ETRX357 ZigBee® Module

Application Note – Power Consumption

Table of Contents

1 Introduction

In order to calculate the average power consumption (or battery life) of an ETRX3 series based product it is necessary to know the exact amount of power consumed for each individual activity.

In case the device is a router, or a coordinator this is straightforward as the unit will mostly consume 26.5mA (28.5mA in boost mode) in receive mode and for short intervals between 30mA and 43.5mA depending on power settings in transmit mode. A router or coordinator is not supposed to go to sleep. When it comes to end devices, estimating the required power becomes more complex.

This document gives an overview of common activities and the resulting power consumption to aid calculating the overall battery life. The firmware used to trigger these activities is based on the ZigBee PRO stack Ember ZNet 4.5.2., and the device type is a sleepy end device. The power consumption shown includes any time taken to wake up from deep sleep and go back to this state after completing the described task.

Please note that in the same way that message transit times over multiple hops are non-deterministic in a mesh network, the power consumption may vary with network utilisation as multiple retries and varying backoff periods may be required. Because of this the document at hand should only be regarded as a guideline. We highly encourage you to do in-system measurements for each individual application in a real life scenario to get most accurate figures.

2 Power Consumption of Different Activities

For all experiments described in this chapter an ETRX357 is set to be a sleepy end device (SED) which is sleeping with just the on chip RC timer running, unless there is a functionality which is to be executed. The main difference to a router is that on an end device even when fully powered, only the microcontroller is running and the radio only gets switched on when in use.

When not part of a network, an ETRX357 will consume 7.5mA whilst being fully awake, because only its microcontroller is running. When scanning for PANs, trying to join a PAN or doing an energy scan, the radio is fully switched on and the unit will consume 26.5mA for most of the duration of the scan (28.5mA during boost mode).

To measure the power consumption, the device under test is attached to a Telegesis Development board and powered with 3.0V provided by the development board. The current consumption is measured in terms of a voltage drop across a 10Ω series resistor.

A Tektronix MSO2012 is used to capture the voltage dropped across the 10 Ohms resistor connected in series with the power supply of the module. The area underneath the resulting waveform was measured using the build in mathematical functions of the MSO2012. The resulting area in mVs (millivolt-Seconds) was converted into Charge (Q) in Coulombs using the following formula:

Q = I x t with $I = \frac{V}{R}$ where R = 10Ω

The charge in Coulomb (Q) is equal to one Ampere flowing for one second. In analogy the charge of one micro Coulomb (µQ) is equivalent to one milliampere for one millisecond.

The measurements were checked using three ETRX357 modules from different batches to verify the repeatability of the results.

2.1 SED Polling -> No Message Present

For this measurement the on chip timer is set up for the end device to poll its parent at regular intervals, whilst it is made sure that there is no data to be polled from the parent. The length of the interval between any two polls has no effect on the amount of energy consumed per poll. It could be seen though that the semi random backoff period inserted by the stack caused polls to require different amounts of time and therefore different amounts of power. The longest and shortest waveforms were captured and the oscilloscope was used to integrate the area under each waveform (see figure 1). The initial in-rush current peak is clipped but it has no significant effect on the integration result.

