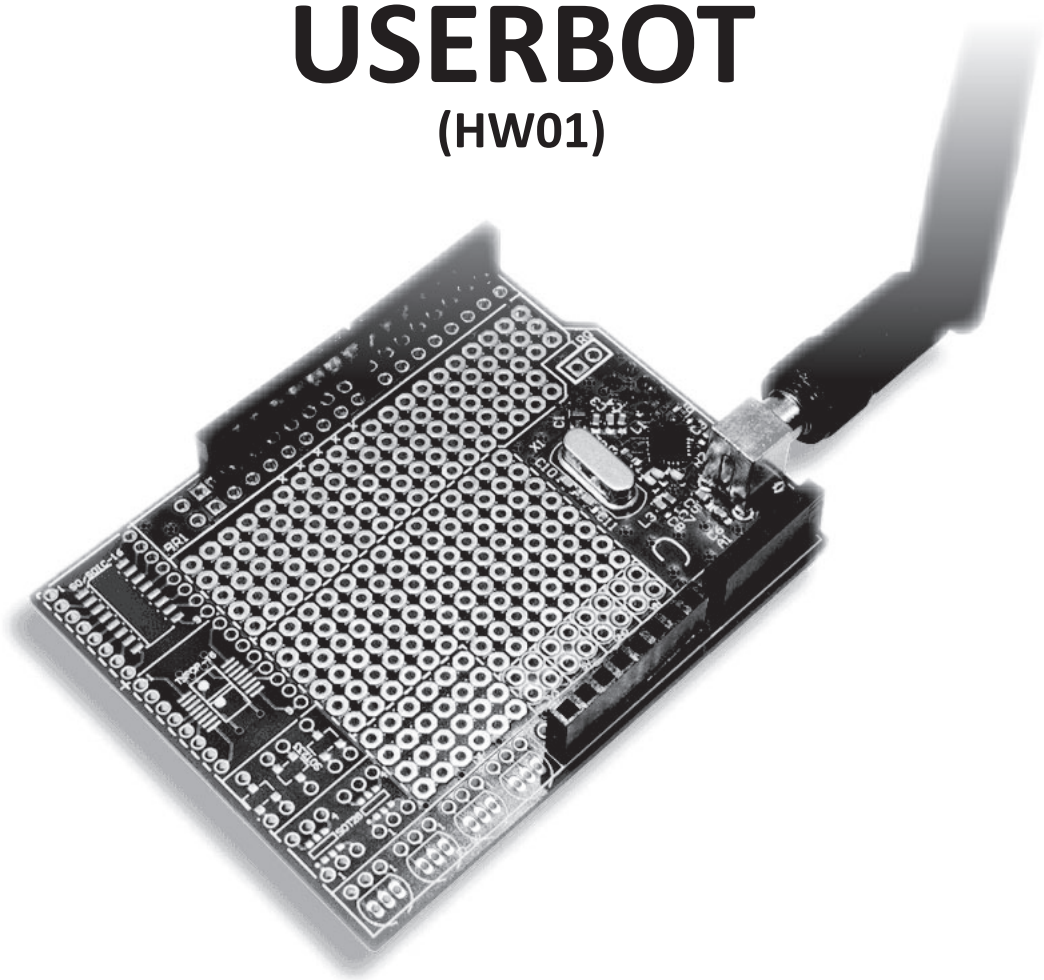
GROLAB®

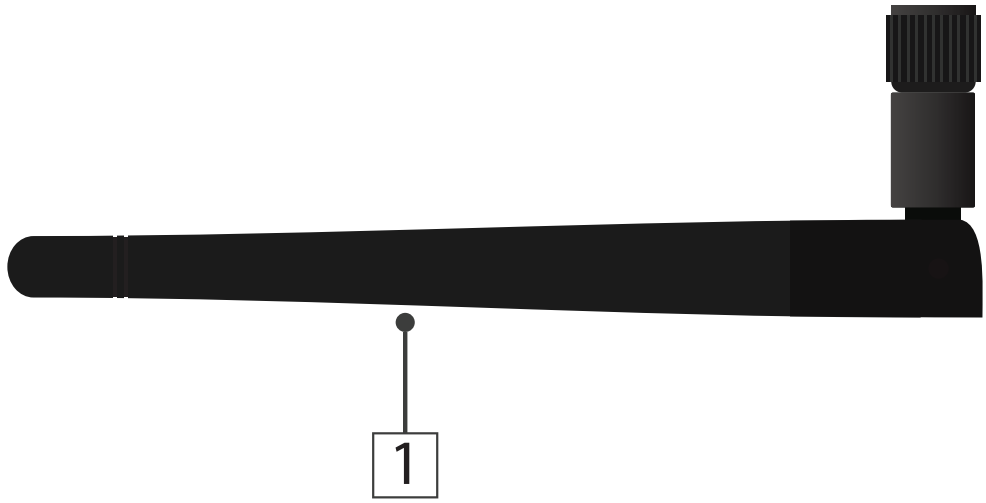
USERBOT
(HW01)

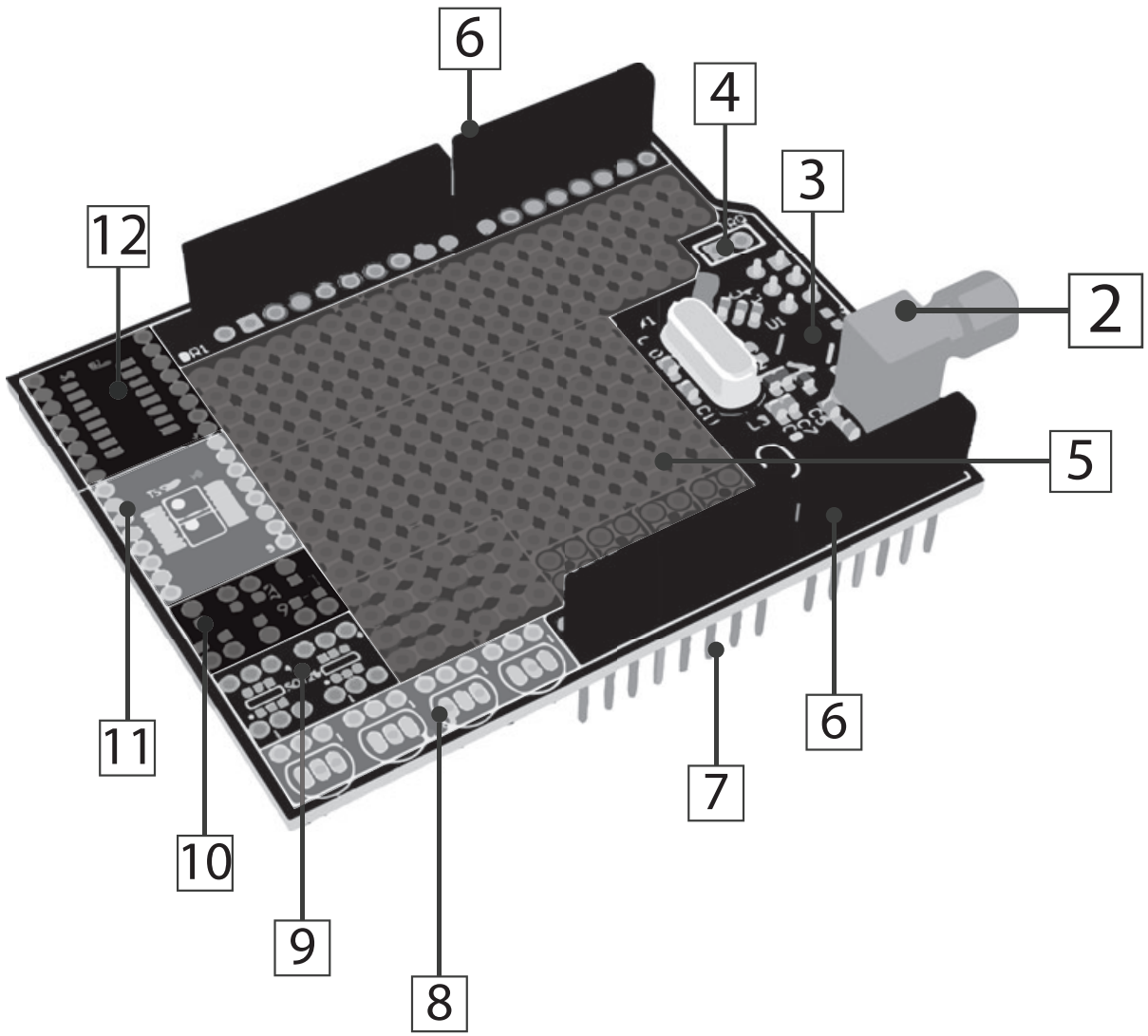


INSTALLATION MANUAL

MANUAL DE INSTALACIÓN
MANUEL D'INSTALLATION
MANUAL DE INSTALAÇÃO
MANUALE DI INSTALLAZIONE

www.opengrow.pt


1


6
4
3
12
2
5
11
10
9
8
7
6

1 RF communication antenna

2 RF antenna outlet

3 RF module

4 RF interrupt pin

5 Universal breakout board

6 Female pin headers

7 Arduino connection pins

8 4 x TO-92

9 2 x SOT26

10 3 x SSOT223

11 2 x TSSOP-8 (=) 1 x TSSOP-16

12 SOIC16

SPECIFICATIONS TABLE

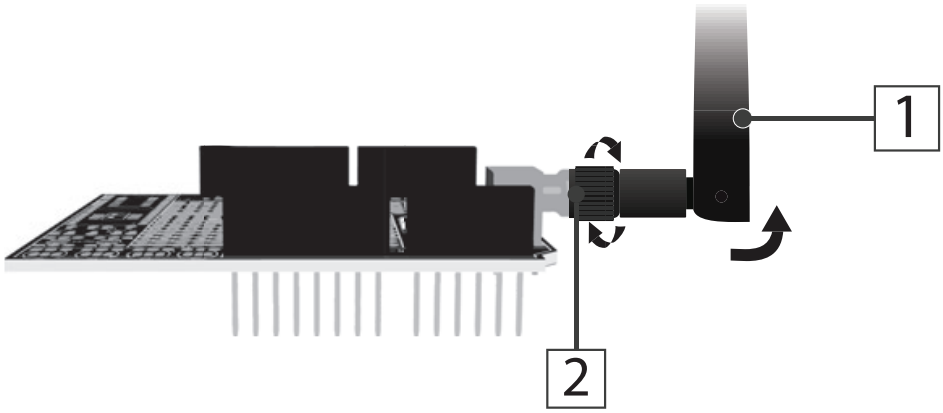
Hardware Version	H01
Dimensions	68.6mm x 53.4mm
Exterior	Material: FR-4 Color: Green
Operating Voltage	+3V3 VDC
Connections	RP-SMA female Female Pin Headers (Arduino connection/extension)
SMD Sockets	1 x SOIC16 2 x TSSOP 8 (=) 1 x TSSOP 16 3 x SSOT223 2 x SOT26
TH Sockets	4 x TO-92
BreakOut Board	DIP/1206/0805/0603
BreakOut Board Spacing	2.54mm - 100mil
Includes	Antenna
Inter-Module Communication	Radio Frequency - 2.4GHz
Warranty	2-years limited hardware warranty

INSTALLATION ①

Base connections

Screw the communication antenna (1) into the module's RF antenna outlet (2).

For correct communication, turn the communications antenna (1) to position it vertically in the direction of the module.



At this point, the UserBot shield is ready to be assembled into the Arduino.

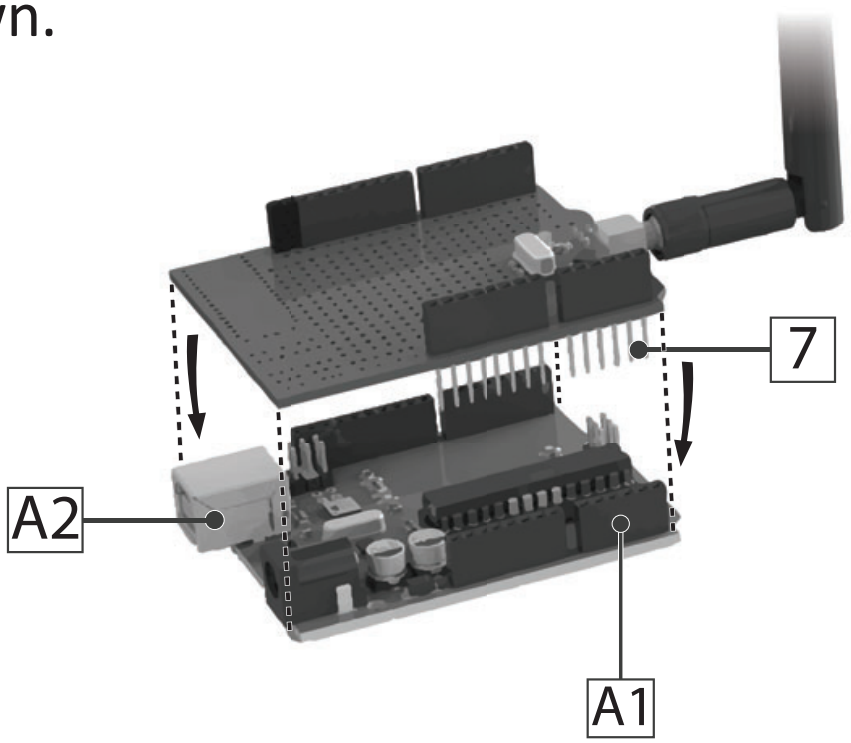
Connection to Arduino

First, check if your Arduino model is compatible with UserBot shield. Compatible models:

- **Arduino UNO;**
- **Arduino Duemilanove;**
- **Arduino Leonardo with headers;**
- **Genuino Zero.**

Before connecting the UserBot shield to the Arduino, make sure the Arduino is correctly assembled and ready to be used (follow the official Arduino instructions manual). Also, make sure your Arduino is not connected to the power supply.

Place the Arduino on a firm surface, far from flood-prone areas. Make sure the Arduino female pin headers (A1) are faced up and the UserBot shield connection pins (7) are faced down.



Insert the UserBot connection pins (7) into the Arduino female header (A1) from top-down as shown in the above picture.

⚠ Avoid the contact between the UserBot shield and the Arduino USB power entry (A2).

Peripherals connection

Since UserBot is an extension for Arduino, it means that any device or sensor that can interface with Arduino is also compatible with UserBot shield.

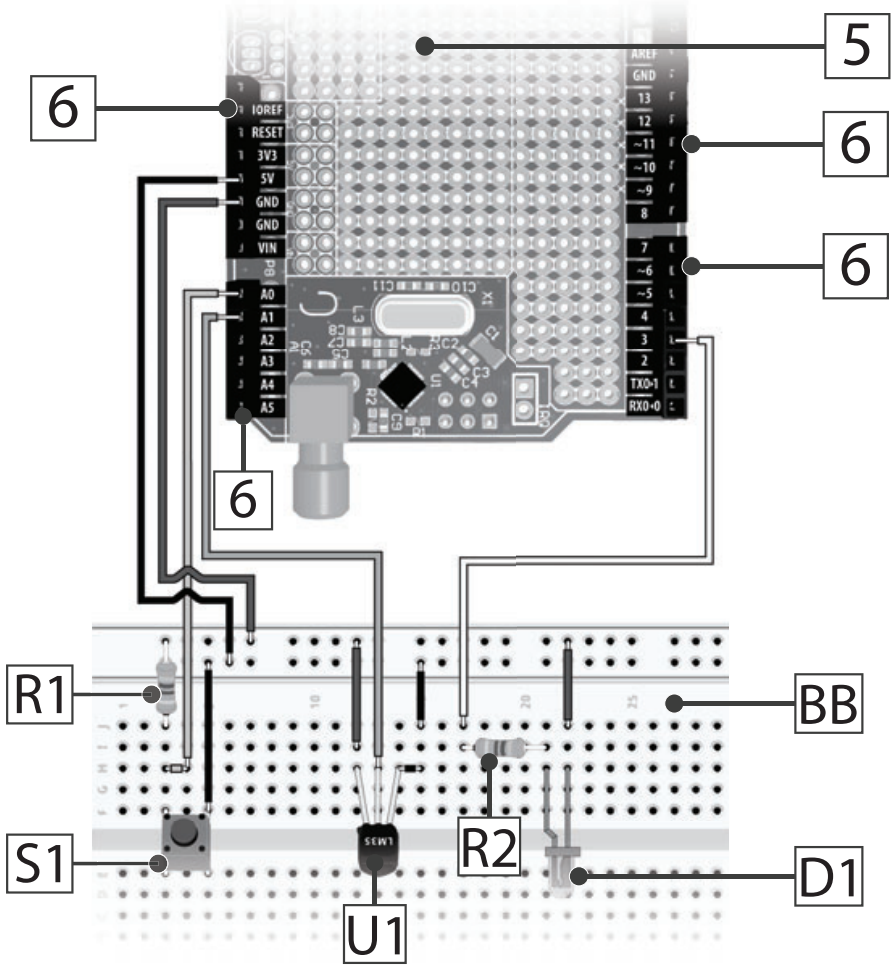
In this way, the list of compatible devices and sensors is extremely long, being impossible to list and explain how to connect all of them.

We will provide two examples: the first one shows how to connect/sample a button and a temperature sensor, while also act on a LED; the second one explains how to act on a DC motor.

⚠ Before assembling any sensor or device, make sure the Arduino is not connected to the power supply. Moreover, it is advisable to carry out the assembly on a firm surface, far from flood-prone areas. Also, use the proper tools for handling electronic components.

Example 1

Connecting one button, one temperature sensor and one LED.



5	Universal breakout board
6	Female pin headers
R1	Pull down resistor 10kΩ
R2	LED resistor 240Ω
S1	Tactile Button SPST
U1	LM35 temperature sensor
D1	LED
BB	External breadboard

The diagram on the previous page shows the connection of some different peripherals to the UserBot shield through its female pin headers (6).

The tactile button SPST (S1) is sampled on Arduino **analog pin A0**; the LM35 temperature sensor (U1) is sampled on Arduino **analog pin A1**; the LED (D1) is controlled through the Arduino **digital pin 3**. To control the 'speed/intensity' of a device, it is required to use a PWM Arduino digital pin (e.g. 3 and 5).

Use the **+5V**, **+3V3** and the **GND pins** on Arduino to power your sensors and/or devices. Please refer to the official Arduino documentation regarding power constraints on these pins.

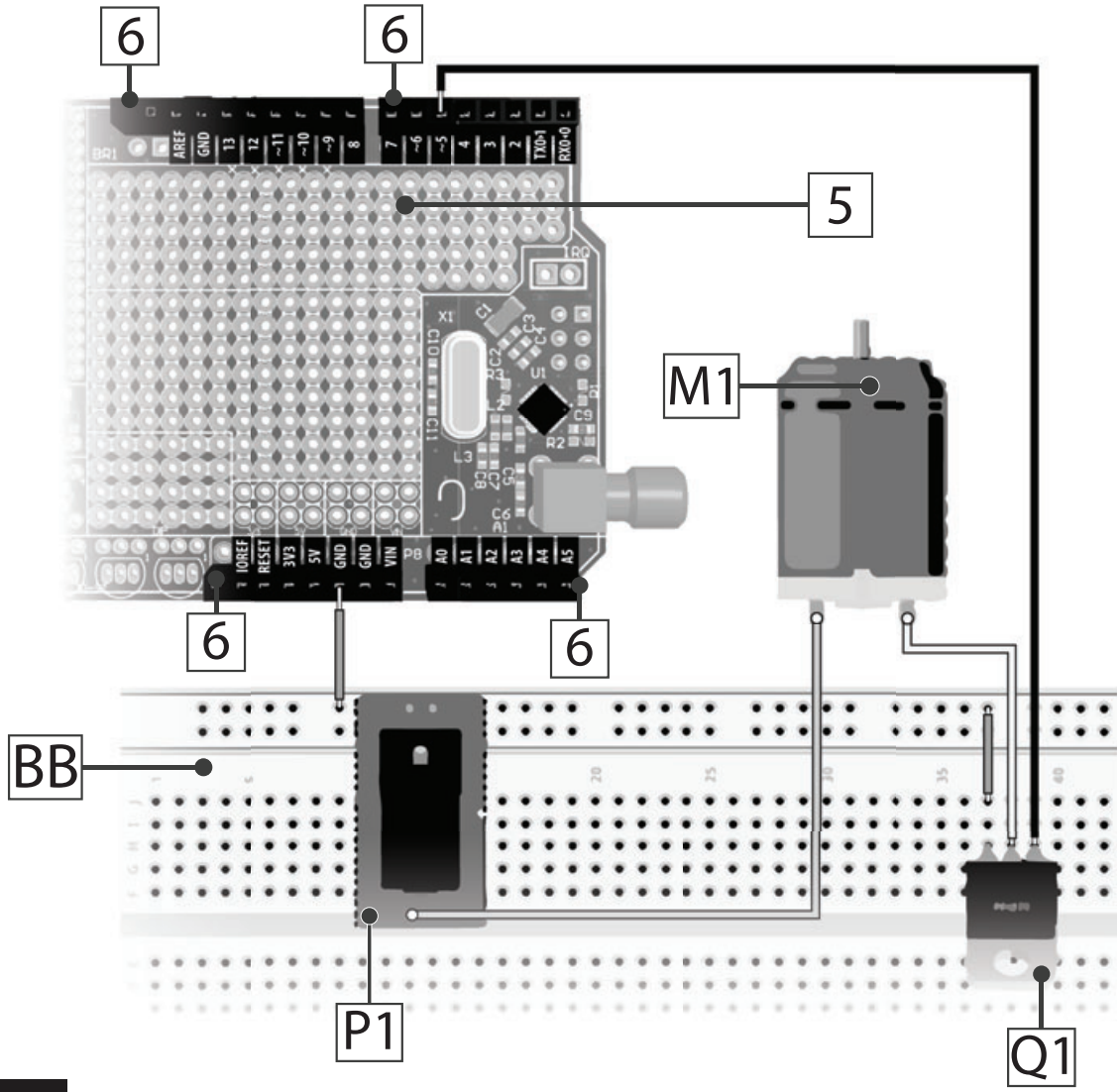
The UserBot shield includes an onboard breadboard (5) that you are free to use. However, for the diagram, we used an external breadboard (BB) for a better understanding.

⚠ It is crucial to pay attention to the polarity of the different components. A wrong connection can damage your components and Arduino/UserBot.

EN ES FR PT IT

Example 2

Connecting a DC motor (peristaltic pump, water pump...).



5 Universal breakout board

6 Female pin headers

M1 DC motor

BB External breadboard

P1 12VDC power entry

Q1 N-type MOSFET (e.g. IRLZ14)

The diagram on the previous page shows an N-type MOSFET (Q1) being used to control the supply voltage to a DC motor (M1).

The DC motor (M1) is powered by an external 12VDC (P1) and has PWM control via the Arduino **digital pin 5**. To control the "speed or intensity" of a device, it is required to use a PWM Arduino digital pin (e.g. 3 and 5).

The 12VDC power entry (P1) and the MOSFET (Q1) also need to be connected to the Arduino **GND pin**.

Use the UserBot female pin headers (6) to connect the 12VDC power entry (P1) and the MOSFET (Q1) to the respective Arduino pins.

The UserBot shield includes an onboard breadboard (5) that you are free to use. However, for the diagram, we used an external breadboard (BB) for a better understanding.

⚠ It is crucial to pay attention to the polarity of the different components. A wrong connection can damage your components and Arduino/UserBot.

Prepare the source code

After assembling all the sensors and devices, it is time to prepare the source code to handle them.

