

# **The CogBag Proximity Sensor**

## **V1.0 – August 3, 2005**

The Cognitive Bag is becoming a remarkable device packed with the latest that technology has to offer. With a built-in iPaq, cellular phone, GPS receiver, RFID tag reader, and at least three Mote modules, this bag packs more computing horsepower (and a higher price tag) than any fashion accessory owned by James Bond.

The reality of any bag, no matter how useful or expensive, is the high probability of being left behind on some days. We all know the feeling of arriving at school only to realize our books were left behind, or leaving a restaurant and leaving our wallet or purse in a booth, or watching the bus fade into the distance with our keys, bicycle, glasses, or whatever still on board.

For the Cognitive Bag, such a scenario would not do. We needed to create a device that would monitor the owner's relation to the bag and, if the bag were in danger of being left behind, warn the owner so that the bag could be retrieved before it was lost forever. Though it might be tempting to simply tether the bag to the owner, such a tether might appear suspicious to the general public. We require a solution that is inconspicuous, allowing the owner to blend in any crowd.

The CogBag Proximity Sensor is our answer to this problem. The sensor is built from a Mica2Dot, modified for use with two CR2354 coin-cell batteries, and packaged to wear around the wrist as one would wear an ordinary sports watch.

### **What it does**

If the CogBag is nearby, the sensor is silent and blinks it's LED once every ten seconds or so. If the CogBag moves beyond a set distance away, the sensor beeps and flashes the LED once every second until the bag is returned.

### **How it works**

First, know that the CogBag contains a MoteGPS, which has been set to broadcast either UTC time or battery millivolts every two seconds.

The Proximity Sensor listens for the GPS broadcasts and determines relation to the bag by measuring the signal strength. If the strength is over a certain threshold, then the bag is considered "close" and the sensor shuts down for 15 seconds. If, on the other hand, the signal strength is too weak or if no signal is present at all, then the sensor warns the owner by flashing (or sounding) an alarm until the bag is brought back within range.

## A few words on battery life

Early in the proposal phase, we identified battery life as our largest obstacle. We wanted a solution that could last for at least a month between battery changes, but knew that battery capacity would be limited in any device that can be worn about the wrist. For this device, meeting both usability and aesthetic requirements would be a challenge.

A CR2354 coin-cell battery is rated for 350mAh. Under normal conditions, a Mica2Dot consumes about 17mA of current. This would give our device a useable life of only 20 hours.

We have developed methods to take advantage of the Mica2Dot's power-down modes so that our useable life is typically 80 hours using a single CR2354 battery. Because we are using two such batteries, life can be extended to 160 hours.

Still, 160 hours (about seven days) is far short of our 31-day mandate.

Fortunately, the sensor need not be active 24 hours a day. Indeed, the sensor should be active only when trips are expected. Because of this, we made the sensor responsive to message that prompt extended deep-sleep periods between trips.

All this means that the device is good for 160 hours of total trip time per month. This equates to an eight hour trip time for every business day, which is more than sufficient in our application.

And, if that isn't enough, we can add at least two more coin-cell batteries without sacrificing appearance.

## The Radio Packet Interface

The proximity sensor has been designed to work hand-in-hand with modes 3 & 4 of the Mote Modem. This makes it easy for people to interact with the sensor without requiring specific knowledge of TinyOS messages.

Using a Mote Modem in mode 3 or 4, you can interact with the sensor using the following queries:

<u>Query:</u>	<u>Response:</u>	<u>Use:</u>
PING	PROXIMITY_PONG	Determines if sensor is in area.
PROXIMITY_POWER	PROXIMITY_MV=nnnn	Reports battery voltage, 2800 typical.
PROXIMITY_SLEEP1	None	Puts the sensor into deep sleep for 1 hour.
PROXIMITY_SLEEP5	None	Puts the sensor into deep sleep for 5 hours.
PROXIMITY_SLEEP10	None	Puts the sensor into deep sleep for 10 hours.
PROXIMITY_ATSLEEPnn	None	Specifies how many seconds unit should remain in deep sleep mode once entered.

The setting PROXIMITY\_ATSLEEP tells the device how many seconds it should stay in a deep sleep state once entered. This command works almost exactly as the ATSLEEP command explained in the Command and Control section of this document. One important difference is that PROXIMITY\_ATSLEEP does not save the value to persistent storage.

Because the sensor normally sleeps when it is near the bag, it will be difficult to interact with the device when a bag is nearby. It is best to remove the bag from the area when interacting with sensor. The GPS unit in the bag can also be suspended for 30 seconds by sending it a GPS\_PAUSE. If neither options are possible, then the message should be sent to the device every second for a period of a minute or more to ensure the message gets through before the sensor returns to a sleep state.

