LoRaTracker

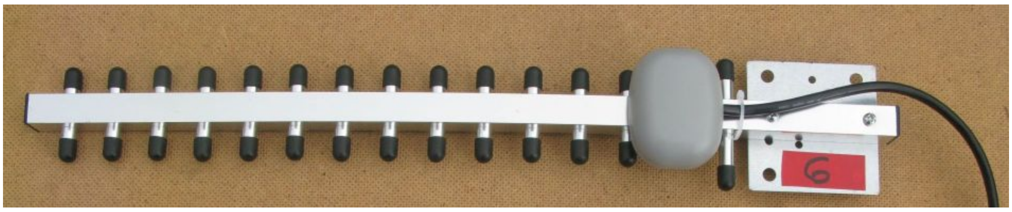
Search …

SX1280 Fast LoRa

30TH APRIL 2019 BY STUARTROBINSON LEAVE A COMMENT

The SX1280s LoRa modules can use far higher data rates than the UHF band SX127Xs. At the fastest settings, spreading factor 5 and bandwidth 1625khz, the effective data rate is **203kbs**.

High gain antennas such as yagis are very low cost for 2.4ghz since so many are produced for use on WiFi. The yagi below cost me £4.50 delivered from a UK supplier.



At what distance would a pair of these yagis, one for each transceiver allow a line of sight (LOS) data link to operate at 203kbs ? I decided to find out.

I configured my custom link test software to run at the 203kbs setting, put the link test transmitter on the mast on my shed and went to the receiving location North of Cardiff, 4.4km away that I have used for the SX1280 ranging tests described in a previous blog.

With the yagi connected to the NiceRF module on my receiver, I was getting packets down to -3dBm.

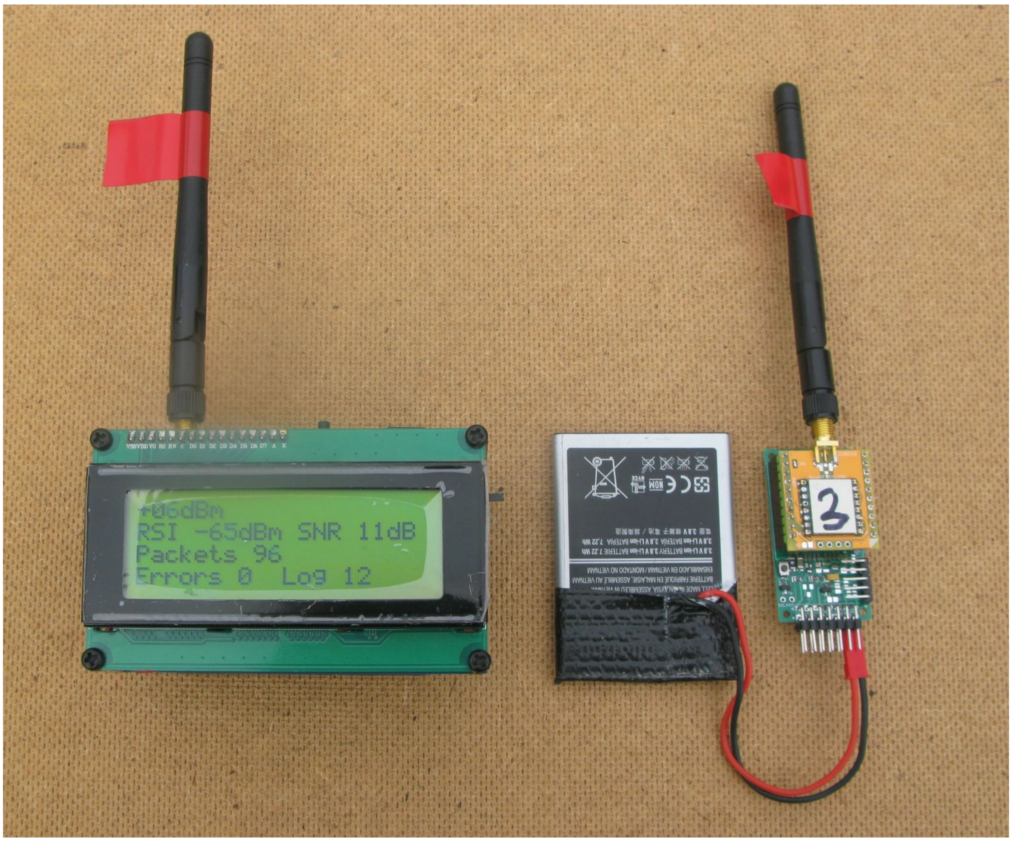
So at the maximum TX power the SX1280 can provide, 12dBm, there would be 15dBm of link gain, which suggests a LOS range at that power of around 25km. Which would be impressive.

I exchanged the yagi antenna I was using for the receiver of the test packets to a simple 2dBi one, see below;

March 2016
November 2015
August 2015
June 2015
January 2015
January 201

META

Log in



I was then receiving packets down to +5dBm, so the receiving end yagi was providing me with 8dBm of gain.

The implication was that the simple 2dBi antennas fitted either end would be on the verge of working over the 4.4km link. So I swapped the transmitting end yagi with a simple 2dBi one and tried again. Much to my surprise it worked, I received all the 12dBm packets. This also shows how useful the decending power method of testing can be in predicting actual link performance.

So with the simplest of antennas a pair of SX1280 LoRa modules can communicate at 4.4km at 203kbs using only 12mW of transmit power, impressive.