Open Grow provides the base firmware code in an open-source format. It is available to download on the GitHub at the following page:

github.com/OpenWeGrow/UserBot

In this way, the first task is to download the base code. Since there are several ways to do it (e.g. using apps to handle repositories or directly download from the GitHub web page), we will not detail any specific method in this manual. However, if you need assistance, please contact our technical support:

www.opengrow.pt @ support@opengrow.pt

NOTE: The examples provided by this manual are based on the code available on August 14, 2019.

Since the essence of the UserBot shield is to integrate the Arduino with the GroLab system, then to prepare the code we will use the Arduino IDE. If at any moment, you find any issues or obstacles with Arduino IDE, please refer to the Arduino official guides.

After downloading the base code, copy/move the contents of the folders **'libraries'** and **'examples'**, to the respective folders (same names) inside the Arduino installation folder on your PC.

Ensure you have the correct Arduino board selected in the Arduino IDE. For that, open the Arduino IDE and navigate to **Tools → Board** and choose the correct Arduino board based on your physical Arduino.

Next step is to open the code with the Arduino IDE, for that navigate to **File → Examples → OpenGrow → UserBot** and click to open it.

Before proceeding to change the code, we will list the file structure and explain it a little:

→ libraries
→ OpenGrow

• ComsTask.cpp • ComsTask.h	**Communication state machine.**
• CRC16.cpp • CRC16.h	**CRC calculator from Tim W. Shilling.**
• EEPROM_Utils.cpp • EEPROM_Utils.h	**NVM handling functions.**
• GroBot_Variables.h	**UserBot inputs/outputs definitions.**
• nRF24L01.h • RF24.cpp • RF24.h • RF24_config.h	**NRF24L01 driver from J. Coliz <maniacbug@ymail.com> with tweaks by Open Grow.**
• OpenBus.cpp • OpenBus.h	**Communication commands handler.**
• SerialTask.cpp • SerialTask.h	**State machine to handle UART commands.**

→ examples
→ OpenGrow
→ UserBot

• SensorsTask.cpp • SensorsTask.h	**A dedicated state machine for sensors sampling and output control.**
• UserBot.ino	**Arduino (.ino) file and base to run the program. This is the main UserBot configuration file. All of your inputs and outputs should be configured in this file.**

→ examples
→ OpenGrow
→ UserBot-DCMotor

(Same file structure as **examples → OpenGrow → UserBot**.)

Changing the code

Now everything is set, so it is time to start changing the code to match the user needs.

In summary, **for the majority of the cases (if not all)**, the **user only needs to change or duplicate the following files**:

→ examples
  → OpenGrow
    → UserBot
      - SensorsTask.cpp
      - SensorsTask.h
      - UserBot.ino

Since the required code changes are dependent on the assembled electronics by the user and the possible combination of sensors and devices is endless, it is not possible to provide the exact solution.

However, we will show the code required to achieve the two example diagrams previously presented.

EN ES FR PT IT

Example 1

Connecting one button, one temperature sensor, and one LED.

To interface the button, the temperature sensor and the LED, open the **UserBot.ino** (**examples → OpenGrow → UserBot**) and navigate to the ***setup function***. This ***function*** contains the configuration for all the inputs and outputs. You should adjust the code to match the electronics you assembled on the board. For the example diagram, the code should be the following:

INPUTS CONFIGURATION

```
inputs[INPUT_INDEX0].arduinoPin = A0;    //Button Pin
inputs[INPUT_INDEX1].arduinoPin = A1;    //LM35 Pin
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = BUTTON;
inputs[INPUT_INDEX1].type = DIG_TEMPERATURE;
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

OUTPUTS CONFIGURATION

```
outputs[OUTPUT_INDEX0].arduinoPin = 3;  //LED Pin
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = LED;
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

The different types of inputs and outputs are declared in the **GroBot_Variables.h** (**libraries → OpenGrow**) below the comments **'Possible Input Types'** and **'Possible Output Types'**.

The changes in this first file are complete and it is time to open the **SensorsTask.cpp** (**examples → OpenGrow → UserBot**).

NOTE: UserBot shield supports up to 10 inputs and 10 outputs.

After opening the **SensorsTask.cpp**, navigate to the ***SensorsTask function*** and search for the comment that says **'* IO Config *'.** Below the comment, you will find an example pin configuration that you should adjust to match your components. For the example diagram the code should be the following:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ***********************
    /*Here you need to set your used pins as inputs or outputs*
    pinMode(inputs[INPUT_INDEX0].arduinoPin, INPUT);
    pinMode(inputs[INPUT_INDEX1].arduinoPin, INPUT);

    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Now, navigate to the ***GoSensorsTask function*** and look at the ***switch***. Inside the ***case GET_IOS***, you should add the code to sample your inputs and update its respective values in the ***inputs array***. For example:

```
if (digitalRead(inputs[INPUT_INDEX0].arduinoPin))
  inputs[INPUT_INDEX0].value = 255;
else
  inputs[INPUT_INDEX0].value = 0;
```

Or if it's an output, inside the ***case ACT_ON_IOS***, you should add the code to act on the outputs. For example:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

You can add more ***cases*** based on your needs. For the example schematics, the code inside the ***switch*** should be the following:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = GET_TEMP;
  break;
case GET_TEMP:
  //Sampling Temperature Sensor
  time = getAnalogRead(inputs[INPUT_INDEX1].arduinoPin);
  inputs[INPUT_INDEX1].value += calcTemp(time);
  inputs[INPUT_INDEX1].value = inputs[INPUT_INDEX1].value/2;
  snsState = GET_IOS;
  break;
case GET_IOS:
  //Polling button
  if(digitalRead(inputs[INPUT_INDEX0].arduinoPin))
    inputs[INPUT_INDEX0].value = 255;
  else
    inputs[INPUT_INDEX0].value = 0x00;

  snsState= ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on LED pin
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1))
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = GET_TEMP;
  break;
```

Note that we added some more code related to the output to support the variable **'speed'** (e.g to control the LED's intensity) and other functionalities already implemented in the GroLab System (such as the security cooldown time for sensitive devices).

NOTE: The inputs and outputs arrays, stores the sample values from sensors, as well as the state of the outputs. Those arrays are crucial to exchange info with the GroLab system.

If you need to change the ***cases*** from the ***switch*** it is also required to change the ***enum*** that contains those ***cases***. The ***enum*** is declared in the **SensorsTask.h** (**examples → OpenGrow → UserBot**).

At this point, the code is prepared to handle the physical peripherals from the diagram shown in example 1. Now it is time to compile and upload the code to your Arduino.

If you need assistance to compile and upload the code, please refer to the official Arduino guides or contact the Arduino support center.

After uploading the code to your Arduino, there is still one step required to make it fully operational and ready to communicate with GroLab system: the **factory settings need to be set** (serial number and communication channel). Please refer to the instructions on **pages 25 and 26**.

Example 2

Connecting a DC motor (peristaltic pump, water pump...).

To interface the DC motor, open the **UserBot.ino** (**examples → OpenGrow → UserBot-DCMotor**) and navigate to the ***setup function***. This ***function*** contains the configuration for all the inputs and outputs. You should adjust the code to match the electronics you assembled on the board. For the example diagram the code should be the following:

INPUTS CONFIGURATION

```
inputs[INPUT_INDEX0].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX1].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX1].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

OUTPUTS CONFIGURATION

```
outputs[OUTPUT_INDEX0].arduinoPin = 5;  //DC Motor as Perista
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = PERISTALTIC_PUMP;  //DC Motor a
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

The different types of inputs and outputs are declared in the **GroBot_Variables.h** (**libraries → OpenGrow**) below the comments **'Possible Input Types'** and **'Possible Output Types'**.

The changes in this first file are complete and it is time to open the **SensorsTask.cpp** (**examples → OpenGrow → UserBot-DCMotor**).

NOTE: UserBot shield supports up to 10 inputs and 10 outputs.

After opening the **SensorsTask.cpp**, navigate to the ***SensorsTask function*** and search for the comment that says **'* IO Config *'.** Below the comment, you will find an example pin configuration that you should adjust to match your components. For the example diagram the code should be the following:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ********************
    /*Here you need to set your used pins as inputs or outpu
    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Now, navigate to the ***GoSensorsTask function*** and look at the ***switch***. Inside the ***case ACT_ON_IOS***, you should add the code to act on the outputs. For example:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

NOTE: The output array stores the state of the outputs. This array is crucial to exchange info with the GroLab system.

You can add more ***cases*** based on your needs. For the example schematics, the code inside the ***switch*** should be the following:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on DC Motor
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1) )
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = INIT_SENSORS;
  break;
```

Note that we added some more code related to the output to support the variable **'speed'** and other functionalities already implemented in the GroLab system (such as the security cooldown time for sensitive devices).

If you need to change the ***cases*** from the ***switch***, it is also required to change the ***enum*** that contains those ***cases***. The ***enum*** is declared in the **SensorsTask.h** (**examples → OpenGrow → UserBot-DCMotor**).

At this point, the code is prepared to handle the physical peripherals from the diagram shown in example 2. Now it is time to compile and upload the code to your Arduino.

If you need assistance to compile and upload the code, please refer to the official Arduino guides or contact the Arduino support center.

Set the factory settings

After uploading the code to your Arduino, there is still one step required to make it fully operational and ready to communicate with GroLab system: the **factory settings need to be set** (serial number and communication channel).

Make sure your Arduino is connected through USB to the PC, and that the serial port is not currently in use. Open the Arduino IDE and select the respective serial port. Access the **'Serial Monitor'** and set the **baud rate to 230400**. Send the **'R'** command with the option **'No line ending'**. Wait a moment until you see **RST** printed in the 'Serial Monitor'. Reboot your Arduino (power interruption or reset button if available).

Open the 'Serial Monitor' again, then send the **'M'** command with the option **'No line ending'**. Then send the communication channel that is currently being used by your GroNode* with the option **'Both NL & CR'**.

***There are 5 communication channels (1 to 5), the default channel is 5.**

Send the **‘S’** command with the option **‘No line ending’**, type the desired serial number (sending **digit by digit**) and when you reach the last number send it with the option **‘Both NL & CR’.** Note the serial number should be composed of 10 numbers and start by the number ‘5’, for example, ‘5123456789’.*

To check if everything is correct, send the **‘D’** command with the option **‘No line ending’**. You should see something similar to the following image through the ‘Serial Monitor’:

```
UserBot! Ready
N:       New Module
S:       5123456789
MID:     2
SAdd:    FF
FW:      1.1.0.8
```

After completing all the previous steps, Arduino is already set up and ready to communicate with GroNode through the UserBot shield.

***The serial number ‘5999999999’ is reserved as the default one and should not be used.**

COMMUNICATIONS ②

Connection to GroNode

Make sure the GroNode is correctly installed and accessible through the GroLab Software. If not, follow the instructions provided inside the GroNode's manual.

After completing all the UserBot shield installation steps, open the GroLab Software, establish a connection with the GroNode and then, access **Modules** through the main menu.

Usually, 2 or 3 minutes are enough for GroNode to detect a newly installed module. Once it is detected, it will automatically appear in the **Modules** menu under the respective module type section.

You can check the list of modules of the same type on the right side of this menu. Note that one GroNode supports a maximum of 4 modules of each type.

Problems or losses in communication

GroNode was designed to communicate with the other modules through radio frequency signals. The **range of action** is **25 meters (82 feet) indoors** and **100 meters (328 feet) outdoors**, depending on space conditions.

Access the **Modules** menu on the GroLab Software and check if your modules are available. If they aren't available or they are experiencing communication losses, you may have exceeded the distance between your modules and GroNode.

Inside the **Modules** menu, you can find a wireless signal icon that indicates if the selected module is connected to the GroNode.

⚠ **NOTICE: Some load-bearing walls and electronic devices may interfere with the signal.**

If the problem persists, perform a test run by placing the module next to the GroNode and check the module's communication status through the software. If communication has been successfully re-established, repeat this operation at different distances until you detect the maximum communication distance.

If the module is not found at all, try to cycle the GroNode through all the available communication channels until you find your modules, allowing 3 minutes between channel changes.

To change the communication channel, open the GroLab Software and connect to the GroNode. Inside the **Main Menu (Configurations)**, go to **Settings (GroNode) → General Settings Tab**. Click the button in the lower right corner to enable editing and change the communication channel to the desired channel. To apply the changes, click the green button in the lower right corner.

If after following the previous steps your modules still don't show up on the GroLab Software, please contact our technical support.

Firmware Updates

Keep in mind that some software updates may require that you update the firmware of your GroLab modules.

All updates provide crucial improvements that ensure optimal system performance.

However, as UserBot firmware is compiled and applied by the user, it is required to keep monitoring the UserBot code repository on GitHub for any code updates. In case some code updates are submitted, the user must apply those changes to their code, re-compile and re-upload it to the Arduino. It is also necessary to reapply the factory settings.

SAFETY INSTRUCTIONS ④

The following general safety guidelines are provided to help ensure your own personal safety and protect your device from potential damage.

•Do not attempt to service the device and never disassemble the device. For some devices with a user-replaceable battery, please read and follow the instructions provided by the installation manual.
•Keep your device away from radiators and heat sources.
•Keep your device away from extremely hot or cold temperatures to ensure that it is used within the specified operating range.
•Do not spill food or liquids on your device.
•Before you clean your device, disconnect it from the electrical outlet. Clean your device with a dry soft cloth. Do not use liquids.
•If your device does not operate normally - in particular, if there are any unusual sounds or smells coming from it - unplug it immediately and contact an authorized dealer or the Open Grow support center.
•To help avoid the potential hazard of electric shock, do not connect or disconnect any cables, or perform maintenance or reconfiguration of your device during an electrical storm.
•Check the voltage rating before you connect the device to an electrical outlet to ensure that the required voltage and frequency match the available power source.
•Also, ensure that your GroLab modules and attached devices are electrically rated to operate with the AC power available in your location.
•Do not plug the power cables into an electrical outlet if the power cables are damaged.
•To prevent electric shock, plug the power cables into properly grounded electrical outlets.
•If you use an extension power cable, ensure that the total ampere rating of the devices plugged into the extension power cable does not exceed the ampere rating of the extension cable.

WARRANTY ⑤

Open Grow, LDA, guarantees the purchaser that the product is free from material and/or manufacturing defects. Open Grow, LDA, liability is limited to the repair or replacement of any defective parts. Do not ship directly to Open Grow, LDA, without first consulting with us to verify the procedure that should be followed.

Contact our technical services at **support@opengrow.pt**.

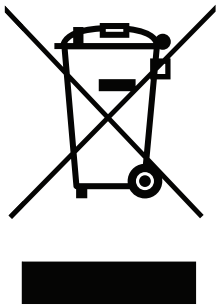
All Open Grow, LDA, products have a 2-year guarantee except for consumables (sensors and/or actuators of any type) and under normal use.

A guarantee claim is non-transferable and only the original purchaser can submit one. To enforce the guarantee the customer must always provide the purchase invoice.

GUARANTEE DISCLAIMER:

Application of the guarantee is excluded should the breakdown of the defective part or parts be a result of the product's inadequate and/or negligent use. Understood as inadequate and/or negligent use is any use other than the one for which the product is meant and/or that is recommended in the instructions manual, not executing the maintenance operations recommended in the instructions manual, carrying out operations that are different from those mentioned and that compromise the quality of the product, modifications that are not performed by authorized repairers and/or with non-original or non-approved parts.

COMPLIANCE ⑥



This symbol on the product or packaging means that according to local laws and regulations this product should not be disposed of in the household waste but sent for recycling. Please take it to a collection point designated by your local authorities once it has reached the end of its life, some will accept products for free. By recycling the product and its packaging in this manner you help to conserve the environment and protect human health.

This symbol on the product or packaging means that this product is compliant with RoHS Regulations of the European Parliament and Council Directive on the Restrictions of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment (2011/65/EU).

This symbol on the product or packaging means that this product is in compliance with the following directives and regulations:

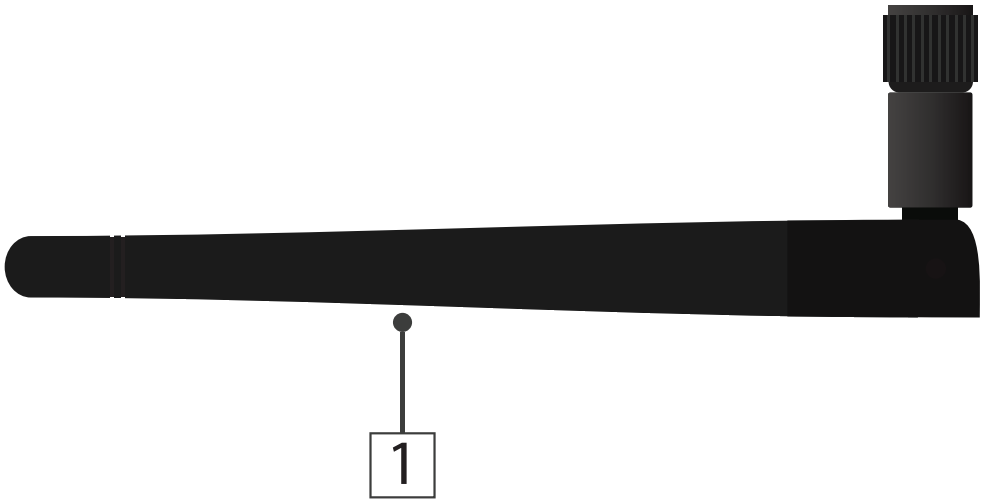
- (2014/53/EU) Radio Equipment Directive.
- (2011/65/EU) RoHS Directive.
- (2014/35/EU) Low Voltage Directive.
- (2014/30/EU) EMC.

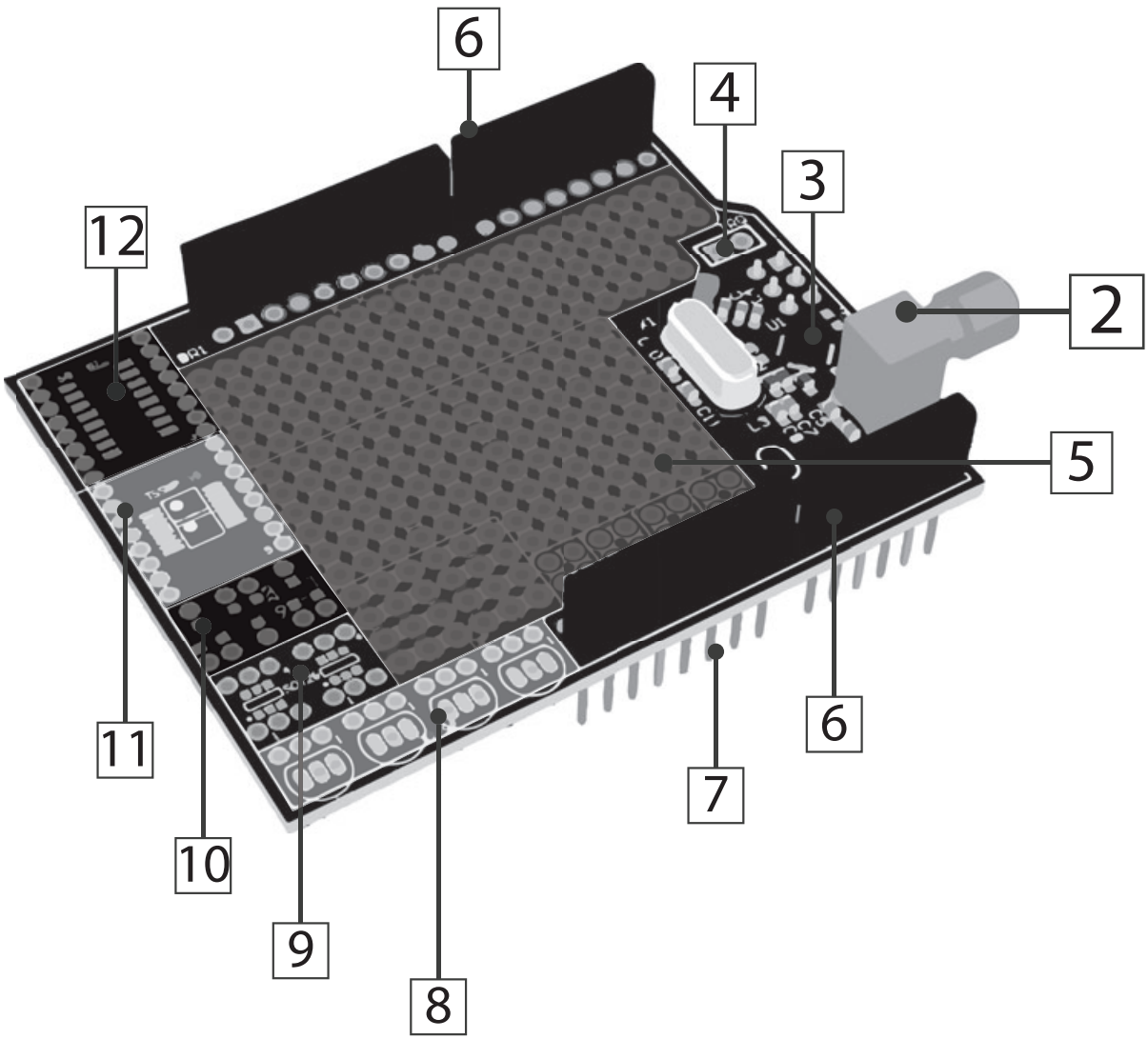
	Frequency Band(s)	Max. Output Power (EIRP)
2.4 G	2.4 - 2.4835 GHz	100 mW

Open Grow, LDA, reserves the right to update and/or modify the content of its products at any time without prior warning. Check out our Terms & Conditions at www.opengrow.pt.

OPENGROW.

NOTES


1


6
4
3
12
2
5
11
10
9
8
7
6

1 Antena de comunicaciones RF

2 Toma para antena de comunicaciones RF

3 Módulo RF

4 Pin de interrupción de RF

5 Placa de desarrollo universal

6 Conectores rápidos hembra

7 Pines de conexión Arduino

8 4 x TO-92

9 2 x SOT26

10 3 x SSOT223

11 2 x TSSOP-8 (=) 1 x TSSOP-16

12 SOIC16

TABLA DE ESPECIFICACIONES

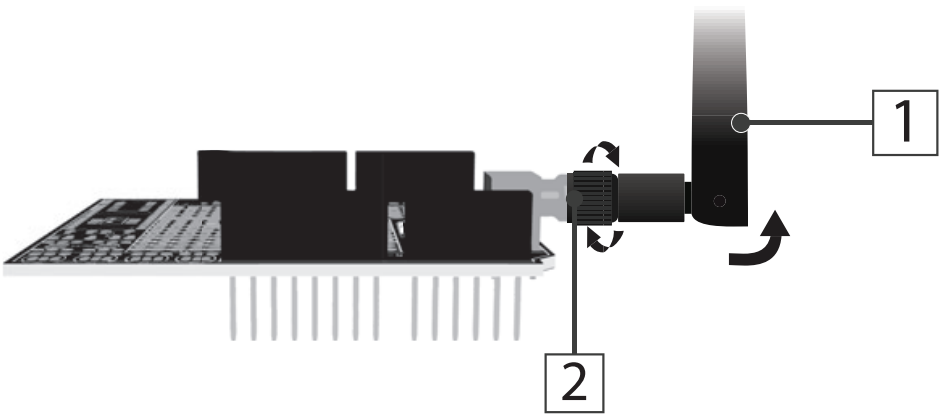
Versión de Hardware	H01
Dimensiones	68.6mm x 53.4mm
Exterior	Material: FR-4 Color: Verde
Tensión de Funcion.	+3V3 VDC
Conexiones	RP-SMA hembra Conectores rápidos hembra (conexión/extensión Arduino)
Sockets SMD	1 x SOIC16 2 x TSSOP 8 (=) 1 x TSSOP 16 3 x SSOT223 2 x SOT26
Sockets TH	4 x TO-92
Placa de Desarrollo	DIP/1206/0805/0603
Espaciado de la Placa de Desarrollo	2.54mm - 100mil
Incluye	Antena
Comunicación entre Módulos	Radio Frecuencia - 2.4GHz
Garantía	Garantía de hardware limitada a 2 años

INSTALACIÓN ①

Conexiones básicas

Atornille la antena de comunicaciones (1) al módulo a través de la entrada roscada (2).

