**INDEX**

1. **General** ................................................................................................................................................. 4  
   1.1. General and safety information .................................................................................................................. 4  
   1.2. Conditions of use ...................................................................................................................................... 4  

2. **Waspmote Plug & Sense!** ................................................................................................................................... 5  
   2.1. Features .................................................................................................................................................. 5  
   2.2. Sensor Probes ....................................................................................................................................... 5  
   2.3. Solar Powered ...................................................................................................................................... 6  
   2.4. Programming the Nodes .......................................................................................................................... 7  
   2.5. Radio Interfaces ...................................................................................................................................... 8  
   2.6. Program in minutes .................................................................................................................................. 9  
   2.7. Data to the Cloud ................................................................................................................................... 9  
   2.8. Meshlium Storage Options ....................................................................................................................... 10  
   2.9. Meshlium Connection Options .................................................................................................................. 10  
   2.10. Models ................................................................................................................................................ 11  
   2.10.1. Smart Parking ................................................................................................................................... 12  

3. **Hardware** .................................................................................................................................................. 13  
   3.1. General Description ................................................................................................................................. 13  
   3.2. Specifications ......................................................................................................................................... 13  
   3.3. Electrical Characteristics ......................................................................................................................... 13  
   3.4. Assembling ............................................................................................................................................ 14  

4. **Sensor** ..................................................................................................................................................... 16  
   4.1. Specifications ......................................................................................................................................... 16  
   4.2. Measurement Process ............................................................................................................................... 16  

5. **Board configuration and programming** .................................................................................................... 18  
   5.1. Application considerations ....................................................................................................................... 18  
   5.1.1. Deployment of the motes ....................................................................................................................... 18  
   5.1.2. Interference of other vehicles ............................................................................................................... 20  
   5.1.3. Variation of the reference value ........................................................................................................... 20  
   5.1.4. Determination of the state of the spot ................................................................................................. 20  
   5.1.5. Network structure ............................................................................................................................... 20  
   5.2. Installation of the mote .............................................................................................................................. 24  
   5.3. Powering the mote .................................................................................................................................. 33  
   5.3.1. Battery options ................................................................................................................................. 33  
   5.3.2. Estimated lifetime ............................................................................................................................... 34  
   5.4. API ......................................................................................................................................................... 34  
   5.5. Example code ....................................................................................................................................... 37
1. General

1.1. General and safety information

- In this section, the term “Waspmote” encompasses both the Waspmote device itself and its modules and sensor boards.
- Read through the document “General Conditions of Libelium Sale and Use”.
- Do not allow contact of metallic objects with the electronic part to avoid injuries and burns.
- NEVER submerge the device in any liquid.
- Keep the device in a dry place and away from any liquid which may spill.
- Waspmote consists of highly sensitive electronics which is accessible to the exterior, handle with great care and avoid bangs or hard brushing against surfaces.
- Check the product specifications section for the maximum allowed power voltage and amperage range and consequently always use a current transformer and a battery which works within that range. Libelium is only responsible for the correct operation of the device with the batteries, power supplies and chargers which it supplies.
- Keep the device within the specified range of temperatures in the specifications section.
- Do not connect or power the device with damaged cables or batteries.
- Place the device in a place only accessible to maintenance personnel (a restricted area).
- Keep children away from the device in all circumstances.
- If there is an electrical failure, disconnect the main switch immediately and disconnect that battery or any other power supply that is being used.
- If using a car lighter as a power supply, be sure to respect the voltage and current data specified in the “Power Supplies” section.
- If using a battery in combination or not with a solar panel as a power supply, be sure to use the voltage and current data specified in the “Power supplies” section.
- If a software or hardware failure occurs, consult the Libelium Web Development section.
- Check that the frequency and power of the communication radio modules together with the integrated antennas are allowed in the area where you want to use the device.
- Waspmote is a device to be integrated in a casing so that it is protected from environmental conditions such as light, dust, humidity or sudden changes in temperature. The board supplied “as is” is not recommended for a final installation as the electronic components are open to the air and may be damaged.

1.2. Conditions of use

- Read the “General and Safety Information” section carefully and keep the manual for future consultation.
- Use Waspmote in accordance with the electrical specifications and the environment described in the “Electrical Data” section of this manual.
- Waspmote and its components and modules are supplied as electronic boards to be integrated within a final product. This product must contain an enclosure to protect it from dust, humidity and other environmental interactions. In the event of outside use, this enclosure must be rated at least IP-65.
- Do not place Waspmote in contact with metallic surfaces; they could cause short-circuits which will permanently damage it.

Further information you may need can be found at: http://www.libelium.com/development/waspmote

The “General Conditions of Libelium Sale and Use” document can be found at: http://www.libelium.com/development/waspmote/technical_service

Acknowledgement:
Libelium wants to thank to the Network Planning and Mobile Communications Group from the University of Cantabria on behalf of the Smart Santander project who contributed to the testing and performance improvement of the sensor board.
2. Waspmote Plug & Sense!

The new Waspmote Plug & Sense! line allows you to easily deploy wireless sensor networks in an easy and scalable way ensuring minimum maintenance costs. The new platform consists of a robust waterproof enclosure with specific external sockets to connect the sensors, the solar panel, the antenna and even the USB cable in order to reprogram the node. It has been specially designed to be scalable, easy to deploy and maintain.