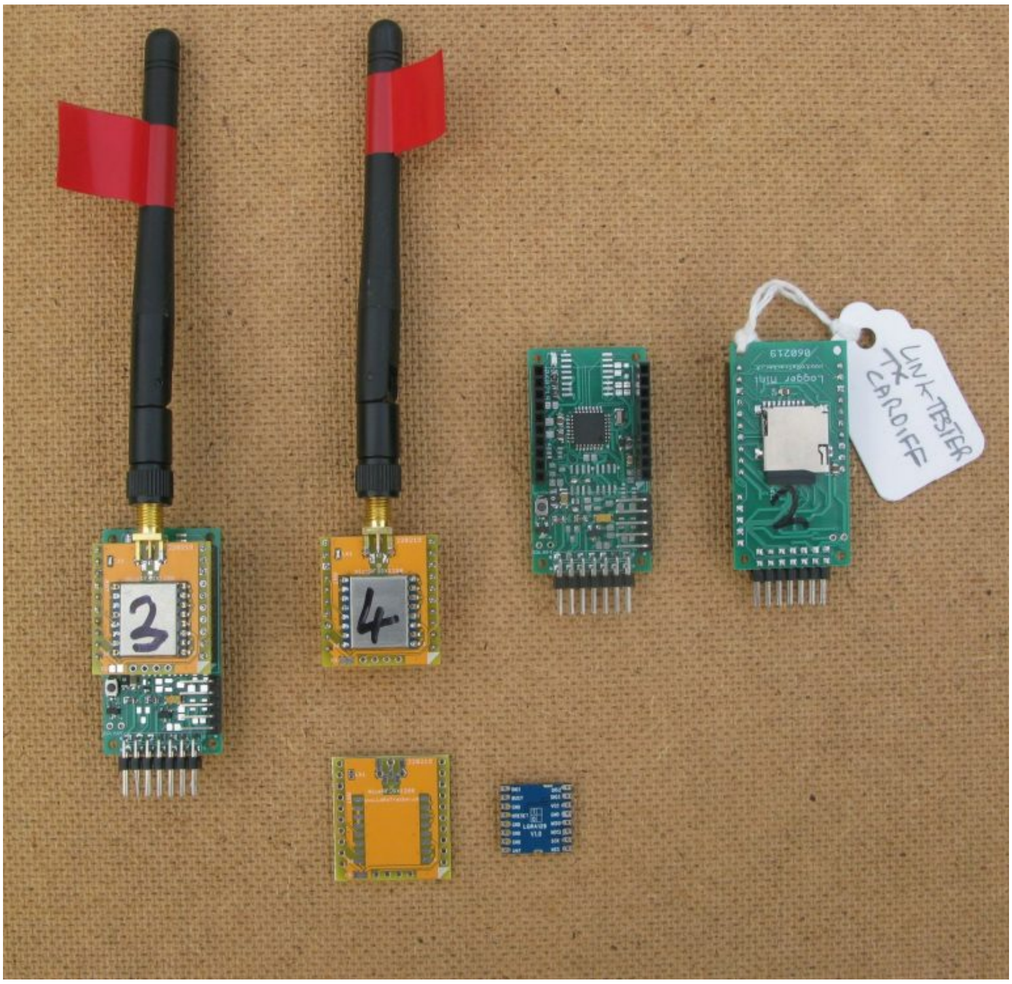
Semtech SX1280 2.4ghz LoRa Ranging Tranceivers

26TH APRIL 2019 BY STUARTROBINSON LEAVE A COMMENT

The Semtech SX1280 LoRa devices bring long range LoRa technology to the 2.4ghz bands. The devices allow for much higher data throughput than the lower frequency LoRa devices, data transfer speeds are up to 203kbps.

The SX1280 also has a ranging mode that can measure distance by recording the time of flight of a special packet exchange between two SX1280s.

I have designed some plug in Mikrobus boards for the Ebyte and G-NiceRF SX1280 LoRa devices. The larger green PCBs are an ATMega328P based Mini Logger I also designed. It can be fitted with a single plug in Mikrobus compatible board, see the picture below;



The Mikrobus PCBs shown, the yellow ones, are breadboard friendly so will plug into a standard 0.1″ breadboard, very useful for prototyping.

A complete Arduino based transceiver fitted with an G-NiceRF SX1280 device is shown on the left of the picture above, complete with 2dBi WiFi antenna.

The board was programmed in the Arduino IDE.

Semtech publish sample code for the SX1280 but its not directly suitable for the Arduino environment so I used the Semtech code as a basis for writing some Arduino library files.

First Ranging tests

I used a large open field to test the ranging, these were the results;.

Actual Distance	Indicated Distance.
0	4.4M
50M	57.6M
100M	103M
150M	148M
200M	201M
250M	253M

So not too bad, the results did seem to be consistent for large open areas. So short distance ranging seems OK, what about longer distances ?

4.4km Ranging test

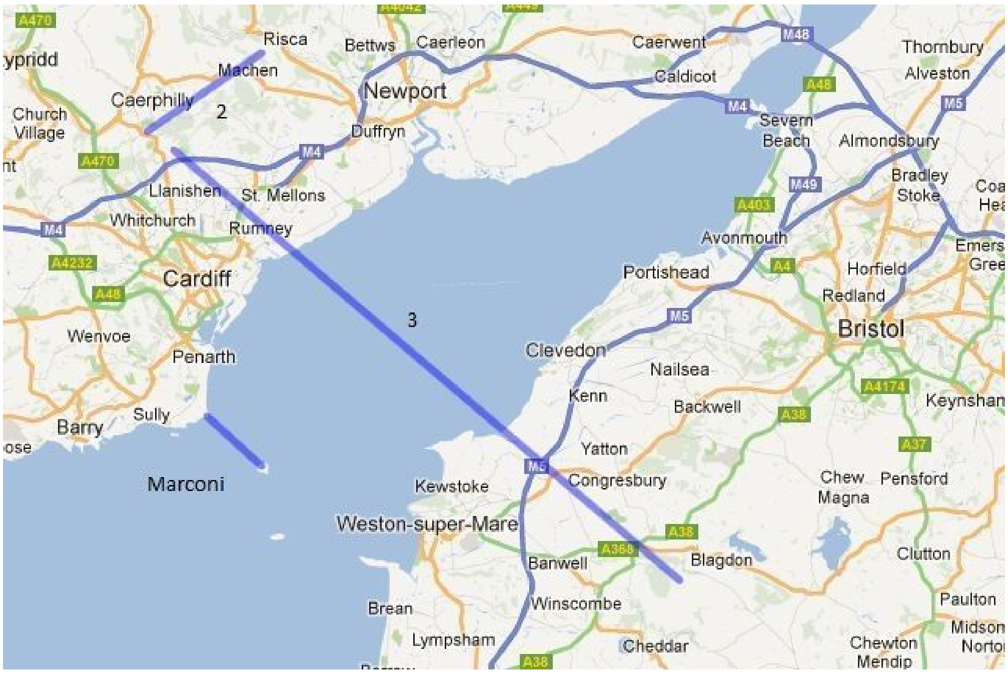
North of the Cardiff where I live there is a ridge and the location is around 150M higher than where my shed is located in the city. With a SX1280 transceiver on top of the 6M mast attached to my shed there is a relatively clear line of sight from the ridge to the device on the mast, the distance is 4.4km.

I wrote ranging software that would first transmit a ranging request at high power (10dBm!) and report whether the ranging request completed. The transmit power was then reduced and another ranging request made.

Over the 4.4km link the ranging worked down to -14dBm. Thus if the full power was used, 10dBm, this test indicates the ranging may have potential to work at over 69km.

40km Ranging Test

There is a location North of Cardiff (altitude 264m) suitable for a 40km link test to Black Down (altitude 325m) in the Mendips, the other side of the Bristol channel.



At the Cardiff end there was a SX1280 ranging receiver fixed to a pole and another for a basic link test. The view from here towards the Mendips is below;

At the Mendips end I had a matching set of devices, one as the ranging test instigator/transmitter and another for receiving basic link test packets. The results were logged to a micro SD card for later analysis. Basic 2dBi antennas were used at both ends.

At the Mendips end the ranging request transmitter would emit a long beep at the start of the test sequence, send a ranging request at 10dBm, then

9dBm, 8dBm etc. If the ranging request was successful then the would be a short beep. Thus to work out the power level when the ranging stopped working you only had to listen for the long beep and then count the short beeps.

