dannyelectronics

October 25, 2015March 29, 2018

A HX711-based MilliOhm Meter

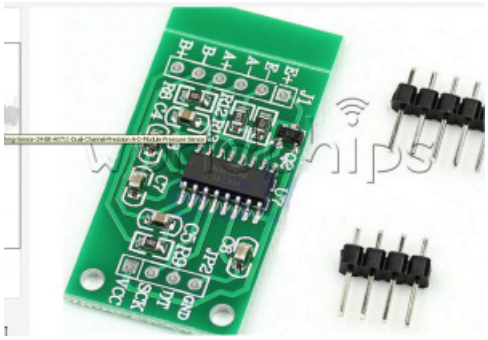
In an earlier post, I discussed how to build an inexpensive milliohm meter based on the crude ADC in an ATtiny25/45/85 (and other AVRs with similar adc modules – some of them have gains into 200x territory). The build-in "PGA" in those modules are quite good for purposes like that.

- code available for download **here (https://github.com/dannyf00/HX711)**.
- A minimalist implementation based on Arduino Leonardo / ATmega32U4 can be found **here (https://dannyelectronics.wordpress.com/2018/03/26/minimalist-milliohm-meter-with-digital-read-out-leonardo-atmega32u4-implementation/)**; An ATmega32-based proof-of-concept can be found **here (https://dannyelectronics.wordpress.com/2018/03/26/minimalist-milliohm-meter-with-digital-read-out-proof-of-concept/)**.

In that post, I mentioned the possibility of extending the concept to a high-resolution external ADC, like LTC244x chips / modules available. Those modules are quite expensive, starting at $20 or so.

- There are actually ways to modify a 7106-based digital multimeter as a milliohm meter -> by generating the reference voltage through a reference resistor. Another post for another day, however.

Fortunately, over the last couple years, we have seen some (surprisingly) inexpensive 24-bit ADC modules available. For example, the HX711 modules go for $1 on eBay.



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-module.png)

HX711 Module

Those are 24-bit ADCs with two sets of differential inputs, with gains ranging from 32x to 128x. With a reference voltage of 1250mv, that means a resolution of 0.002uv/LSB, in theory.

I experimented with this module a while back and shared my findings. I thought this post as an aggregation of my experience there.

Concept:

There are a couple basic approaches to milliohm measurements:

1. Driving the DUT with a precision constant current source and measure the voltage drop off the DUT: from a measurement

perspective, this is the simplest approach – as measuring voltage drop is fairly straightforward. However, generating a precise current is very difficult.
2. Applying an arbitrary voltage across a reference resistor in serial with the DUT: by comparing the voltage drop over the two resistors, you can calculate the DUT's resistance – this is called "ratio-metric".

The ratio-metric approach has the advantage of not needing a precision current source, but requires a reference resistor -> generally available but can be expensive: 0.1% resistors sell for <$10 and 0.001% resistors less than $100 in singles. However, if you have access to calibrated resistors, lowly metal film resistors can also serve as the reference resistors for most applications.

Most ratio-metric implementation requires a chip with (differential) ADC inputs + differential reference voltage inputs: the reference resistor is in serial with the DUT. The current through the reference resistor generates a differential signal to serve as the external reference generator: you can find countless ADC chips that can be configured as such.

That is the approach we will follow, but with a twist.

HX711:

There is very little information available about the HX711. All we know is that it is a 24-bit ADC, made by a Chinese firm, with the weight scale as the intended application.

Here is a link for its datasheet: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf

The datasheet shows a typical application and I have also highlighted for you the corresponding pins on the eBay-sourced HX711 modules:

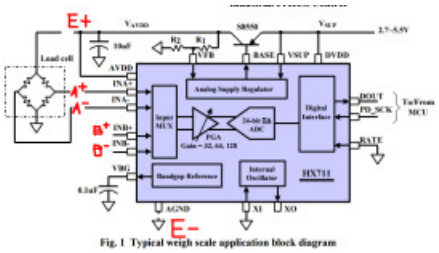


Fig. 1 Typical weigh scale application block diagram

(https://dannyelectronics.files.wordpress.com/2015/10/hx711-typical-application.png)

HX711 typical application

You will need to note the following:

1. The chip is intended for a bridge application;
2. The module has 3 sets of terminals to interface with the bridge:
   - Energizing terminals (E+ and E-): they are required to charge up the bridge. Please note that E- in the module is not connected with AGND/GND. ***YOU WILL NEED TO MAKE THAT CONNECTION***;
   - Channel A differential inputs (A+ and A-): those are for the Ch A inputs of the ADC. They can have a gain of 64x and 128x, user-configurable. Also note that most of the eBay modules have no input resistor on the A- pin but a 1k resistor on the A+ pin. Those two terminals also have a filter cap between them;
   - Channel B differential inputs (B+ and B-): those are for the Ch B inputs. They have a fixed gain of 32x. Like Ch A termianals, the B+ input has a 1k resistor but the B- input does not. A filter cap is present.
3. The interface with a mcu is digital, through DOUT and PD_SCK pins on the chip, and correspondingly DT and SCK pins on the module.
4. The module requires a power supply of 5v, necessitated by the values of R1 (20K) and R2 (8.2K) used. If you wish to use the module under lower voltages, you can either decrease R1 or increase R2.

Design Goal:

Eventually, I want to be able to build a milliohm meter with digital display. As minimal processing is needed here, I thought a 8pdip mcu, like PIC12F675 or ATtiny25/45/85 driving both the HX711 module and a I2C LCD would be great.

In the meantime, we will prototype the design on a much more powerful platform, the LM4F120 launchpad, 🙂

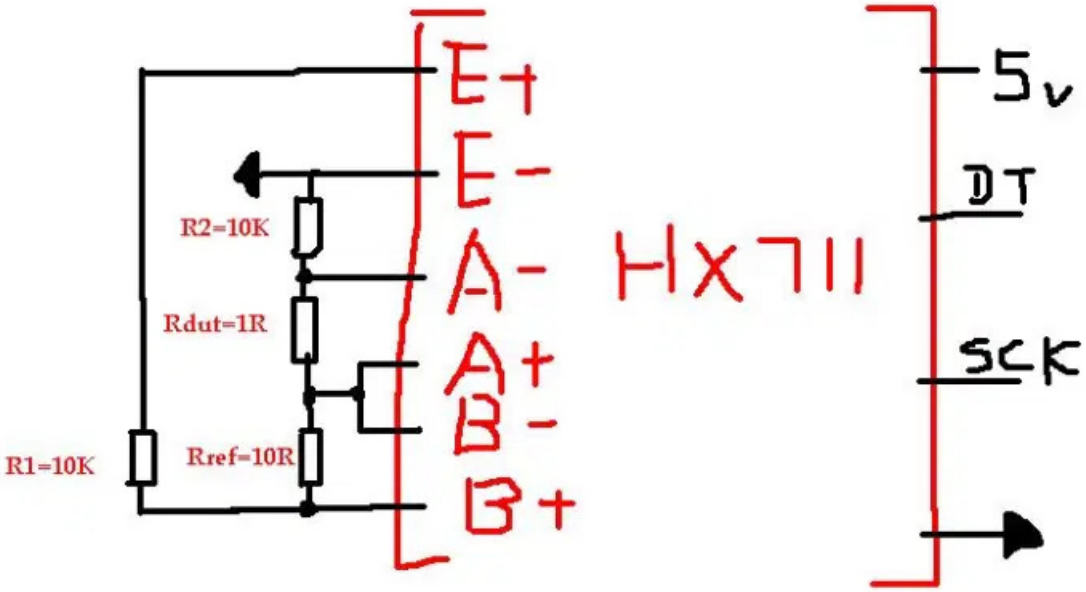
Prototype:

The launchpad is used here for its simple serial interface: I can just use a virtual serial terminal to monitor the output on a PC. The same code can be then recompiled for the PIC or ATtiny later on.

So, Here it is, a HX711 module driven by a LM4F120 Launchpad. The 5v source came from the USB bus on the LM4F120 (noisy).

HX711 Milliohm Meter on LM4F120

The wiring is simple here:



HX711 milliohm meter – LM4F120 wiring

The code is even simpler:

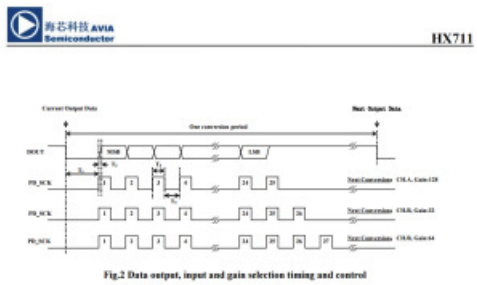
```
//read dut on Ch A
while (hx711_busy()) continue; //wait for hx711 to be ready

tmp_dut=hx711_readB32(); //read ref on CH A, 64x gain. Set next conversion to be ch B, 32x gain

//read ref on Ch B
while (hx711_busy()) continue; //wait for hx711 to be ready
tmp_ref=hx711_readA64(); //read dut on CH B, 32x gain. Set next conversion to be Ch A, 64x gain

//calculate uOHM
tmp_uOHM = (double) tmp_dut * uOHM_REF / tmp_ref;
tmp_uOHM /= 2; //correct for gain differentials: dut on 64x gain, and ref on 32x gain
```

- I was asked to explain the strange way the above code works, particularly with regards to the comments: "tmp_ref=hx711_readA64();" is associated with a comment about reading Channel B at 32x gain, not Channel A at 64x gain, as the routine's name may have suggested.



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-timing.png)

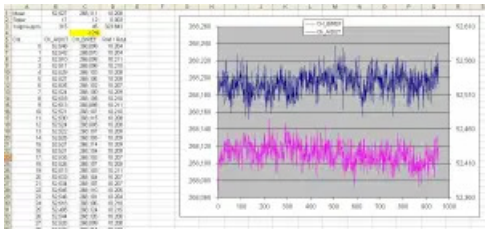
HX711 timing

- The issue lies in the way the chip works. If you look at the timing on Page 5 of the datasheet, reproduced below, you will find that the gain and channel setting are determined for the next conversion, by sending 25, 26 or 27 pulses, depending on the desired channel / gain combination. Because of that, our code had a dummy read up front, and followed it with a pair of reads for Ch A and Ch B. An alternative is to have a dummy read each time you access the chip. Obviously, this approach is more robust but it slows down the chip considerably.

Because of the riometric nature of the algorithm, we don't need to any complicated calculation: the ratio between the two channels' adc readings, corrected for gain differentials, is all we need.

The two ADC readings, one for Ch_A and another for Ch_B, are then sent via a virtual comm port to the PC and captured via HyperTerminal (or anything like that).

Here is the plot of about 1,000 data points (1-2 seconds per two conversions):



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-milliohm-meter-lm4f120-readings.jpg)

HX711 Milliohm meter – readings

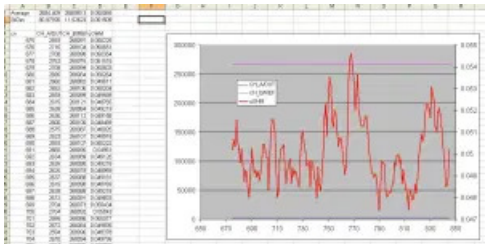
A few quick observations:

1. Some variability in readings off both channels: more so for the DUT / Ch_A than for the Ref / Ch_B.
2. The resistance (ratio) remains quite stable, to the 5th digit (1-sigma), or 4th digit (3-sigma): we used a 10ohm reference resistor here so the actual resistance of the DUT would be 10ohm / 10.208x = 0.980ohm, assuming that the 10ohm reference resistor is

accurate.

3. The two channels are negatively correlated: Cell C4, highlighted in yellow, shows the correlation between the ADC readings of both channels and it is -12%. This is probably the worst part of this particular chip: for a ratiometric approach to work, you want the two channels to be as much positively correlated as possible. Negative correlation widens the range of output errors.
4. not a true 24-bit adc: the noise free output for Ch_A is about 52527/17=3171, or 11.6 digits. So the HX711 is more like a 14-bit ADC. One way to (optically) reduce that variability is to filter the output. If we have enough space, we will do that in the final product.

As another test, I ran the meter on a 12" 22g copper wire:



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-milliohm-meter-lm4f120-short.jpg)

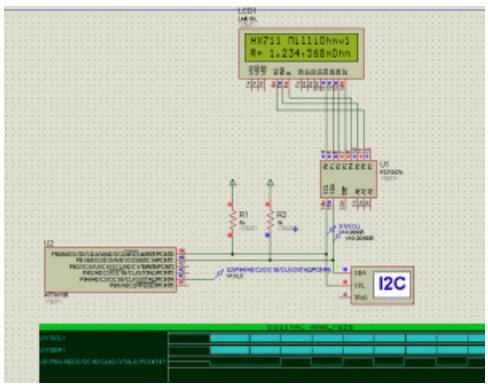
HX711 milliohm meter – short wire

It has a resistance of 50mohm, varying from 48mohm to 53mohm, likely due to the contact resistance in a breadboard. Not a bad result.