**Note:** For a complete reference guide download the “Waspmote Plug & Sense! Technical Guide” in the Development section of the Libelium website.

2.1. Features

- Robust waterproof IP65 enclosure
- Add or change a sensor probe in seconds
- Solar powered with internal and external panel options
- Radios available: Zigbee, 802.15.4, Wifi, 868MHz, 900MHz and 3G/GPRS
- Over the air programming (OTAP) of multiple nodes at once
- Special holders and brackets ready for installation in street lights and building fronts
- Graphical and intuitive programming interface

2.2. Sensor Probes

Sensor probes can be easily attached by just screwing them into the bottom sockets. This allows you to add new sensing capabilities to existing networks just in minutes. In the same way, sensor probes may be easily replaced in order to ensure the lowest maintenance cost of the sensor network.

*Figure 1: Connecting a sensor probe to Waspmote Plug & Sense!*
2.3. Solar Powered

Battery can be recharged using the internal or external solar panel options. The external solar panel is mounted on a 45° holder which ensures the maximum performance of each outdoor installation.

For the internal option, the solar panel is embedded on the front of the enclosure, perfect for use where space is a major challenge.
2.4. Programming the Nodes

Waspmote Plug & Sense! can be reprogrammed in two ways:

The basic programming is done from the USB port. Just connect the USB to the specific external socket and then to the computer to upload the new firmware.
Over the Air Programming is also possible once the node has been installed. With this technique you can reprogram wirelessly one or more Waspmote sensor nodes at the same time by using a laptop and the Waspmote Gateway.

Figure 6: Typical OTAP process

### 2.5. Radio Interfaces

<table>
<thead>
<tr>
<th>Model</th>
<th>Protocol</th>
<th>Frequency</th>
<th>txPower</th>
<th>Sensitivity</th>
<th>Range *</th>
</tr>
</thead>
<tbody>
<tr>
<td>XBee-802.15.4-Pro</td>
<td>802.15.4</td>
<td>2.4GHz</td>
<td>100mW</td>
<td>-100dBm</td>
<td>7000m</td>
</tr>
<tr>
<td>XBee-ZB-Pro</td>
<td>ZigBee-Pro</td>
<td>2.4GHz</td>
<td>50mW</td>
<td>-102dBm</td>
<td>7000m</td>
</tr>
<tr>
<td>XBee-868</td>
<td>RF</td>
<td>868MHz</td>
<td>315mW</td>
<td>-112dBm</td>
<td>12km</td>
</tr>
<tr>
<td>XBee-900</td>
<td>RF</td>
<td>900MHz</td>
<td>50mW</td>
<td>-100dBm</td>
<td>10Km</td>
</tr>
<tr>
<td>Wifi</td>
<td>802.11b/g</td>
<td>2.4GHz</td>
<td>0dBm - 12dBm</td>
<td>-83dBm</td>
<td>50m-500m</td>
</tr>
<tr>
<td>GPRS</td>
<td>-</td>
<td>850MHz/900MHz/1800MHz/1900MHz</td>
<td>2W(Class4) 850MHz/900MHz, 1W(Class1) 1800MHz/1900MHz</td>
<td>-109dBm</td>
<td></td>
</tr>
<tr>
<td>3G/GPRS</td>
<td>-</td>
<td>Tri-Band UMTS 2100/1900/900MHz Quad-Band GSM/EDGE, 850/900/1800/1900 MHz</td>
<td>UMTS 900/1900/2100 0,25W GSM 850MHz/900MHz 2W DCS1800MHz/PCS1900MHz 1W</td>
<td>-106dBm</td>
<td></td>
</tr>
</tbody>
</table>

* Line of sight, Fresnel zone clearance and 5dBi dipole antenna.
2.6. Program in minutes

In order to program the nodes an intuitive graphic interface has been developed. Developers just need to fill a web form in order to obtain the complete source code for the sensor nodes. This means the complete program for an specific application can be generated just in minutes. Check the Code Generator to see how easy it is at:

http://www.libelium.com/development/plug_&_sense/sdk_and_applications/code_generator

![Waspmote - Plug & Sense! - Code Generator](image)

Figure 7: Code Generator

2.7. Data to the Cloud

The Sensor data gathered by the Waspmote Plug & Sense! nodes is sent to the Cloud by Meshlium, the Gateway router specially designed to connect Waspmote sensor networks to the Internet via Ethernet, Wifi and 3G interfaces.

Thanks to Meshlium's new feature, the Sensor Parser, now it is easier to receive any frame, parse it and store the data into a local or external Data Base.

![Meshlium](image)

Figure 8: Meshlium
2.8. Meshlium Storage Options

- Local Data Base
- External Data Base

2.9. Meshlium Connection Options

- ZigBee → Ethernet
- ZigBee → Wifi
- ZigBee → 3G/GPRS
2.10. Models

There are some defined configurations of Waspmote Plug & Sense! depending on which sensors are going to be used. Waspmote Plug & Sense! configurations allow to connect up to six sensor probes at the same time.