Para una mejor comunicación, se recomienda que la antena (1) esté perpendicular al módulo y sin obstrucciones.



Después de colocar la antena, el UserBot *shield* está listo para ser montado en Arduino.

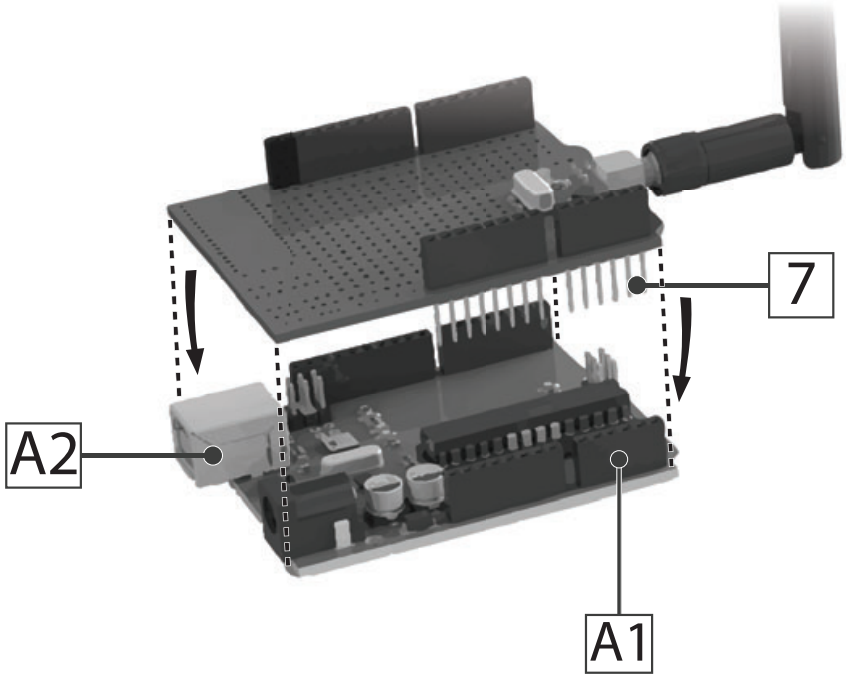
Conexión con Arduino

Antes de continuar con el montaje, verifique la compatibilidad de su Arduino con UserBot. Lista de modelos compatibles:

- **Arduino UNO;**
- **Arduino Duemilanove;**
- **Arduino Leonardo con conectores hembra;**
- **Genuino Zero.**

Antes de montar el UserBot *shield* en Arduino, confirme que el Arduino está correctamente ensamblado y listo para usar (siga el manual de instrucciones oficial de Arduino). También asegúrese de que el Arduino no esté conectado a la fuente de alimentación.

Coloque el Arduino sobre una superficie firme, lejos de áreas propensas a inundaciones. Asegúrese de que los conectores rápidos hembra de Arduino (A1) estén hacia arriba y los pines del conector UserBot (7) hacia abajo.



Inserte los pines del conector UserBot (7) de arriba hacia abajo en los conectores hembra Arduino (A1) como se muestra en la figura anterior.

⚠ **Evite el contacto entre el UserBot *shield* y la entrada de alimentación USB de Arduino (A2).**

Conexión de periféricos

Como UserBot es una extensión para Arduino, significa que cualquier dispositivo o sensor que pueda interactuar con Arduino también es compatible con el UserBot *shield*.

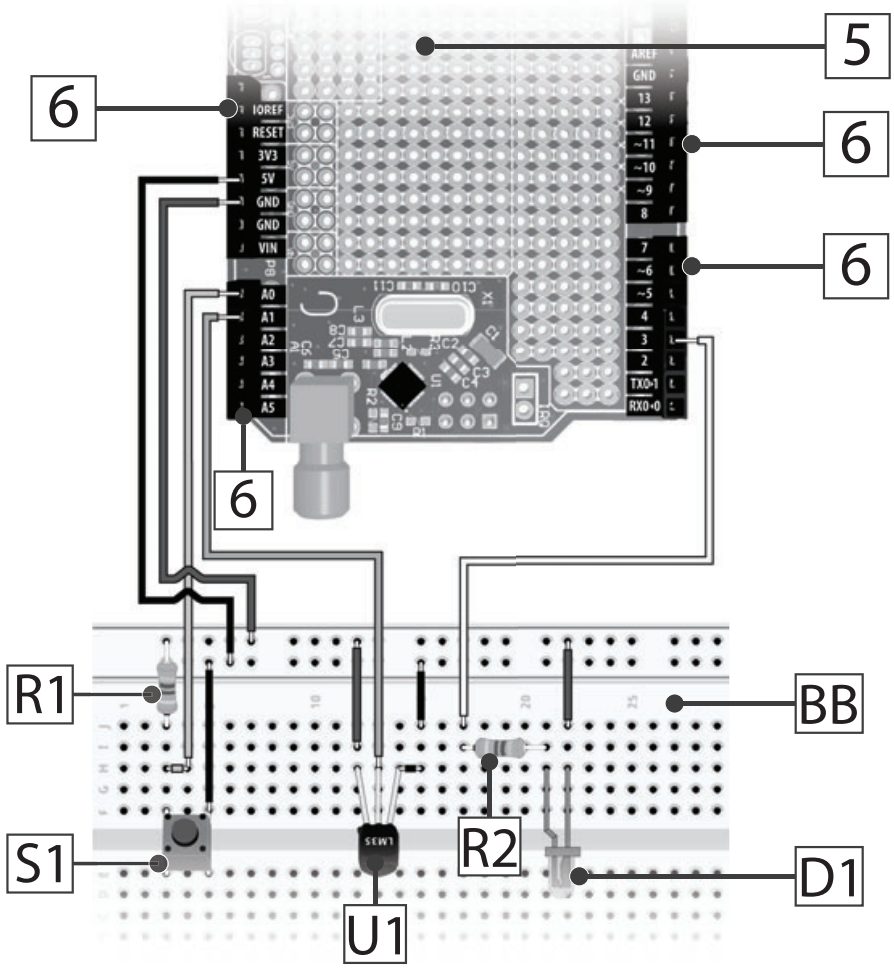
Por lo tanto, la lista de dispositivos y sensores compatibles es extremadamente larga y es imposible enumerarlos y explicar su conexión.

Aún así, cubriremos dos ejemplos: el primero muestra cómo encender/muestrear una perilla y un sensor de temperatura, así como actuar sobre un LED; El segundo explica cómo operar en un motor de corriente continua (CC).

⚠ Antes de conectar cualquier sensor o dispositivo, asegúrese de que Arduino no esté conectado a la fuente de alimentación. Además, es aconsejable montar sobre una superficie firme, lejos de áreas propensas a inundaciones. Use herramientas apropiadas para manejar componentes electrónicos.

Ejemplo 1

Conexión de un botón, un sensor de temperatura y un LED.



5	Placa de desarrollo universal
6	Conectores rápidos hembra
R1	Resistencia de 10kΩ (pull down)
R2	Resistencia de 240Ω para LED
S1	Botón táctil SPST
U1	Sensor de temperatura LM35
D1	LED
BB	Placa de desarrollo externa

El diagrama que se muestra en la página anterior muestra la conexión de algunos componentes del UserBot *shield* a través de los conectores rápidos hembra (6).

El botón táctil SPST (S1) se muestrea en el **pin analógico A0** de Arduino; el sensor de temperatura LM35 (U1) se muestrea en el **pin analógico A1** de Arduino; el LED (D1) se controla a través del **pin digital 3** de Arduino. Para controlar la 'velocidad/intensidad' de un dispositivo, debe usar un pin digital Arduino PWM (por ejemplo, los pines 3 y 5).

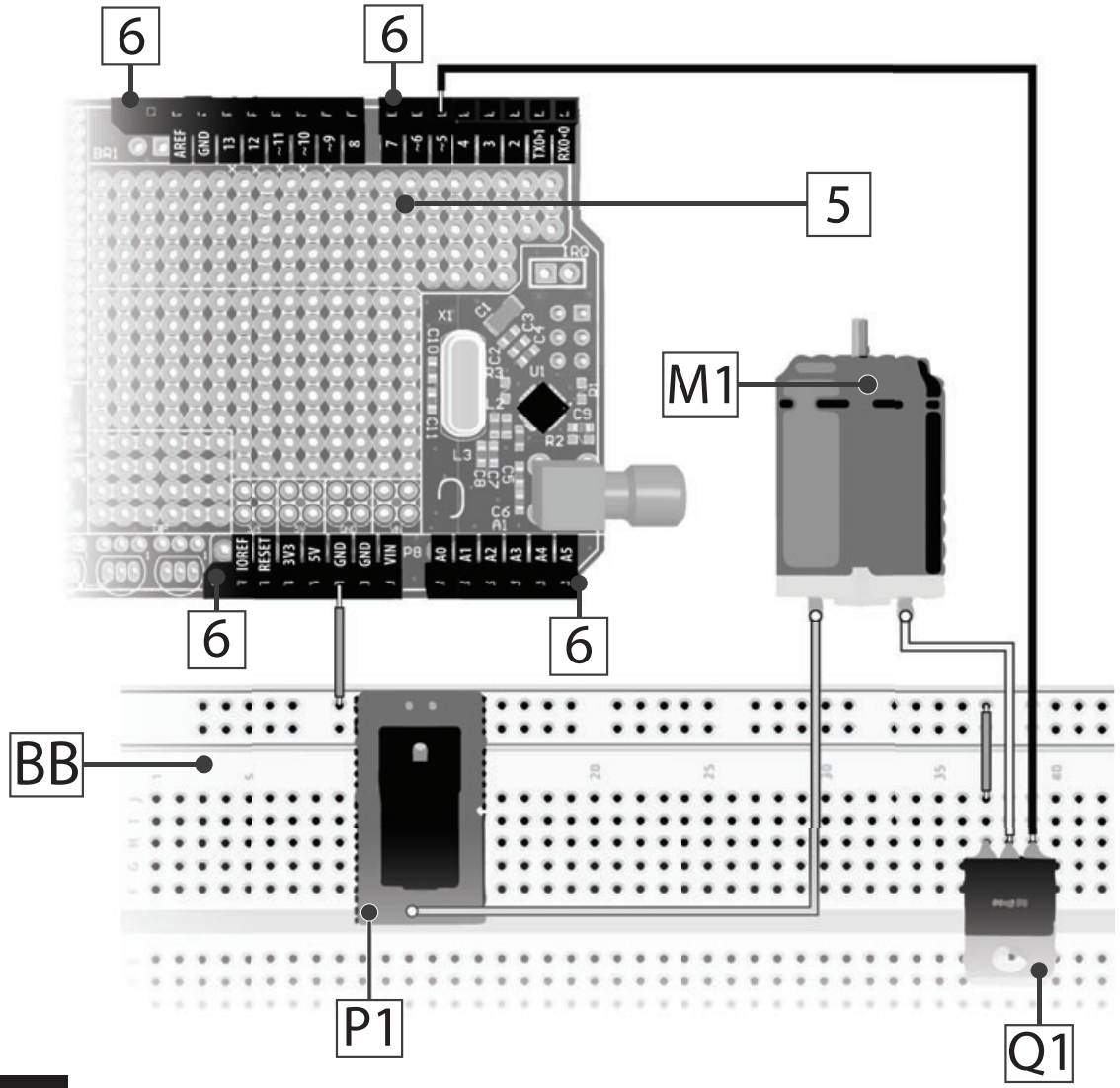
Use los **pines +5V**, **+3V3** y **GND** de Arduino, para alimentar los sensores y/o dispositivos. Consulte las restricciones de alimentación de estos pines en la documentación oficial de Arduino.

El UserBot *shield* ofrece una placa de desarollo incorporada (5) que puede usar libremente. Sin embargo, para el diagrama, utilizamos una placa externa (BB) para una mejor comprensión.

⚠ Es crucial prestar atención a la polaridad de los diferentes componentes. Una conexión incorrecta puede dañar los componentes y/o Arduino/UserBot.

Ejemplo 2

Conexión de un motor CC (bomba peristáltica, bomba de agua...).



5 Placa de desarrollo universal

6 Conectores rápidos hembra

M1 Motor de corriente continua (CC)

BB Placa de desarollo externa

P1 Entrada de energía de 12VDC

Q1 MOSFET tipo N (p. ej. IRLZ14)

El diagrama mostrado en la página anterior, muestra el uso de un MOSFET de tipo N (Q1) para controlar la tensión de alimentación de un motor de corriente continua (M1).

El motor de CC (M1) está alimentado por 12VDC externo (P1) y tiene control PWM a través del **pin digital 5** de Arduino. Para controlar la "velocidad/intensidad" de un dispositivo, debe usar un pin digital Arduino PWM (por ejemplo, los pines 3 y 5).

La entrada de alimentación de 12VDC (P1) y el MOSFET (Q1) también deben conectarse al **pin GND** de Arduino.

Utilice los conectores rápidos hembra del UserBot (6) para conectar la entrada de alimentación de 12VDC (P1) y el MOSFET (Q1) a los respectivos pines de Arduino .

El UserBot *shield* ofrece una placa de desarollo incorporada (5) que puede usar libremente. Sin embargo, para el diagrama, utilizamos una placa externa (BB) para una mejor comprensión.

⚠ **Es crucial prestar atención a la polaridad de los diferentes componentes. Una conexión incorrecta puede dañar los componentes y/o Arduino/UserBot.**

Preparar el código fuente

Después de montar los sensores y dispositivos, es hora de preparar el código fuente para controlarlos.

Open Grow proporciona el código de *firmware* base en un formato de código abierto. El código está disponible para descargar desde la siguiente página de GitHub:

github.com/OpenWeGrow/UserBot

De esta manera, la primera tarea es descargar el código. Dado que hay varias formas de hacer esto (por ejemplo, a través de aplicaciones de administración de repositorio o descargando directamente desde la página *web* de GitHub), no detallaremos ningún método específico. Sin embargo, si necesita ayuda, póngase en contacto con nuestro soporte técnico:

www.opengrow.pt @ support@opengrow.pt

NOTA: Los ejemplos proporcionados en este manual se basan en el código disponible el 14 de agosto de 2019.

Dado que la esencia del UserBot *shield* es integrar el Arduino con el sistema GroLab, para preparar el código, utilizaremos el IDE de Arduino. Si en algún momento encuentra algún problema u obstáculo con el IDE de Arduino, consulte la documentación oficial de Arduino.

Después de descargar el código base, debe copiar/mover el contenido de las carpetas **'libraries'** y **'examples'**, a sus respectivas carpetas (mismo nombre) dentro de la carpeta de instalación de Arduino en su PC.

Asegúrese de que la tarjeta Arduino seleccionada en el IDE sea la correcta. Para hacer esto, abra el IDE de Arduino y navegue a **Herramientas → Placa** y seleccione el tablero Arduino correcto según su Arduino físico.

El siguiente paso es abrir el código con el IDE de Arduino navegando a **Archivo → Ejemplos → OpenGrow → UserBot** y haciendo clic para abrirlo.

Antes de continuar con el cambio de código, enumeraremos la estructura del archivo y explicaremos un poco:

→ libraries
→ OpenGrow

• ComsTask.cpp • ComsTask.h	**Máquina de estados para gestionar tareas de comunicación.**
• CRC16.cpp • CRC16.h	**Calculadora CRC por Tim W. Shilling.**
• EEPROM_Utils.cpp • EEPROM_Utils.h	**Funciones de gestión de NVM.**
• GroBot_Variables.h	**Definiciones de entrada/salida de UserBot.**
• nRF24L01.h • RF24.cpp • RF24.h • RF24_config.h	***Driver* NRF24L01, por J. Coliz <maniacbug@y-mail.com> con ajustes realizados por Open Grow.**
• OpenBus.cpp • OpenBus.h	**Gestor de comandos de comunicación.**
• SerialTask.cpp • SerialTask.h	**Máquina de estados para gestionar comandos UART.**

→ examples
→ OpenGrow
→ UserBot

• SensorsTask.cpp • SensorsTask.h	**Una máquina de estados dedicada para muestreo de sensores y control de salida.**
• UserBot.ino	**Archivo Arduino (.ino) y el código base para ejecutar el programa. Este es el archivo de configuración principal para UserBot. Todas las entradas y salidas deben configurarse en este archivo.**

→ examples
→ OpenGrow
→ UserBot-DCMotor

(Misma estructura de archivo que: **examples → OpenGrow → UserBot**.)

Cambiar el código

Una vez que se ha preparado el código base, debe cambiarse para que coincida con los componentes ensamblados.

En resumen, **para la mayoría de los casos (si no todos), solo necesita cambiar o duplicar los siguientes archivos:**

→ examples
  → OpenGrow
    → UserBot
      - SensorsTask.cpp
      - SensorsTask.h
      - UserBot.ino

Como los cambios de código requeridos dependen de los componentes ensamblados por el usuario y la variedad de combinaciones de sensores y dispositivos es infinita, no es posible proporcionar la solución exacta para todos los casos.

Sin embargo, presentaremos el código necesario para los dos diagramas de ejemplo que se muestran arriba.

Ejemplo 1

Conexión de un botón, un sensor de temperatura y un LED.

Para interconectar el código con el botón, el sensor de temperatura y el LED, abra **UserBot.ino** (**examples → OpenGrow → UserBot**) y navegue a la ***función setup***. Esta ***función*** contiene la configuración para todas las entradas y salidas. Deberá ajustar el código para que coincida con la electrónica ensamblada. Para el diagrama de ejemplo, el código debe ser el siguiente:

CONFIGURACIÓN DE ENTRADAS

```
inputs[INPUT_INDEX0].arduinoPin = A0;    //Button Pin
inputs[INPUT_INDEX1].arduinoPin = A1;    //LM35 Pin
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = BUTTON;
inputs[INPUT_INDEX1].type = DIG_TEMPERATURE;
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURACIÓN DE SALIDAS


```
outputs[OUTPUT_INDEX0].arduinoPin = 3;  //LED Pin
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = LED;
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Los diferentes tipos de entradas y salidas se declaran en el archivo **GroBot_Variables.h** (**libraries → OpenGrow**) debajo de los comentarios **'Possible Input Types'** y **'Possible Output Types'**.

Los cambios en el primer archivo están completos. El siguiente paso es abrir el archivo **SensorsTask.cpp** (**examples → OpenGrow → UserBot**).

NOTA: El UserBot soporta hasta 10 entradas y 10 salidas.

Después de abrir el archivo **SensorsTask.cpp**, navegue a la ***función SensorsTask*** y busque el comentario **'* IO Config *'.** Debajo del comentario encontrará un ejemplo de configuración de pines, esta configuración debe ajustarse para que coincida con los componentes ensamblados. Para el diagrama del ejemplo 1, el código debe ser el siguiente:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ***********************
    /*Here you need to set your used pins as inputs or outputs*/
    pinMode(inputs[INPUT_INDEX0].arduinoPin, INPUT);
    pinMode(inputs[INPUT_INDEX1].arduinoPin, INPUT);

    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Luego navegue a la ***función GoSensorsTask*** y observe el ***switch***. Dentro del ***case GET_IOS***, debe agregar el código para muestrear las entradas y actualizar sus valores en el ***array inputs***. Por ejemplo:

```
if (digitalRead(inputs[INPUT_INDEX0].arduinoPin))
  inputs[INPUT_INDEX0].value = 255;
else
  inputs[INPUT_INDEX0].value = 0;
```

En el caso de las salidas, debe agregar el código para actuar sobre ellas, dentro del ***case ACT_ON_IOS***. Por ejemplo:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

Puede agregar más ***cases*** de acuerdo a sus necesidades. Para el esquema del ejemplo 1, el código dentro del ***switch*** debe ser el siguiente:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = GET_TEMP;
  break;
case GET_TEMP:
  //Sampling Temperature Sensor
  time = getAnalogRead(inputs[INPUT_INDEX1].arduinoPin);
  inputs[INPUT_INDEX1].value += calcTemp(time);
  inputs[INPUT_INDEX1].value = inputs[INPUT_INDEX1].value/2;
  snsState = GET_IOS;
  break;
case GET_IOS:
  //Polling button
  if(digitalRead(inputs[INPUT_INDEX0].arduinoPin))
    inputs[INPUT_INDEX0].value = 255;
  else
    inputs[INPUT_INDEX0].value = 0x00;

  snsState= ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on LED pin
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1))
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = GET_TEMP;
  break;
```

Tenga en cuenta que hemos agregado más código relacionado con la actuación de salida. Estos cambios agregan soporte para la variable **'speed'** (p. ej. para control de intensidad de LED) y otras características implementadas en el sistema GroLab (como el tiempo de enfriamiento para dispositivos sensibles).

NOTA: Los *arrays* de entrada y salida almacenan valores de muestra del sensor, así como el estado de salida. Estos *arrays* son cruciales para intercambiar información con el sistema GroLab.

Si necesita cambiar los ***cases*** del ***switch***, debe cambiar el ***enum*** que contiene los ***cases***. Esto ***enum*** se declara en el archivo **SensorsTask.h** (**examples → OpenGrow → UserBot**).

Después de completar todos los pasos anteriores, el código estará listo para manejar los componentes ensamblados en el diagrama del ejemplo 1. El siguiente paso es compilar y cargar el código en su Arduino.

Si necesita ayuda para compilar y cargar el código, consulte la documentación oficial de Arduino o póngase en contacto con el centro de soporte de Arduino.

Después de cargar el código en su Arduino, todavía hay un paso necesario para que esté operativo y listo para comunicarse con el sistema GroLab: **establecer la configuración de fábrica** (número de serie y canal de comunicación). Consulte las instrucciones en las **páginas 25 y 26**.

Ejemplo 2

Conexión de un motor CC (bomba peristáltica, bomba de agua...).

Para interconectar el código con el motor CC, abra el archivo **UserBot.ino** (**examples → OpenGrow → UserBot-DCMotor**) y navegue a la ***función setup***. Esta ***función*** contiene la configuración para todas las entradas y salidas. Deberá ajustar el código para que coincida con la electrónica ensamblada. Para el diagrama de ejemplo, el código debe ser el siguiente:

CONFIGURACIÓN DE ENTRADAS

```
inputs[INPUT_INDEX0].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX1].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX2].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;    //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;    //Unused Input

inputs[INPUT_INDEX0].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX1].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;   //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;   //Unused Input
```

CONFIGURACIÓN DE SALIDAS

```
outputs[OUTPUT_INDEX0].arduinoPin = 5;  //DC Motor as Perista
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = PERISTALTIC_PUMP;  //DC Motor a
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Los diferentes tipos de entradas y salidas se declaran en el archivo **GroBot_Variables.h** (**libraries → OpenGrow**) debajo de los comentarios **'Possible Input Types'** y **'Possible Output Types'**.

Los cambios en el primer archivo están completos. El siguiente paso es abrir el archivo **SensorsTask.cpp** (**examples → OpenGrow → UserBot-DCMotor**).

NOTA: El UserBot soporta hasta 10 entradas y 10 salidas.

Después de abrir el archivo **SensorsTask.cpp**, navegue a la ***función SensorsTask*** busque el comentario **'* IO Config *'.** Debajo del comentario encontrará un ejemplo de configuración de pin, esta configuración debe ajustarse para que coincida con los componentes ensamblados. Para el diagrama del ejemplo 2, el código debe ser el siguiente:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ********************
    /*Here you need to set your used pins as inputs or output
    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Luego navegue a la ***función GoSensorsTask*** y observe el ***switch***. Dentro del ***case ACT_ON_IOS***, debe agregar el código para actuar en las salidas. Por ejemplo:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

NOTA: El *array* de salidas almacena los estados de las salidas. Esto *array* es crucial para intercambiar información con el sistema GroLab.