Our next task is to move it to an ATtiny85 and instrumentationize it.

HX711 on ATtiny85:

I have moved the code to an ATtiny85:



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-attiny85.png)

HX711 on ATtiny85

It is the same code as on the LM4F120 but driving an I2C LCD (wiring consistent with the typical eBay modules). No UART output implemented. HX711 routines are implemented but not shown.

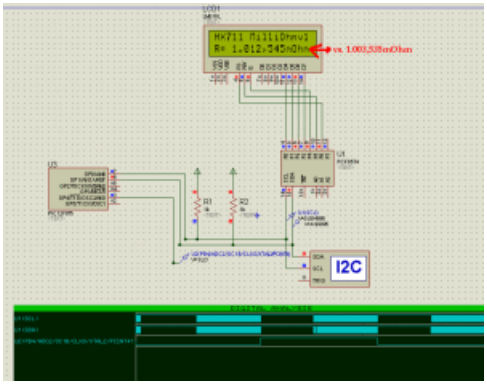
The resistance is simulated to be 1.234,567Ohm, with an 82Ohm reference resistor.

Next, run it on a real ATtiny85.

BTW, the code compiles to 1.2KB flash in debug mode. So the hex should fit into a ATtiny25/45 as well. And I suspect that a 12F675 implementation is totally within reach.

- It turns out that porting it to a 12F675 is more difficult than I thought originally. Yes, it is doable, as shown below. The flash is 994 bytes, or 97.1% of total flash size of 0x400 bytes on a 12F675. RAM is 100% used up.
- Update [10/27/2015]: worked a little bit more to optimize the code. Flash utilization now is 971 bytes (94.8%), and RAM

utilization is 57 bytes (89.1%). A lot of work for a few bytes' savings. Exhausting but satisfying.



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-12f6751.png)

HX711 Milliohm on 12F675

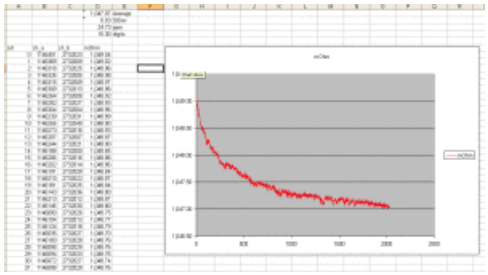
- However, because of the limited ram / rom, you have to utilize some creative math to fit the code into a 12F675, leading to some inaccuracies. For example, the particular case had a thoretical resistance of 1.003535ohm, yet the "creative" math got to 1.012545ohm, an error rate of 0.9%. Whether that kind of errors is acceptable will dependent on your specific application.

For now, I will focus on ATtiny85 implementation instead.

Update [10/27/2015]:

I decided to run the test circuits a little bit hotter, at 2ma, vs. 0.2ma previously, by using two 1Kohm current limiting / padding resistors. This greatly improved the performance of the circuit:

- Channel A (the DUT channel) is running at 128x gain now, vs. 64x before.
- Experiment done again on the LM4F120 Launchpad.



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-module-at-2ma.png)

HX711 at 2ma

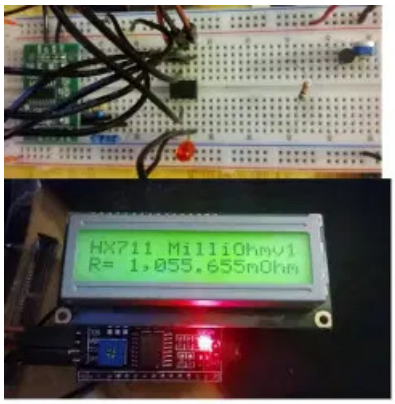
The chart showed approximately the first 2000 measurements of our DUT's resistance (1ohm nominal). A few quick observations:

1. The measured value started at 1.049R and declined gradually to 1.047R -> given the low current levels, I suspect that it is more of the chip itself heating up – by putting my finger on the chip, I can create an increase of approximately 0.0007R in a short period of time.
2. If you take the last 100 or so measurements (to reduce the impact of heating), the measurement variations are minimal – approximately 0.03mOhm within 1-sigma. Meaning that 99% of the time, the measurement will be within a +/- 0.05mOhm band. That is way beyond our ability to calibrate any precision resistors.
3. The effective resolution of the circuit is about 15.4 bits, equivalent to that of a 16-18 bit ADC. Not too bad for something sold for a dollar on eBay.
4. For precision measurements, a Kelvin clip is a must.
5. This thing screams to live in a temperature controlled environment.
6. Gain inconsistencies are significant so it is best to run the meter at a fixed gain, once it is calibrated.