Each model takes a different conditioning circuit to enable the sensor integration. For this reason each model allows to connect just its specific sensors.

This section describes each model configuration in detail, showing the sensors which can be used in each case and how to connect them to Waspmote. In many cases, the sensor sockets accept the connection of more than one sensor probe. See the compatibility table for each model configuration to choose the best probe combination for the application.

It is very important to remark that each socket is designed only for one specific sensor, so they are not interchangeable. Always be sure you connected probes in the right socket, otherwise they can be damaged.

Figure 11: Identification of sensor sockets
2.10.1. Smart Parking

Smart Parking allows to detect available parking spots by placing the node under the pavement. It works with a magnetic sensor which detects when a vehicle is present or not. Waspmote Plug & Sense! can act as a repeater for a Smart Parking node.

Figure 12: Smart Parking enclosure

Sensor sockets are no used for this model.
3. Hardware

3.1. General Description

The Waspmote’s Smart Parking Board principle of operation is based on the variation of the magnetic field above the board caused by the chassis of a vehicle placed on the parking lot where the mote has been deployed. This change is detected using a permalloy magnetic field sensor (MFS), a material whose electric resistance varies with the magnetic field through it, measured and processed to provide the system with an answer that can be used to determine the state of the parking spot.

3.2. Specifications

Weight: 20gr  
Dimensions: 73.5 x 51 x 1.3 mm  
Temperature Range: [-20ºC, 65ºC]

3.3. Electrical Characteristics

Board Power Voltages: 5V  
Sensor Power Voltages: 5V  
Maximum admitted current (continuous): 200mA  
Maximum admitted current (peak): 400mA
3.4. Assembling

The MFS includes a module attached perpendicularly to the board, utilized to monitor the vertical axis (Z axis) of the magnetic field. Every board is prepared to operate with one single module, so it is important to maintain the correspondence between them, as it is to maintain the relationship between the board and the Wasmote, prepared for the temperature compensation of a single given board. All the boards, modules and Wasmotes are tagged to allow their identification. In the images below the assembling process is shown.

*Figure 15: Image of the Smart Parking board, the Z axis sensor and the corresponding Wasmote with the labels highlighted*

*Figure 16: Image of the Smart Parking board with the Z axis sensor attached*
Figure 17: Image of the complete system 2. Sensor
4. Sensor

4.1. Specifications

Three axis magnetic field sensor

Maximum Supply Voltage: 12V
Operation Temperature: -40 ~ 125°C
Bridge Resistance: 600 ~ 1200Ω
Typical Output Voltage: 3.5mV/V/gauss
Average consumption: 15mA
Maximum consumption (peak): 500mA

4.2. Measurement Process

The MFS basically consists of a thin film of permalloy whose resistance is a function of the magnetic field through it (intensity and orientation). This film is integrated in a Wheatstone bridge of resistance between 600Ω and 1200Ω (typically 850Ω), thus between the two output terminals of the sensor we have a voltage in the order of the 3.5mV/V/gauss with a supply voltage of 5V.

Since the MFS is a three axis sensor it allows to monitor the magnetic field in any spatial direction. The output of each of the axis is amplified using an instrumentation amplifier and filtered to prevent glitches caused by external magnetic fields. The outputs of the three axis are directly read with the Wasmote’s microprocessor’s analog-to-digital converter in the analog input pins ANALOG1, ANALOG2 and ANALOG5.

To read the outputs it is enough with executing the commands readParking or readParkingSetReset from the board’s library, in which the analog voltage in the input of the analog-to-digital converter is captured. In the second of these functions a small current pulse (500mA of peak, one microsecond wide) is applied to an internal circuit before reading the sensor so the molecules of the permalloy film are realigned with the magnetic field, eliminating non-desired effects owed to the hysteresis of the material.

The board also incorporates a temperature sensor with the purpose of compensating the effects that this parameter has on the sensor. The output of the sensor at pin ANALOG7 can be read using the function readTemperature, assigning its output to an integer variable. Regarding the temperature compensation, Libelium facilitates the sensors calibrated for an operating range between 0º and 45º, with the parameters necessary to calculate the reference in function of the new temperature stored in the EEPROM memory of the mote.

Below, the resulting graphs of two different applications are shown. In figure 16 the output of one axis sampled once every 10ms appears, with one car entering and leaving the spot twice. In figure 17 the outputs of the three axis sampled every 10 seconds are shown. In this case we can see four different vehicles entering and leaving the lot.
Figure 18: Image of one of the axis of the MFS sampled every 10ms

Figure 19: Image of the outputs of the three axis of the MFS sampled every 10 seconds

More information about the functions of the library, as long as an example code for reading the sensors, can be found in section "API" of this guide.
5. Board configuration and programming

5.1. Application considerations

5.1.1. Deployment of the motes

The optimum deployment point will be the one where the probability of detection is maximum, which means minimizing the probabilities of false detection (caused by other vehicles or objects near the lot under control) and false rejection (owed to a not high enough variation in the magnetic field above the mote with a vehicle parked in the spot).

This optimum deployment spot will depend on the kind of parking lot that we are going to monitor. In the case of parallel parking lots the mote should be deployed below one of the car sides, as shown in figure 20, while for perpendicular parking spots the most adequate place will be the one nearest to the center of the motor or the backside of the vehicle (see figure 21).