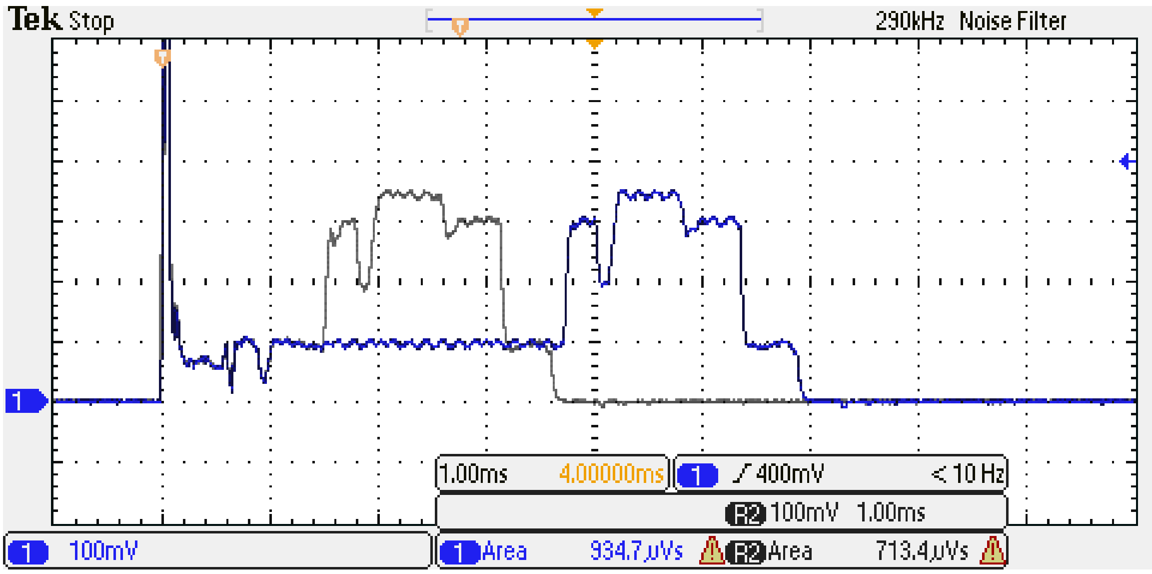


Figure 1: Polling

As seen in the figure above a poll consists of 5 phases. In the first phase the on-chip microcontroller wakes up and the power consumption goes to about 9.5mA. After that the unit switches the radio on to send a poll, during which the power consumption first goes up to 30mA to listen for a clear channel, then to 34mA to transmit. Next, the radio is switched back to receive mode to listen for a response. Finally there is a short period at 9mA to prepare the device for sleep mode. Table 1 below gives an indication of how long the individual phases are and the corresponding current consumption.

Mode	Current Consumption	Shortest	Longest
Sleep	1µA	?	?
Wakeup	9.5mA	1.5ms	3.7ms
Listen	30mA	360µs	340µs
Send Poll	34.5mA	760us	780µs
Listen for reply	30mA	520µs	500µs
Processing	9.5mA	460µs	500µs
Sleep	1µA	?	?
Total charge		71µC	93µC

Table 1: Poll for Data

“Longest” and “Shortest” refer to the total length of the on-period, but it is clear from the table that the initial wakeup period dominates and the lengths of the individual phases do not always correlate with each other. All samples were taken with no additional traffic on the air, as additional traffic could potentially increase the power consumption (collision prevention, and actual collisions taking place).

The total charge taken from the waveform integration shown in figure 1 (93µC and 71µC) can be seen to differ hardly from the summed approximate figures in table 1.

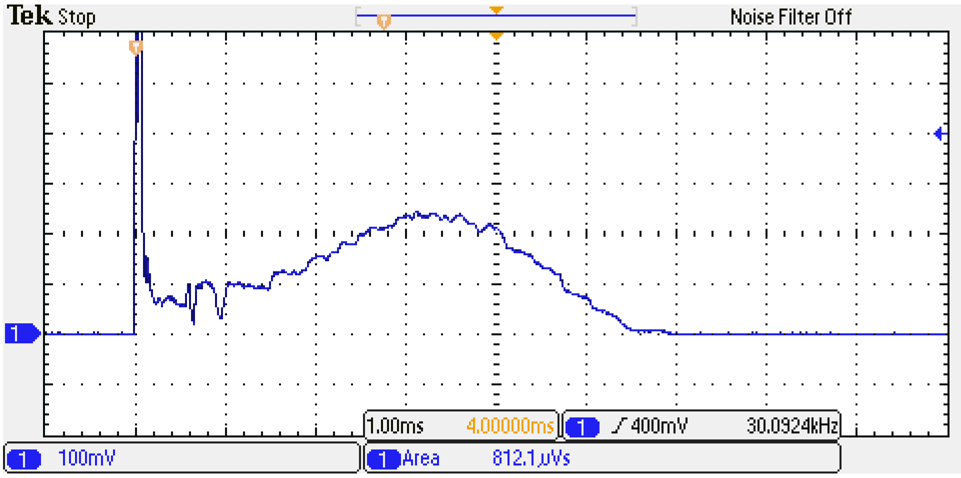


Figure 2: Average of 128 polls

The maximum and minimum consumption figures are of less interest than the long-term average. Rather than extrapolate from Table 1, the waveform was averaged over 128 polling events to get a result that makes no assumptions about the lengths of the individual phases or their statistical distribution (figure 2). The result is **81µC** per poll, which is actually very close to the mean of the longest and shortest measurement.

Please note that when polling in intervals of less than about 5 seconds only broadcasts, which are sent from the parent using just in time messaging, will be reliably buffered and passed on to the end device, but unicasts may be lost.

