface circuit consisting of two general purpose NPN transistors and four resistors. If you chose the Pro Micro 3.3V/8 MHz and 3.3V LCD, this interface circuit probably should not be needed. Minor software changes would be needed to compensate for the clock and data inversion.

The 5V power comes in via a USB cable but the processor can handle up to 12V. It feeds 5V to the interface circuit and to the display. The caliper has its own 1.55V battery.

Details of the caliper's data port can be found in:

http://rick.sparber.org/electronics/hf6.pdf

You will find ground, clock, and data.



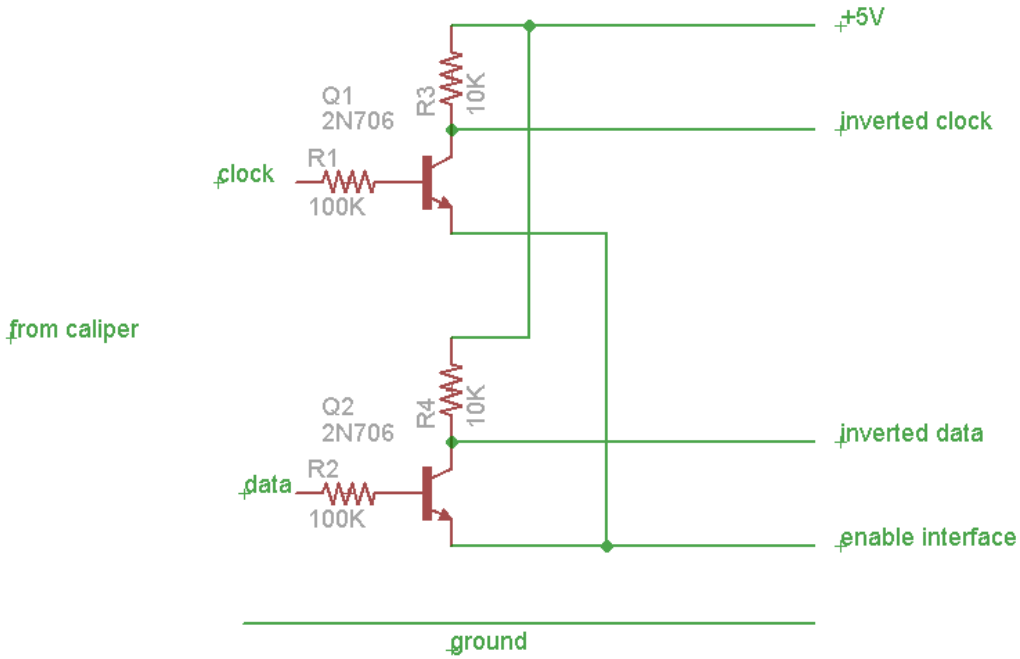
Ground, inverted clock, and inverted data then feed from the interface circuit to the processor.

While the logic levels coming in to the interface circuit are 0 and 1.5V, what comes out is 0 and 5V. When the processor wants to read the clock and data leads, it enables the interface circuit. Otherwise, it disables the circuit which reduces current drain on the caliper's battery.

The processor drives 6 wires which connect to the display along with power and ground.