The LoRa settings for the ranging were SF10 and bandwidth 406khz, the longest range settings that the SX1280 allows for ranging mode.

Ranging calculations

The path used for the ‘4.4km Ranging test’ mentioned above was used to test and calibrate the ranging function. The ground distance was 4.42km as measured on Google maps. The average ranging result from the SX1280 was 24515. This gave a conversion factor to metres of 0.1803.

The average ranging result for the Cardiff to Mendips link was 225982. Applying the 0.1803 conversion factor gives a distance of **40.745km**. The distance taken from Google Maps was **40.650km**. So the SX1280 ranging produced a measurement of +0.2% over actual. Not bad.

Summary – Distance Potential of the SX1280 ranging and point to point.

The ranging requests were received down to 4dBm over the 40km link. This would imply a potential range of **80km** LOS at 10dBm. Even modestly improved antennas at both ends (see antenna comparisons above) could double this range.

The longest range settings for point to point mode, SF12 and bandwidth 203khz should have a link advantage of 8dBm over the ranging settings described above, SF10 and bandwidth 406khz. Converting this 8dBm link advantage over the ranging settings, suggests a potential range for point to point LoRa in the **200km** region.

SX1280 Program Code

Semtech publish source code in C++ and a development kit for the SX1280 based on ARM\MBED.   I have adapted the code to work under the Arduino IDE, its been tested on ATMega328P and ATMega1284P. There is still some work to do on the ranging part of this code, when it it ready it is my intention to publish it as an Arduino library.

All the parts used in these tests were self funded, no complimentary parts were used.

GPS Performance Comparisons

1ST FEBRUARY 2019 BY STUARTROBINSON LEAVE A COMMENT

I have carried out a performance review of the common available GPSs that you might use for a LoRa tracker.

In most cases you will run the GPS in hot fix mode, and unfortunately most of the common GPS breakout boards that you get on eBay etc, are not suited for this in a power miser tracker that needs to survive on batteries for a month or longer. The reasons are explained in the article.

There is data of the performance of 12 common GPSs, signal strength (antenna effectiveness), real world power consumption and hot fix performance over an extended period. There are notes on methods of reducing tracker power consumption.

Choose the wrong GPS and your tracker, powered by a 2800mAhr battery, might only last 14 days. Choose the GPS wisely and the same battery can last 200 days.

The report is at the link below;

GPS Performance Comparisons

GPS backup issues 3 – Whats the alternative ?

9TH JANUARY 2019 BY STUARTROBINSON LEAVE A COMMENT

So if we don’t use a a super capacitor or a small lithium rechargeable, then what is the alternative ?

We could of course use a non-recharegeable lithium battery such as a CR1216 but in the holders these have much the same issue as the super capacitor, they are physically large, see picture.

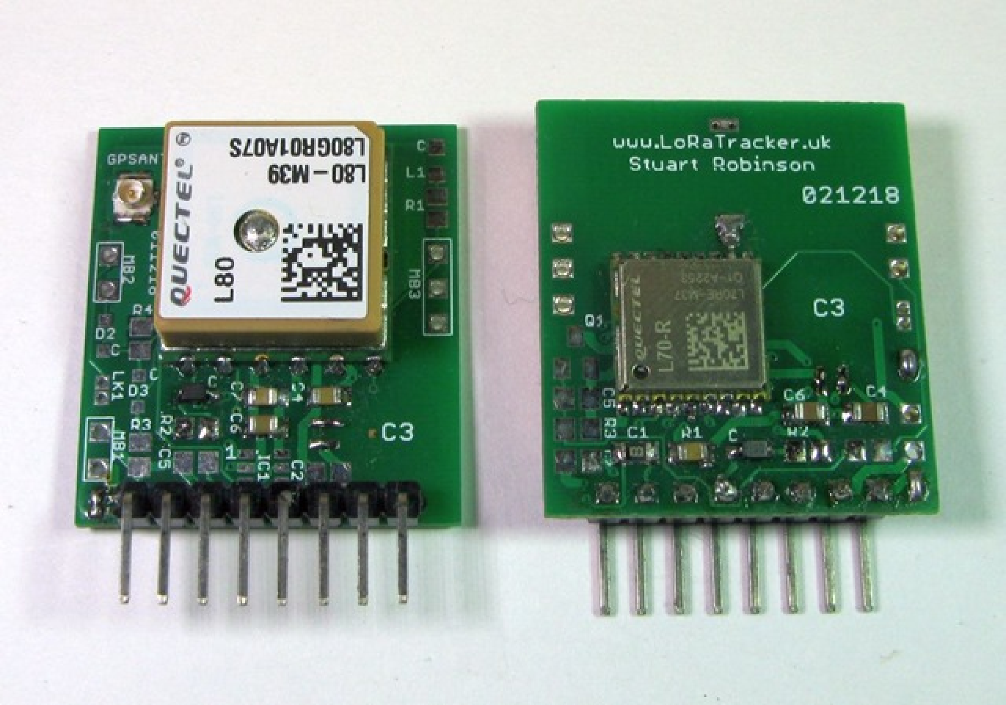
In addition even a battery as physically big as the CR1216 only has a capacity of circa 25mAhr, so that would only keep the GPS (@15uA backup current) going for 70 days. A tracker based on this setup would need it's GPS battery replaced around every two months.

The answer to the problem is quite simple really. With the small lithium rechargeable or supercapacitor options, then the backup power for the GPS is actually coming from the projects main battery, so why not use the main battery as the backup power source for the GPS ?

All we need to do is take a 3.3V supply from the project and feed this voltage via a diode to the GPS Vbackup supply. We then need to ensure that the 3.3V supply is present when the project (using the GPS) is in sleep mode. This is not difficult even if the project is using something like a TPL5110 to switch off \on the battery supply to the project. A low drop out regulator such as the MCP1711 has a quiescent current of 0.6uA, so that can be used to go between the projects battery and the GPS Vbackup supply without having a significant impact on total GPS backup current consumption.

Whilst the solution to providing a long term GPS backup current is simple to implement, very few GPS breakout boards provide access to the GPS Vbackup pin, so you cannot feed in an external backup supply. For this reason I designed my own GPS breakout boards for the Quectel L70 and L80 GPS. These are low cost high performing GPSs with substantially lower power consumption than most other GPS modules, see a detailed report here;

GPS performance comparisons



The L80 GPS has a slightly worse power performance than the L70, but it has its own ceramic patch antenna and easy to include in PCB designs and solder in place, I like it. The L70 has a balloon mode, for use up to 80km.

GPS backup issues 2 – Super Capacitor ?

9TH JANUARY 2019 BY STUARTROBINSON LEAVE A COMMENT

If a small lithium battery is not suitable; what about a super capacitor that can at least be charged fairly quickly so it might accrue enough charge during a short hot fix power on to last for several hours ?