Due to the internals of the stack it is to be expected that the measurements detailed above also apply to a mobile end device, given it has not moved away from the parent it has previously polled and it is polling within the timeout period.

Finally, polling triggered by external interrupt in sleep modes is expected to produce the same results.

2.2 SED Polling -> Single 1-Byte Unicast Message Present

As in the previous section the end device is polling for data. The only difference is that now there is a unicast message waiting to be passed on to the child. The application payload of the message is 1 byte (plus 8 bytes containing the sender's EUI64). As before the longest and shortest possible exchanges were captured in figure 3.

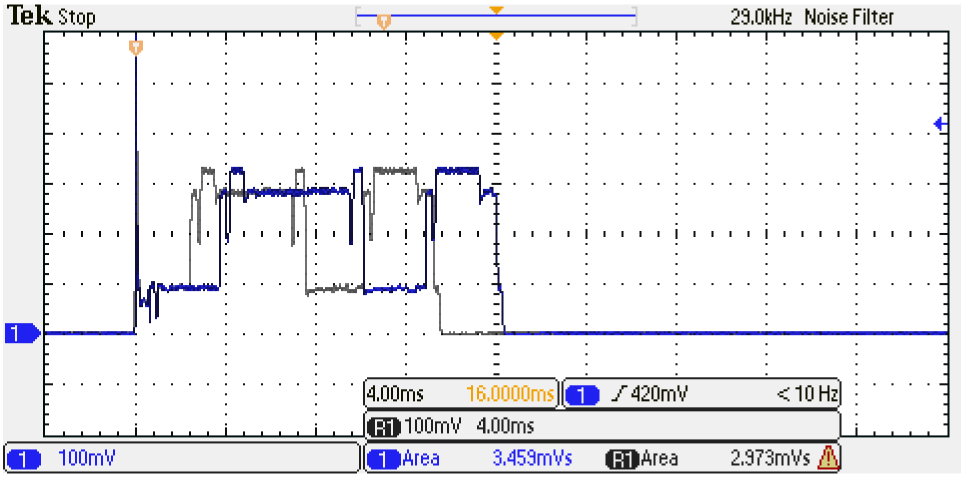


Figure 3: Poll for 1-Byte Unicast

The waveform shown in figure 3 captures the entire transaction including transmission of the aps acknowledgement.
In the initial example it was shown that the power consumption based on the scope's integration of the waveform is reliable and that the average consumption is close to the mean of the maximum and minimum. Therefore it can be concluded from figure 3 that the power consumed for polling a single byte unicast is between 297µC and 345µC with the mean being **321 µC.**

The current consumed when receiving messages will vary considerably not just according to the message length itself, but by the way it is processed by the receiver before going back to sleep. For example, in **Error! Reference source not found.** a broadcast text message was received containing a 40-byte string. The two traces were selected so that the current profiles in the initial phases had the same duration, but with one trace the text is sent to the module's serial port with a prefix, whereas with the other all the text is discarded so that the serial port is silent. The longer trace consumes 84% more power than the shorter one.

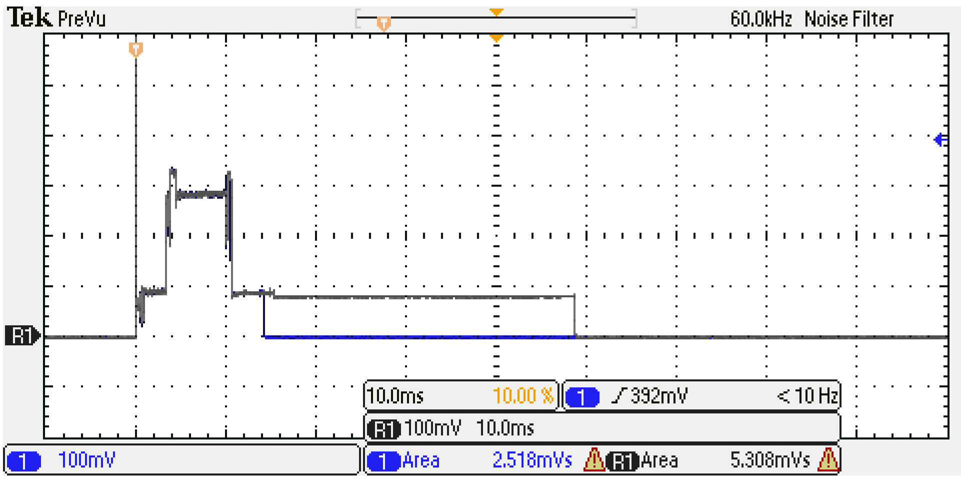


Figure 4: Different ways of processing a Message

2.3 ED Polling -> Single 50-Byte Unicast Message Present

In analogy to the previous chapter a unicast with an application payload of 50 bytes (plus 8 bytes containing the sender's EUI64) is polled form the parent. Again the longest and shortest possible exchanges were captured in figure 5.