Puede agregar más ***cases*** de acuerdo a sus necesidades. Para el esquema del ejemplo 2, el código dentro del ***switch*** debe ser el siguiente:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on DC Motor
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1) )
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = INIT_SENSORS;
  break;
```

Tenga en cuenta que hemos agregado más código relacionado con la actuación de salida. Estos cambios agregan soporte para la variable **'speed'** y otras características implementadas en el sistema GroLab (como el tiempo de enfriamiento para dispositivos sensibles).

Si necesita cambiar los ***cases*** del ***switch***, debe cambiar el ***enum*** contiene los ***cases***. Esto ***enum*** se declara en el archivo **SensorsTask.h** (**examples → OpenGrow → UserBot-DCMotor**).

Después de completar todos los pasos anteriores, el código estará listo para administrar los componentes ensamblados en el diagrama del ejemplo 2. El siguiente paso es compilar y cargar el código en su Arduino.

Si necesita ayuda para compilar y cargar el código, consulte la documentación oficial de Arduino o póngase en contacto con el centro de soporte de Arduino.

Establecer config. de fábrica

Después de cargar el código en su Arduino, todavía hay un paso necesario para que esté operativo y listo para comunicarse con el sistema GroLab: **establecer la configuración de fábrica** (número de serie y canal de comunicación).

Asegúrese de que su Arduino esté conectado a su PC a través de USB y que el puerto serie no esté en uso por otra aplicación. Abra el IDE de Arduino y seleccione su puerto serie. Acceda al **'Monitor Serie'** de Arduino y configure la **velocidad en baudios en 230400**. Envíe el comando **'R'** con la opción **'Sin ajuste de linea'**. Espere un momento hasta que vea la respuesta **RST** en el 'Monitor Serie'. Reinicie el Arduino (corte de energía o botón de reinicio, si está disponible).

Abra el 'Monitor Serie' nuevamente y envíe el comando **'M'** con la opción **'Sin ajuste de linea'**. Luego envíe el número de canal de comunicación que su GroNode está utilizando*, con la opción **'Ambos NL & CR'**.

***Los canales de comunicación se configuran de 1 a 5, siendo 5 el canal predeterminado.**

Envíe el comando **‘S’** con la opción **‘Sin ajuste de linea’**, ingrese el número de serie deseado (**envíe un número a la vez**) y cuando llegue al último número, envíelo con la opcíon **‘Ambos NL & CR’.** El número de serie debe constar de 10 números y comenzar con el número '5', por ejemplo, '5123456789'.*

Para verificar que la configuración se haya aplicado correctamente, envíe el comando **‘D’** con la opción **‘Sin ajuste de linea’**. Debería ver, a través del ‘Monitor Serie’, algo similar a la siguiente imagen:

```
UserBot! Ready
N:       New Module
S:       5123456789
MID:     2
SAdd:    FF
FW:      1.1.0.8
```

Después de completar todos los pasos anteriores, su Arduino estará completamente configurado y listo para comunicarse con GroNode a través del UserBot *shield*.

***El número de serie ‘5999999999’ está reservado como número predeterminado y no debe utilizarse.**

COMUNICACIONES ②

Conexión con GroNode

Asegúrese de que GroNode esté correctamente instalado y accesible a través del *software* GroLab. Si no, sigue las instrucciones proporcionadas en el manual de GroNode.

Después de completar todos los pasos de instalación del UserBot *shield*, abra el *software* GroLab y conéctese a GroNode. Después de conectarse, debe acceder al panel **Módulos** a través del menú de configuración principal.

Normalmente, 2 o 3 minutos son suficientes para que GroNode detecte un módulo recién instalado. Una vez detectado, aparecerá automáticamente en el panel **Módulos** en la sección correspondiente al tipo de módulo.

Puede consultar la lista de módulos del mismo tipo en el lado derecho del panel **Módulos**. Un GroNode admite un máximo de 4 módulos de cada tipo.

Problemas o fallas de comunicación

GroNode está diseñado para comunicarse con los demás módulos mediante señales de radiofrecuencia. El **rango máximo de acción** es de **25 metros (82 pies) en interiores** y **100 metros (328 pies) en campo abierto** dependiendo de las condiciones del espacio.

Acceda al panel **Módulos** a través del *software* GroLab y asegúrese de que sus módulos sean accesibles. Si no son accesibles o tienen fallas/pérdidas de comunicación, es posible que haya excedido la distancia entre sus módulos y GroNode.

Dentro del panel de **Módulos**, puede encontrar un icono de señal *wireless* que indica si el módulo seleccionado está conectado a GroNode y se comunica correctamente.

⚠ ATENCIÓN: Algunos muros de carga y aparatos electrónicos pueden interferir en la señal.

Si el problema persiste, haga lo siguiente: coloque el módulo al lado de GroNode y verifique el estado de la comunicación a través del *software* GroLab. Si la comunicación se restablece con éxito, repita el proceso a diferentes distancias hasta que se identifique la distancia máxima de comunicación.

Si el módulo no puede comunicarse con GroNode, intente navegar a través de los diferentes canales de comunicación hasta que encuentre sus módulos (debe esperar 3 minutos entre cada cambio de canal).

Para cambiar el canal de comunicación, vaya al panel **Configuración** (de GroNode) y luego a la sección **Configuración General.** En esta sección encontrará el campo de configuración para el canal de comunicación. Haga clic en el botón en la esquina inferior derecha para activar la edición y cambiar el canal de comunicación al canal deseado. Para aplicar los cambios, haga clic en el botón verde en la esquina inferior derecha.

Si, después de seguir los pasos anteriores, sus módulos aún no aparecen en el *software* GroLab, comuníquese con nuestro soporte técnico.

Actualización de firmware

Tenga en cuenta que algunas actualizaciones de *software* pueden requerir la actualización del *firmware* de sus módulos GroLab.

Todas las actualizaciones proporcionan mejoras importantes que aseguran el mejor rendimiento del sistema.

Sin embargo, dado que el *firmware* de UserBot es compilado y aplicado por el usuario, es necesario seguir el repositorio de código de UserBot en GitHub. Si se envían actualizaciones de código, debe aplicar esas actualizaciones a su código. También deberá volver a compilar el código, cargarlo a Arduino y volver a aplicar la configuración de fábrica.

INSTR. DE SEGURIDAD ④

Las siguientes pautas generales de seguridad se proporcionan para ayudar a garantizar su propia seguridad personal y para proteger su dispositivo de posibles daños.

•No intente reparar o desmontar el dispositivo. Para algunos dispositivos con batería reemplazable por el usuario, lea y siga las instrucciones proporcionadas en el manual de instalación.
•Mantenga el dispositivo alejado de radiadores y fuentes de calor.
•Mantenga su equipo alejado de temperaturas extremadamente altas o bajas para garantizar que se utilice dentro del rango operativo especificado.
•No derrame alimentos o líquidos sobre su dispositivo.
•Antes de limpiar su dispositivo, desenchúfelo de la toma de corriente. Limpie su dispositivo con un paño suave y seco. No usar líquidos.
•Si su dispositivo no funciona normalmente, en particular, si hay sonidos u olores inusuales, desconéctelo de inmediato y comuníquese con un distribuidor autorizado o con el centro de soporte de Open Grow.
•Para ayudar a prevenir el riesgo potencial de descarga eléctrica, no conecte ni desconecte cables, ni mantenga o reconfigure su dispositivo durante tormentas eléctricas o durante períodos de rayos.
•Compruebe el voltaje antes de enchufar el dispositivo a una toma de corriente eléctrica para asegurarse de que el voltaje y la frecuencia requeridos coincidan con la fuente de alimentación disponible.
•Además, asegúrese de que sus módulos GroLab y los dispositivos conectados a ellos estén clasificados eléctricamente para funcionar con la alimentación disponible en su ubicación.
•No enchufe los cables de alimentación a una toma de corriente si los cables de alimentación están dañados.
•Para evitar descargas eléctricas, conecte los cables de alimentación a tomas eléctricas con conexión a tierra.
•Si utiliza un cable de extensión de alimentación, asegúrese que el amperaje total de los dispositivos conectados no exceda la capacidad total de resistencia del cable de extensión.

GARANTÍA ⑤

Open Grow, LDA, garantiza al consumidor que el producto está libre de defectos de material y/o fabricación. La responsabilidad de Open Grow, LDA, se limita a la reparación o reemplazo de piezas defectuosas. Antes de enviarnos cualquier pieza defectuosa, comuníquese con nuestro centro de soporte para verificar el procedimiento.

Puede encontrar nuestro centro de asistencia en **opengrow.pt/support** o envíe un correo electrónico a **support@opengrow.pt**.

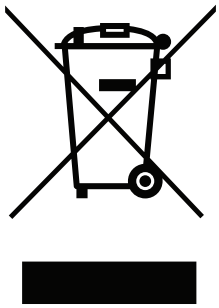
Todos los productos Open Grow, LDA, tienen una garantía de 2 años, excepto los consumibles (sensores y/o actuadores de cualquier tipo), mediante el uso apropiado.

La solicitud de aplicación de la garantía es intransferible y solo se podrá llevar a cabo por el consumidor original. Para hacer efectiva la garantía el consumidor deberá aportar siempre la factura de compra.

EXCLUSIÓN DE LA GARANTÍA:

La aplicación de la garantía está excluida en caso de que la avería de la pieza o piezas defectuosa/s se derive de un uso inadecuado y/o negligente del producto. Se entenderá por uso inadecuado y/o negligente cualquier uso distinto al propio de la naturaleza del producto y/o al recomendado en el manual de instrucciones, no realizar las operaciones de mantenimiento recomendadas en el manual de instrucciones, o realizar otras distintas a las mencionadas y que comprometer la calidad del producto, realizar modificaciones fuera de los talleres autorizados y/o con piezas no originales o que no estén homologadas.

CONFORMIDAD ⑥



Este símbolo en el producto o empaque significa que de acuerdo con las leyes y regulaciones locales, este producto no debe desecharse con la basura doméstica, sino que debe enviarse para su reciclaje. Por favor, llévelo a un punto de recolección designado por sus autoridades locales una vez que haya llegado al final de su vida útil, algunos aceptarán productos de forma gratuita. Al reciclar el producto y su empaque de esta manera, usted ayuda a conservar el medio ambiente y proteger la salud humana.

Este símbolo en el producto o empaque significa que este producto cumple con las regulaciones RoHS de la Directiva del Parlamento Europeo y del Consejo sobre las Restricciones de Uso de Ciertas Sustancias Peligrosas en Equipos Eléctricos y Electrónicos (2011/65/EU).

Este símbolo en el producto o empaque significa que este producto cumple con las siguientes directivas y regulaciones:

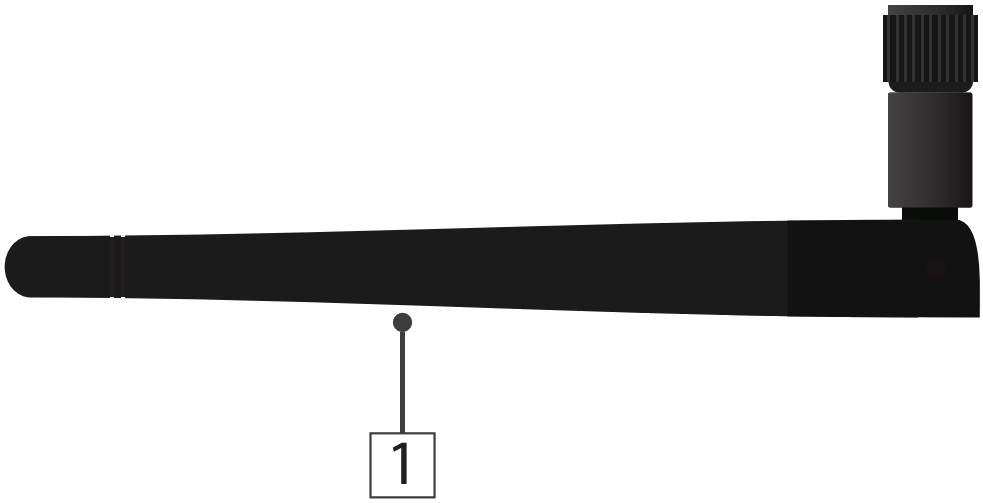
- (2014/53/EU) Directiva de Equipos Radioeléctricos.
- (2011/65/EU) Directiva RoHS.
- (2014/35/EU) Directiva de Baja Tensión.
- (2014/30/EU) EMC.

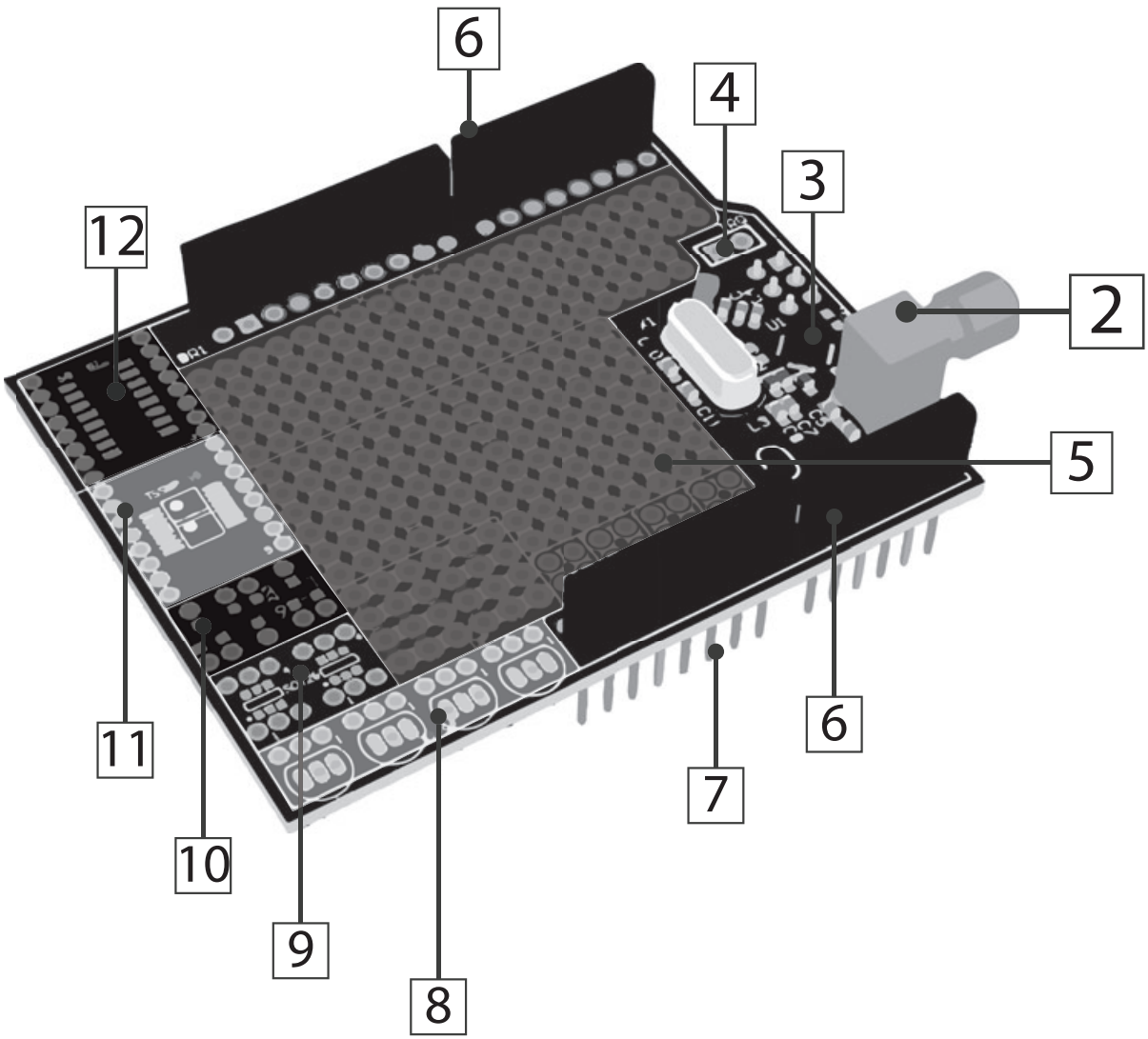
	Banda(s) de frecuencia	Potencia de salida máxima (EIRP)
2.4 G	2.4 - 2.4835 GHz	100 mW

Open Grow, LDA, se reserva el derecho de actualizar y/o modificar el contenido de sus productos en cualquier momento sin previo aviso. Consulte nuestros Términos y Condiciones en www.opengrow.pt.

OPENGROW.

ANOTACIONES


1


6
4
3
12
2
5
11
10
9
8
7
6

1 Antenne de communications RF

2 Prise pour antenne de communications RF

3 Module RF

4 Pin d'interruption RF

5 Carte de développement universel

6 Connecteurs rapides femelles

7 Pins de connexion Arduino

8 4 x TO-92

9 2 x SOT26

10 3 x SSOT223

11 2 x TSSOP-8 (=) 1 x TSSOP-16

12 SOIC16

TABLEAU DE SPÉCIFICATIONS

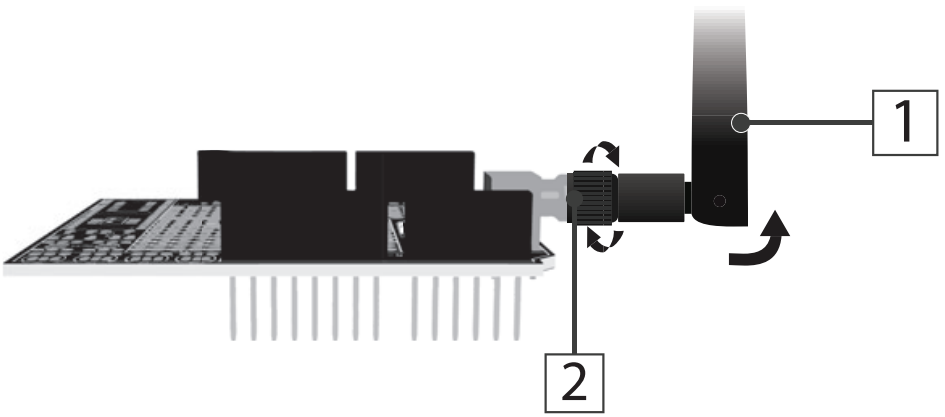
Version du Hardware	H01
Dimensions	68.6mm x 53.4mm
Extérieur	Matériel: FR-4 Couleur: Vert
Tension de Fonctionn.	+3V3 VDC
Connexions	RP-SMA femelle Connecteurs rapides femelles (Connexion/extension Arduino)
Sockets SMD	1 x SOIC16 2 x TSSOP 8 (=) 1 x TSSOP 16 3 x SSOT223 2 x SOT26
Sockets TH	4 x TO-92
Carte de Dévelopg.	DIP/1206/0805/0603
Espacement de la carte de développement	2.54mm - 100mil
Comprend	Antenne
Communication entre modules	Radio Fréquence - 2.4GHz
Garantie	Garantie du hardware limitée à 2 ans

INSTALACION ①

Connexions de base

Vissez l'antenne de communication (1) sur le module à travers l'entrée filetée (2).

Pour une meilleure communication, il est recommandé que l'antenne (1) soit perpendiculaire au module et non obstruée.



Après avoir placé l'antenne, le UserBot *shield* est prêt à être monté sur Arduino.

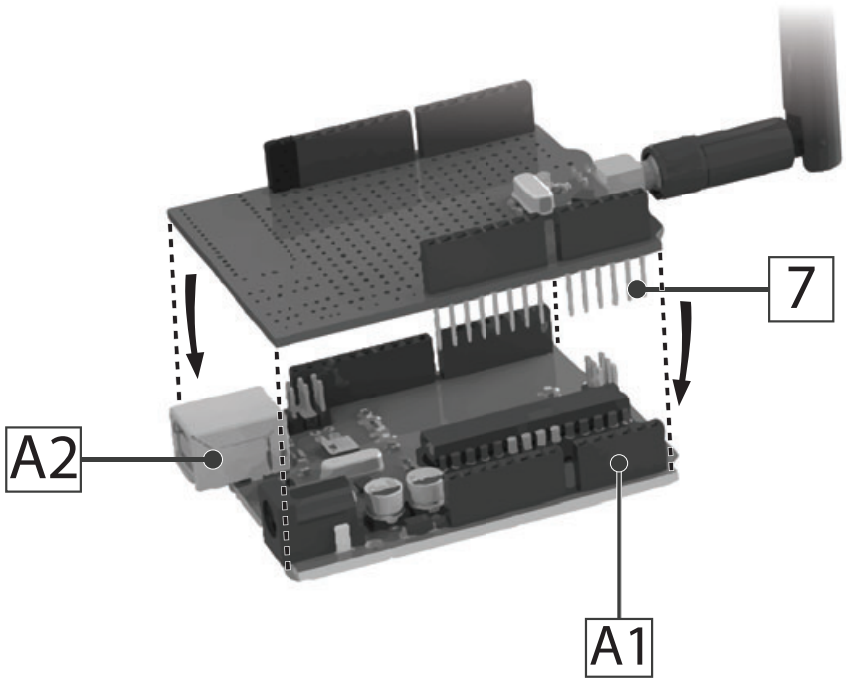
Connexion avec Arduino

Avant de poursuivre l'assemblage, vérifiez la compatibilité de votre Arduino avec UserBot. Liste des modèles compatibles:

- **Arduino UNO;**
- **Arduino Duemilanove;**
- **Arduino Leonardo avec connecteurs femelles;**
- **Genuino Zero.**

Avant de monter le UserBot *shield* sur Arduino, vérifiez que l'Arduino est correctement assemblé et prêt à être utilisé (suivez le manuel d'instruction officiel Arduino). Assurez-vous également que l'Arduino n'est pas branché à l'alimentation.

Placez l'Arduino sur une surface ferme, à l'écart des zones exposées aux inondations. Assurez-vous que les connecteurs rapides femelles Arduino (A1) sont tournés vers le haut et les broches du connecteur UserBot (7) vers le bas.



Insérez les pins du connexion du UserBot (7) de haut en bas dans les connecteurs femelles Arduino (A1) comme indiqué dans la figure précédente.

⚠ **Evitez tout contact entre l'utilisateur UserBot *shield* et l'entrée d'alimentation USB d'Arduino (A2).**

Connexion périphérique

Etant donné que UserBot est une extension pour Arduino, cela signifie que tout dispositif ou capteur pouvant interagir avec Arduino est également compatible avec UserBot *shield*.

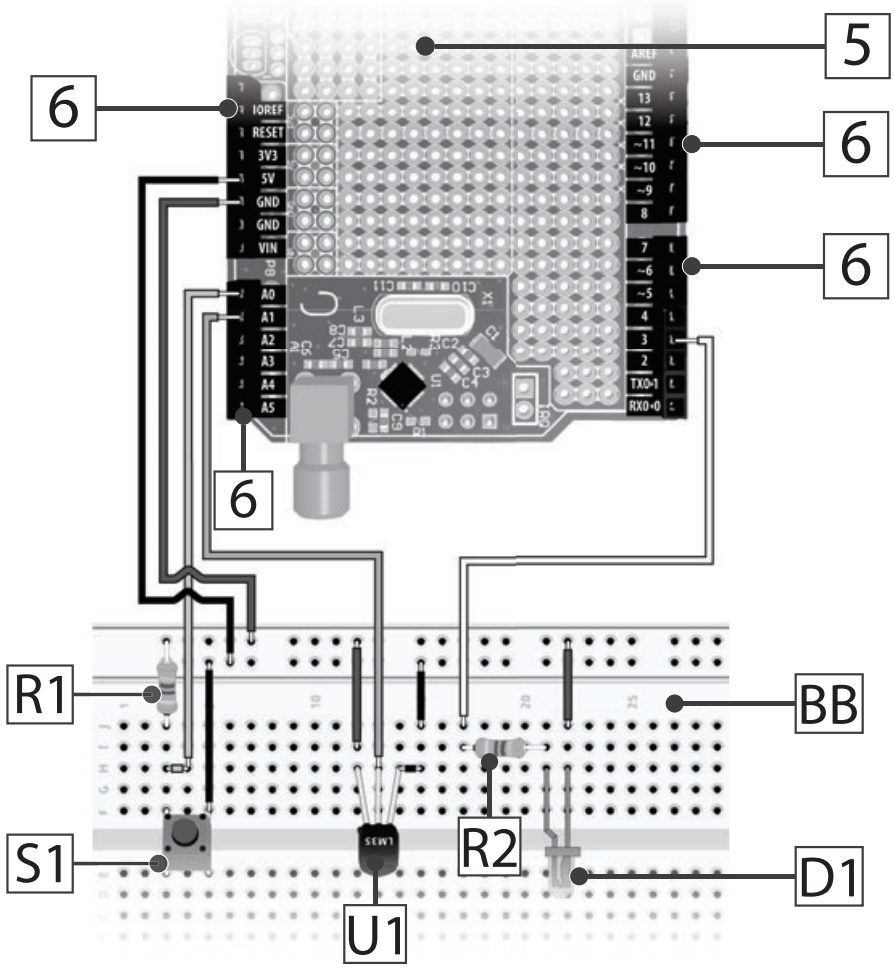
Par conséquent, la liste des appareils et capteurs compatibles est extrêmement longue et il est impossible de les lister et d'expliquer leur connexion.

Néanmoins, nous allons couvrir deux exemples: le premier montre comment allumer/échantillonner un bouton et un capteur de température, ainsi que pour agir sur une LED; La seconde explique comment utiliser un moteur à courant continu.

⚠ Avant de connecter un capteur ou un périphérique, assurez-vous qu'Arduino n'est pas branché sur le secteur. De plus, il est conseillé de rouler sur une surface ferme, loin des zones inondables. Utilisez des outils appropriés pour manipuler les composants électroniques.

Exemple 1

Connexion d'un bouton, d'un capteur de température et d'une LED.



5	Carte de développement universel
6	Connecteurs rapides femelles
R1	Résistance 10kΩ (pull down)
R2	Résistance 240Ω pour LED
S1	Bouton tactile SPST
U1	Capteur de température LM35
D1	LED
BB	Carte de développement externe

Le diagramme présenté à la page précédente montre la connexion de certains composants UserBot *shield* via les connecteurs rapides femelles (6).

Le bouton tactile SPST (S1) est échantillonné dans le **pin analogique A0** d'Arduino; le capteur de température LM35 (U1) est échantillonné dans le **pin analogique A1** d'Arduino; La LED (D1) est contrôlée par le **pin numérique 3** d'Arduino. Pour contrôler la 'vitesse/intensité' d'un périphérique, vous devez utiliser une pin numérique Arduino PWM (par exemple, les pins 3 et 5).

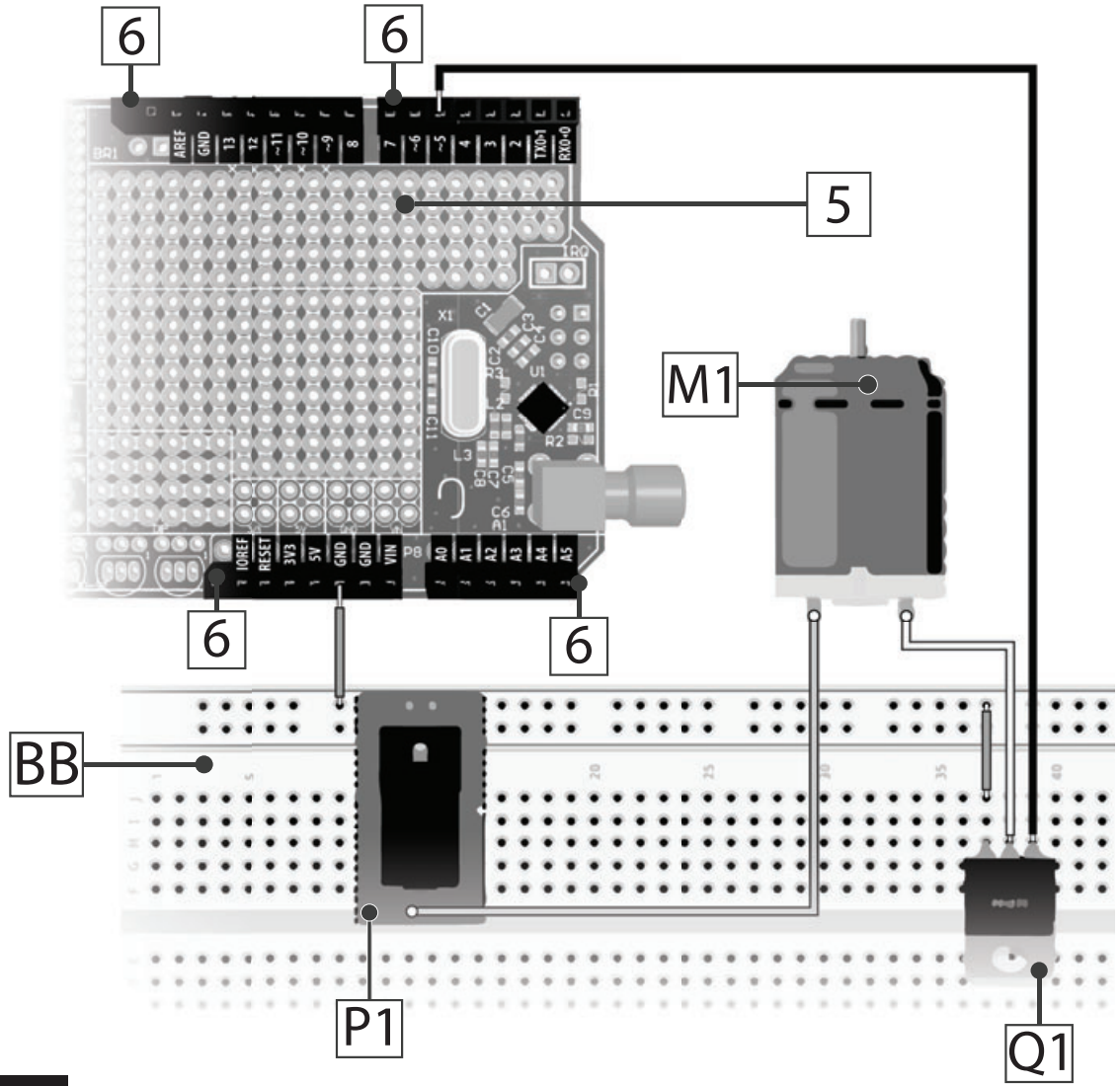
Utilisez les pins **+5V**, **+3V3** et **GND** d'Arduino pour alimenter les capteurs et/ou les périphériques. Vérifiez les restrictions d'alimentation de ces pins dans la documentation officielle d'Arduino.

UserBot *shield* offre une carte de développement intégrée (5) que vous pouvez utiliser librement. Cependant, pour le diagramme, nous utilisons une plaque externe (BB) pour une meilleure compréhension.

⚠ Il est essentiel de faire attention à la polarité des différents composants. Une connexion incorrecte peut endommager les composants et/ou Arduino/UserBot.

Exemple 2

Connexion d'un moteur CC (pompe péristaltique, pompe à eau...).



5 Carte de développement universel

6 Connecteurs rapides femelles

M1 Moteur à courant continu (CC)

BB Carte de développement externe

P1 Puissance d'entrée 12VDC

Q1 MOSFET type N (p. ex. IRLZ14)

Le diagramme présenté à la page précédente montre l'utilisation d'un MOSFET de type N (Q1) pour contrôler la tension d'alimentation d'un moteur à courant continu (M1).

Le moteur à courant continu (M1) est alimenté par une source externe 12VDC (P1) et est commandé par PWM via le **pin numérique 5** d'Arduino. Pour contrôler la "vitesse/intensité" d'un périphérique, vous devez utiliser une pin numérique Arduino PWM (par exemple, les pins 3 et 5).

L'entrée d'alimentation 12VDC (P1) et le MOSFET (Q1) doivent également être connectés dans le **pin GND** d'Arduino .

Utilisez les connecteurs rapides femelles de l'utilisateur (6) pour connecter l'entrée 12VDC (P1) et le MOSFET (Q1) aux broches Arduino correspondantes.

UserBot *shield* offre une carte de développement intégrée (5) que vous pouvez utiliser librement. Cependant, pour le diagramme, nous utilisons une plaque externe (BB) pour une meilleure compréhension.

⚠ **Il est essentiel de faire attention à la polarité des différents composants. Une connexion incorrecte peut endommager les composants et/ou Arduino/UserBot.**

Prépare le code source

Après avoir monté les capteurs et les appareils, il est temps de préparer le code source pour les contrôler.

Open Grow fournit le code du micrologiciel de base dans un format open source. Le code est disponible au téléchargement à partir de la page GitHub suivante:

github.com/OpenWeGrow/UserBot

De cette manière, la première tâche consiste à télécharger le code. Comme il existe plusieurs manières de le faire (par exemple, via des applications d'administration de référentiel ou en téléchargeant directement à partir du site *web* de GitHub), nous ne détaillerons aucune méthode spécifique. Toutefois, si vous avez besoin d'aide, contactez notre support technique:

www.opengrow.pt @ support@opengrow.pt

REMARQUE: Les exemples fournis dans ce manuel sont basés sur le code disponible le 14 août 2019.

Puisque le bouclier UserBot consiste essentiellement à intégrer Arduino au système GroLab, nous utiliserons l'IDE Arduino pour préparer le code. Si à un moment quelconque vous rencontrez un problème ou un obstacle avec l'IDE Arduino, consultez la documentation officielle Arduino.

Après avoir téléchargé le code de base, vous devez copier/déplacer le contenu des dossiers **'libraries'** et **'examples'**, dans leurs dossiers respectifs (même nom) dans le dossier d'installation Arduino de votre ordinateur.

Assurez-vous que la carte Arduino sélectionnée dans l'EDI est correcte. Pour ce faire, ouvrez l'EDI Arduino, accédez à **Outils → Type de carte**, puis sélectionnez la carte Arduino appropriée en fonction de votre Arduino physique.

L'étape suivante consiste à ouvrir le code avec l'IDE Arduino en sélectionnant **Fichier → Exemples → OpenGrow → UserBot** yet en cliquant pour l'ouvrir.

Avant de continuer avec le changement de code, nous allons lister la structure du fichier et expliquer un peu:

→ libraries
→ OpenGrow

• ComsTask.cpp • ComsTask.h	**Machine d'état pour gérer les tâches de communication.**
• CRC16.cpp • CRC16.h	**Calculatrice CRC par Tim W. Shilling.**
• EEPROM_Utils.cpp • EEPROM_Utils.h	**Fonctions de gestion NVM.**
• GroBot_Variables.h	**Définitions d'entrée/sortie d'UserBot.**
• nRF24L01.h • RF24.cpp • RF24.h • RF24_config.h	***Driver* NRF24L01, par J. Coliz <maniacbug@ymail.com> avec des réglages effectués par Open Grow.**
• OpenBus.cpp • OpenBus.h	**Gestionnaire de commandes de communication.**
• SerialTask.cpp • SerialTask.h	**Machine d'état pour gérer les commandes UART.**

→ examples
→ OpenGrow
→ UserBot

• SensorsTask.cpp • SensorsTask.h	**Une machine d'état dédiée pour l'échantillonnage de capteurs et le contrôle de sortie.**
• UserBot.ino	**Fichier Arduino (.ino) et le code de base pour exécuter le programme. Ceci est le fichier de configuration principal pour UserBot. Toutes les entrées et sorties doivent être configurées dans ce fichier.**

→ examples
→ OpenGrow
→ UserBot-DCMotor

(Même structure de fichier que: **examples → OpenGrow → UserBot**.)

Changer le code

Une fois que le code de base a été préparé, il doit être modifié pour correspondre aux composants assemblés.

En résumé, **dans la plupart des cas (sinon tous), il vous suffit de modifier ou de dupliquer les fichiers suivants:**

→ examples
  → OpenGrow
    → UserBot
      - SensorsTask.cpp
      - SensorsTask.h
      - UserBot.ino

Comme les modifications de code requises dépendent des composants assemblés par l'utilisateur et que le nombre de combinaisons de capteurs et de dispositifs est infini, il n'est pas possible de fournir la solution exacte dans tous les cas.

Cependant, nous allons présenter le code nécessaire pour les deux exemples de diagramme ci-dessus.

Exemple 1

Connexion d'un bouton, d'un capteur de température et d'une LED.

Pour interconnecter le code avec le bouton, le capteur de température et le voyant, ouvrez **UserBot.ino** (**examples → OpenGrow → UserBot**) et accédez à la ***fonction setup***. Cette ***fonction*** contient la configuration pour toutes les entrées et sorties. Vous devez ajuster le code pour correspondre à l'électronique assemblée. Pour l'exemple de diagramme, le code doit être le suivant:

CONFIGURATION D'ENTRÉE