There is a web site which is an online super capacitor charge and discharge calculator;

Super Capacitor Calculator

First lets see what size of super capacitor we need. Assume a maximum backup time of one day, a fully charged voltage of 3.0V and a minimum volts for GPS backup of 1.5V, a GPSs backup current of 15uA which is typical for a Ublox GPS for instance. The calculator suggests a capacitor of 1Farad would last a couple of hours longer than one day.

If the hot fix gap was 10 minutes and the hot fix takes 5 seconds with a GPS backup current of 15uA, then assuming no charge\discharge efficiency losses the charge current needs to be 600/5 times the backup current or 1.8mA. That's manageable.

So if we are to use a 1F super capacitor how big would it be ?

Checking on the Farnell web site a 1Farad 3.6V super capacitor would be around 20mm, perhaps a bit more, in diameter and 5mm high. That's really quite big, see picture for how big the 5.5V version 1F capacitor is. The GPS shown is one of the larger 25mm ceramic patch antenna types.

There are some small super capacitors around the same size as the Seiko MS621, but these super capacitors are of very limited capacity and would only provide a couple of hours backup (based on practical tests).

So if a super capacitor is too big to be practical and we have allready dismissed small lithium rechargeables, then what is the alternative ?

GPS backup issues 1

2ND JANUARY 2019 BY STUARTROBINSON LEAVE A COMMENT

A modern GPS will normally get a fix from cold in 30 – 60 seconds when it has a good view of the sky. Once it has downloaded enough information on the GPS satellites the main GPS power can be turned off and as long as a voltage is supplied to the GPS backup pin it will retain the satellite information in GPS memory. When the GPS power is then turned back on the GPS can acquire an updated fix in as little as 2 – 5 seconds, this is called 'hot fixing'.

In backup mode the GPS may consume only 5uA to 20uA, so clearly there are significant power savings to be had if we only want intermittent fixes from the GPS. A GPS that only needs to run for a few seconds to get an updated fix every 10 minutes or longer will use a lot less power that an GPS that is left running all the time.

A very common way to supply the backup supply to a GPS is to use a small Lithium battery, typically a Seiko MS621. However this is not without problems, the MS621 has a capacity of only 5.5mAhr and a max charge current of 100uA. So when first used with the GPS you need to leave it powered for 55 hours, just over two days, to fully charge the battery which is not very convenient. In addition you need to consider the cycle time. A typical GPS will consume 15uA in backup mode, so the MS621 will only last 15 days or so when fully charged. If the tracker is to use hot fixing for longer than 15 days then the backup battery needs to be charged, or it will just run out and hot fixing will fail.

If we are taking a GPS fix every 10 minutes and the hot fix takes 5 seconds, there is only 100uA x 5 seconds = 500uA seconds going into the battery. But in 10 minutes there is 10min x 60seconds x 15uA = 9000uA seconds going out. Thus the battery is draining 18 times faster than it is charging and will inevitably discharge quite quickly.

Although fitting an MS621 battery is common with GPS modules, its not suitable as a backup method if the GPS is to be used for a couple of weeks or more. A different backup strategy is needed.

Stuart Robinson

January 2019

Very Low Power Nodes

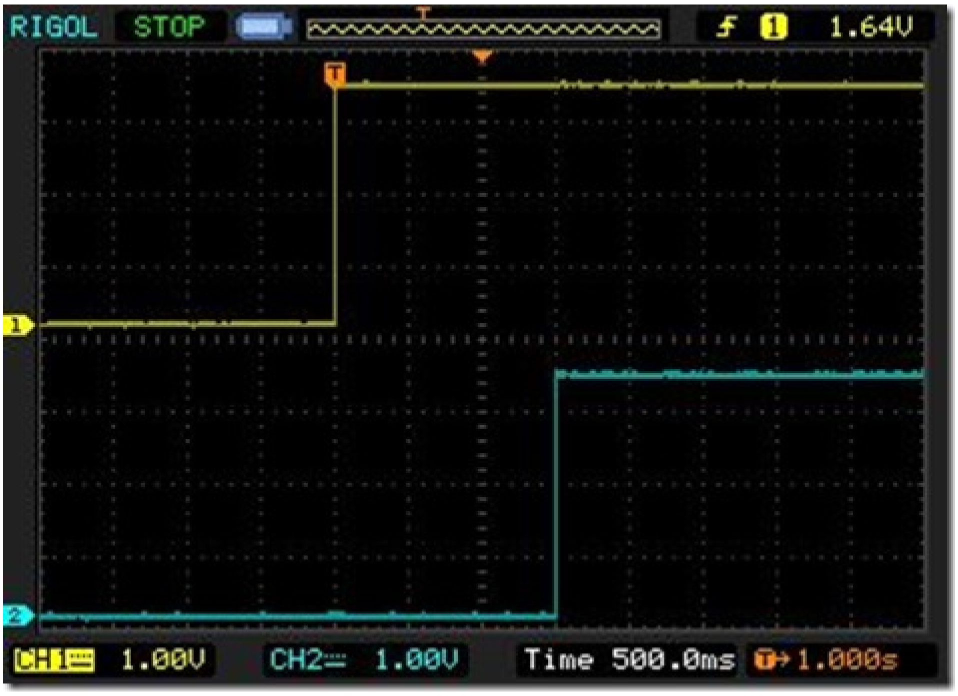
24TH DECEMBER 2018 BY STUARTROBINSON LEAVE A COMMENT

There is quite a lot of significance being attached to reducing sleep current of LoRa projects or nodes to very low levels, the lower the better apparently, but is sleep current alone a significant factor ?

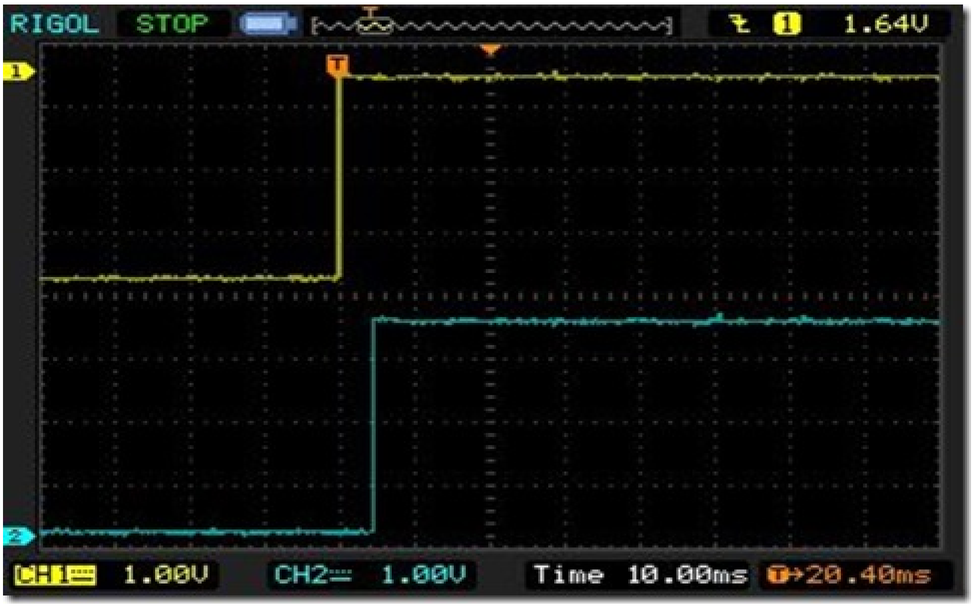
A bare bones ATmgea328P or 1284P based node with a LoRa module can be used in normal deep sleep mode and the sleep current is around 0.2uA, you can wake up from interrupt. Add a regulator such as MCP1700 if you need one, and total sleep current is around 1.5uA with a **TPL5010** acting as a watchdog and interrupt wake up device.