Detailed Hardware View

Interface Circuit

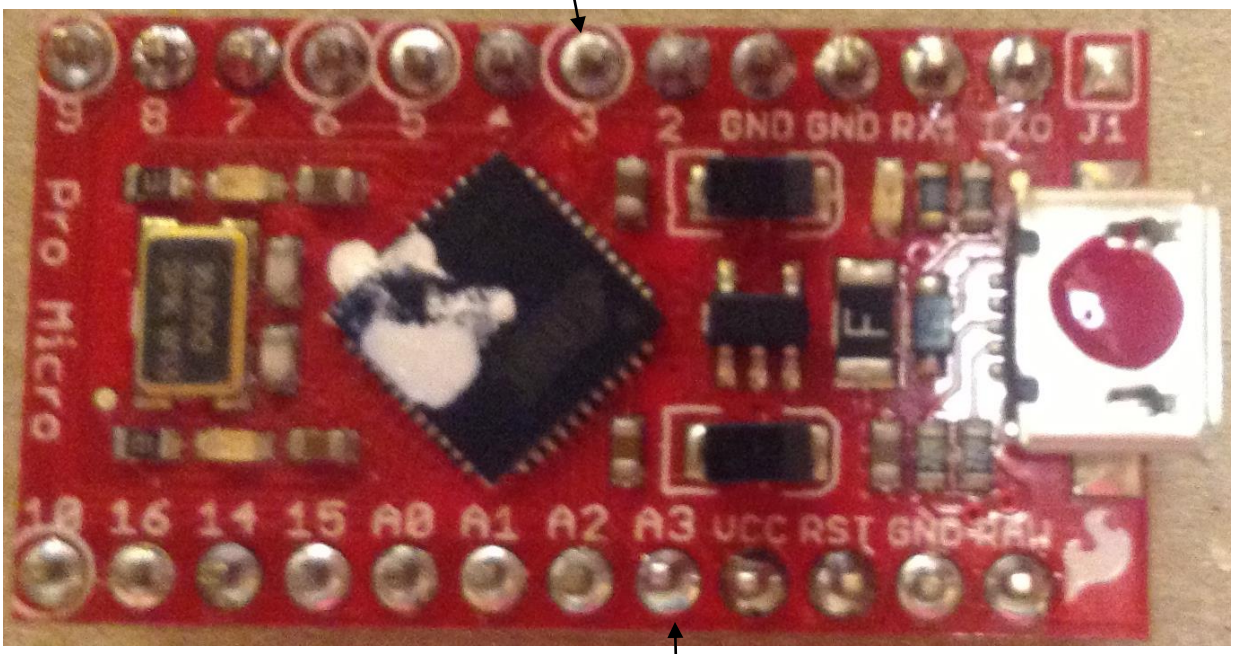


All resistors are 1/8 watt 10%. The two NPN transistor can be any general purpose devices.

When the processor is not reading this interface, the enable interface lead is high. This causes both transistors to become high impedance. That minimizes the load on the caliper's clock and data leads. When enable interface is low, a low on clock turns off Q1 and lets inverted clock rise to +5V. When clock is high, Q1 turns on and inverted clock is pulled down to near ground. Q2 and data works the same way.

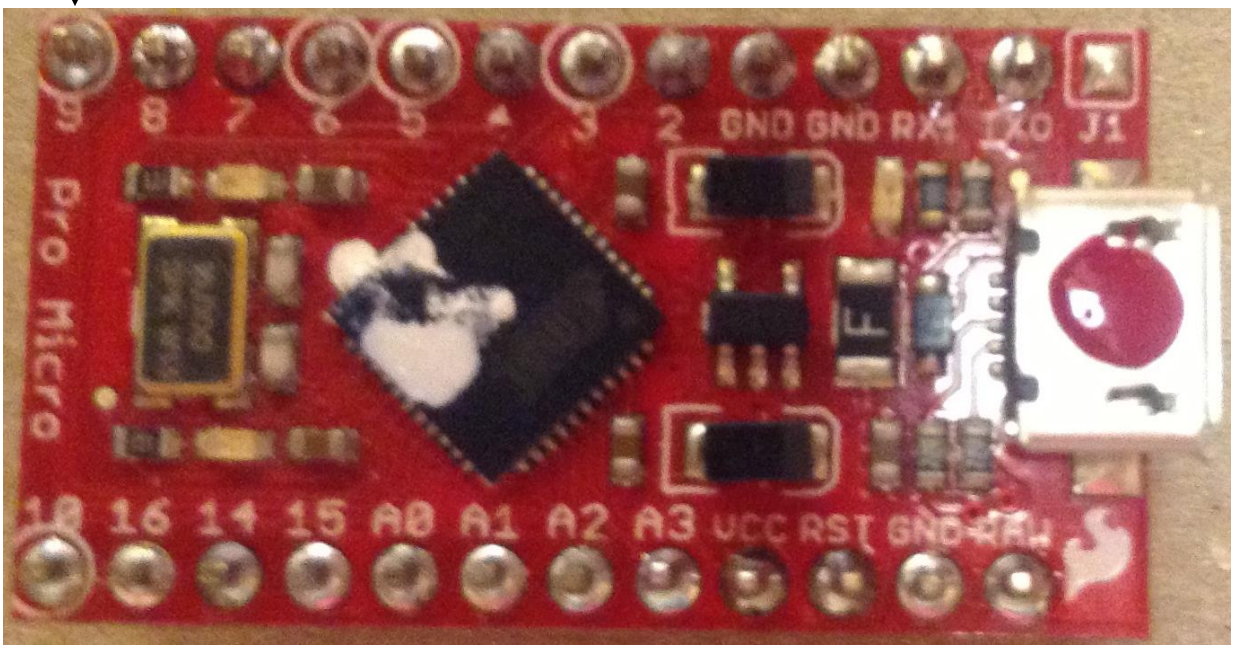
Interface Circuit to Pro Micro Connections