![Diagram of the deployment points of the motes for parallel parking lots](image-url)
Other consideration to be taken into account in mote deployment is the communication between this one and the gateway or router that will receive the data and process or redirect it. This is a very variable issue that will have to be analyzed independently for each scenario.
5.1.2. Interference of other vehicles

As pointed in section “Deployment of the motes”, the presence of other vehicles in contiguous spots or near places may influence in the detection, modifying the detection threshold. This influence earns special importance in perpendicular lots, where the distance between the mote and the contiguous vehicles is shortest, and in the non-delimited parallel lots. The best way to avoid this disturbance is to take into account the state of the near spots in the detection decision when the variation in the magnetic field is very close to the detection threshold.

5.1.3. Variation of the reference value

Owing to the variation of the environmental conditions (such as temperature, presence of external fields or small changes in the magnetic field of the earth) the reference value from which the state of the lot is determined doesn’t remain constant, but shows a small oscillation that must be eliminated to maintain the reliability of the system. In the case of the small variations owed to the hysteresis of the permalloy under parasitic magnetic fields, the sensor is endowed with a readjustment system that permits to realign the molecules of the permalloy film with the present magnetic field through a small current pulse. In the case of the error for variations in the temperature of the sensor, Libelium supplies the sensors calibrated for an operation range between 0º and 45º approximately, so this alteration can be compensated through a degree two approximation that updates the reference value in function of the measured temperature in the moment of determining the magnetic field.

5.1.4. Determination of the state of the spot

The influence of external factors mentioned before make more difficult to make a decision about the state of the spot in the mote itself, lacking the information about the environmental conditions, such as the presence of vehicles in contiguous lots, which may lead to errors in the estimation. Owing to this, it may be advisable to carry out the calculation of the state of the spot on the routers or gateways which collect the information from all the motes at a zone. This entails an increase in the radio-frequency transmissions from the sensor motes, which involves a decrease of the battery’s life. Given the wide casuistry of this kind of applications, a deep study of the conditions of the location will be necessary to assure this procedure is necessary.

5.1.5. Network structure

There are two possible Network topologies depending on the frequency and protocol used:

- 2.4GHz - Multihop ZigBee/Digimesh
- 868/900MHz Direct Access

a) 2.4GHz - Multihop ZigBee/Digimesh Network

With this frequency band we need to use WaspMote nodes acting as repeaters in the nearby street lights. Once they get the information they send using multihop topology to Meshlium, the Gateway of the network.

In order to communicate at this frequency band the high power module version (100mW) is recommended.

- 1st Step (Figure 24): The nodes send the information to the repeaters.
- 2nd Step (Figure 25): The repeaters perform multihop routing and send the information to Meshlium.
- 3rd Step (Figure 26): Meshlium sends the information to the Internet using the Wifi or GPRS radio.
Figure 24: 2.4 GHz - Multihop ZigBee / Digimesh Network - The motes send the information to the repeaters

Figure 25: 2.4 GHz - Multihop ZigBee / Digimesh Network - The repeaters send the data using a multihop topology
Figure 26: 868 / 900 MHz Direct Access Network - The Gateway sends the data to the Internet

Figure 27: Meshlium: the ZigBee Internet Gateway

b) 868/900MHz - Direct Access Network

With this frequency band we can send directly the data from Wasp mote to the Gateway located in the same street.

- 1st Step (Figure 28): The nodes send the information to Meshlium, the Gateway of the network.
- 2nd Step (Figure 29): Meshlium sends the information to the Internet using the Wifi or GPRS radio.
Figure 28: 868 / 900 MHz Direct Access Network - The nodes send directly the data to the Gateway

Figure 29: 868 / 900 MHz Direct Access Network - The Gateway sends the data to the Internet


5.2. Installation of the mote

Parking detection applications require a casing with more restrictive characteristics than other wireless sensor networks applications, owing to the need of lasting for a long time buried in asphalt, where they must resist the entry of water, owing to the possibility of puddles of water formed above it, and high pressures on it caused by vehicles parked or driven upon.

Libelium includes a PVC casing compliant with the IP-67 and IK-10 standards that may be installed in any environment of this characteristics.

Besides, Libelium provides a special non-conductive gel to avoid any damage caused by corrosion, humidity and bad environmental conditions. This gel is made of a two component silicone which becomes solid when these two components are mixed.

Next to these lines, a few recommendations are given in order to ensure right installation procedures, avoiding bad environmental conditions which can affect the Wasp mote enclosure and the electronic equipment inside. Try to follow next steps carefully to achieve a right installation.

**Important:** Before installation, make sure enough tests have been performed in order to achieve a 100% functional code and that all the necessary information related to the mote, such as identification numbers or XBee MAC addresses, has been compiled and stored, since once the mote is installed, the access to it will be very limited. Be sure to add in your code a delay (or the possibility to make calibration using an external command) before calibration processes in order to carry out the case installation without affecting the sensor calibration (see section "Example code" for a complete example of calibration). Owing to this, it is highly recommended to carry out the gel preparation in the deployment location to minimize the battery consumption before operation.