A popular alternative to the processor deep sleep approach appears to be to use the **TPL5110** device which can be used to completely power down a node, from seconds to 2 hours, with a deep sleep current of less than 0.1uA. This appears to be a significant improvement on the 1.5uA of a bare bones ATmega328 in deep sleep mode, or is it ? Is concentrating or being obsessed with the deep sleep current the best way of reducing overall power consumption ?

Take a look at the scope plot below, it shows (top trace) the power to a The Thing Network GPS tracker node based on an Arduino Pro Mini being turned on. The lower trace shows a logic pin being sent high at the point the tracker node software queued the packet ready to send, so the difference is the node wakeup time. The delay from power on to ready to send the packet was 1512mS, mainly caused by the Arduino bootloader delay. Power consumption during this 1512mS was 5.42mA.



Now compare a similar situation where the tracker node is put into deep sleep and woken up by an interrupt timer such from the TPL5010. The bare bones Arduino with decent regulator and LoRa device has a deep sleep current of circa 1.5uA. Note carefully the time it takes to wake up from deep sleep and be ready to send another packet;



Here the top trace the interrupt waking the tracker node from deep sleep and the bottom trace shows the same logic pin going active just before packet transmission. The delay here is a meagre 4.5mS.

So the deep sleep wakeup is very much quicker (4.5mS versus 1512mS) but does the quicker ‘boot’ time save power?

Its easier to consider the numbers here in terms of uA/Seconds rather than mAhr and look at the numbers over a day. Lets assume our tracker node sends a transmission every 10 minutes or 144 times a day.

TPL5110 Power Down Node

The TPL5110 equipped node has a sleep current of 0.1uA so uses 0.1 x 60 x 60 X 24 = 8640uA/Seconds per day during sleep.

At every wakeup it uses 1.512 x 5420 = 8185uA/Seconds per wakeup.

There are 144 of these transmissions, so that is 1180086uA/Seconds per day.

So our TPL5110 node uses 8640 + 1180086 = 1188726uA/Seconds per day sleeping and waking up, or **0.33mAhr per day**.

TPL5010 Deep Sleep Interrupt Wakeup Node

A TPL5010 equipped node has a sleep current of 1.5uA so uses 1.5 x 60 x 60 x 24 = 129600uA/Seconds per day during sleep.

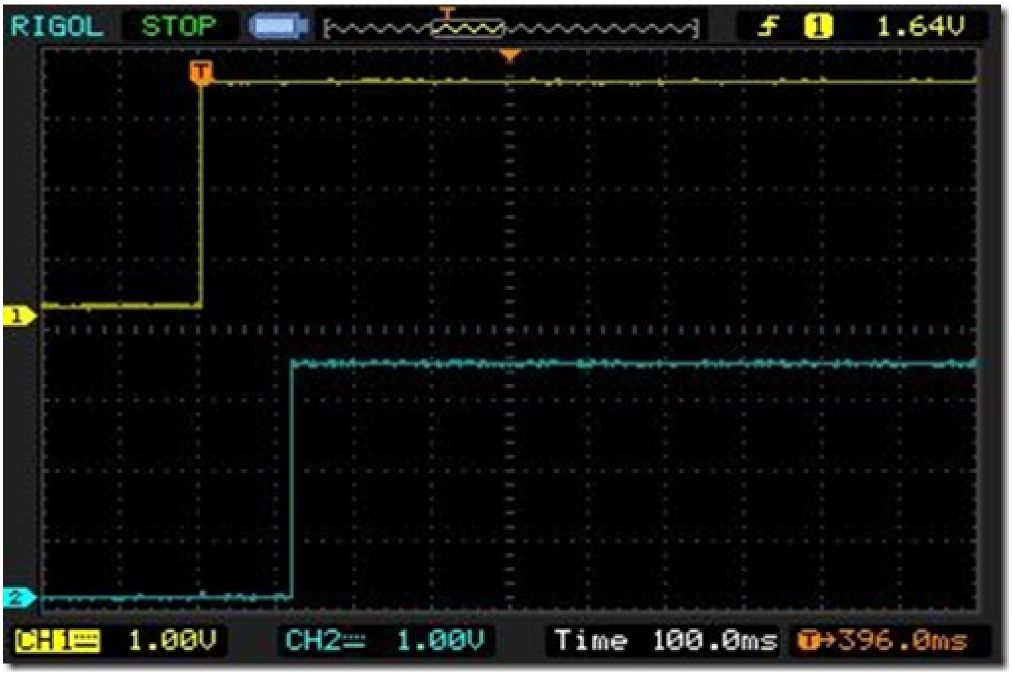
At every wakeup it uses 0.0045 x 5420 = 24.4uA/Seconds per wakeup.

There are 144 of these transmissions, so that is 3513uA/Seconds per day.

So node uses 129600 + 3513 = 133113uA/Seconds per day sleeping and waking up, or **0.037mAhr per day**.

Arduino Boot Loader

Of course we don’t have to use the Arduino bootloader to load sketches, we can use an ISP programmer although its less convenient.



Here the delay between power applied and ready to send a packet is much less; 120mS.

Other Advantages of the TPL5010?

Now the TPL5010 can act as a watchdog device, if you don't respond to the periodic interrupt in a timely fashion then there will be a reset of the processor, which is useful protection against software crashes or power supply induced errors.

One additional advantage capitalises on the Arduinos quick wakeup form deep sleep. At 4.5mS it becomes feasible to set the TPL5010 to say a one minute interrupt wakeup and then count the interrupts to get in effect a programmable wakeup time. A simple loop counting 10 of the interrupt wake ups then gets you a 10 minute wakeup, without the need to change the TPL5010 timing resistor.

Conclusion?

Using the **TPL5110** to power down node a node (with Arduino serial bootloader) uses **0.33mAhr** per day sleeping and waking and the **TPL5010** uses **0.037mAhr** per day. So they are significant power savings to be had using a bare bones Arduino setup and the **TPL5010** as a wakeup timer. If you remove the Arduino serial bootloader then there is not a lot to choose between the two methods, although the TPL5010 does provide added flexibility.