Inverted clock connects to data pin 3 on the Pro Micro. Note that this pin is marked

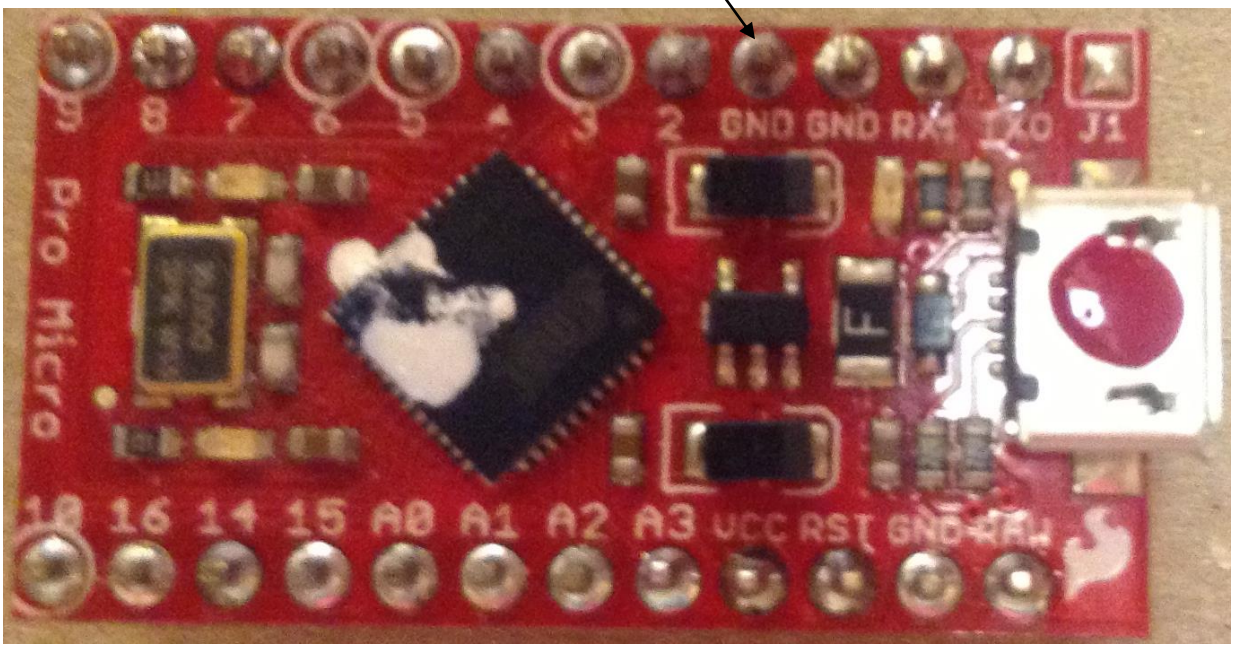


with just the number 3 and is not A3.