Moreover, Gel manipulation is simple and must be done just before placing the parking case in its final location.

The necessary elements for one complete parking node are:

1. 1 x Wasp mote + Smart Parking board + Communication module
2. 1 x non rechargeable battery
3. 1 x PVC casing
4. Gel packs: component A bottles + component B bottles (see note below) *
5. 2 x pink filling foam

![Figure 30: Necessary elements for a parking node](image)

---

v4.1
**Note:** Each enclosure needs 650 ml of mixed gel to cover all electronic parts sensitive to humidity, that means 325 ml of component A and 325 ml of component B. However, gel is sold in packs of 500 ml (250 ml of component A + 250 ml of component B) so please check if you have enough gel for all of your parking motes. Next table shows necessary gel depending how many motes are you planning to install.

<table>
<thead>
<tr>
<th>Number of Parking motes</th>
<th>Necessary packs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>20</td>
<td>26</td>
</tr>
</tbody>
</table>

Once you have checked all necessary elements, next steps must be followed.

**Step 1: Making a hole into pavement.**

Use a hole saw or other dedicated tool to drill a hole into the pavement bigger enough for the PVC case. Try to perforate it deep enough to bury the mote completely without rising over the asphalt surface but trying to keep the top of the case as up as possible in order to have the best RF signal possible.

![Figure 31: Making a hole into pavement](image1.png)

![Figure 32: Final pavement hole](image2.png)
**Step 2: Put Waspmote inside PVC enclosure**

When Waspmote is placed into the parking enclosure, battery should be placed in the dedicated hole in the pink filling foam, taking care that the mote rests in horizontal position and with the antenna in vertical position.

**IMPORTANT:** Remember to **stick the filling foam** to the enclosure. Otherwise it will float when adding the protective gel.

*Figure 33: Diagram of dimensions of the case*

*Figure 34: Preparing the sticking foam*
Step 3: Mix the gel

Gel is prepared by mixing the two components (A and B) in a 1:1 proportion in a vessel. Use a graduated recipient like shown in figures below to ensure you add the exactly required quantities. For one parking mote 650 ml of mixed gel are needed, therefore you should add 325 ml of component A and 325 ml of component B. Stir the mixture for a while ensuring the components are completely mixed, since it will not become solid otherwise. Once an homogeneous and without bubbles medley is got, pour carefully into the casing.
Step 4: Turn Waspmote ON

Ensure that Waspmote has the desired code and **that it is switched ON** before pouring the gel.

Step 5: Fill enclosure with gel

The gel should fill all space of the enclosure till Waspmote and the Smart Parking board are completely covered. Pay special attention till cover the Smart Parking Board sensor. This process will ensure protection against water, moisture and bad environmental conditions (see figures below to ensure you add the right quantity of gel). Be sure to include the pink filling foam to ensure that the provided gel is enough to cover all the electronic parts. Also be sure pink foam is firmly stuck to the enclosure, otherwise it could drift when gel is added, which would move Waspmote to a bad position.

Moreover, it is highly recommended to carry out the gel preparation in the deployment location to minimize the battery consumption before operation.
Step 6: Close PVC enclosure

After the gel is poured, close the PVC enclosure firmly.

Step 7: Place enclosure into pavement hole

Be sure that the enclosure remains under pavement line but no more than one centimeter. Following images show the mote before and after the installation.
Step 8: Fixing enclosure to pavement

Now, the PVC enclosure has to be fixed to the pavement using Epoxy resin (commonly used for filling road fissures). This resin should fill all the space between enclosure and pavement and also should cover completely the top of the enclosure, ensuring protection against water, bad environmental conditions and vandalism.

Figure 43: Fixing enclosure to pavement

Figure 44: Epoxy filling. (Good installation)

Figure 45: No filling. (Bad installation)
**Step 9: Recommended Parking node calibration**

Once the node is placed into the pavement, it needs to be calibrated while the parking spot is empty. Use a laptop with a gateway connected to communicate with the parking mote and start calibration process.

To communicate with the node, a serial monitor should be used: Hyperterminal in Windows, Realterm in Mac or Cutecom in Linux are some examples. In this guide, Waspmote IDE serial monitor will be used. To use it, just open the IDE, select serial port where the Waspmote Gateway is connected and open the serial monitor.

Uploading a code into Waspmote like the provided in section “Example code”, next communication sequence would be carried out for node calibration.

1. After the node is turned on, it sends to the gateway a command asking for calibration and it enters in a waiting loop. See figure 47.

2. Once the node is installed and the spot is empty, a command is sent from the serial monitor which controls the gateway to start calibration process in this node (this command will depend on the used XBee module and its configuration, and also how data reception is done in the code. For example code provided on section “Example code”, any data packet coming from a network node is valid to start the node calibration). See figure 48

3. The node starts calibration process and it answers when calibration is done. Answer is shown in serial monitor. See figure 49.

4. Once the node has been calibrated, it sends periodically the normal operation commands. See figure 50.

Next four figures are screenshots which match with each described step.

---

**Figure 46: Node calibration with an empty spot**

To communicate with the node, a serial monitor should be used: Hyperterminal in Windows, Realterm in Mac or Cutecom in Linux are some examples. In this guide, Waspmote IDE serial monitor will be used. To use it, just open the IDE, select serial port where the Waspmote Gateway is connected and open the serial monitor.

