An introduction to SquidBee

Emanuele Goldoni[*]
TLC & Networking Lab, University of Pavia
Via Scarsellini 2, 46100 Mantova, Italy
emanuele.goldoni@unipv.it

March 15, 2008

Abstract

Quoting its website, SquidBee is *an Open Hardware and Source wireless sensor device*. In this article we examine the SquidBee platform and explain how to use it to build a wireless sensor network.

1 Hardware platform

A SquidBee wireless sensor network is composed of several SquidBee *sensor nodes*, which acquire data in the field and transmit it (using ZigBee[1]) to a *gateway node*, for further processing and data collection. The SquidBee platform is based on open hardware components; the building blocks of a sensor node are

- an Arduino board (see [2]), equipped with a programmable AVR microcontroller, several I/O ports and a USB connector;
- an XBee shield, which implements a ZigBee serial modem;
- several sensors, attached to the Arduino I/O ports; a standard SquidBee sensor node has light, temperature and humidity sensors.

The AVR microcontroller is user programmable with standard opensource tools (see Section **??**), and its code can be customized to easily acquire and process data from the sensors, and broadcast it on the air.

1.1 Interpreting data

A Squidbee node should send, by default, a stream of data like this:

[*]Special thanks to Davide Cavalca (*davide.cavalca01@ateneopv.it*) for his contribution to 1 and **??**

[1]ZigBee is a low-cost, low-power, wireless mesh networking standard, targeted at radio-frequency applications which require a low data rate, long battery life, and secure networking. For these reasons it is widely used in wireless sensor networks. For more information see [1].

```
@1|0|data0-692|data1-414|data2-42
@1|1|data0-680|data1-414|data2-43
@1|2|data0-668|data1-419|data2-43
@1|3|data0-659|data1-416|data2-43
@1|4|data0-991|data1-416|data2-44
@1|5|data0-992|data1-415|data2-43
@1|6|data0-644|data1-423|data2-43
...
```

Where the data pattern is the following:

```
@node-id|counter|(light val.)|(humidity val.)|(temperature val.)
```

As you can imagine, sensors give an analog value but the micro converts volts into an integer number ranging from 0 to 1023 (the ADC used inside the AVR is a 10-bit one). However in an heterogeneous wsn we ca not assume that all the nodes use the same kind of sensor or that they all have a 10-bit-ADC. The solution for that it is quite simple: the sensors must send data using standard units, like Celsius degrees for temperature or a percentage as far as relative humidity is concerned. Using the sensors datasheets, we have found how to obtain meaningful values with Squidbee nodes and then we have modified the code to convert measurements before sending them.

1.1.1 Temperature

The temperature sensor (LM35) gives 10 mV/°C but the AVR converts volts into a number: 0 for 0V and 1023 for 5V. To obtain Celsius degrees again, you have to multiply the value from the sensor by 0.4878 (that comes from $\frac{5}{1024 \cdot 0.01}$).

1.1.2 Humidity

The humidity sensor used (808H5V5) in Squibee nodes has (approx) a linear response to relative humidity: the voltage output ranges from 0.8V to 3.9V responding to 0-100% RH. Given the digitalized value h from the sensor, you can to convert it in a RH percentage using the formula $\frac{h \cdot 0.4878 - 80}{3.1}$.

1.1.3 Light

The Light Sensor is a LDR (Light Dependent Resistor) standard 10K one, as stated on Squidbee website. However it is unclear how the digital output relates to environment light: a simple test can only show that the sensor output decreases as the light intensity increases. For that reason we carried out a number of experiments to throw light on this issue. The sensor has been exposed to natural light source under different conditions (ranging from 3 to 3000 lux[2]) and put next to a light meter, which was set to measure real light intensity values in lux. The results we obtained in our lab[3] show that the relationship between voltage output and lux is

[2]lux is the SI unit of illuminance and luminous emittance, used in photometry as a measure of the intensity of light. One lux is equal to one lumen per square metre, where 4π lumens is the total luminous flux of a light source of one candela of luminous intensity.

[3]Thanks to Marco Bianchera (*marco.bianchera01@ateneopv.it*) for his help during tests.

non-linear, as suggested also by many websites (for e.g. see [13]). If l is the 10-bit digital value provided by the node, light intensity in lux can be computed using the formula $\left(\frac{1024 \cdot 1}{l} - 1\right) \cdot 500$ However you need to be aware that this only seems to be the best (and simplest) fitting curve - all the values are based on empirical results. In addition to this, it necessary to keep in mind that the LDR output may additionally be influenced by wavelength, distance or relative position of the source, dust, settling time after a very bright exposure etc...

References

[1] ZigBee, from Wikipedia: `http://en.wikipedia.org/w/index.php?title=ZigBee&oldid=194597719`

[2] Arduino project main page: `http://www.arduino.cc/` `http://reactivated.net/writing_udev_rules.html`

[3] Measure Light Intensity using LDR: `http://togglezero.blogspot.com/2007/11/ldr-light-depender`