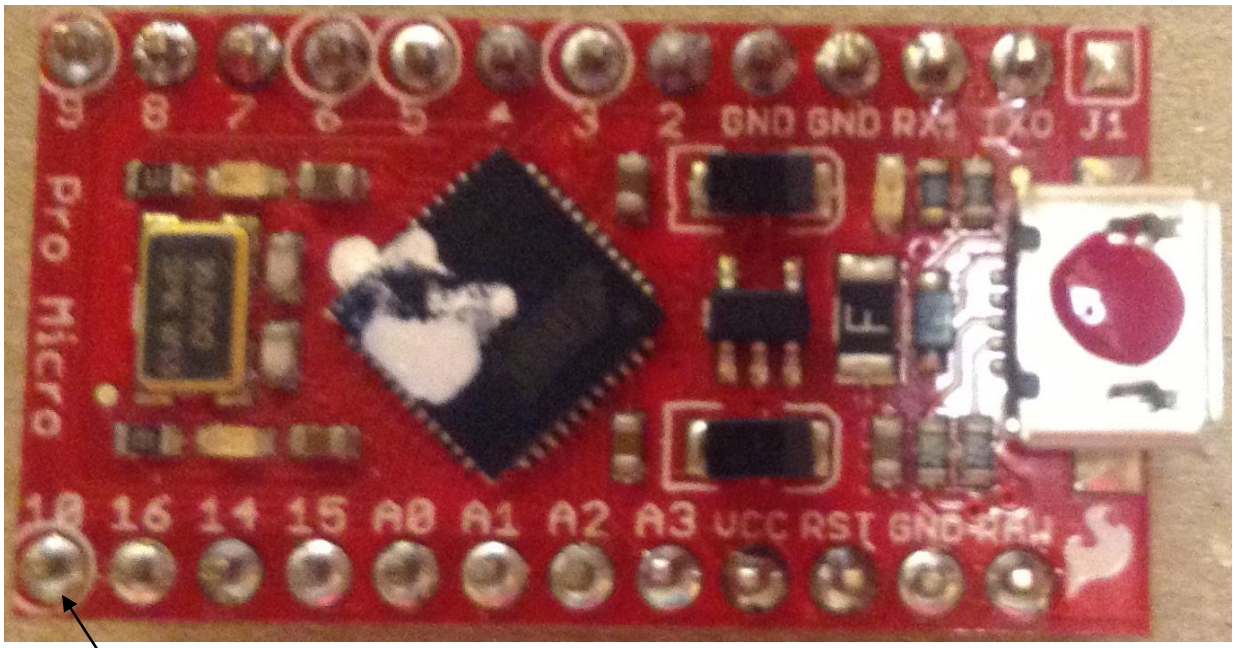
Inverted data connect to data pin 9.



Ground from the caliper connects to GND on the Pro Micro.

