A New Way to Measure the Power Consumption of Really Low-Power Hardware (for Software People)

How to Make Sure Your Software Runs for Years on Tiny Batteries

We recently got a new tool at Thingsquare, which helps us ensure that we have extremely low power consumption of our software: the Otii Arc (https://www.qoitech.com/) power box by Sony (https://www.sonymobile.com/global-en/smart-solutions/otii/). This little box quickly lets us – a bunch of software persons – measure and understand the power consumption of the software that we run on our low-power hardware.
Today we use this to build an wireless dimmer and make it extremely low-power: we want to run it on one single coin-cell battery.

(We have no affiliation with the creators of the Otii Arc, we just happened to stumble upon their product and fell in love with it!)

Measuring Power Consumption – Old vs New

The lifetime of an IoT device is determined by its power consumption, but measuring power consumption was always tricky.

Old

New
The traditional way to measure power consumption of a device is to connect multimeters, oscilloscopes, or specialized power analyzers to the hardware that is to be tested. These tools are slow, difficult to set up, and make it difficult to measure the power consumption for more than a few seconds. Also, they typically require a fair amount of experience with electronics and electrical engineering.
The new way, enabled by tools like the Otii power box, consists of plugging in a few cables and immediately seeing the results on the screen – simple!

An Ultra-Low-Power Wireless Dimmer

To test out our new equipment, we build a wireless dimmer for our industrial lamps in the office.

Our aim is to make the wireless dimmer run with an extremely low power consumption. We would like to be able to run it off of a coin-cell battery.

We build our dimmer with a Launchpad board, one slider potentiometer, and one coin-cell battery. We load the Launchpad board with a custom application, shown below, that periodically reads the slider and sends a signal to lights in the same wireless network that tells them to dim to the value given by the slider.

To measure the power consumption of the setup, we simply disconnect the coin-cell battery and connect the power pins to the Otii power box instead.
The Hardware Challenge: The Power Cost of the Slider

Our wireless dimmer use a slider for the user to set the dim level of the lamps. Although we have a good grasp of the power consumed by the wireless System-on-a-Chip and on-board peripherals (read more about it in this article (/blog/articles/sensortag-power/)), we do not know how much power our slider hardware uses, nor how much additional power is needed to periodically sample the value of the slider.

Fortunately, we can measure this with our new tool.

We program the device with our custom program, and let it run for a while to allow it to settle with the wireless network and go into its most low-power mode.

We then let it run for a while to get a good, representative average of its power consumption.
The Thingsquare Blog: A New Way to Measure the Power Consumption of Really Low-Power Hardware (for Software People)

Measuring the steady-state power consumption of the hardware. This is the cost of simply existing and will always be there. This mainly consists of sleep currents for the System-on-Chip (SoC) and peripherals and the slider. We see an average current of around 7.6 uA.

The Software Challenge: the Power Cost of Best-effort vs Reliable Reporting

We use the Otii power box to investigate the difference between two versions of the code with one very small change: in one version of the code, the application sends the value of the slider without making sure that the value was received. In the other version, the system makes sure that the value is received.

In terms of code, a small change. In terms of behavior, a fundamental difference. In terms of power consumption – let’s find out!
This is a very small change in the code, but the change alters the behavior of the system because the device will have to do more work to ensure that the new value was received. In particular, the device will have to be awake to hear that the backend sent an acknowledgment packet to convince the device that the value was received.

Reliable

```c
thsq_sset_printf_important("slider", "%d", value);
thsq_push();
```

Best-effort

```c
thsq_sset_printf("slider", "%d", value);
thsq_push();
```
We can now quantify the effect of our code: we see that the cost of the transmission in the best-effort case consumes about 0.71 uWh whereas the reliable version consumes some 2.74 uWh.

We can also see why the reliable case costs more: the reliable report requires the radio to be on when listening for the acknowledgment from the backend. Moreover, if this acknowledgment is not received, the device will have to send the data again, until the acknowledgment is received. This will inevitably consume more power.
The Details: the Actual Code

The Thingsquare system has several features convenient for lighting applications. It is very easy to make a low-power dimmer that can control the light settings for a group of lamps. The code for this dimmer unit can be found below (direct link here (https://gist.github.com/msloth/c93a1fbd051990efc1b40e56b3ce969f)). It periodically reads the position of a potentiometer and if it has changed, it sends this to the network so the lamps can react accordingly.

- The code (click to open)

Do it Yourself!

Interested in setting this up yourself? You can follow our getting started guides here (/start/)!