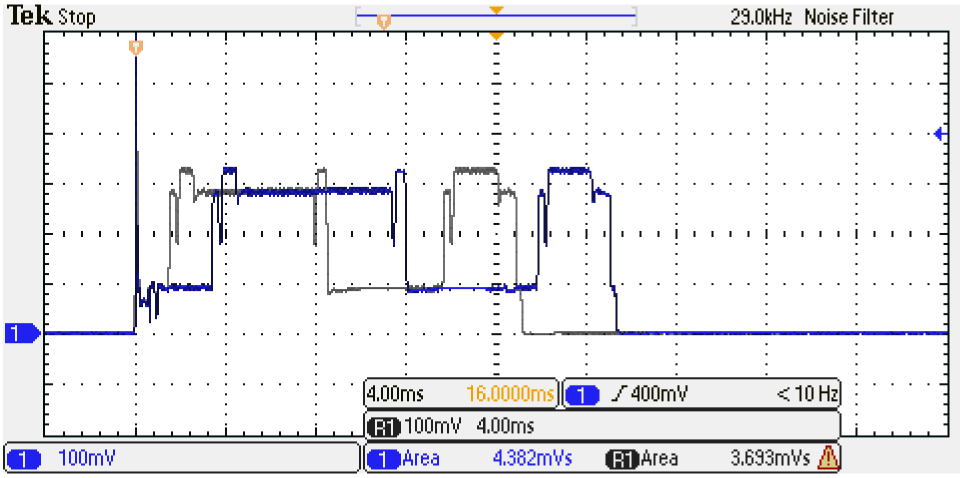


Figure 5: Poll for 50-Byte Unicast

Looking at figure 5 it can be seen that the charge consumed for this transaction is between 369µC and 438µC with the mean being **404µC**.

2.4 ED Polling -> Single 1-Byte Broadcast Message Present

The measurement is now repeated when polling a Broadcast with 1-Byte application payload (plus 8 bytes containing the sender's EUI64).

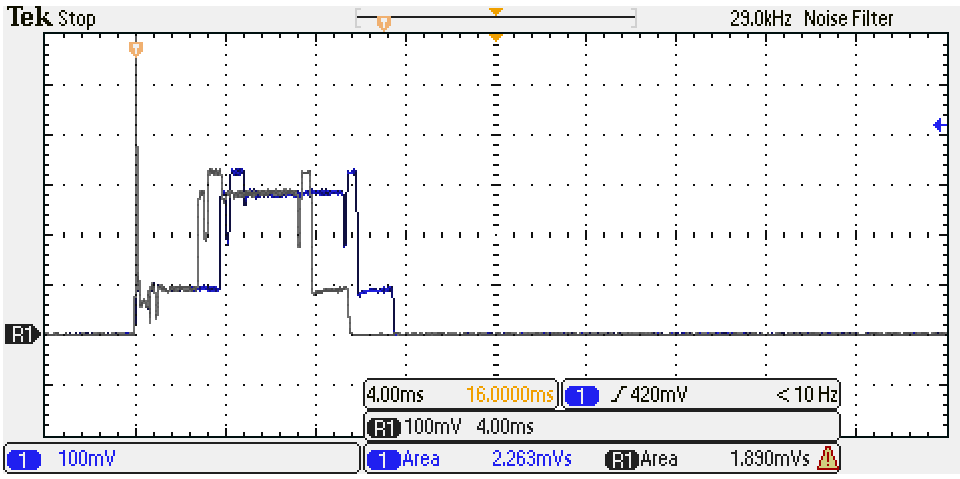


Figure 6: Poll for 1-Byte Broadcast

It can be seen that the power consumed for this transaction is between 189µC and 263µC with the mean being **226µC**. This is less than the power required to receive a 1-Byte unicast because there is no aps acknowledgement being sent as a reply to receiving a broadcast.

2.5 ED Polling -> Single 50-Byte Broadcast Message Present

As in the previous test a broadcast is polled from the parent, this time with a payload of 50 bytes (plus 8 bytes containing the sender's EUI64).