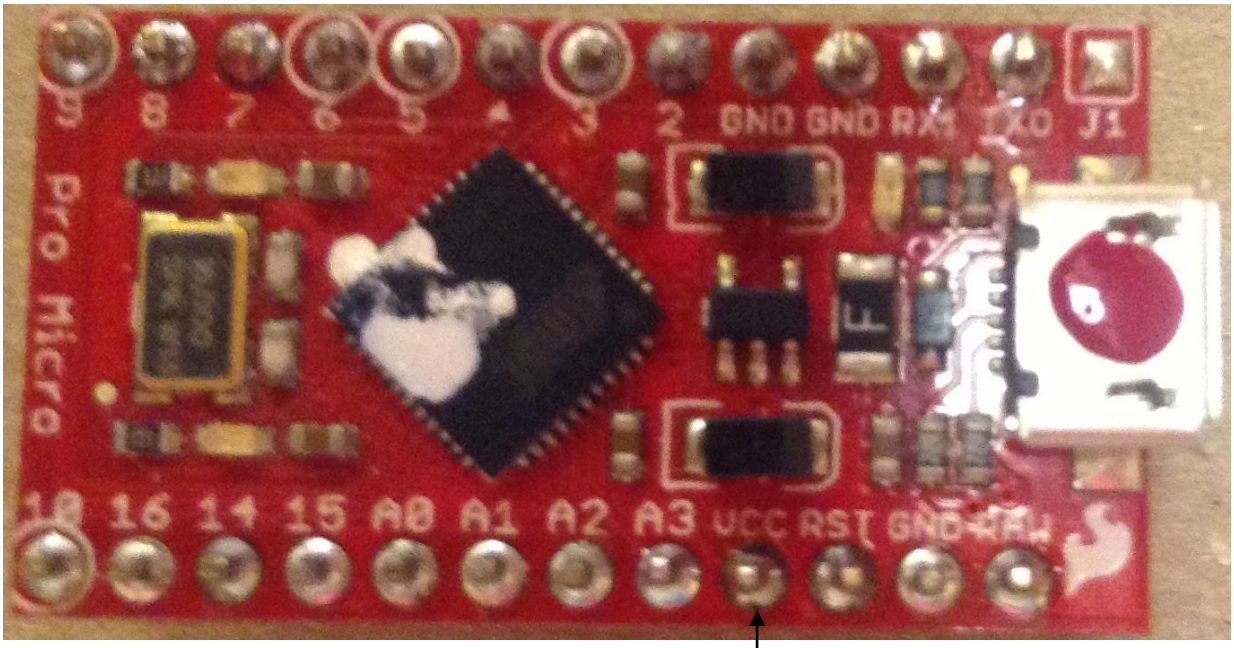


Enable Interface connects to data pin 10.



Data pin 10.

+5V to the caliper interface connects to VCC.