Stuart Robinson

December 2018

LoRa – does CAD work at weak signal levels ?

28TH SEPTEMBER 2018 BY STUARTROBINSON 2 COMMENTS

The SX127x data sheet defines CAD as;

When in CAD mode, the device will check a given channel to detect LoRa preamble signal

And;

Principle of Operation

The channel activity detection mode is designed to detect a LoRa preamble on the radio channel with the best possible power efficiency. Once in CAD mode, the SX1276/77/78/79 will perform a very quick scan of the band to detect a LoRa packet preamble.

There is little to suggest that CAD works on the body of the packet, which in practice is the major par of the transmit time. So whilst CAD is supposed to detect the pre-amble of a packet, does it detect a packet at other times ?

To test this the idea would be to transmit long packets (of several seconds) and have a receiver constantly running the CAD process. When each CAD period was finished the CAD detected flag would be read and a LED turned on or logic level set if the flag was active. Thus by observing the LED so that you could see for how long a packet was detected by CAD and how consistent the detect was.

The packets would be sent in sequence of descending powers (17dBm down to 2dBm) so it would be possible to see by observing the LED how well CAD performs as the signals got weaker. The link test software I use was changed to send packets that were close to 5 seconds long at SF12 and bandwidth 125khz.

The test transmitter was fitted with an SMA terminator on its antenna input, this should cut the actual transmitter power (and received distance) significantly.

I did an initial test with transmitter and receiver close. It's clear that the CAD worked on the entire packet. In my test the LED on the transmitter (indicating a packet send in progress) closely followed the CAD detect on the receiver. There was one curious exception and it was consistent, approximately 360mS into the 5079mS packet there would be one CAD detect that failed. It was also noted later that this gap occurred with weak signals too.

With the principle of CAD working on the entire packet established I moved on to seeing if it also worked on weak packets.

The transmitter and receiver were loaded with my link test software and I adjusted the locations of both so that the packets stopped being received at around the 9dBm point.

Here is the link test software output;

```
Mode1 17dBm,11 16dBm,11 15dBm,11 14dBm,11 13dBm,10 12dBm,10
11dBm,11 10dBm,10 9dBm,8 8dBm,4 7dBm,5 6dBm,0 5dBm,1 4dBm,0
3dBm,0 2dBm,0
```

```
Mode1,CSV,11,11,11,11,10,10,11,10,8,4,5,0,1,0,0,0
```

You can see from the counts that packet reception starts to fail at around the 9dBm point and stops almost completely after 7dBm.

Once the test link had been characterised the receiver was loaded with the CAD detect test software. The positions of the transmitter and receiver were kept the same to ensure consistent link performance.

With the receiver LED being on when CAD succeeds, we can easily tell when a packet is being received. At the start of the descending power sequence

(17dBm packet) there is an audio tone transmitted which can be heard on a UHF handheld. When the tone from the remote transmitter is heard you can now count the arriving packets by observing the CAD detect LED. By counting the packets you know which power is in use when the CAD starts to fail, indicated by the LED flickering.

The CAD process was consistent (apart form the gap @ 360mS mentioned above) and started breaking up at around 14 to 13dBm dBm so within about 4 or 5dBm of link failure, this is better performance than expected.

So in approximate terms CAD detect does appear to work on the entire packet, but is only consistent up to around half the distance that the weakest packets are detected.

In the CAD detect software used each CAD detect and print of information to console was taking approximately 62mS. I left the receiver running for 10 minutes to see if there were any false CADs. There were 119 false CADs in 10 minutes or about 1 every 5 seconds or about 1 : 80 CADs are false.

What of the gap in the CAD at the beginning of the packet? This would appear to be at the time the preamble pattern ends and the packet data starts, so it's possible CAD will see the preamble, or the packet, but not when both are mixed together.

More ESP32 adventures

9TH SEPTEMBER 2018 BY STUARTROBINSON LEAVE A COMMENT

I have shown in a previous post that it had been possible to cut the deep sleep current of a ESP32 module, with a LoRa device, FRAM and display down to the 8uA region by careful attention to pull up resistors and type of display.

I thought it would be useful to be able to include a Micro SD card into my projects. For the trackers it would be very handy to be able to log GPS data to an SD card. This could be useful if for some reason the LoRa radio link stopped working. For the same reason this would be a useful backup for remote sensor or data gathering applications.

The current taken by micro SD cards seems to vary greatly, the one I was using was consuming 200uA or more when inactive, which would make battery life poor.

I tried switching off the power to just the SD card, but that had the effect of dragging pins on the SPI bus low which increased the sleep current of the LoRa device.

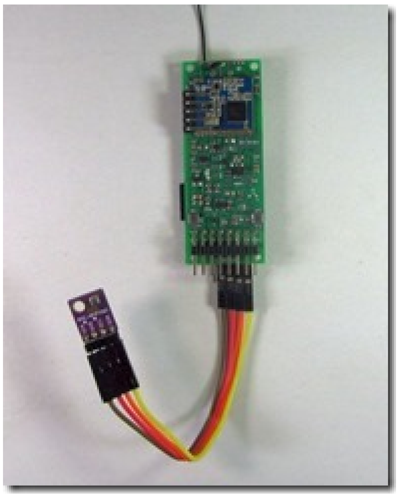
A different approach was needed.

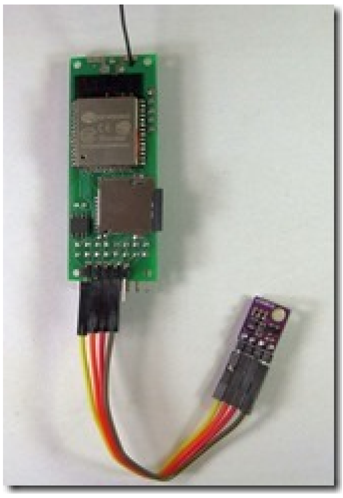
I built up an ESP32 board that used a MOSFET to turn off the VCC power rail to all the devices external to the ESP32, LoRa module, display, FRAM, BME280 sensor and micro SD card.

The picture shows the prototype in action, sleep current was measured at 11uA, and that includes a resistor network to measure the battery voltage and a charger IC to allow for solar battery charging.

This power switching approach also made the software side an easier, it was not necessary to take all the attached devices through a shutdown sequence.

Of course this does mean that all devices need to be re-initialised every time the ESP32 comes out of sleep mode but this is not a difficult issue to deal with.