```
inputs[INPUT_INDEX0].arduinoPin = A0;    //Button Pin
inputs[INPUT_INDEX1].arduinoPin = A1;    //LM35 Pin
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = BUTTON;
inputs[INPUT_INDEX1].type = DIG_TEMPERATURE;
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURATION DES SORTIES

```
outputs[OUTPUT_INDEX0].arduinoPin = 3;  //LED Pin
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = LED;
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Les différents types d'entrées et de sorties sont déclarés dans le fichier **GroBot_Variables.h** (**libraries → OpenGrow**) sous les commentaires **'Possible Input Types'** et **'Possible Output Types'**.

Les modifications dans le premier fichier sont terminées. L'étape suivante consiste à ouvrir le fichier **SensorsTask.cpp** (**examples → OpenGrow → UserBot**).

REMARQUE: Le UserBot prend en charge jusqu'à 10 entrées et 10 sorties.

Après avoir ouvert le fichier **SensorsTask.cpp**, accédez à la ***fonction SensorsTask*** et recherchez le commentaire **'* IO Config *'.** En dessous du commentaire, vous trouverez un exemple de configuration des pins. Cette configuration doit être ajustée pour correspondre aux composants assemblés. Pour le diagramme de l'exemple 1, le code doit être le suivant:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ***********************
    /*Here you need to set your used pins as inputs or outputs*/
    pinMode(inputs[INPUT_INDEX0].arduinoPin, INPUT);
    pinMode(inputs[INPUT_INDEX1].arduinoPin, INPUT);

    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Accédez ensuite à la ***fonction GoSensorsTask*** et observez le ***switch***. Dans le ***case GET_IOS***, vous devez ajouter le code pour échantillonner les entrées et mettre à jour leurs valeurs dans le ***array inputs***. Par exemple:

```
if (digitalRead(inputs[INPUT_INDEX0].arduinoPin))
  inputs[INPUT_INDEX0].value = 255;
else
  inputs[INPUT_INDEX0].value = 0;
```

Dans le cas des exits, vous devez ajouter le code pour agir sur eux, dans le ***case ACT_ON_IOS***. Par exemple:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

Vous pouvez ajouter plus de ***cases*** en fonction de vos besoins. Pour le schéma de l'exemple 1, le code à l'intérieur du ***switch*** doit être le suivant:


```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = GET_TEMP;
  break;
case GET_TEMP:
  //Sampling Temperature Sensor
  time = getAnalogRead(inputs[INPUT_INDEX1].arduinoPin);
  inputs[INPUT_INDEX1].value += calcTemp(time);
  inputs[INPUT_INDEX1].value = inputs[INPUT_INDEX1].value/2;
  snsState = GET_IOS;
  break;
case GET_IOS:
  //Polling button
  if(digitalRead(inputs[INPUT_INDEX0].arduinoPin))
    inputs[INPUT_INDEX0].value = 255;
  else
    inputs[INPUT_INDEX0].value = 0x00;

  snsState= ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on LED pin
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1))
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = GET_TEMP;
  break;
```

Notez que nous avons ajouté plus de code lié aux performances de sortie. Ces modifications ajoutent la prise en charge de la variable **'speed'** (p. ex. pour le contrôle d'intensité de LED) et d'autres fonctionnalités implémentées dans le système GroLab (telles que le temps de refroidissement des périphériques sensibles).

REMARQUE: Les *arrays* d'entrée et de sortie stockent les valeurs d'échantillons de capteur, ainsi que l'état de la sortie. Ces *arrays* sont essentiels pour l'échange d'informations avec le système GroLab.

Si vous devez modifier les ***cases*** de ***switch***, vous devez modifier l'***enum*** contenant les ***cases***. Cette ***enum*** est déclarée dans le fichier **SensorsTask.h** (**examples → OpenGrow → UserBot**).

Une fois toutes les étapes précédentes terminées, le code sera prêt à gérer les composants assemblés dans le diagramme de l'exemple 1. L'étape suivante consiste à compiler et à charger le code dans votre Arduino.

Si vous avez besoin d'aide pour la compilation et le chargement du code, consultez la documentation officielle Arduino ou contactez le centre de support Arduino.

Après avoir chargé le code dans votre Arduino, il reste une étape nécessaire pour le rendre opérationnel et prêt à communiquer avec le système GroLab: **définissez les paramètres d'usine** (numéro de série et canal de communication). Voir les instructions aux **pages 25 et 26**.

Exemple 2

Connexion d'un moteur CC (pompe péristaltique, pompe à eau...).

Pour interconnecter le code avec le moteur CC, ouvrez le fichier **UserBot.ino** (**examples → OpenGrow → UserBot-DCMotor**) et accédez à la ***fonction setup***. Cette ***fonction*** contient la configuration pour toutes les entrées et sorties. Vous devez ajuster le code pour correspondre à l'électronique assemblée. Pour l'exemple de diagramme, le code doit être le suivant:

CONFIGURATION D'ENTRÉE

```
inputs[INPUT_INDEX0].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX1].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX1].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURATION DES SORTIES

```
outputs[OUTPUT_INDEX0].arduinoPin = 5;  //DC Motor as Perista
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = PERISTALTIC_PUMP;  //DC Motor a
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Les différents types d'entrées et de sorties sont déclarés dans le fichier **GroBot_Variables.h** (**libraries → OpenGrow**) sous les commentaires **'Possible Input Types'** et **'Possible Output Types'**.

Les modifications dans le premier fichier sont terminées. L'étape suivante consiste à ouvrir le fichier **SensorsTask.cpp** (**examples → OpenGrow → UserBot-DCMotor**).

REMARQUE: Le UserBot prend en charge jusqu'à 10 entrées et 10 sorties.

Après avoir ouvert le fichier **SensorsTask.cpp**, accédez à la ***fonction SensorsTask*** et recherchez le commentaire **'* IO Config *'.** En dessous du commentaire, vous trouverez un exemple de configuration des pins. Cette configuration doit être ajustée pour correspondre aux composants assemblés. Pour le diagramme de l'exemple 2, le code doit être le suivant:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ********************
    /*Here you need to set your used pins as inputs or output
    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Accédez ensuite à la ***fonction GoSensorsTask*** et observez le ***switch***. Dans le ***case ACT_ON_IOS***, vous devez ajouter le code pour agir sur les sorties. Par exemple:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

REMARQUE: L'*array* de sortie stocke les états des sorties. Cet *array* est essentiel pour échanger des informations avec le système GroLab.

Vous pouvez ajouter plus de ***cases*** en fonction de vos besoins. Pour le schéma de l'exemple 2, le code à l'intérieur du ***switch*** doit être le suivant:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on DC Motor
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1) )
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = INIT_SENSORS;
  break;
```

Notez que nous avons ajouté plus de code lié aux performances de sortie. Ces modifications ajoutent la prise en charge de la variable **'speed'** et d'autres fonctionnalités implémentées dans le système GroLab (telles que le temps de refroidissement des périphériques sensibles).

Si vous devez modifier les ***cases*** de ***switch***, vous devez modifier l'***enum*** contenant les ***cases***. Cette ***enum*** est déclarée dans le fichier **SensorsTask.h** (**examples → OpenGrow → UserBot-DCMotor**).

Une fois toutes les étapes précédentes terminées, le code sera prêt à gérer les composants assemblés dans le diagramme de l'exemple 2. L'étape suivante consiste à compiler et à charger le code dans votre Arduino.

Si vous avez besoin d’aide pour la compilation et le chargement du code, consultez la documentation officielle Arduino ou contactez le centre de support Arduino.

Définir les paramètres d'usine

Après avoir chargé le code dans votre Arduino, il reste une étape nécessaire pour le rendre opérationnel et prêt à communiquer avec le système GroLab: **définissez les paramètres d'usine** (numéro de série et canal de communication).

Assurez-vous que votre Arduino est connecté à votre PC via USB et que le port série n'est pas utilisé par une autre application. Ouvrez l'IDE Arduino et sélectionnez son port série. Accédez au **'Moniteur série'** d'Arduino et définissez le **débit en bauds sur 230400**. Envoyez la commande **'R'** avec l'option **'Pas de fin de ligne'**. Attendez un moment jusqu'à ce que la réponse **RST** apparaisse sur le 'Moniteur série'. Redémarrez l'Arduino (panne de courant ou bouton de réinitialisation, si disponible).

Ouvrez à nouveau le 'Moniteur série' et envoyez la commande **'M'** avec l'option **'Pas de fin de ligne'**. Envoyez ensuite le numéro du canal de communication que votre GroNode utilise*, avec l'option **'Les deux, NL & CR'**.

***Les canaux de communication sont réglés de 1 à 5, 5 étant le canal par défaut.**

Envoyez la commande **‘S’** avec l’option **‘Pas de fin de ligne’**, entrez le numéro de série souhaité (**n’envoyez qu’un numéro à la fois**). Lorsque vous atteignez le dernier numéro, envoyez-le avec l’option **‘Les deux, NL & CR’.** Le numéro de série doit comporter 10 chiffres et commencer par le chiffre '5', par exemple '5123456789'.*****

Pour vérifier que la configuration a été appliquée correctement, envoyez la commande **‘D’** avec l’option **‘Pas de fin de ligne’**. Vous devriez voir, à travers le ‘Moniteur série’, quelque chose de similaire à l'image suivante:

```
UserBot! Ready
N:       New Module
S:       5123456789
MID:     2
SAdd:    FF
FW:      1.1.0.8
```

Une fois toutes les étapes ci-dessus terminées, votre Arduino sera entièrement configuré et prêt à communiquer avec GroNode via UserBot *shield*.

***Le numéro de série ‘5999999999’ est réservé comme numéro par défaut et ne doit pas être utilisé.**

COMMUNICATIONS ②

Connexion avec GroNode

Assurez-vous que GroNode est correctement installé et accessible via le logiciel GroLab. Sinon, suivez les instructions fournies dans le manuel GroNode.

Une fois toutes les étapes d'installation de UserBot *shield* terminées, ouvrez le logiciel GroLab et connectez-vous à GroNode. Une fois la connexion établie, vous devez accéder au panneau **Modules** via le menu de configuration principal.

En général, 2 à 3 minutes suffisent à GroNode pour détecter un module nouvellement installé. Une fois détecté, il apparaît automatiquement dans le panneau **Modules** dans la section correspondant au type de module.

Vous pouvez consulter la liste des modules du même type sur le côté droit du panneau **Modules**. Un GroNode supporte un maximum de 4 modules de chaque type.