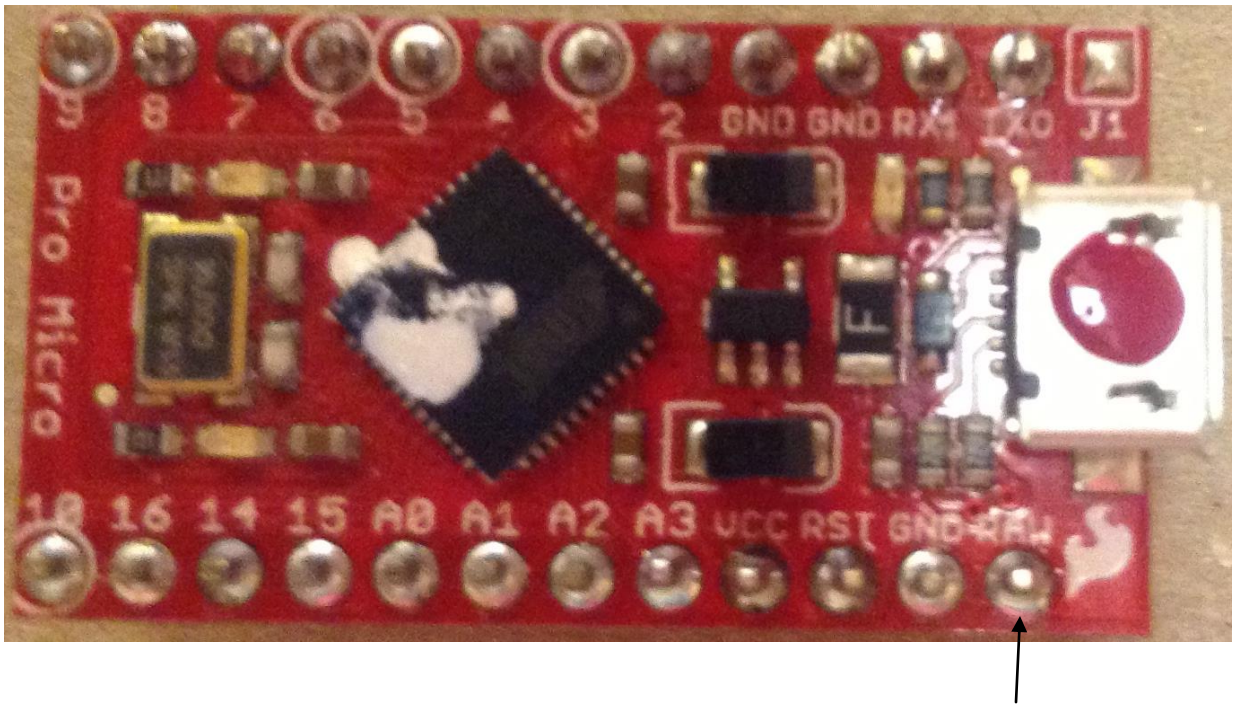


VCC

Power

If power is coming from the USB cable, the Pro Micro takes care of it.

If you want to power the system from a battery, then you have two choices. If you have a source of regulated 5V, then connect it to the VCC pin of the Pro Micro. If



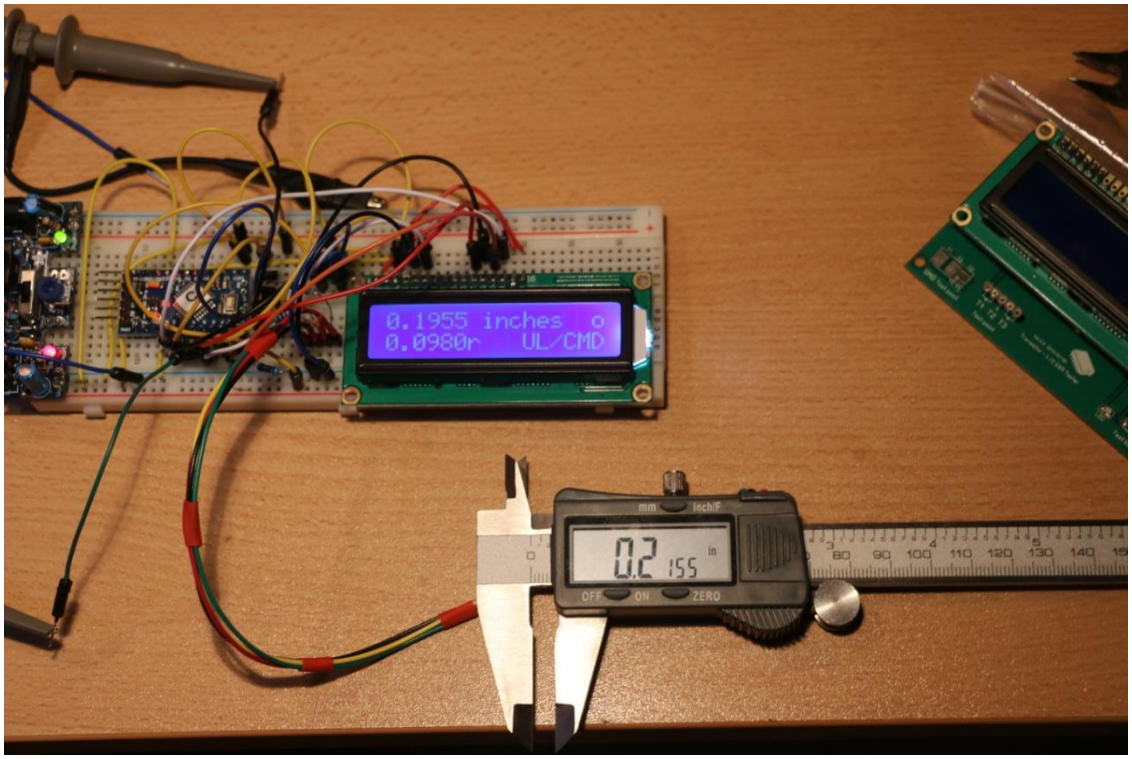
you want to power it from about 6V to 12V, feed it in to the RAW pin.

Current drain is less than 60 mA.