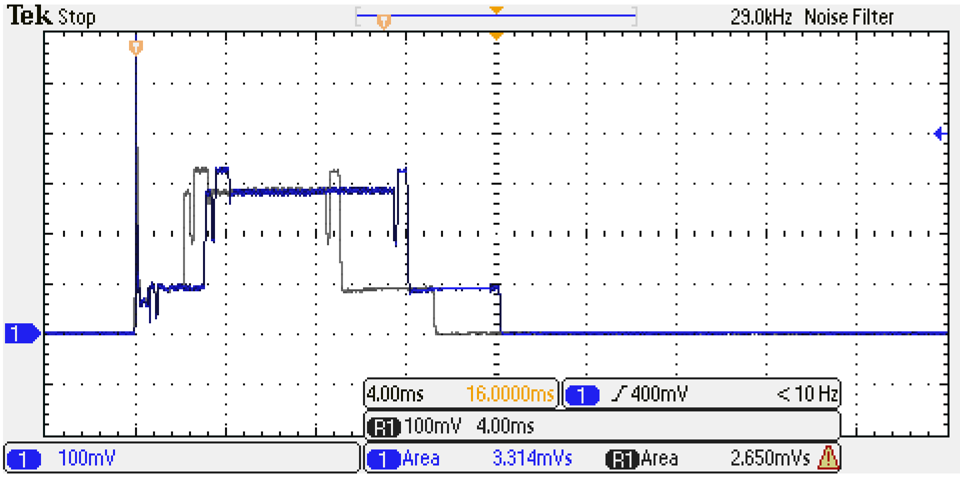


Figure 7: Poll for 50-Byte Broadcast

In the same way as before the power consumed for this transaction was calculated from the integration result of the oscilloscope. The power consumption is between 265µC and 331µC with the mean being **298µC**.

2.6 SED Sending a Unicast with 1 byte payload

Now we are sending a unicast from the sleepy end device to another node in the network. The payload of this unicast is only one byte plus the 8 byte EUI of the sender which is attached to the frame (optional).

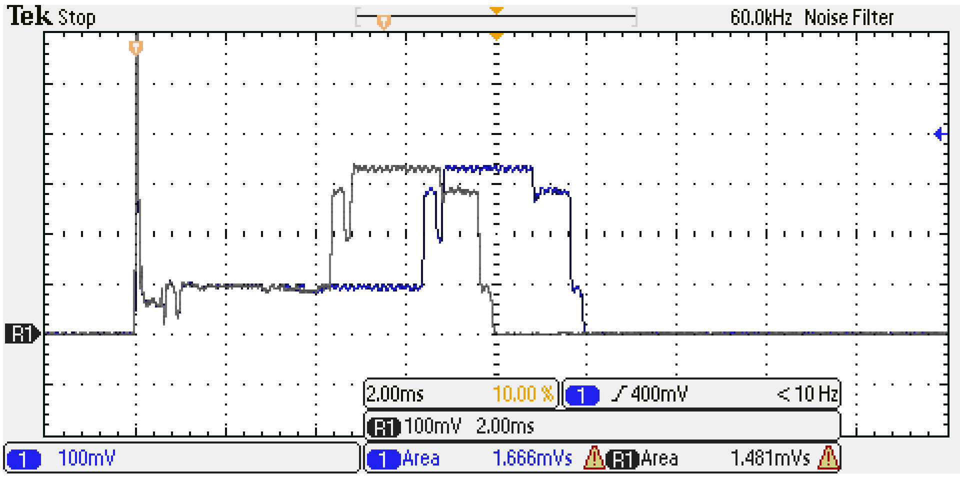


Figure 8: Sending a 1-byte Unicast

It can be seen that the power required for this transaction is between 148µC and 166µC with the mean being **157µC**.

Please note that this trace only includes sending the unicast itself (and the mac acknowledgement), but not the aps acknowledgement. Because of this an identical waveform is expected for sending a 1-byte broadcast. When requiring an APS acknowledgement, the overall operation involves three current pulses in the sleepy end device, and the operations are

1. Send a unicast message and receive a MAC ACK (as shown in figure 8)
2. Send a poll, receive the APS ACK and send a MAC ACK
3. (optionally) Send a poll in case there are incoming messages and receive a MAC ACK

Depending on the timing it is also possible to receive the mac ack and the aps ack in the same transaction as shown in figure 9. This behaviour can be seen in case the aps ack is available at the parent very quickly, e.g. in case the child is addressing its parent itself.