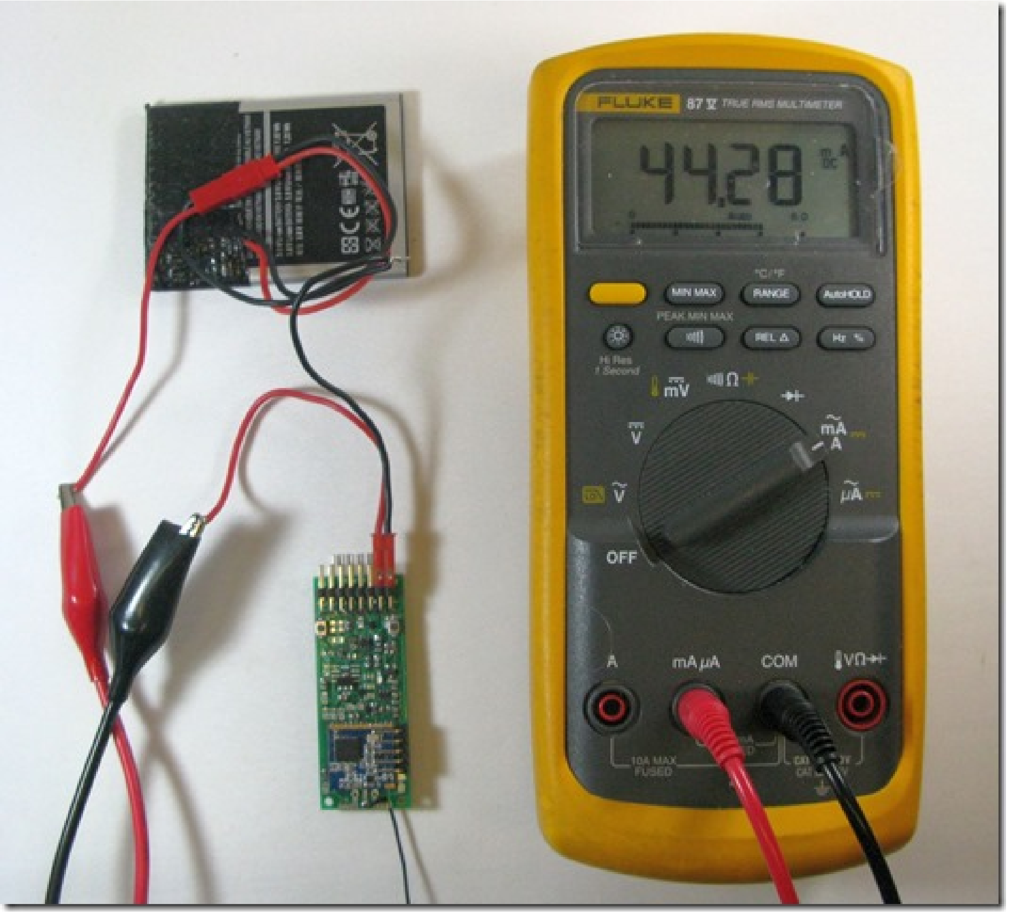


Measuring Low Currents

9TH SEPTEMBER 2018 BY STUARTROBINSON 2 COMMENTS

You may find you need to measure the current consumption of your latest cunning project.

If this current is in the mA range, say from 2mA up to 500mA or more then a standard multimeter will normally do the job. All you need to do it put the multimeter on a suitable current range in series with the power supply, as in the picture;



One issue that can affect the measurements is the voltage drop across the multimeter. The multimeter measures current by allowing the current to flow through a low value resistor and then measures the voltage drop across that resistor (normally called the shunt). The multimeter can then calculate the current in the resistor.

This voltage drop across the resistor\multimeter is the **burden voltage** of the multimeter and its consequence is that the project whose current consumption you are measuring is now lower than before.

A Fluke 87 for instance has a burden voltage of 1.8mV per mA on the mA range so if the current flow is 100mA, there is an 180mV voltage drop across the multimeter and the projects supply voltage is now 180mV lower.

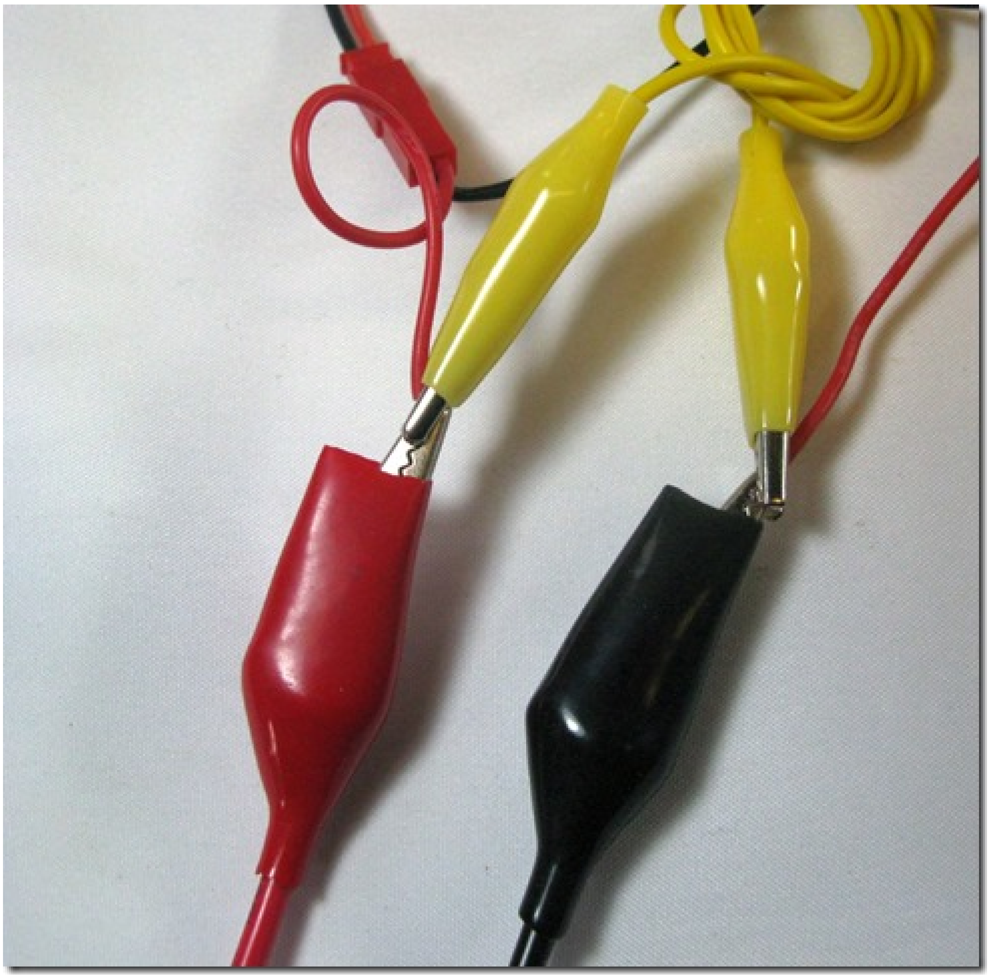
What do you do if you want to measure the sleep current of a project which is likely in the tens of uA range? If you have the multimeter on the mA range there won't be enough resolution to give a good indication of a current in the low uA range.