One simple option is to use the USB cable fed from a USB power pack. You get the portability of a battery without the complexity of building your own power source.

Replication

Rob Reeves contacted me on October 27, 2015 asking for the code. On November 1, 2015 he sent me this picture of a working system. Way to go Rob!



Future Effort

With this platform in place, I plan to add more functionality. One thing that would be easy to add would be a go/no-go feature. It would be similar to calibrating with gage blocks but would drive the display to just say "Under", "OK", and "Over". Given the local maximum and local minimum functionality and this simpler display, a lower skill worker could do the job.

A feature with a lot of promise, suggest by JT of the Atlas/Craftsman Yahoo group, would be to implement the averaging procedure used in the appendix. The user would take 10 measurements and the program would throw out the largest and smallest values. The remaining 8 measurements would be averaged.

Acknowledgements

Thanks to Paul Rayes for his invaluable help with both the Arduino hardware and software.

Thanks to Rob Reeves for confirming my documentation and design. In less than a week he built a copy of the system.

Thanks to JT of the Atlas/Craftsman group for helping me see the various error sources plus suggesting a new feature that would essentially implement the calculations done in the appendix in order to generate a more accurate value.

I welcome your comments and questions.

Rick Sparber
Rgsparber@aol.com
Rick.Sparber.org

Appendix: Evaluation Raw Data

In this test the range was verified from 0.1000" to 0.1090" in steps of 0.0010". Worst case error was ± 0.0005".

A 0.1000" and a 0.1010" +/- 0.0001" spacer block were used to calibrate readings between 0.1000" and 0.10095". The expected corrected result should be within +/- 0.0005" of the block value.

Block	Raw data from caliper	Corrected result (taken consecutively)	Throw out max and min and average the rest	Maximum tolerance
0.1000	0.0990	0.0995, 0.0995, 0.0995, 0.1000, 0.1000, 0.1000, 0.1000, 0.1000, 0.1000, 0.0995, 0.1000	0.0998	+0 -0.0005
0.1010	0.1000	0.1010, 0.1015, 0.1010, 0.1010, 0.1010, 0.1010, 0.1010, 0.1010, 0.10100, 0.1010	0.1010	+0.0005 -0
0.1020	0.1010	0.1020, 0.1020, 0.1020, 0.1020, 0.1020, 0.1020, 0.1020, 0.1020, 0.1020, 0.1020	0.1020	0
0.1030	0.1020	0.1030, 0.1030, 0.1030, 0.1030, 0.1030, 0.1030, 0.1030, 0.1030, 0.1030, 0.1030	0.1030	0
0.1040	0.1030	0.1040, 0.1040, 0.1040, 0.1040, 0.1040, 0.1040, 0.1040, 0.1040, 0.1045, 0.1040	0.1040	+0.0005 -0
0.1050	0.1040	0.1050, 0.1055, 0.1045, 0.1050, 0.1045, 0.1045, 0.1050, 0.1050, 0.1045, 0.1045	0.1048	+0.0005 -0.0005
0.1060	0.1045	0.1055, 0.1055, 0.1055, 0.1060, 0.1060, 0.1055, 0.1055, 0.1060, 0.1065, 0.1060	0.1058	+0.0005 -0.0005
0.1070	0.1060	0.1070, 0.1065, 0.1065, 0.1065, 0.1065, 0.1070,0.1070, 0.1065, 0.1070, 0.1070	0.1068	+0 -0.0005
0.1080	0.1070	0.1080, 0.1080, 0.1085, 0.1085, 0.1085, 0.1075, 0.1080, 0.1080, 0.1075, 0.1075	0.1080	+0.0005 -0.0005
0.1090	0.1080	0.1090, 0.1090, 0.1085, 0.1090, 0.1090, 0.1085, 0.1085, 0.1090, 0.1085	0.1088	+0 -0.0005

Note that the software can be designed to average 10 readings, throw away the smallest and largest, and average the rest as done above. It could then round the result to the nearest half thou. Note that if that was done with this set of data, the results would all agree with the gage block values.