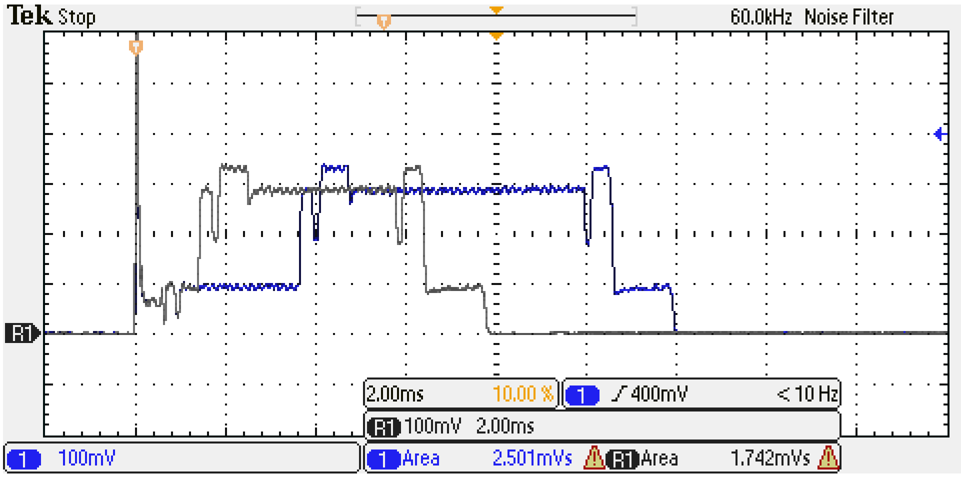


Figure 9. Sending a 1-byte Unicast with immediate APS ack

In this particular case the power consumption is between 174µC and 250µC depending on the CSMA backoff period, with the mean being **212µC** for the entire transaction including the aps ack.

Receiving the APS ACK as part of a separate poll looks very similar to receiving a short message as detailed in chapter 2.2. The third (optional) operation, sending a final poll, is the same as chapter 2.1.

2.7 SED Sending a Unicast with 50 byte payload

In the same way as in the previous chapter we are now sending a unicast from the sleepy end device to another node in the network. The payload of the unicast is 50 bytes plus the 8 byte EUI of the sender which is attached to the frame (optional).

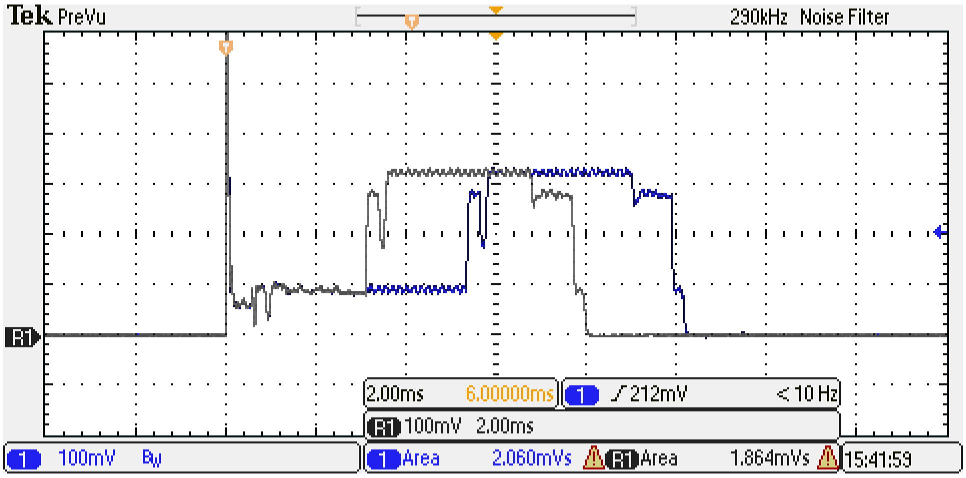


Figure 10: Sending a 50 Byte broadcast

From figure 10 it can be seen that the overall power consumption for sending a 50 byte unicast/broadcast excluding the reception of an aps ack is between 186µC and 206µC, with the mean being **196µC**.

As previously also the scenario of receiving the aps ack as part of the same transaction is examined. Figures 11 and 12 are showing the longest and shortest waveforms required for this transaction.