Uploading a code into Waspmote like the provided in section “Example code”, next communication sequence would be carried out for node calibration.

1. After the node is turned on, it sends to the gateway a command asking for calibration and it enters in a waiting loop. See figure 47.

2. Once the node is installed and the spot is empty, a command is sent from the serial monitor which controls the gateway to start calibration process in this node (this command will depend on the used XBee module and its configuration, and also how data reception is done in the code. For example code provided on section “Example code”, any data packet coming from a network node is valid to start the node calibration). See figure 48

3. The node starts calibration process and it answers when calibration is done. Answer is shown in serial monitor. See figure 49.

4. Once the node has been calibrated, it sends periodically the normal operation commands. See figure 50.

Next four figures are screenshots which match with each described step.
It is recommended to test deeply this part before final installation, to ensure node remains well calibrated and ready for a good operation. Ask in Libelium technical forum in our website if you have questions about how to use parking API functions.
5.3. Powering the mote

Since the mote in a parking monitoring application will be buried underground with no access to a system to recharge the battery, it is recommended to use a battery with the highest capacity available to extend as much as possible the life of the device. Under these conditions, Libelium recommends the utilization of one of the two Lithium non-rechargeable batteries described in next section (to obtain more information about them please contact Libelium’s sales department at commercial@libelium.com).

5.3.1. Battery options

Libelium provides two different Lithium non-rechargeable batteries for powering Waspmote in parking applications:

- Battery of 26Ah
- Battery of 52Ah

Main differences between them are battery capacity and self discharge current. Next pictures show both possibilities.

Figure 51: Image of the 26Ah battery

Figure 52: Image of the 52Ah battery
5.3.2. Estimated lifetime

This section describes estimated lifetime for a parking application in which the state of the spot is measured every five minutes and transmitted via XBee Digimesh 2.4GHz and 868MHz.

Consumption table:

<table>
<thead>
<tr>
<th></th>
<th>XBee Digimesh 2.4GHz</th>
<th>XBee RF 868MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep Time and consumption</td>
<td>298.1 seconds, 64μA</td>
<td>297.3 seconds, 64μA</td>
</tr>
<tr>
<td>Measurement time and consumption</td>
<td>380 miliseconds, 33mA</td>
<td>380 miliseconds, 33mA</td>
</tr>
<tr>
<td>Transmission time and consumption</td>
<td>1.52 seconds, 72mA</td>
<td>2.32 seconds, 90.5mA</td>
</tr>
<tr>
<td>Complete cycle average consumption</td>
<td>0.470mA</td>
<td>0.805mA</td>
</tr>
</tbody>
</table>

Battery lifetime estimation table:

<table>
<thead>
<tr>
<th></th>
<th>XBee Digimesh 2.4GHz</th>
<th>XBee RF 868MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>26Ah battery approximate self discharge current</td>
<td>88μA</td>
<td>88μA</td>
</tr>
<tr>
<td>52Ah battery approximate self discharge current</td>
<td>176μA</td>
<td>176μA</td>
</tr>
<tr>
<td>26Ah battery estimated lifetime</td>
<td>5.3 years</td>
<td>3.3 years</td>
</tr>
<tr>
<td>52Ah battery estimated lifetime</td>
<td>9.1 years</td>
<td>6.0 years</td>
</tr>
</tbody>
</table>

5.4. API

The functions to handle all the features of the board, included in the WaspSensorParking API library, are detailed below. The functions necessary to control the state of the spot in all kind of situations have been implemented, but take into account that not all of them will be needed in all the situations:

Library values:

- **Compensation coefficients:** this library includes those coefficients, in floating point format, necessary to estimate the compensation for the temperature variation using the function calculateReference. These coefficients are unique for each sensor board, are stored in the Wasp mote’s EEPROM memory and can be read using the function loadReference. The list of the names of the coefficients is shown below:
  - coefX2
  - coefY2
  - coefZ2
  - coefX
  - coefY
  - coefZ
  - constX
  - constY
  - constZ

- **Sensors’ initial values:** after carrying out the calibration of the initial state of the parking lot the values read from the magnetic field and temperature sensors are stored in four integer variables that will be used to calculate the reference of the magnetic field for every measurement:
  - initialX
  - initialY
  - initialZ
  - initialT
- **Sensors’ read values**: the values of the sensors read in the execution of functions readParking or readParkingSetReset are stored in three integer variables that will be used to estimate the state of the spot:
  - valueX
  - valueY
  - valueZ

- **Sensors’ reference values**: after executing the calculateReference function the result is stored in three integer variables that contain the reference for each of the three sensors on the board.
  - referenceX
  - referenceY
  - referenceZ

- **Sensors’ compensation indexes**: from the calibration temperature and the compensation coefficients three indexes, one for each sensor, that will be used in the calculation of the sensor’s reference are extracted and stored as floating point variables.
  - indexX
  - indexY
  - indexZ

- **THRESHOLD**: defined as a constant instead of a variable, it defines the threshold used in function estimateState to determine the state of the parking lot. In case its value is to be modified, it must be done in the file WaspSensorParking.h of the library.