Problèmes ou échecs de communication

GroNode est conçu pour communiquer avec les autres modules en utilisant des signaux de radiofréquence. La **portée maximale de l'action** est de **25 mètres (82 pieds) à l'intérieur** et à **100 mètres (328 pieds) en plein champ** en fonction des conditions de l'espace.

Accédez au panneau **Modules** via le logiciel GroLab et assurez-vous que vos modules sont accessibles. S'ils ne sont pas accessibles ou présentent des problèmes de communication, vous avez peut-être dépassé la distance entre vos modules et GroNode.

Dans le panneau **Modules**, vous trouverez une icône de signal sans fil qui indique si le module présenté est connecté à GroNode et communique correctement.

⚠ **ATTENTION: Certains murs de charge et appareils électroniques peuvent interférer avec le signal.**

Si le problème persiste, procédez comme suit: placez le module à côté de GroNode et vérifiez le statut de la communication via le logiciel GroLab. Si la communication est rétablie avec succès, répétez le processus à différentes distances jusqu'à ce que la distance de communication maximale soit identifiée.

Si le module ne peut pas communiquer avec GroNode, essayez de naviguer dans les différents canaux de communication jusqu'à ce que vous trouviez vos modules (vous devez attendre 3 minutes entre chaque changement de canal).

Pour modifier le canal de communication, accédez au panneau **Paramètres** (de GroNode), puis à la section **Paramètres Généraux**. Dans cette section, vous trouverez le champ de configuration du canal de communication. Cliquez sur le bouton dans le coin inférieur droit pour activer l'édition et changer le canal de communication en canal souhaité. Pour appliquer les modifications, cliquez sur le bouton vert dans le coin inférieur droit.

Si, après avoir suivi les étapes ci-dessus, vos modules n'apparaissent toujours pas dans le logiciel GroLab, contactez notre support technique.

Mise à jour de firmware

Notez que certaines mises à jour logicielles peuvent nécessiter la mise à jour du micrologiciel de vos modules GroLab.

Toutes les mises à jour apportent des améliorations importantes qui garantissent les meilleures performances du système.

Cependant, étant donné que le micrologiciel UserBot est compilé et appliqué par l'utilisateur, il est nécessaire de suivre le référentiel de code UserBot sur GitHub. Si des mises à jour de code sont envoyées, vous devez les appliquer à votre code. Vous devez également recompiler le code, le télécharger sur Arduino et réappliquer les paramètres d'usine.

CONSIG. DE SECURITE ④

Les consignes de sécurité générales suivantes sont fournies pour vous aider à assurer votre sécurité personnelle et à protéger votre appareil contre les dommages éventuels.

•N'essayez pas de réparer ou de démonter l'appareil. Pour certains appareils dotés d'une batterie remplaçable par l'utilisateur, lisez et suivez les instructions fournies dans le manuel d'installation.
•Gardez l'appareil à l'écart des radiateurs et des sources de chaleur.
•Maintenez votre équipement à l'abri de températures extrêmement élevées ou basses pour vous assurer qu'il est utilisé dans la plage de fonctionnement spécifiée.
•Ne renversez pas de nourriture ou de liquide sur votre équipement.
•Avant de nettoyer votre appareil, débranchez-le de la prise de courant. Nettoyez votre appareil avec un chiffon doux et sec. Ne pas utiliser de liquides.
•Si votre appareil ne fonctionne pas normalement, en particulier en cas de sons ou d'odeurs inhabituels, débranchez-le immédiatement et contactez un revendeur agréé ou le centre de support Open Grow.
•Pour éviter tout risque d'électrocution, ne connectez et ne déconnectez pas les cordons, n'effectuez aucune opération de maintenance ni de reconfiguration de votre équipement pendant les orages ou les éclairs.
•Vérifiez la tension avant de brancher l'appareil sur une prise électrique afin de vous assurer que la tension et la fréquence requises correspondent à la source d'alimentation disponible.
•Assurez-vous également que vos modules GroLab et les périphériques connectés sont qualifiés sur le plan électrique pour fonctionner avec le courant disponible sur votre site.
•Ne branchez pas les cordons d'alimentation dans une prise électrique si les cordons sont endommagés.
•Pour éviter tout risque d'électrocution, connectez les cordons d'alimentation dans des prises électriques mises à la terre.
•Si vous utilisez un cordon d'alimentation supplémentaire, assurez-vous que l'intensité nominale totale des produits connectés au cordon d'alimentation supplémentaire ne dépasse pas la capacité nominale en ampères de la extension.

GARANTIE ⑤

Open Grow, LDA, garantit au consommateur que le produit est exempt de défauts de matériau et/ou de fabrication. La responsabilité de Open Grow, LDA, se limite à la réparation ou au remplacement des pièces défectueuses. Avant de nous envoyer des pièces défectueuses, contactez notre centre de support pour vérifier la procédure.

Vous pouvez trouver notre centre d'aide à **opengrow.pt/support** ou envoyer un courrier électronique à **support@opengrow.pt**.

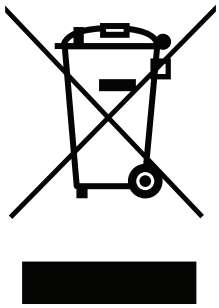
Tous les produits Open Grow, LDA, bénéficient d'une garantie de 2 ans, à l'exception des consommables (capteurs et/ou actionneurs de tout type), pour une utilisation appropriée.

La demande de garantie n'est pas transférable et ne peut être effectuée que par le consommateur d'origine. Pour que la garantie soit effective, le consommateur doit toujours fournir la facture d'achat.

EXCLUSION DE LA GARANTIE:

L'application de la garantie est exclue au cas où la panne de la pièce ou des rebuts est due à un usage inadéquat et/ou négligent du produit. Un usage inadéquat et/ou négligent est tout usage différent et non conforme à la nature du produit et/ou à ce qui est recommandé dans le manuel d'instructions, ne pas réaliser les opérations de maintenance recommandées dans le manuel d'instructions, ou en réaliser d'autres différentes de celles qui sont mentionnées et qui compromettent la qualité du produit, faire des modifications en dehors des établissements autorisés et/ou en utilisant des pièces qui ne sont pas d'origine ou qui ne sont pas homologuées.

CONFORMITÉ ⑥



Ce symbole sur le produit ou son emballage signifie que, conformément à la législation et aux réglementations locales en vigueur, ce produit ne doit pas être jeté avec les ordures ménagères, mais doit être envoyé au recyclage. S'il vous plaît apportez-le à un point de collecte désigné par vos autorités locales une fois que vous avez atteint la fin de sa vie utile, certains accepteront les produits gratuitement. En recyclant ainsi le produit et son emballage, vous contribuez à la protection de l'environnement et à la santé humaine.

Ce symbole sur le produit ou son emballage signifie que ce produit est conforme à la réglementation RoHS de la directive du Parlement Européen et du Conseil sur les Restrictions d'Utilisation de Certaines Substances Dangereuses dans les Équipements Électriques et Électroniques (2011/65/EU).

Ce symbole sur le produit ou l'emballage signifie que ce produit est conforme aux directives et réglementations suivantes:

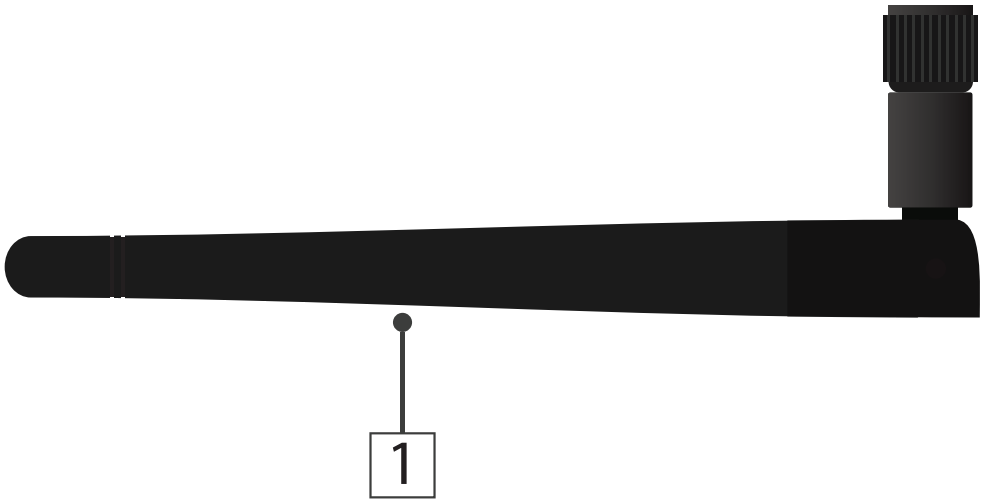
- (2014/53/EU) Directive des Équipements Radio.
- (2011/65/EU) Directive RoHS.
- (2014/35/EU) Directive Basse Tension.
- (2014/30/EU) EMC.

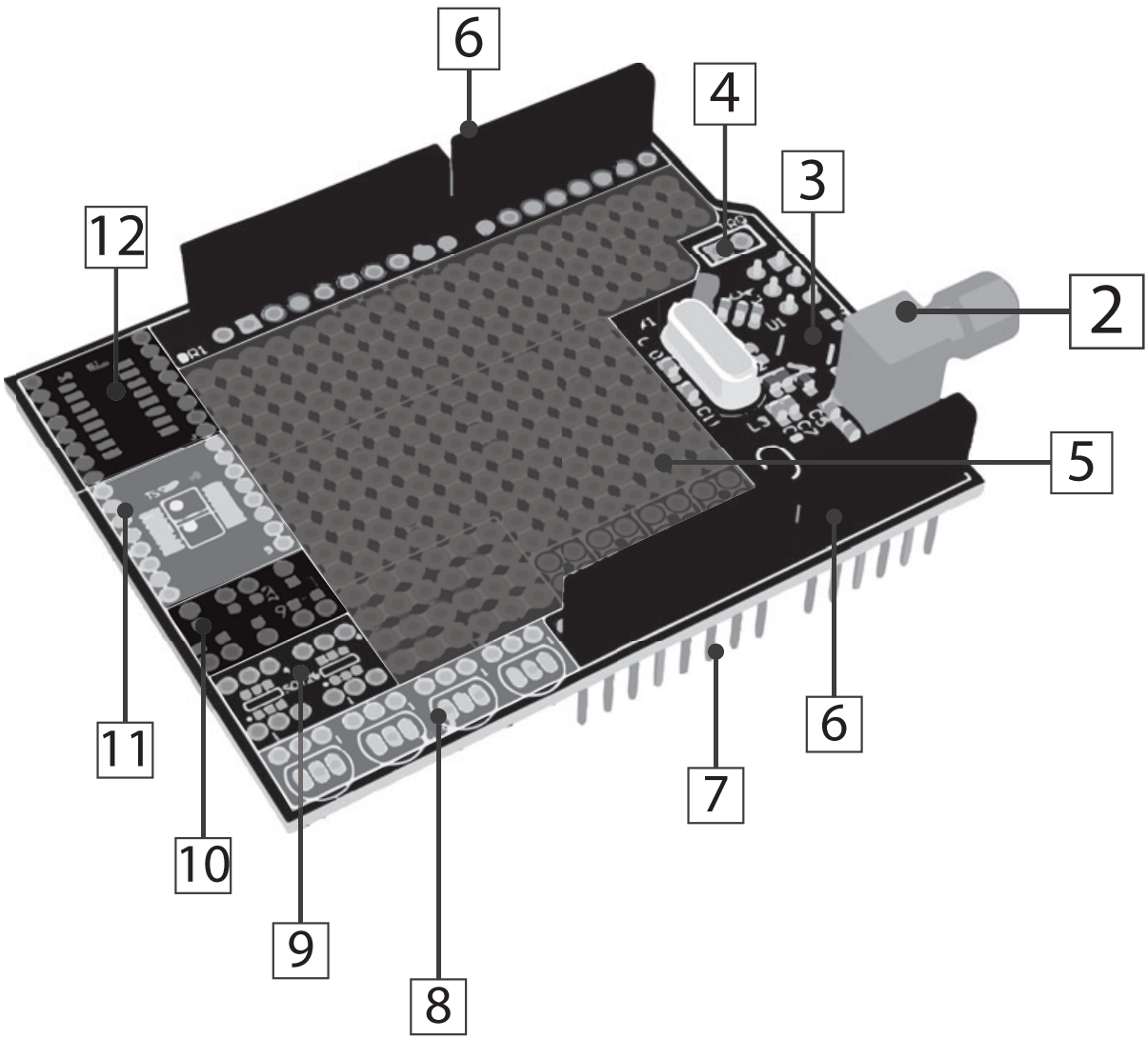
	Bande(s) de fréquence	Puissance de sortie maximale (EIRP)
2.4 G	2.4 - 2.4835 GHz	100 mW

Open Grow, LDA, se réserve le droit de mettre à jour et/ou de modifier le contenu de ses produits à tout moment et sans préavis. Consultez nos conditions générales sur www.opengrow.pt.

OPENGROW.

INSCRIPTIONS


1


6
4
3
12
2
5
11
10
9
8
7
6

1 Antena de comunicação RF

2 Entrada de rosca para antena RF

3 Módulo RF

4 Pino de interrupção de RF

5 Placa de prototipagem universal

6 Conectores rápidos fêmea

7 Pinos de ligação ao Arduino

8 4 x TO-92

9 2 x SOT26

10 3 x SSOT223

11 2 x TSSOP-8 (=) 1 x TSSOP-16

12 SOIC16

TABELA DE ESPECIFICAÇÕES

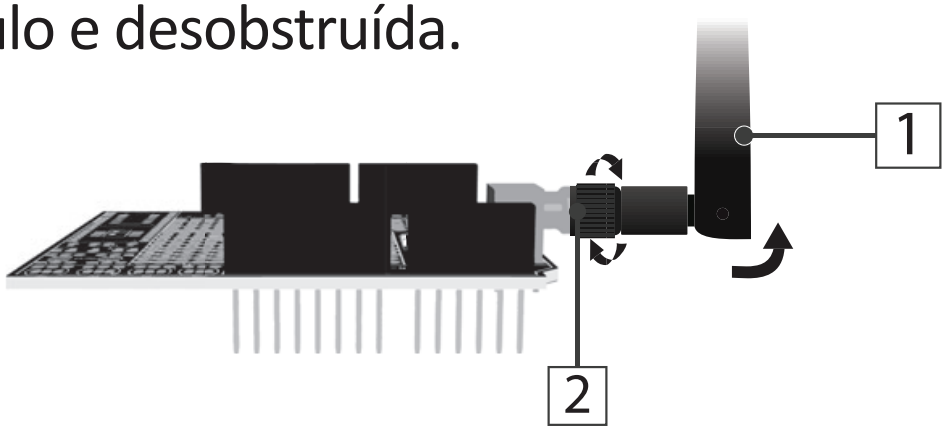
Versão do Hardware	H01
Dimensões	68.6mm x 53.4mm
Exterior	Material: FR-4 Cor: Verde
Tensão de Funcion.	+3V3 VDC
Ligações	RP-SMA fêmea Conectores rápidos fêmea (ligação/extensão ao Arduino)
Sockets SMD	1 x SOIC16 2 x TSSOP 8 (=) 1 x TSSOP 16 3 x SSOT223 2 x SOT26
Sockets TH	4 x TO-92
Placa de Prototipagem	DIP/1206/0805/0603
Espaçamento da Placa	2.54mm - 100mil
Inclui	Antena
Comunicação entre Módulos	Radiofrequência - 2.4GHz
Garantia	Garantia de hardware limitada a 2 anos

INSTALAÇÃO ①

Ligações iniciais

Enrosque a antena de comunicações (1) ao módulo, através da entrada de rosca para a mesma (2).

Para uma melhor comunicação, é recomendado que a antena (1) esteja perperdicular ao módulo e desobstruída.



Após a colocação da antena, o UserBot *shield* está pronto a ser montado no Arduino.

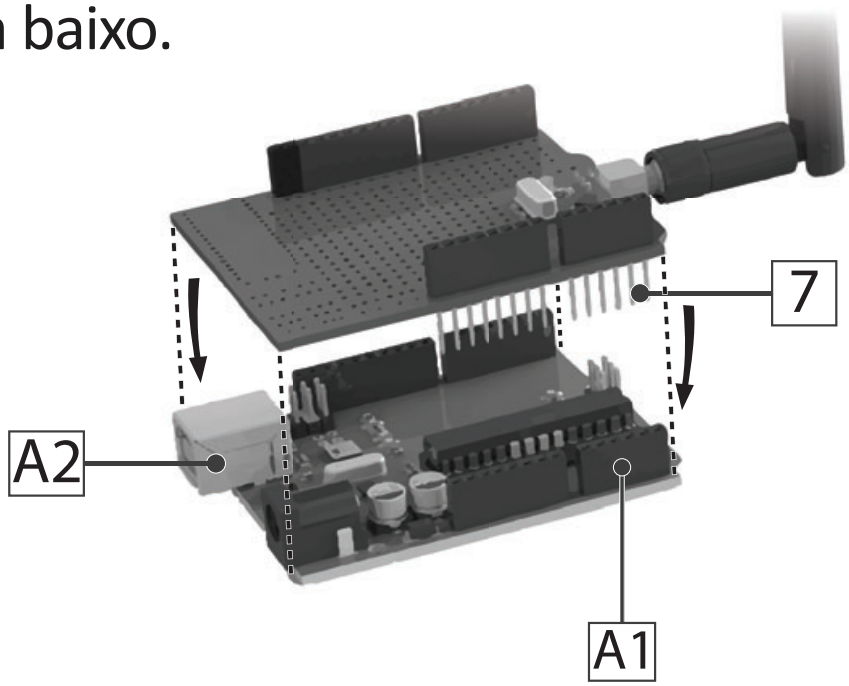
Ligação ao Arduino

Antes de prosseguir com a montagem, verifique a compatibilidade do seu Arduino com o UserBot. Lista de modelos compatíveis:

- **Arduino UNO;**
- **Arduino Duemilanove;**
- **Arduino Leonardo com conectores fêmea;**
- **Genuino Zero.**

Antes de montar o UserBot *shield* no Arduino, confirme que o Arduino está corretamente montado e pronto a utilizar (siga o manual de instruções oficial do Arduino). Verifique também se o Arduino não está ligado à fonte de alimentação.

Coloque o Arduino numa superfície firme, longe de áreas propensas a inundações. Certifique-se que os conectores rápidos fêmea do Arduino (A1) estão voltados para cima e que os pinos de ligação do UserBot (7) estão voltados para baixo.



Introduza de cima para baixo, os pinos de conexão do UserBot (7) nos conectores fêmea do Arduino (A1), tal como mostrado na figura acima.

⚠ **Evite o contato entre o UserBot *shield* e a entrada de energia USB do Arduino (A2).**

Ligação de periféricos

Como o UserBot é uma extensão para o Arduino, significa que qualquer dispositivo ou sensor que possa interagir com o Arduino também é compatível com o UserBot *shield*.

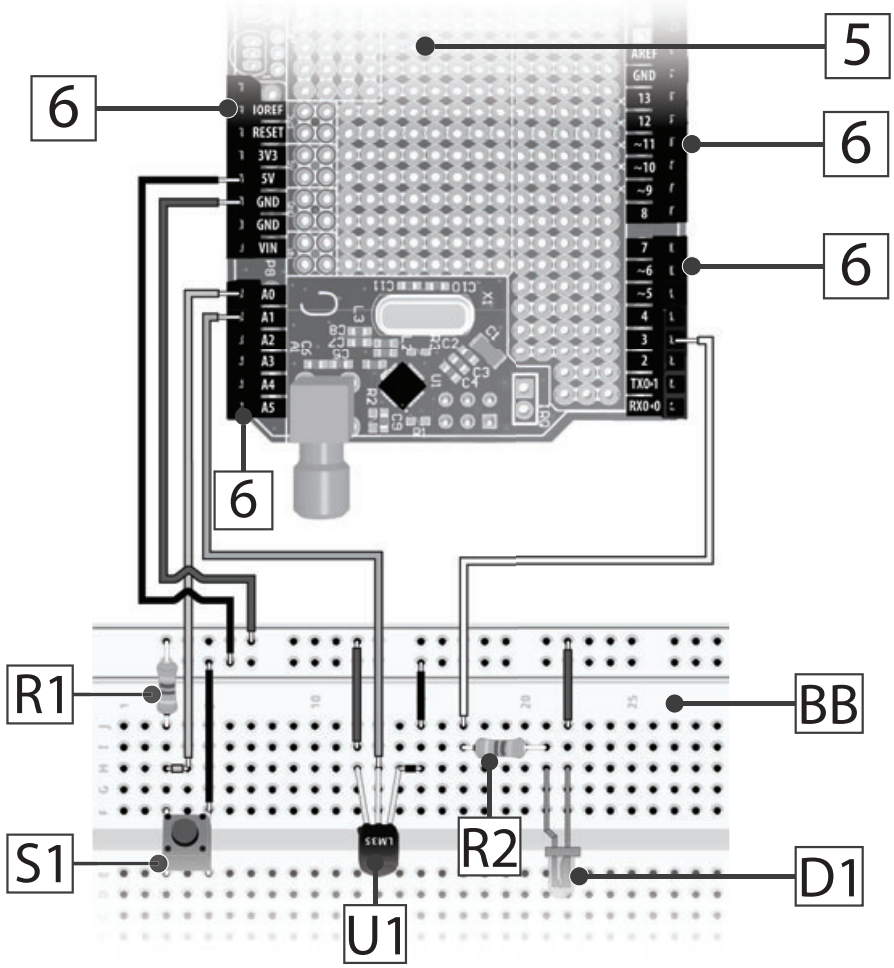
Desta forma, a lista de dispositivos e sensores compatíveis é extremamente longa, sendo impossível listá-los, assim como explicar a ligação dos mesmos.

Ainda assim, iremos abordar dois exemplos: o primeiro mostra como ligar/amostrar um botão e um sensor de temperatura, assim como atuar num LED; o segundo explica como atuar num motor de corrente contínua (CC).

⚠ Antes de ligar qualquer sensor ou dispositivo, verifique se o Arduino não está ligado à fonte de alimentação. Além disso, é aconselhável realizar a montagem numa superfície firme, longe de áreas propensas a inundações. Use ferramentas adequadas para manejar componentes eletrónicos.

Exemplo 1

Ligação de um botão, um sensor de temperatura e um LED.



5	Placa de prototipagem universal
6	Conectores rápidos fêmea
R1	Resistência de 10kΩ (pull down)
R2	Resistência de 240Ω para LED
S1	Botão táctil SPST
U1	Sensor de temperatura LM35
D1	LED
BB	Placa de prototipagem externa

O diagrama apresentado na página anterior, mostra a ligação de alguns componentes ao UserBot *shield*, através dos conectores rápdios fêmea (6).

O botão táctil SPST (S1) é amostrado no **pino analógico A0** do Arduino; o sensor de temperatura LM35 (U1) é amostrado no **pino analógico A1** do Arduino; o LED (D1) é controlado através do **pino digital 3** do Arduino. Para controlar a 'velocidade/intensidade' de um dispositivo, é necessário usar um pino digital PWM do Arduino (por exemplo, os pinos 3 e 5).