Also you probably cannot put the multimeter on the mA and then switch to uA as the switching can disconnect power from your project and cause a reset.

So what happens if you start with the multimeter on the uA range? The Fluke 87 has a burden voltage of 100uV per uA on this range so if the project is drawing 100mA when its not in sleep mode the burden voltage of the multimeter will be;

100,000 * 0.0001 = 10v !

Oh dear, with a 10V voltage drop across the multimeter your project is unlikely to run in at all.



So what to do ?

The answer is fairly straight forward. Start with the multimeter on the uA range and short across it with a bit of wire.

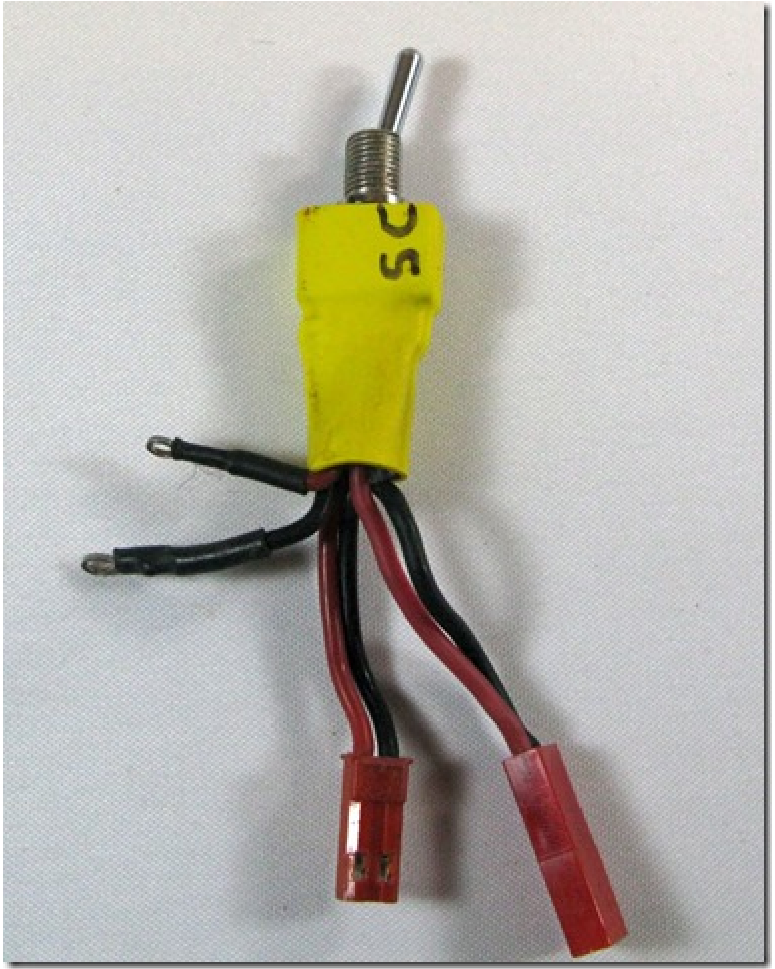
See picture, the multimeter connections are the red and black crocodile clips, the shorting wire is the yellow one.

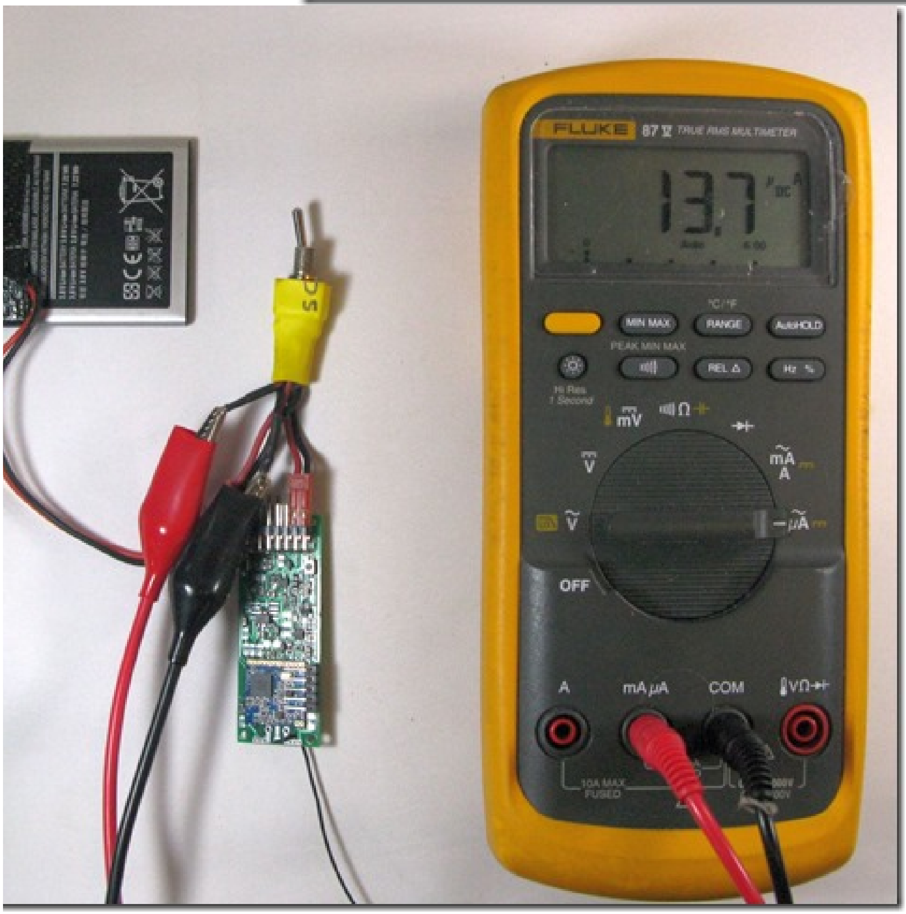
Then when your project puts itself into low current mode, remove the wire. The burden voltage of 100uV per uA should not cause problems with most projects and you should be able to accurately measure currents down to 1 or 2 uA.

The project shown in the first picture has a sleep current 12.9uA, so the burden voltage is only 1.29mV, which is unlikely to cause a problem. My OWAN B35T (much cheaper than a Fluke !) has a burden voltage of 50uV per uA.

Rather than use a bit of wire to short out the multimeter, make up a switch unit that has power in and out connectors (see picture) with tags for the multimeter connections in the positive line. The switch shorts across the multimeter connections. Short out the multimeter with the switch put the multimeter on the uA range and when the project is in low current mode open the switch. Simple.

← OLDER POSTS


SC


FLUKE 87 V TRUE RMS MULTIMETER
13.7
MIN MAX
RANGE
AutoHOLD
REL Δ
Hz %
OFF
A
mA µA
COM