**Reserved Memory Zone:**

A zone of the EEPROM memory of the microcontroller between addresses 186 and 222 has been reserved to store the coefficients used in temperature compensation. **Under no circumstances should this zone of the memory be overwritten**, as long as it could lead to an irreparable error in most applications.

When using the Smart Parking Sensor Board on Wasp mote PRO, remember it is mandatory to include the SensorParking library by introducing the next line at the beginning of the code:

```
#include <WaspSensorParking.h>
```

**Library functions:**

- **SensorParking.ON()**
  
  Turns on the sensor board by activating the 5V supply line.

- **SensorParking.OFF()**
  
  Turns off the sensor board by cutting the 5V supply line.

- **SensorParking.setBoardMode(MODE)**
  
  The setBoardMode function of this board activates or deactivates completely the supply voltage of the board assigning the values SENS_ON or SENS_OFF to the parameter MODE. In the case of the parking detection board only the 5V power supply is used, and all the electronics contained in it, included the sensors, are turned on or off using this function.

- **SensorParking.calibration()**
  
  This function calculates the initial values of magnetic field for each of the three sensors in the empty parking lot, which are stored in variables initialX, initialY and initialZ, and the compensation indexes of each sensor from the temperature measured in that moment, staying those values stored in variables indexX, indexY and indexZ.
• SensorParking.loadReference()

In those applications that require temperature compensation this function loads the compensation coefficients stored in the EEPROM memory, characteristic of a specific parking board, in the variables mentioned before.

• SensorParking.calculateReference(TEMPERATURE)

The calculateReference function is used in applications that require temperature compensation to update the reference value in function of the current temperature of the mote. It utilizes the values stored in variables initialX, initialY, initialZ, indexX, indexY, indexZ, the compensation coefficients and the temperature, introduced in Celsius degree as an integer variable through the parameter TEMPERATURE, returning the new values in variables referenceX, referenceY and referenceZ.

• SensorParking.readParking()

The readParking function is used to read the output of each of the sensors at the input of the microcontroller's analog-to-digital converter. When it is executed the voltage values at the analog inputs of the three sensors (ANALOG1, ANALOG2 and ANALOG5) are captured and stored in variables valueX, valueY and valueZ respectively.

• SensorParking.readParkingSetReset()

The function behaves in a very similar way to the former one, with the only difference that this one applies a current pulse in the internal circuit of every sensor to realign the molecules of the permalloy film, as indicated in section "Measurement Process".

• SensorParking.readTemperature()

The temperature sensor attached to the Smart Parking board can be read through the readTemperature function that, with the purpose of obtaining a more accurate measurement, averages eight captures of the voltage value at the analog pin corresponding to the sensor output (ANALOG7). The value is returned in Celsius degree as an integer variable.

• SensorParking.estimateState()

This function uses the values read from the sensors and stored in variables valueX, valueY and valueZ and the reference values calculated with the function to estimate the state of the spot, returning a boolean variable ('1' in the case of an occupied spot, '0' in the case of an empty one). This calculation consists of a comparison between the difference between the value read from each sensor and its respective reference and a threshold stored in constant THRESHOLD. In the case any of these differences surpasses given threshold, the lot will be considered occupied.
5.5. Example code

We provide below with a code suitable for a simple parking detection application. Take into account that depending on the conditions of the application a more complex calculation in the decision of the state of the lot may be necessary.

```c
/* ------------Smart Parking Sensor Board example---------------
www.Libelium.com
*/

// Inclusion of the Frame library
#include <WaspFrame.h>

// Inclusion of the Smart Parking Sensor Board v20 library
#include <WaspSensorParking.h>

// Inclusion of the XBee Digimesh library
#include <WaspXBeeDM.h>

// Sleep time between measures
#define SLEEP_TIME "00:00:00:05"

// Variable to store temperature
int temperature;

// Variable to store status
boolean status;

// Gateway address
#define ADDRESS "0013A2004070DA3E"

packetXBee* packet;
int error;

void setup()
{
    
    // ---SCALE--- STEP 1 ---SCALE---
    // Turn On XBee and wait calibration command //
    
    // Turn ON the XBee Digimesh module
    xbeeDM.ON();
    delay(1000);
    packet=(packetXBee*) calloc(1,sizeof(packetXBee)); // memory allocation
    packet->mode=UNICAST; // set Unicast mode
    
    // Send a broadcast message of “awaiting calibration” command
    xbeeDM.setDestinationParams(packet, ADDRESS, "SPOT1 Awaiting calibration command\n\r", MAC_TYPE);
    do
    {
        error = xbeeDM.sendXBee(packet);
    }while( error );
    free(packet);
    packet = NULL;

    // Note: Code hangs here until an answer is received. The user can
    // add here his own command and also his own waiting period, etc.
    Utils.setMuxXBEE();
    delay(100);
    while( xbeeDM.available() == 0 )
```


```c
{
   // Turn off the XBee Digimesh module
   xbeeDM.OFF();
   delay(100);

   //-------------------------------------------------------------
   // STEP 2
   // Calibration process.
   //-------------------------------------------------------------

   // Once “start calibration” command is received, calibration starts.
   SensorParking.loadReference();
   SensorParking.ON();
   delay(100);
   SensorParking.calibration();
   SensorParking.OFF();

   // Send a broadcast message with calibrated values
   xbeeDM.ON();
   delay(1000);
   packet=(packetXBee*) calloc(1,sizeof(packetXBee)); // memory allocation
   packet->mode=UNICAST; // set Broadcast mode

   // Create new frame (ASCII) with the sensor data
   frame=createFrame(ASCII,"Waspmote Pro");
   frame.addSensor(SENSOR_STR, "SPOT1 initial values");
   initialZ);

   xbeeDM.setDestinationParams( packet, ADDRESS, frame.buffer, frame.length);
   do
   {
      error = xbeeDM.sendXBee(packet);
   }while( error );
   xbeeDM.OFF();

   free(packet);
   packet = NULL;
}

void loop()
{

   //-------------------------------------------------------------
   // STEP 3
   // Normal operation: Measurement
   //-------------------------------------------------------------

   // Turn on the board
   SensorParking.ON();
   delay(100);

   // Read the sensors
   SensorParking.readParkingSetReset();
   temperature = SensorParking.readTemperature();

   // Estimate the state and turn off the board
   SensorParking.calculateReference(temperature);
   status = SensorParking.estimateState();
   SensorParking.OFF();
```
STEP 4