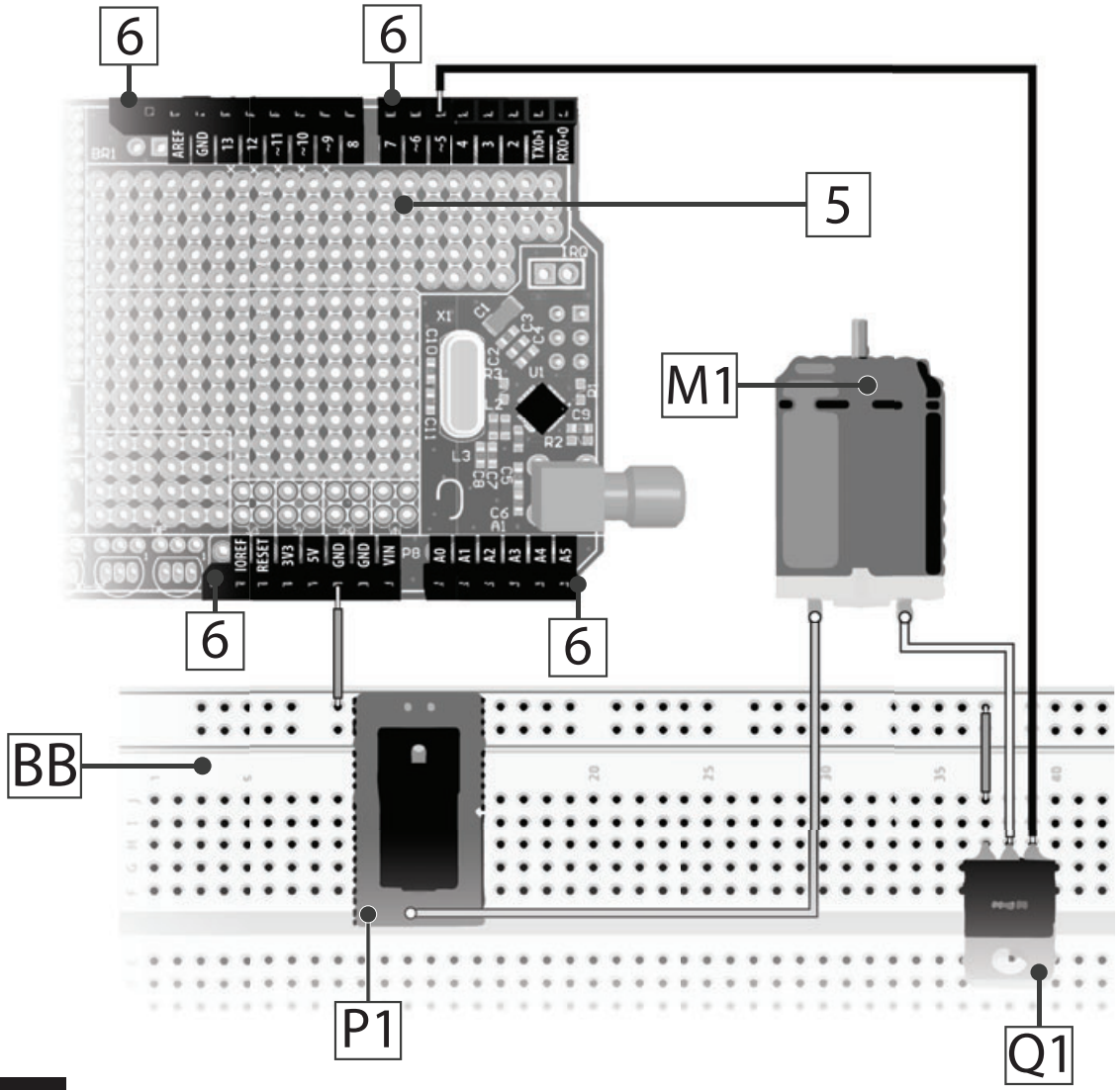
Use os **pinos +5V**, **+3V3** e **GND** do Arduino, para alimentar os sensores e/ou dispositivos. Por favor, consulte as restrições de energia desses pinos, na documentação oficial do Arduino.

O UserBot *shield* oferece uma placa de protipagem embutida (5), a qual pode utilizar livremente. No entanto, para o diagrama, usámos uma placa externa (BB) para uma melhor compreensão.

⚠ **É crucial prestar atenção à polaridade dos diferentes componentes. Uma ligação incorreta pode danificar os componentes e/ou o Arduino/UserBot.**

Exemplo 2

Ligação de um motor CC (bomba peristáltica, bomba de água...).



5 Placa de prototipagem universal

6 Conectores rápidos fêmea

M1 Motor de corrente contínua (CC)

BB Placa de prototipagem externa

P1 Entrada de alimentação 12VDC

Q1 MOSFET tipo N (p. ex. IRLZ14)

O diagrama apresentado na página anterior, mostra a utilização de um MOSFET tipo N (Q1) para controlar a tensão de alimentação de um motor CC (M1).

O motor CC (M1) é alimentado por 12VDC externos (P1) e tem controlo PWM control através do **pino digital 5** do Arduino. Para controlar a "velocidade/intensidade" de um dispositivo, é necessário usar um pino digital PWM do Arduino (por exemplo, os pinos 3 e 5).

A entrada de alimentação 12VDC (P1) e o MOSFET (Q1) também precisam estar ligados ao **pino GND** do Arduino.

Use os conectores rápidos fêmea do UserBot (6) para ligar a entrada de alimentação 12VDC (P1) e o MOSFET (Q1) ao respectivos pinos do Arduino.

O UserBot *shield* oferece uma placa de protipagem embutida (5), a qual pode utilizar livremente. No entanto, para o diagrama, usámos uma placa externa (BB) para uma melhor compreensão.

⚠ **É crucial prestar atenção à polaridade dos diferentes componentes. Uma ligação incorreta pode danificar os componentes e/ou o Arduino/UserBot.**

Preparar o código-fonte

Depois de montar os sensores e dispositivos, é altura de preparar o código-fonte para os controlar.

A Open Grow fornece o código de *firmware* base em formato de código aberto. O código está disponível para *download* na seguinte página do GitHub:

github.com/OpenWeGrow/UserBot

Desta forma, a primeira tarefa é fazer o *download* do código. Como existem várias formas de o fazer (ex.: através de aplicações de gestão de repositórios ou, *download* direto da página *web* do GitHub), não iremos detalhar nenhum método específico. No entanto, se precisar de ajuda, entre em contato com nosso suporte técnico:

www.opengrow.pt @ support@opengrow.pt

NOTA: Os exemplos fornecidos por este manual baseiam-se no código disponível a 14 de agosto de 2019.

Como a essência do UserBot *shield* é integrar o Arduino com o sistema GroLab, então para preparar o código, iremos utilizar o IDE do Arduino. Se a qualquer momento encontrar algum problema ou obstáculo com o IDE do Arduino, por favor, consulte a documentação oficial do Arduino.

Após *download* do código base, deverá copiar/mover o conteúdo das pastas **'libraries'** e **'examples'**, para as respectivas pastas (mesmo nome) dentro da pasta de instalação do Arduino no seu PC.

Certifique-se que a placa Arduino selecionada no IDE é a correta. Para isso, abra o IDE do Arduino e navegue até **Ferramentas → Placa** e selecione a placa Arduino correta baseada no seu Arduino físico.

O próximo passo é abrir o código com o IDE do Arduino, para isso navegue até **Ficheiro* → Exemplos → OpenGrow → UserBot** e clique para abri-lo.

***'Ficheiro'* em Português de Portugal é equivalente a *'arquivo'* em Português do Brasil.**

Antes de prosseguir com a alteração do código, listaremos a estrutura do arquivo e explicaremos um pouco:

→ libraries
→ OpenGrow

• ComsTask.cpp • ComsTask.h	**Máquina de estados para a gestão de tarefas de comunicação.**
• CRC16.cpp • CRC16.h	**Calculadora CRC por Tim W. Shilling.**
• EEPROM_Utils.cpp • EEPROM_Utils.h	**Funções de gestão de NVM.**
• GroBot_Variables.h	**Definições de entradas/saídas do UserBot.**
• nRF24L01.h • RF24.cpp • RF24.h • RF24_config.h	***Driver* NRF24L01, por J. Coliz <maniacbug@ymail.com> com ajustes feitos pela Open Grow.**
• OpenBus.cpp • OpenBus.h	**Gestor de comandos de comunicação.**
• SerialTask.cpp • SerialTask.h	**Máquina de estados para a gestão comandos UART.**

→ examples
→ OpenGrow
→ UserBot

• SensorsTask.cpp • SensorsTask.h	**Uma máquina de estados dedicada para amostragem de sensores e controle de saída.**
• UserBot.ino	**Ficheiro do tipo Arduino (.ino) e o código base para executar o programa. Este é o ficheiro de configuração principal do UserBot. Todas as entradas e saídas devem ser configuradas neste ficheiro.**

→ examples
→ OpenGrow
→ UserBot-DCMotor

(Mesma estrutura de ficheiros que: **examples → OpenGrow → UserBot**.)

Alterar o código

Após ter o código base preparado, é necessário alterar o mesmo para estar de acordo com os componentes montados.

Resumindo, **para a maioria dos casos (se não todos)**, o **utilizador só precisa de alterar ou duplicar os seguintes ficheiros:**

→ examples
  → OpenGrow
    → UserBot
      - SensorsTask.cpp
      - SensorsTask.h
      - UserBot.ino

Como as alterações necessárias ao código, dependem dos componentes montados pelo utilizador e, a variedade de combinações de de sensores e dispositivos é infinita, não é possível apresentar a solução exata para todos os casos.

No entanto, iremos apresentar o código necessário para os dois diagramas exemplo mostrados anteriormente.

Exemplo 1

Ligação de um botão, um sensor de temperatura e um LED.

Para fazer a interface do código com o botão, o sensor de temperatura e o LED, abra o **UserBot.ino** (**examples → OpenGrow → UserBot**) e navegue até à ***função setup***. Esta ***função*** contém a configuração para todas as entradas e saídas. Deverá ajustar o código para corresponder à eletrónica montada. Para o diagrama de exemplo, o código deverá ser o seguinte:

CONFIGURAÇÃO DE ENTRADAS

```
inputs[INPUT_INDEX0].arduinoPin = A0;    //Button Pin
inputs[INPUT_INDEX1].arduinoPin = A1;    //LM35 Pin
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = BUTTON;
inputs[INPUT_INDEX1].type = DIG_TEMPERATURE;
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURAÇÃO DE SAÍDAS


```
outputs[OUTPUT_INDEX0].arduinoPin = 3;  //LED Pin
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = LED;
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Os diferentes tipos de entradas e saídas, estão declarados no ficheiro **GroBot_Variables.h** (**libraries → OpenGrow**) abaixo dos comentários **'Possible Input Types'** e **'Possible Output Types'**.

As alterações ao primeiro ficheiro estão concluídas. O próximo passo é abrir o ficheiro **SensorsTask.cpp** (**examples → OpenGrow → UserBot**).

NOTA: O UserBot *shield* suporta até 10 entradas e 10 saídas.

Depois de abrir o ficheiro **SensorsTask.cpp**, navegue até à ***função SensorsTask*** e procure o comentário **'* IO Config *'.** Abaixo do comentário, encontrará um exemplo de configuração de pinos, a qual deverá ser ajustada para corresponder aos componentes montados. Para o diagrama do exemplo 1, o código deverá ser o seguinte:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ***********************
    /*Here you need to set your used pins as inputs or outputs*/
    pinMode(inputs[INPUT_INDEX0].arduinoPin, INPUT);
    pinMode(inputs[INPUT_INDEX1].arduinoPin, INPUT);

    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

De seguida, navegue até à ***função GoSensorsTask*** e observe o ***switch***. Dentro do ***case GET_IOS***, deverá adicionar o código para amostrar as entradas e, atualizar os respetivos valores no ***array inputs***. Por exemplo:

```
if (digitalRead(inputs[INPUT_INDEX0].arduinoPin))
  inputs[INPUT_INDEX0].value = 255;
else
  inputs[INPUT_INDEX0].value = 0;
```

No caso das saídas, deverá adicionar o código para atuar nas mesmas, dentro do ***case ACT_ON_IOS***. Por exemplo:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

Pode adicionar mais ***cases*** consoante as suas necessidades. Para o esquema do examplo 1, o código dentro do ***switch*** deverá ser o seguinte:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = GET_TEMP;
  break;
case GET_TEMP:
  //Sampling Temperature Sensor
  time = getAnalogRead(inputs[INPUT_INDEX1].arduinoPin);
  inputs[INPUT_INDEX1].value += calcTemp(time);
  inputs[INPUT_INDEX1].value = inputs[INPUT_INDEX1].value/2;
  snsState = GET_IOS;
  break;
case GET_IOS:
  //Polling button
  if(digitalRead(inputs[INPUT_INDEX0].arduinoPin))
    inputs[INPUT_INDEX0].value = 255;
  else
    inputs[INPUT_INDEX0].value = 0x00;

  snsState= ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on LED pin
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1))
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = GET_TEMP;
  break;
```

Observe que adicionámos mais código relativo à atuação na saída. Estas alterações adicionam suporte à variável **'speed'** (p. ex. para controlo da intensidade do LED) e a outras funcionalidades implementadas no sistema GroLab (como o tempo de arrefecimento para dispositivos sensíveis).

NOTA: Os *arrays* de entradas e saídas armazenam os valores amostrados pelos sensores, bem como o estado das saídas. Estes *arrays* são cruciais para a troca de informações com o sistema GroLab.

Se precisar de alterar os ***cases*** do ***switch***, será necessário alterar o ***enum*** que contém os ***cases***. Este ***enum*** está declarado no ficheiro **SensorsTask.h** (**examples → OpenGrow → UserBot**).

Após completar todos os passos anteriores, o código estará pronto a manejar os componentes montados no diagrama do exemplo 1. O próximo passo é compilar e fazer *upload* do código para o seu Arduino.

Se precisar de ajuda para compilar e fazer o *upload* do código, consulte a documentação oficial do Arduino ou entre em contato com o centro de suporte do Arduino.

Após o *upload* do código para o seu Arduino, ainda há um passo necessário para que este esteja operacional e pronto a comunicar com o sistema GroLab: **definir as configurações de fábrica** (número de série e canal de comunicação). Por favor, consulte as instruções nas **páginas 25 e 26**.

Exemplo 2

Ligação de um motor CC (bomba peristáltica, bomba de água...).

Para fazer a interface do código com o motor CC, abra o ficheiro **UserBot.ino** (**examples → OpenGrow → UserBot-DCMotor**) e navega até à ***função setup***. Esta ***função*** contém a configuração para todas as entradas e saídas. Deverá ajustar o código para corresponder à eletrónica montada. Para o diagrama do exemplo 2, o código deverá ser o seguinte:

CONFIGURAÇÃO DE ENTRADAS

```
inputs[INPUT_INDEX0].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX1].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX1].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURAÇÃO DE SAÍDAS

```
outputs[OUTPUT_INDEX0].arduinoPin = 5;  //DC Motor as Perista
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = PERISTALTIC_PUMP;  //DC Motor a
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

Os diferentes tipos de entradas e saídas, estão declarados no ficheiro **GroBot_Variables.h** (**libraries → OpenGrow**) abaixo dos comentários **'Possible Input Types'** e **'Possible Output Types'**.

As alterações ao primeiro ficheiro estão concluídas. O próximo passo é abrir o ficheiro **SensorsTask.cpp** (**examples → OpenGrow → UserBot-DCMotor**).

NOTA: O UserBot *shield* suporta até 10 entradas e 10 saídas.

Depois de abrir o ficheiro **SensorsTask.cpp**, navegue até à ***função SensorsTask*** e procure o comentário **'* IO Config *'.** Abaixo do comentário, encontrará um exemplo de configuração de pinos, a qual deverá ser ajustada para corresponder aos componentes montados. Para o diagrama do exemplo 2, o código deverá ser o seguinte:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ********************
    /*Here you need to set your used pins as inputs or outputs
    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

De seguida, navegue até à ***função GoSensorsTask*** e observe o ***switch***. Dentro do ***case ACT_ON_IOS***, deverá adicionar o código para atuar nas saídas. Por exemplo:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

NOTA: O *array* de saídas armazena os estados das saídas. Este *array* é crucial para a troca de informações com o sistema GroLab.

Pode adicionar mais ***cases*** consoante as suas necessidades. Para o esquema do examplo 2, o código dentro do ***switch*** deverá ser o seguinte:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on DC Motor
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1) )
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = INIT_SENSORS;
  break;
```

Observe que adicionámos mais código relativo à atuação na saída. Estas alterações adicionam suporte à variável **'speed'** e a outras funcionalidades implementadas no sistema GroLab (como o tempo de arrefecimento para dispositivos sensíveis).

Se precisar de alterar os ***cases*** do ***switch***, será necessário alterar o ***enum*** que contém os ***cases***. Este ***enum*** está declarado no ficheiro **SensorsTask.h** (**examples → OpenGrow → UserBot-DCMotor**).

Após completar todos os passos anteriores, o código estará pronto para fazer a gestão dos componentes montados no diagrama do exemplo 2. O próximo passo é compilar e fazer *upload* do código para o seu Arduino.

Se precisar de ajuda para compilar e fazer o *upload* do código, consulte a documentação oficial do Arduino ou entre em contato com o centro de suporte do Arduino.

Definir configurações de fábrica

Após o *upload* do código para o seu Arduino, ainda há um passo necessário para que este esteja operacional e pronto para comunicar com o sistema GroLab: **definir as configurações de fábrica** (número de série e canal de comunicação).

Certifique-se de que o seu Arduino está ligado ao PC via USB e que a porta série não está a ser utilizada por outra aplicação. Abra o IDE do Arduino e selecione a respectiva porta série. Acesse o **'Monitor Série'*** do Arduino e defina a **taxa de transmissão para 230400**. Envie o comando **'R'** com a opção **'Sem final de linha'**. Espere um momento, até ver a resposta **RST** no 'Monitor Série'. Reinicie o Arduino (interrupção de energia ou botão de *reset*, se disponível).

Abra novamente o 'Monitor Série' e envie o comando **'M'** com a opção **'Sem final de linha'**. Em seguida, envie o número relativo ao canal de comunicação em uso pelo seu GroNode**, com a opção **'Nova linha e retorno de linha'**.

***'Monitor Série' em Português de Portugal, 'Monitor Serial' em Português do Brasil.**

****Os canais de comunicação estão definidos de 1 a 5, sendo o 5 o canal padrão.**

Envie o comando **'S'** com a opção **'Sem final de linha'**, digite o número de série desejado (**envie um número por vez**) e quando chegar ao último número, envie-o com a opção **'Nova linha e retorno de linha'.** O número de série deve ser composto por 10 números e começar pelo número '5', por exemplo, '5123456789'.*

Para verificar se as configurações foram corretamente aplicadas, envie o comando **'D'** com a opção **'Sem final de linha'**. Deverá ver, através do 'Monitor Série', algo semelhante à seguinte imagem:

```
UserBot! Ready
N:      New Module
S:      5123456789
MID:    2
SAdd:   FF
FW:     1.1.0.8
```

Após a conclusão de todas os passos anteriores, o seu Arduino estará totalmente configurado e pronto a comunicar com o GroNode através do UserBot *shield*.

***O número de série '5999999999' está reservado como número padrão e não deverá ser utilizado.**

COMUNICAÇÃO ②

Ligação ao GroNode

Certifique-se que o GroNode está corretamente instalado e acessível através do *software* GroLab. Caso contrário, siga as instruções disponíveis no manual do GroNode.

Após concluir todos os passos de instalação do UserBot *shield*, abra o *software* GroLab e estabeleça ligação com o GroNode. Após estabelecer ligação, deverá aceder ao painel dos **Módulos** através do menu principal de configurações.

Normalmente, 2 ou 3 minutos são suficientes para o GroNode detectar um módulo recém-instalado. Uma vez detectado, aparecerá automaticamente no painel dos **Módulos**, na secção respectiva ao tipo de módulo.

Pode verificar a lista de módulos do mesmo tipo no lado direito do painel dos **Módulos**. Um GroNode suporta um máximo de 4 módulos de cada tipo.

Problemas ou falhas de comunicação

O GroNode foi desenhado para comunicar com os restantes módulos GroLab, através de sinais de radiofrequência. O **raio de ação** é de **25 metros (82 pés) em espaços interiores** e **100 metros (328 pés) em espaços abertos**, dependendo das condições do espaço.

Acesse o painel dos **Módulos** através do *software* GroLab e verifique se os seus módulos estão acessíveis. Caso não estejam acessíveis ou apresentem falhas/perdas de comunicação, é possível que tenha excedido a distância entre os seus módulos e o GroNode.

Dentro do painel dos **Módulos**, poderá encontrar um ícone de sinal sem fios, o qual indica se o módulo apresentado está ligado ao GroNode e a comunicar corretamente.

⚠ AVISO: Algumas paredes de suporte de carga e dispositivos eletrónicos podem causar interferência no sinal.

Se o problema persistir, faça o seguinte: coloque o módulo próximo ao GroNode e verifique o estado da comunicação através do *software* GroLab. Se a comunicação for restabelecida com sucesso, repita o processo a distâncias diferentes até identificar a distância máxima de comunicação.

Se o módulo não consegue estabelecer comunicação com o GroNode, tente percorrer os diferentes canais de comunicação até encontrar os seus módulos (deve aguardar 3 minutos entre cada mudança de canal).

Para alterar o canal de comunicação, aceda ao painel de **Configurações** (do GroNode) e, de seguida, à seção **Configurações Gerais**. Nesta seção, encontrará o campo de configuração relativo ao canal de comunicação. Clique no botão no canto inferior direito para ativar a edição e mudar o canal de comunicação para o canal desejado. Para aplicar as alterações, clique no botão verde no canto inferior direito.

Se, após seguir as etapas anteriores, os seus módulos ainda não aparecem no *software* GroLab, por favor, entre em contato com nosso suporte técnico.

Atualizações de firmware

Tenha em atenção que algumas atualizações de software, podem exigir a atualização do *firmware* dos seus módulos GroLab.

Todas as atualizações fornecem melhorias importantes, as quais asseguram o melhor desempenho do sistema.

No entanto, como o *firmware* do UserBot é compilado e aplicado pelo utilizador, é necessário que o utilizador esteja atento ao repositório de código do UserBot no GitHub. Caso sejam submetidas atualizações ao código, o utilizador deverá aplicar essas atualizações ao seu código. Será também necessário voltar a compilar o código, fazer *upload* do mesmo para o Arduino e, reaplicar as configurações de fábrica.

INSTR. DE SEGURANÇA ④

As seguintes diretrizes gerais de segurança são fornecidas para ajudar a garantir sua própria segurança pessoal e proteger o seu dispositivo contra possíveis danos.

•Não tente reparar ou desmontar o dispositivo. Para alguns dispositivos com bateria substituível pelo utilizador, leia e siga as instruções fornecidas pelo manual de instalação.
•Mantenha o dispositivo longe de radiadores e fontes de calor.
•Mantenha o dispositivo longe de temperaturas extremamente quentes ou frias para garantir que seja utilizado dentro da faixa de operação especificada.
•Não derrame alimentos ou líquidos no seu dispositivo.
•Antes de limpar seu dispositivo, desconecte-o da tomada elétrica. Limpe o seu dispositivo com um pano macio e seco. Não use líquidos.
•Se o seu dispositivo não funcionar normalmente - em particular, se houver sons ou odores incomuns - desconecte-o imediatamente e entre em contato com um revendedor autorizado ou com o centro de suporte da Open Grow.
•Para ajudar a evitar o risco potencial de choque elétrico, não conecte ou desconecte cabos, nem faça manutenção ou reconfiguração do seu dispositivo durante tempestades ou em períodos propícios à ocorrência de trovoada.
•Verifique a voltagem antes de ligar o dispositivo a uma tomada elétrica para garantir que a voltagem e a frequência necessárias correspondam à fonte de energia disponível.
•Além disso, certifique-se de que os seus módulos GroLab e dispositivos conectados aos mesmos, estejam classificados eletricamente para operar com a energia CA disponível na sua localização.
•Não ligue os cabos de energia a uma tomada elétrica se os cabos de energia estiverem danificados.
•Para evitar choque elétrico, ligue os cabos de energia em tomadas elétricas corretamente aterradas.
•Se usar um cabo extensor de alimentação, confirme que a amperagem total dos dispositivos ligados não excede a amperagem do cabo extensor.