### **The Serial Interface on USART0**

By default, the sensor's USART0 has been programmed to operate at 9600 baud, N81. You can connect the device to a PC by using the Crossbow MIB510 interface board and a program such as HyperTerminal.

Alternatively, a TTL connection is available on J21 or J22. Attach to pin 28 as transmit data (data from the mote to an external device). This connection is meant for embedding the unit into a larger system, as you cannot connect directly to a PC using this method—a TTL to RS-232 adapter is required. If you really want to connect to a PC and not use the MIB510, adapters are readily available from HVW Technologies ([www.HVWtech.com](http://www.HVWtech.com)) for only \$10.

When the sensor first powers-up, the following sign-on message is displayed:

```
Extended TOS Platform.  
APP: PROXIMITY  
Jul 28 2005 at 13:56:02
```

The sign-on message is the best opportunity to verify that the serial connection has been made correctly, and that the sensor has been programmed with the correct version of software.

Every second, the sensor will print "Tick."

If the CogBag is in nearby, the device will print "Sleep" and go to sleep. When the device wakes up again, the sign-on message is printed and the process starts over again.

### **Command and Control**

The Proximity Sensor can be controlled through USART0 using familiar AT-style command strings, similar in spirit to a Hayes smart modem. When the device is not sleeping, it can be put into a "command" mode by sending it the string "ATCOMMAND" (without the quotes) on USART0, followed by a carriage return [cr]

character.

Note that the sensor is not responsive when in deep sleep, so the bag should be removed from the area to prevent the deep sleep mode from being entered.

It is envisioned that a systems integrator will configure the device using a terminal program such as HyperTerminal. When the integrator enters the command string “ATCOMMAND” (without the quotes) followed by a carriage return, the following is displayed:

```
CMD MODE BEGIN
OK
```

The integrator could now get the menu of commands available. This is accomplished by typing “ATH” followed by a carriage return. The following is displayed:

```
COMMANDS :
ATBAUD4800 .. SETS 4800 BAUD
ATBAUD9600 .. SETS 9600 BAUD
ATBAUD19200 .. SETS 19200 BAUD
ATBAUD38400 .. SETS 38400 BAUD
ATBAUD57600 .. SETS 57600 BAUD
ATMODEn .. SETS OPERATING MODE 0-4
ATSLEEPnn .. SETS M2 SLEEP PERIOD, IN SECONDS
ATTIMEOUTnmm .. SETS ALARM TIMEOUT, IN SECONDS (hex)
ATLEDn .. SETS LED MODE 1=ENABLED 0=DISABLE
ATTHRESHnnn .. SETS THE POWER THRESHOLD FOR ACTION (hex)
ATDEFAULT .. RESTORES SYSTEM DEFAULTS
ATSHOW .. SHOW CURRENT SETTINGS
ATSAVE .. EXIT COMMAND MODE & SAVE
ATEXIT .. EXIT COMMAND MODE WITHOUT SAVING
ATH OR AT? .. THIS SCREEN
OK
```

Baud rate commands are self explanatory. The others we’ll discuss here.

Use ATMODE to set the operating mode. Mode 2 is the default mode, which has been explained in detail thus far. Modes 0 and 1 are used for special cases only.

#### Mode 0

Causes the sensor to run at full power and never sleep, and also to spontaneously transmit the PROXIMITY\_MV=nnnnn message every five seconds.

#### Mode 1

Causes the sensor to run at reduced power but never fully sleep.

In modes 0 and 1, the device is always receptive to radio and serial interaction. This can ease configuration and development.

The setting `ATSLEEP` tells the device how many seconds it should stay in a deep sleep state once entered. The value is expressed in hexadecimal. Longer periods increase battery life, but increase the time it takes to notify a user that the bag is no longer nearby. Shorter periods decrease battery life, but make the sensor more responsive. The default value is 9 seconds.

Use `ATTIMEOUT` to set the maximum amount of time, in seconds, that the alarm will sound before the unit enters a 1-hour deep sleep mode. Default is 60 seconds.

The setting `ATTHRESH` sets the threshold RSSI value used to determine CogBag proximity, expressed in hexadecimal. Use lower values to enforce tighter proximity, larger values for looser. Default value is 1C0, which is a distance of several meters.

Setting `ATLED` either enables or disables the LEDs. Though it is usually most desirable to have LEDs enabled, it might be advantageous to disable them in some situations to save power.

Use `ATSHOW` to display current settings. An example show is as follows:

```
SETTINGS :  
MODE=2  
THRESHOLD=1C0  
TIMEOUT=3C  
SLEEP=9  
BAUD=9600  
ENABLE LEDS=1  
OK
```

Command `ATSAVE` is used to save the settings into flash and exit command mode. The unit will automatically reset and the new settings will take effect.

Command `ATEXIT` is used to exit command mode without saving the settings into flash. The unit will automatically reset and the old settings will be restored.