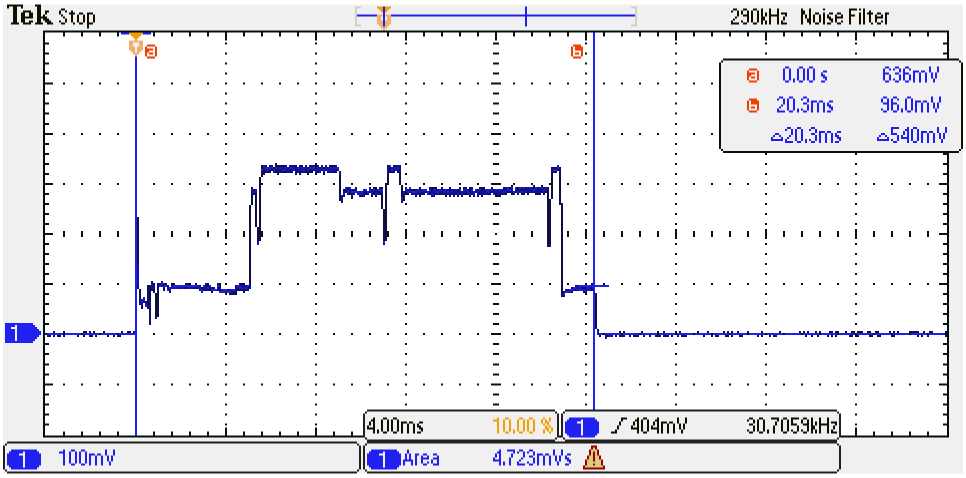


Figure 11: Sending a 50-Byte unicast (short backoff)

As observed previously the semi-random delay put in by the CSMA mechanism can cause different timings for this operation.

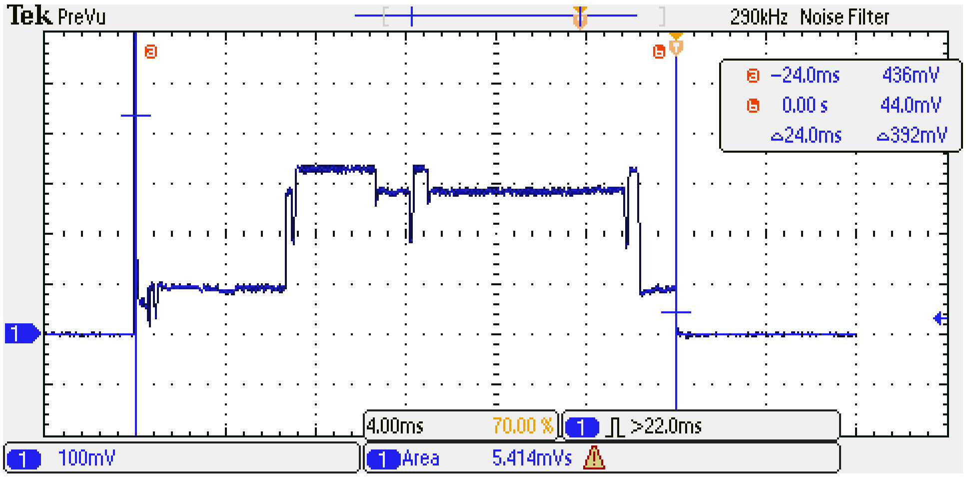


Figure 12: Sending a 50-Byte unicast (long backoff)

Looking at the figures above it can be seen that sending a 50-Byte unicast (including sender's EUI) including reception of the mac ack requires a charge between 472µC and 541µC, with the mean being 506µC.

3 Conclusions and Discussion

The power consumption of an end device can rely on a variety of factors, therefore whenever doing battery life calculations the actual power consumption should be measured and averaged over time in the real life system to get accurate data.
Factors do not have to be as obvious as payload lengths, poll rates and hop counts, but also the overall network traffic (or in-band noise) will have an impact on the power consumption of a wireless mesh network.
In this document some guidelines are provided to give you a rough estimate to see if the project you are planning is feasible with a given power supply (battery).

Please note that in order to achieve ultra low power consumption it is required to either define all I/Os to be outputs, or to pull all inputs to a defined level as floating input pins will increase the current consumption. Using the build in pull-ups will also increase the overall current consumption.

