INA219 Current sensing chip

Understanding the chip's equations and how to use them

Using INA219 Equations

The following paragraphs from the datasheet talk a lot but are not easy to understand. If you just apply them then you get the result they suggest (they do work but require floating point multiplication to process the results).

The following operational description is complicated so...

The exception is if you are designing a bespoke system where you need a specific current resolution and/or range - then you will have to understand internal chip operation.

One issue is: To process the register results from the INA219 chip you have to use floating point calculations to output a meaningful result.

The INA219 does not output useful data directly. So, if you have to use floating point anyway, then the internal calculation processing within the chip seems a little redundant.

In fact, for space constrained chips, you can save 3kbytes of memory by not using floating point and instead use fixed point as explored later.

There seems to be no point in letting the chip do half the work and the microcontroller do the rest; other than the fact that it relieves the microcontroller from a little processing. The other reason could be that the chip operates in the background so doing a minimal calculation may be quicker when retrieving results.

Chip equations

Note: The text (below) "...**provide direct decimal equivalents of the values being measured**" is not easy to figure out. That text means you can directly read the current register for a measurement in amps. This is not usually done since the equations are complex to follow.

Datasheet paragraphs:

"The Calibration Register enables the user to scale the Current Register (04h) and Power Register (03h) to the most useful value for a given application. For example, set the Calibration Register such that the largest possible number is generated in the Current Register (04h) or Power Register (03h) at the expected full-scale point"

"This approach yields the highest resolution using the previously calculated minimum Current_LSB in the equation for the Calibration Register."

"The Calibration Register can also be selected to provide values in the Current Register (04h) and Power Register (03h) that either **provide direct decimal equivalents of the values being measured**, or yield a round LSB value for each corresponding register."

Then the equations follow:

$$cal = trunc\left[\frac{0.04096}{CurrentLSB * Rshunt}\right]$$

$$CurrentLSB = \frac{Maxiumexpectedcurrent}{2^{15}}$$

$$PowerLSB = 20 * CurrentLSB$$

$$CurrentRegister = \frac{Shuntvoltageregister * calibrationregister}{4096}$$

$$PowerRegister = \frac{Currentregister * Busvoltageregister}{5000}$$

$$CorrectedFullScaleCal = \left[\frac{Cal * Measuredshuntcurrent}{INA219Current}\right]$$

Other simpler equations from text in the datasheet

RealShuntVoltage = ShuntVoltageRegister * 10e-6

RealBusVoltage = BusVoltageRegister * 4e-3

From these datasheet statements:

"Shunt voltage is calculated by multiplying the Shunt Voltage Register contents with the shunt Voltage LSB of 10uV."

"The bus voltage bus voltage register (shifted right 3 bits) is multiplied by the Bus Voltage LSB of 4mV to compute the bus voltage.".

So you can calculate the real voltages using the following equations:

```
Vshunt = Shunt_register * 10uV
Vbus   = (Bus_register>>3) * 4mV
```

Using Standard Equations

The data sheet goes through use of the equations to give an optimal resolution for the LSB of the ADC when you use the maximum expected current. When you set this arbitrary limit the resolution of the result is optimised.

Here's the process for choosing a 16V maximum and 400mA limit (the same as one example in the Adafruit library code).

Rshunt = 0.1
VshuntMax = 40mV (Choose this by setting the PGA gain).
VbusMax = 16V (Select voltage range 23V or 16V)

MaxPossibleCurrent = VshuntMax/Rshunt = 0.4A
MaxExpectedCurrent = 400mA

The following gives example current for maximum resolution (averaging enabled) and 12 bit operation. Setup for 12bit or averaging are selected by programming the registers.

MinimumPossibleCurrentLSB15Bit = 400e-3/pow(2,15) = 12.2uA per bit
MinimumPossibleCurrentLSB12Bit = 400e-3/pow(2,12) = 97.6uA per bit

You can choose any value between those results (choose for easy maths is suggested). Adafruit chose 50uA.

CurrentLSB = 50uA

Now you can figure out the cal register contents using the provided equation:

CalReg = trunc(0.04096 / (CurrentLSB * Rshunt))
CalReg = trunc(0.04096 / (50e-6 * 0.1)) = 8192.0

At this point the device will return values that you can read. The problem is that they need to be processed by the microcontroller to give actual values of current and power. (This is why the above process seems a little pointless i.e. why build a chip that does all the processing in the background but then requires you to do floating point operations to get the "real" values out. See fixed point operation on how to do it using the raw values. Note the only real reason is that you get optimal resolution - so maybe that is a good reason!).

Real Current Computation

Calculate register correction factors for real value output:

The full scale Current output is the current register value multiplied by the CurrentLSB (chosen earlier) i.e. multiply by 50e-6 giving a result in Amps.

realCurrent = CurrentReg * 50e-6

A more convenient result is mA so multiply CurrentLSB by 1000 to get the current output in mA:

realCurrent = CurrentReg * 50e-6* 1000.

However it is easier to turn it upside-down (instead of multiplying by a fraction divide by an integer):

realCurrent = CurrentReg / (1000 * 50e-6) = CurrentReg / 20

Assume you have the full current flowing and 40mV across Rshunt then:

ShuntVoltageReg = ShuntVoltage / 10e-6 = 40e-3/10e-6 = 4000

Current reg = ShuntVoltageReg *calreg / 4096 = 4000 * 8192 / 4096 = 8000

Then

realCurrent = 8000 / 20 = 400 = 400mA

To get the real current you divide the "Current register value" by 20 (in the microcontroller). Note that this value of 20 is a direct result of choosing a round number for CurrentLSB.

Real Power computation

Calculate register correction factors for real value output:

The equation for power is:

PowerLSB = 20 *CurrentLSB

PowerRegister = CurrentReg * BusVoltageReg / 5000

Assume 400mA flows, (power will be 0.4*10 = 4W)

RealBusVoltage = 10V

busVoltageRegister = RealBusVoltage/ 4e-3 = 10 / 4e-3 = 2500
(Since the datasheet says 4mV per bus voltage LSB).
CurrentReg = 8000 (as before)

so

PowerRegister = (8000 * 2500) / 5000 = 4000

PowerLSB = 20 * 50e-6 = 0.001W per LSB

RealPower = 0.001 * 4000 = 4 = 4W

To get the real power you multiply by 0.001 (in the microcontroller) or you can forget the 0.001 multiplication leaving the result in mW.

RealPower = 4000mW