Data Transmission and sleep

Create new frame (ASCII) with the sensor data
frame.createFrame(ASCII,"Waspmote Pro");
frame.addSensor(SENSOR_MF, SensorParking.valueX, SensorParking.valueY, SensorParking.valueZ);
frame.addSensor(SENSOR_PS, status);

// Turn On XBee and send State
xbeeDM.ON();
delay(1000);
packet=(packetXBee*) calloc(1,sizeof(packetXBee)); // memory allocation
packet->mode=UNICAST; // set Broadcast mode
xbeeDM.setDestinationParams( packet, ADDRESS, frame.buffer, frame.length);
do
{  
    error = xbeeDM.sendXBee(packet);
}while( error );
xbeeDM.OFF();
free(packet);
packet = NULL;

// Go to sleep.
RTC.ON();
PWR.deepSleep(SLEEP_TIME,RTC_OFFSET,RTC_ALM1_MODE1,ALL_OFF);

You can also find a simple test code where the data is printed through the USB port here:
http://www.libelium.com/development/waspmote/examples/pa-1-smart-parking-test-example/

The files related to this sensor board are: WaspSensorParking.cpp, WaspSensorParking.h
They can be downloaded from: http://www.libelium.com/development/waspmote/sdk_and_applications
6. Consumption

6.1. Table of consumption

In the case of the Smart Parking board, the power supply is only controlled by the general switch of the 5V source in Waspmote, so none of the components will be active once the board has been disconnected, so the consumption in this situation will be null.

<table>
<thead>
<tr>
<th>Consumption</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Board Off</td>
<td>0μA</td>
</tr>
<tr>
<td>Board On (Average)</td>
<td>25mA</td>
</tr>
<tr>
<td>Board On (Peak)</td>
<td>500mA (1 microsecond)</td>
</tr>
</tbody>
</table>
# API Changelog

<table>
<thead>
<tr>
<th>Function / File</th>
<th>Changelog</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#include</code></td>
<td>Remember to include the WaspSensorParking library in the top of your pde</td>
<td>V0.31 → v1.0</td>
</tr>
<tr>
<td><code>SensorParking.ON()</code></td>
<td>New function to turn on the board</td>
<td>V0.31 → v1.0</td>
</tr>
<tr>
<td><code>SensorParking.OFF()</code></td>
<td>New function to turn off the board</td>
<td>V0.31 → v1.0</td>
</tr>
<tr>
<td><code>PARKING_ADDRESS_COEFX2</code></td>
<td>Though it does not affect regular programming, take into account that the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EEPROM addresses in which the calibration coefficients are stored have</td>
<td></td>
</tr>
<tr>
<td></td>
<td>changed from 256<del>292 to 186</del>222</td>
<td>V0.31 → v1.0</td>
</tr>
<tr>
<td><code>PARKING_ADDRESS_CONSTZ</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
8. Documentation changelog

- Added references to 3G/GPRS Board in section: Radio Interfaces.
9. Maintenance

- In this section, the term "Waspmote" encompasses both the Waspmote device itself as well as its modules and sensor boards.
- Take care with the handling of Waspmote, do not drop it, bang it or move it sharply.
- Avoid putting the devices in areas of high temperatures since the electronic components may be damaged.
- The antennas are lightly threaded to the connector; do not force them as this could damage the connectors.
- Do not use any type of paint for the device, which may damage the functioning of the connections and closure mechanisms.
10. Disposal and recycling

- In this section, the term “Waspmote” encompasses both the Waspmote device itself as well as its modules and sensor boards.
- When Waspmote reaches the end of its useful life, it must be taken to a recycling point for electronic equipment.
- The equipment has to be disposed on a selective waste collection system, different to that of urban solid waste. Please, dispose it properly.
- Your distributor will inform you about the most appropriate and environmentally friendly waste process for the used product and its packaging.