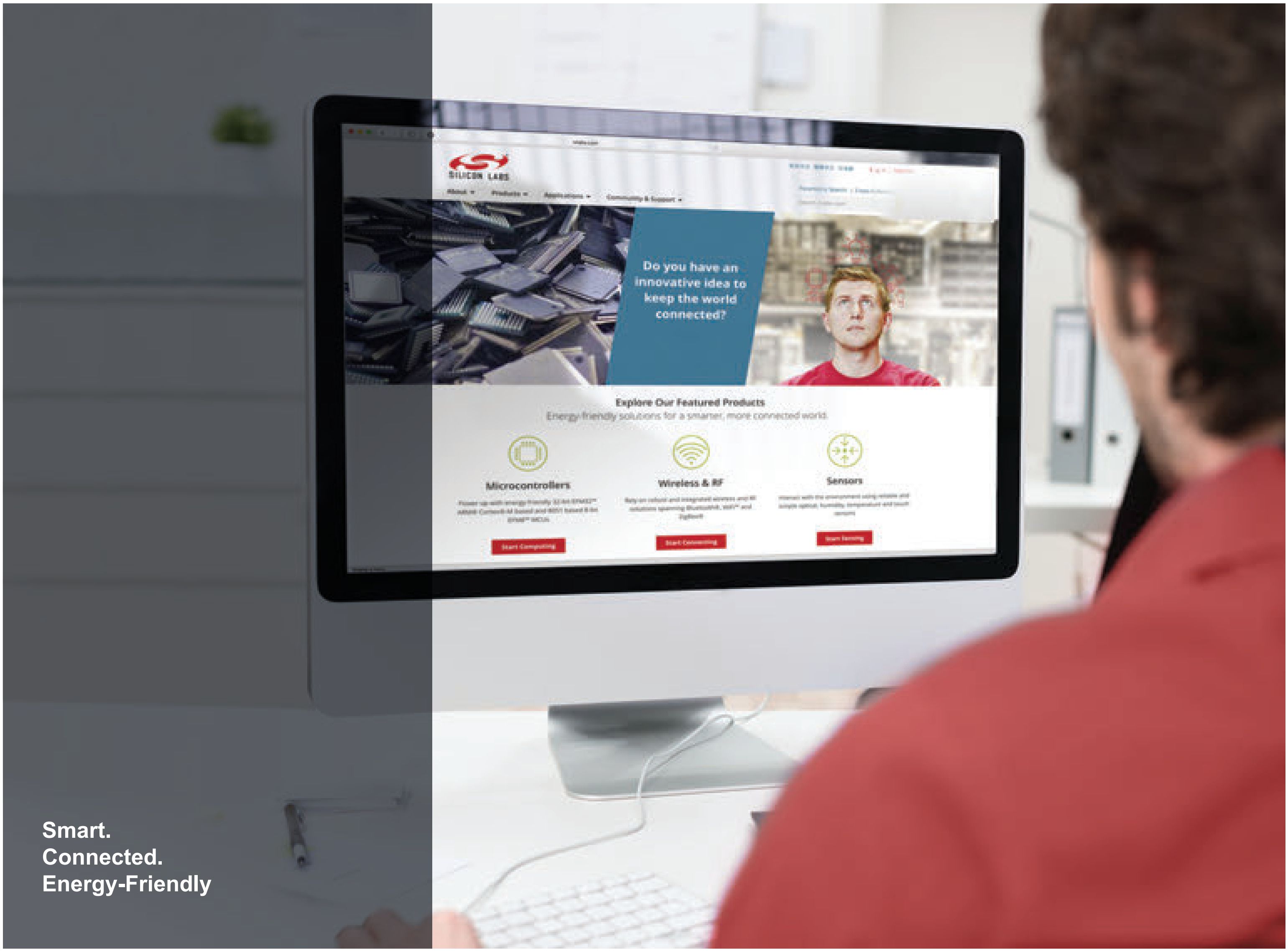
Finally, it needs to be taken into account that the figures are relating to EmberZnet4.5.2 and may vary with different/newer stack versions.

3.1 Summary of Measurements

Table 2 provides an overview of the mean power requirements for different activities which were detailed in chapter 2 of this document.

Activity	Mean power Consumption
Poll, no message present	81µC
Poll, 1-byte unicast message	321µC
Poll, 50-byte unicast message	404µC
Poll, 1-byte broadcast message	226µC
Poll, 50-byte broadcast message	298µC
Sending 1-byte Unicast	157µC
Sending 1-byte unicast with APS ack reception	212µC
Sending 50-byte Unicast	196µC
Sending 50-byte unicast with APS ack reception	506µC

Table 2: Mean Power Consumptions



Products
www.silabs.com/products

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com