The work to move it to a ATtiny85 remains to be done.

Porting to ATtiny85:

After diner, I got some time to port it to an ATtiny85:



(https://dannyelectronics.files.wordpress.com/2015/10/hx711-attiny85-lcd.jpg)

HX711 MilliOhm Meter on ATtiny85

Over a period of 15 minutes, my observations suggest that the readings are stable to the 4th digit: 1.055Ohm here, and fluctuate mostly 40mOhm.

Readings refresh about once per second. They do seem to exhibit a slow downward drift, from 1.055500R to about 1.054950R 45 minutes later. Totally acceptable to me.

- Compiled under gcc-avr, flash usage is 5Kb, and RAM usage is 300+ bytes. So putting it into an ATtiny25/45 is out of question now.
  - I managed to reduce the flash to just 2.6KB. So fitting it onto a ATtiny45 is a real possibility now.
- Those I2C LCD modules seem to have different wiring. The first one I used followed the wiring as I posted earlier in simulation. The 2nd one I tried used a different kind of wiring, with the D4..D7 on the LCD aligned with D4..D7 on the I2C expander, essentially swapped the wiring with the first module. Took me a while to figure that out.
- Update: With a little bit of software filtering, I was able to reduce the fluctuation to within 20mOhm generally and 10mOhm mostly, corresponding to +/- 10mOhm and +/-5mOhm. The first 4 digits are rock-solid, and the 5th digit fluctuates only if the measurement happens to be around 100s of mOhm.
- Tried the meter on some copper wires. Here is a shot of it measuring a 4" 22awg wire.

(https://dannyelectronics.files.wordpress.com/2015/10/hx711-attiny85-4in22awg1.jpg)

- HX711 on 4in 22awg copper wire

I will do a little bit more analysis tomorrow – calling it a day now.

Posted in **analog**, **digital**, **MCU**, **milliohm**, **test and measurement11 Comments**

11 thoughts on “A HX711-based MilliOhm Meter”

1. Pingback: **A MCU-based milliOhm Meter | dannyelectronics**
2. Pingback: **My first post | dannyelectronics**
3. **David Pilling says:**
   December 3, 2015 at 5:10 pm
   Interesting and impressive stuff. “The code is even simpler…” had me wondering where all the code was – maybe it is for later release.

4. **Reply**
   Shanky says:
   July 13, 2016 at 3:54 pm
   Hi Danny very good post about Hx711 ADC, I am interested HX711 to measure a very small voltage range of 2mV – 20 mV. For testing purposes I am using voltage divider circuit to measure 20mV. But I am getting the reading like -8388608 and so on.
   1. How to convert it into equivalent voltage?
   2. My approach of voltage divider is correct or should I use bridge?(for testing purpose only)

   Reply
   1. **dannyf00 says:**
      August 19, 2016 at 1:47 pm
      Shanky, the HX711 data sheet has a demo code that converts the negative numbers to positive and you may want to take a look at. Hope it helps.

      Reply
5. Pingback: **HX711 – code available – dannyelectronics**
6. Pingback: **Minimalist MilliOhm Meter with Digital Read-out – Proof of Concept – dannyelectronics**
7. Pingback: **Minimalist MilliOhm Meter with Digital Read-out – Leonardo / ATmega32U4 Implementation – dannyelectronics**
8. Pingback: **Minimalist Milliohm Meter – Arduino + HX711 – dannyelectronics**
9. **Parth says:**
   December 17, 2018 at 6:26 pm
   How can I connect 2 wire 2 strain gauge of 350 ohms connect with HX711 on channels A+, A- and B+, B- and also setting gain of module to 32 can you please send me connection details and what to do for E+ and E-. please email me if you have Answer on email id.

10. **Reply**
    GB Clark says:
    July 22, 2019 at 7:15 am
    It may not be a .001 resistor, but they are better than a .1%.
    A 100K, .05%, 5ppm resistor in a 0805 package costs $1.60 qty 1.
    You can also get a 470 ohm, .05%, 25ppm in a 1210 package for $1.06 qty 1.
    These are both made by KOA Speer and sold by Mouser. Digikey also stocks them.

    Reply