GARANTIA ⑤

A Open Grow, LDA, garante ao consumidor que o produto está livre de defeitos de material e/ou fabrico. A responsabilidade da Open Grow, LDA, é limitada à reparação ou substituição de peças defeituosas. Antes de nos enviar qualquer peça defeituosa, por favor, entre em contacto com o nosso centro de suporte, de forma a verificar o procedimento a seguir.

Contacte o nosso centro de suporte através da página **opengrow.pt/support** ou através do email **support@opengrow.pt**.

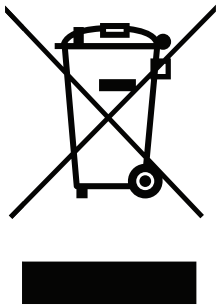
Todos os produtos da Open Grow, LDA, têm garantia de 2 anos, exceto consumíveis (sensores e/ou atuadores de qualquer tipo) e mediante correta utilização.

A reivindicação de garantia é intransferível e somente poderá ser exercidada pelo consumidor original. Para fazer cumprir a garantia, o consumidor deverá sempre fornecer a fatura de compra.

ISENÇÃO DE GARANTIA:

A aplicação da garantia é excluída se a avaria da peça ou peças defeituosas for resultado do uso inadequado e/ou negligente do produto. Entende-se como uso inadequado e/ou negligente, a qualquer uso diferente daquele para o qual o produto é destinado e/ou que é recomendado no manual de instruções, a não execução de operações de manutenção recomendadas no manual de instruções, a realização de operações que são diferentes daqueles mencionados e que comprometem a qualidade do produto, modificações que não são realizadas por técnicos autorizados e/ou com peças não originais ou não aprovadas pelo fabricante.

CONFORMIDADE ⑥



Este símbolo no produto ou embalagem significa que, de acordo com as leis e regulamentações locais, este produto não deve ser descartado no lixo doméstico, mas sim enviado para reciclagem. Por favor, uma vez que tenha atingido o fim de sua vida útil, leve-o para um ponto de recolha designado pelas autoridades locais, alguns aceitarão recolher os produtos gratuitamente. Ao reciclar o produto e sua embalagem dessa maneira, estará a ajudar a preservar o meio ambiente e a proteger a saúde humana.

Este símbolo no produto ou embalagem significa que este produto está em conformidade com os regulamentos RoHS da Diretiva do Parlamento e do Conselho Europeu sobre as Restrições do Uso de Determinadas Substâncias Perigosas em Equipamentos Elétricos e Eletrónicos (2011/65/EU).

Este símbolo no produto ou embalagem significa que este produto está em conformidade com as seguintes diretrizes e regulamentos:

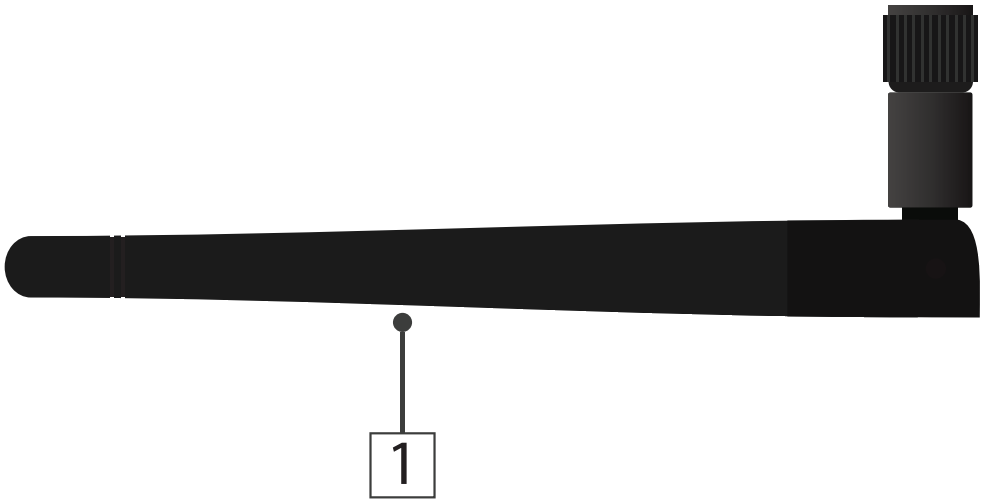
- (2014/53/UE) Diretiva Relativa aos Equipamentos de Rádio.
- (2011/65/EU) Diretiva RoHS.
- (2014/35/EU) Diretiva Relativa à Baixa Tensão.
- (2014/30/EU) Diretiva Relativa à Compatibilidade Eletromagnética.

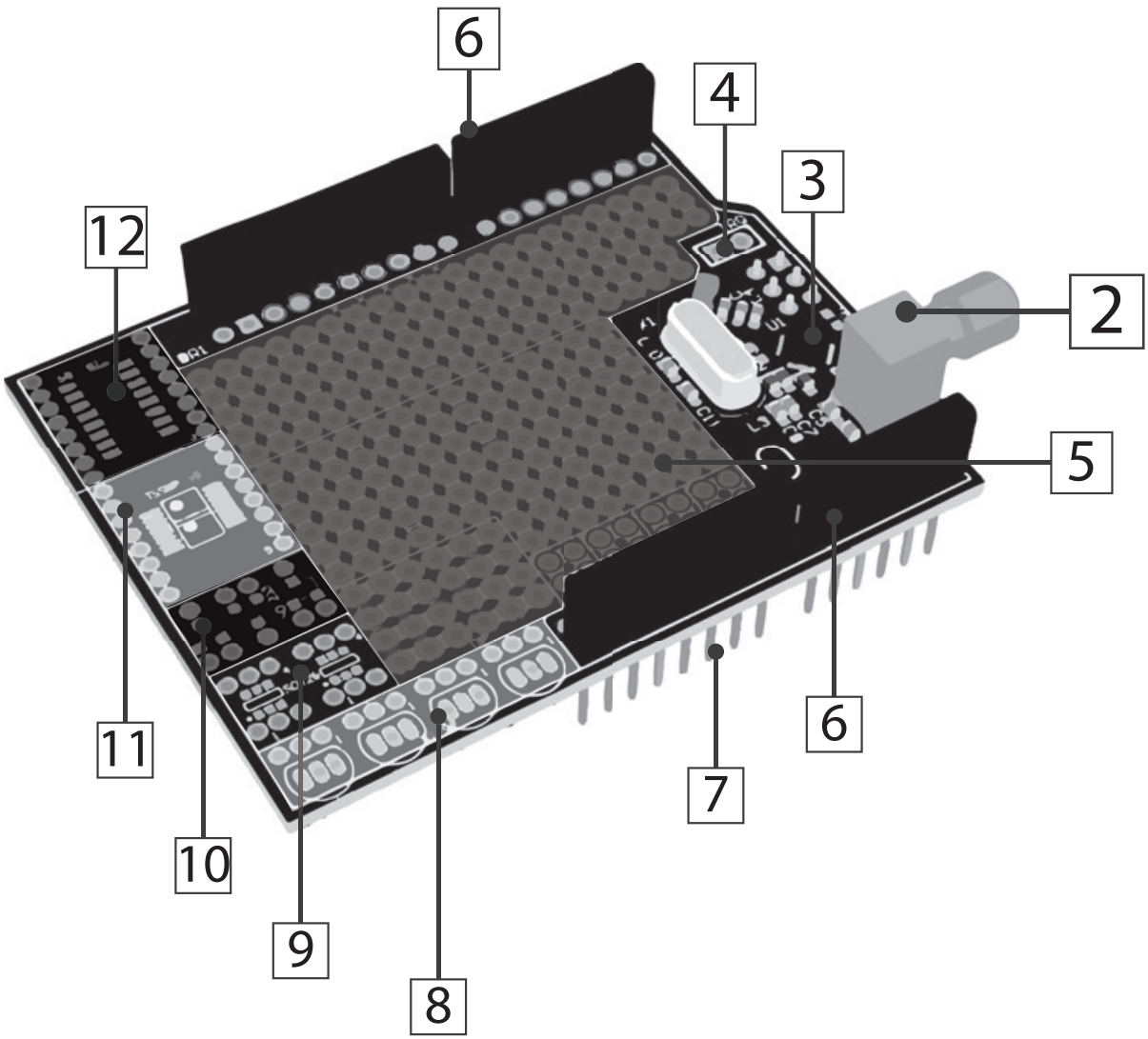
	Banda(s) de Frequência	Max. Potência de Saída (EIRP)
2.4 G	2.4 - 2.4835 GHz	100 mW

A Open Grow, LDA, reserva-se ao direito de atualizar e/ou modificar o conteúdo de seus produtos a qualquer momento, sem aviso prévio. Consulte os nossos Termos e Condições em www.opengrow.pt.

OPENGROW.

NOTAS


1


6
4
3
12
2
5
11
10
9
8
7
6

1 Antenna per comunicazioni RF

2 Presa per antenna di comunicazione RF

3 Modulo RF

4 Pin di interruzione RF

5 Scheda di sviluppo universale

6 Connettori rapidi femmina

7 Piedini di connessione Arduino

8 4 x TO-92

9 2 x SOT26

10 3 x SSOT223

11 2 x TSSOP-8 (=) 1 x TSSOP-16

12 SOIC16

TABELLA DELLE SPECIFICHE

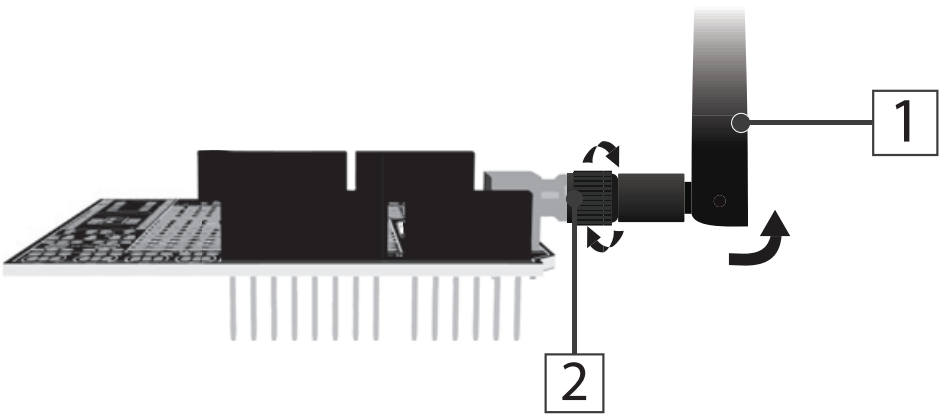
Versione hardware	H01
Dimensioni	68.6mm x 53.4mm
Esterno	Materiale: FR-4 Colore: Verde
Tensione di Funzion.	+3V3 VDC
Connessioni	RP-SMA femmina Connettori femmina rapidi (connessione/estensione Arduino)
Sockets SMD	1 x SOIC16 2 x TSSOP 8 (=) 1 x TSSOP 16 3 x SSOT223 2 x SOT26
Sockets TH	4 x TO-92
Consiglio di Sviluppo	DIP/1206/0805/0603
Spaziatura della Scheda di Sviluppo	2.54mm - 100mil
Include	Antenna
Comunicazione tra i Moduli	Radiofrequenza - 2.4GHz
Garanzia	Garanzia hardware limitata a 2 anni

INSTALLAZIONE ①

Connessioni di base

Avvitare l'antenna di comunicazione (1) al modulo attraverso l'ingresso filettato (2).

Per una migliore comunicazione, si raccomanda che l'antenna (1) sia perpendicolare al modulo e non ostruita.



Dopo aver posizionato l'antenna, UserBot *shield* è pronto per essere montato su Arduino.

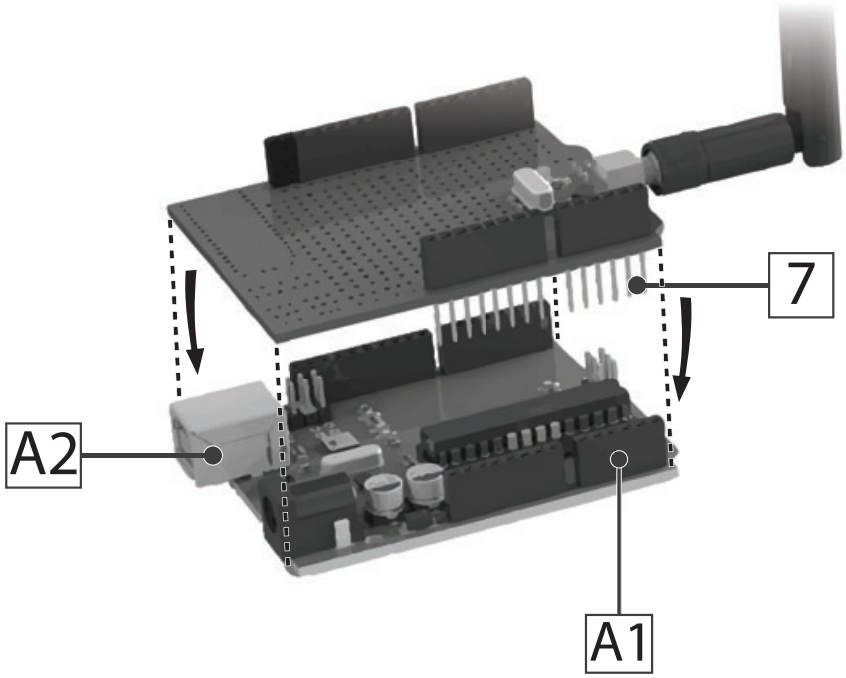
Connessione con Arduino

Prima di continuare con l'assemblaggio, verificare la compatibilità di Arduino con UserBot. Elenco dei modelli compatibili:

- **Arduino UNO;**
- **Arduino Duemilanove;**
- **Arduino Leonardo con connettori femmina;**
- **Genuino Zero.**

Prima di montare UserBot *shield* su Arduino, confermare che Arduino sia correttamente assemblato e pronto per l'uso (seguire il manuale di istruzioni Arduino ufficiale). Assicurarsi inoltre che Arduino non sia collegato all'alimentazione.

Posizionare l'Arduino su una superficie solida, lontano da aree soggette a inondazioni. Assicurarsi che i connettori rapidi femmina Arduino (A1) siano rivolti verso l'alto e i pin del connettore UserBot (7) rivolti verso il basso.



Inserire i pin del connettore UserBot (7) dall'alto verso il basso nei connettori femmina Arduino (A1) come mostrato nella figura precedente.

⚠ Evitare il contatto tra UserBot *shield* e l'ingresso di alimentazione USB di Arduino (A2).

Connessione periferica

Poiché UserBot è un'estensione per Arduino, significa che qualsiasi dispositivo o sensore in grado di interagire con Arduino è anche compatibile con UserBot *shield*.

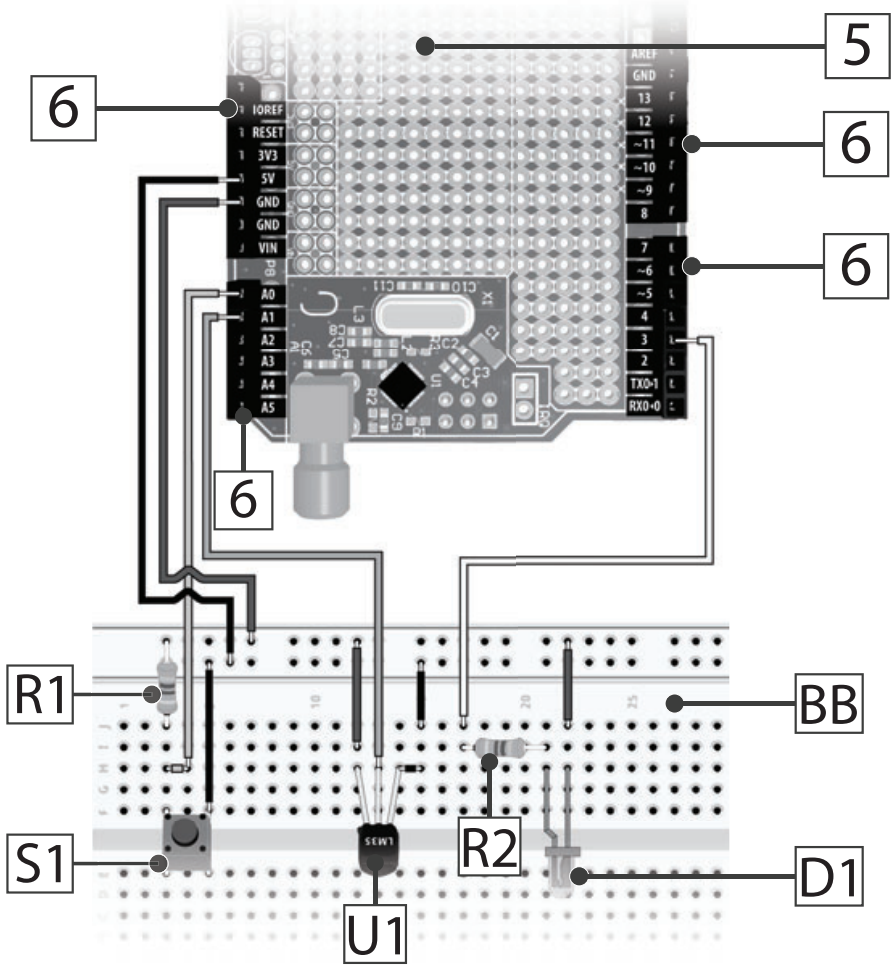
Pertanto, l'elenco di dispositivi e sensori compatibili è estremamente lungo ed è impossibile elencarli e spiegarne la connessione.

Tratteremo comunque due esempi: il primo mostra come accendere/campionare una manopola e un sensore di temperatura, oltre ad agire su un LED; Il secondo spiega come operare in un motore CC.

⚠ Prima di collegare qualsiasi sensore o dispositivo, assicurarsi che Arduino non sia collegato all'alimentazione. Inoltre, è consigliabile guidare su una superficie solida, lontano da aree soggette a inondazioni. Utilizzare strumenti adeguati per gestire i componenti elettronici.

Esempio 1

Collegamento di un pulsante, un sensore di temperatura e un LED.



5	Scheda di sviluppo universale
6	Connettori rapidi femmina
R1	Resistenza 10kΩ (pull down)
R2	Resistenza 240Ω per LED
S1	Pulsante a sfioramento SPST
U1	Sensore di temperatura LM35
D1	LED
BB	Scheda di sviluppo esterno

Il diagramma mostrato nella pagina precedente mostra la connessione di alcuni componenti UserBot *shield* tramite i connettori rapidi femmina (6).

Il pulsante a sfioramento SPST (S1) è campionato sul **pin analogico A0** di Arduino; il sensore di temperatura LM35 (U1) è campionato sul **pin analogico A1** di Arduino; il LED (D1) è controllato tramite il **pin digital 3** di Arduino. Per controllare la 'velocità/intensità' di un dispositivo, è necessario utilizzare un pin digitale PWM Arduino (ad esempio, pin 3 e 5).

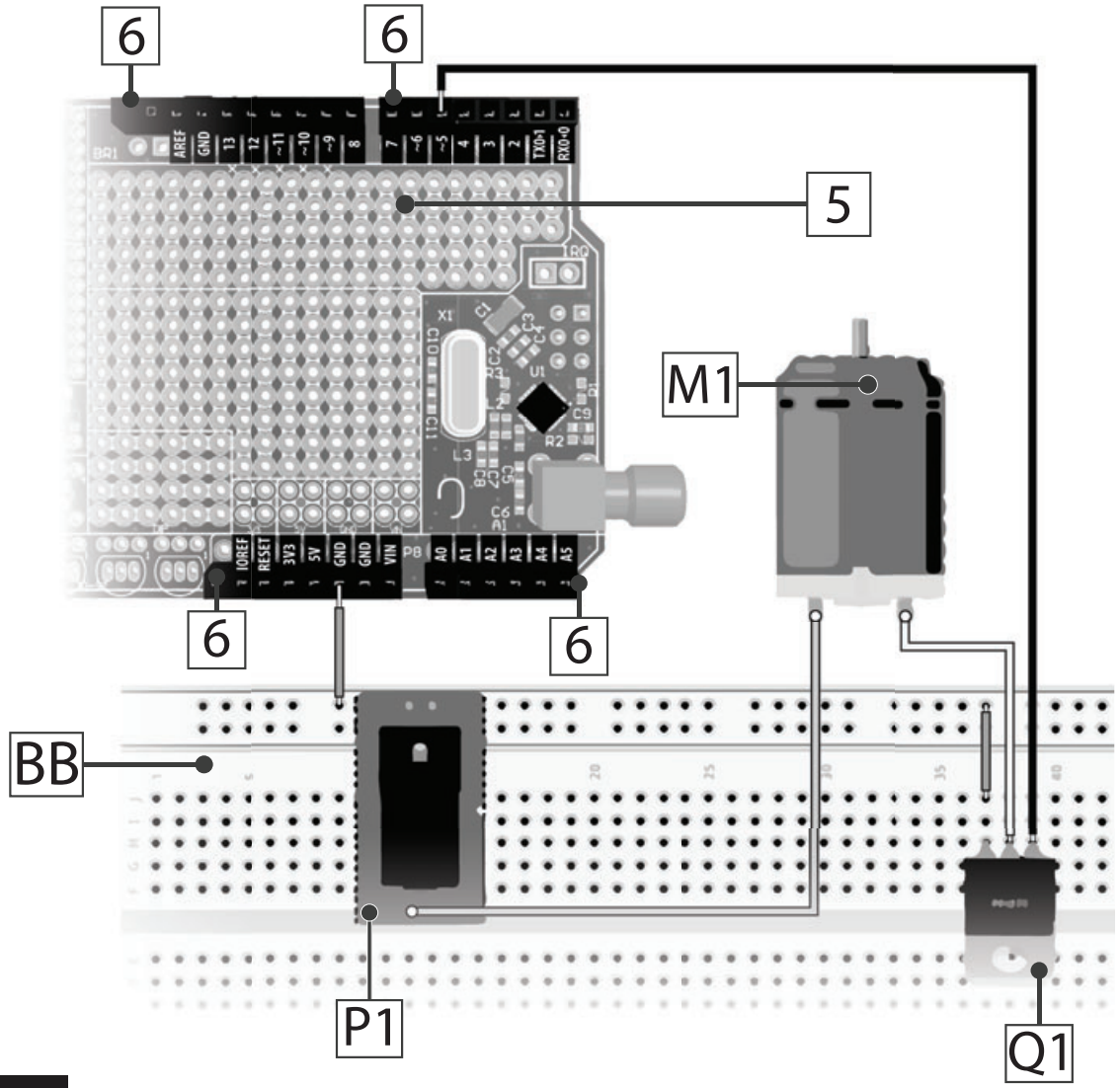
Utilizzare i **piedini +5V**, **+3V3** e **GND** di Arduino, per alimentare i sensori e/o i dispositivi. Controlla le restrizioni di alimentazione di questi pin nella documentazione ufficiale di Arduino.

UserBot *shield* offre una scheda di sviluppo integrata (5) che puoi usare liberamente. Tuttavia, per il diagramma, utilizziamo una scheda esterna (BB) per una migliore comprensione.

⚠ È fondamentale prestare attenzione alla polarità dei diversi componenti. Una connessione errata può danneggiare i componenti e/o Arduino/UserBot.

Esempio 2

Collegamento di un motore CC (pompa peristaltica, pompa dell'acqua...).



5 Scheda di sviluppo universale

6 Connettori rapidi femmina

M1 Motore a corrente continua (CC)

BB Scheda di sviluppo esterno

P1 Modulo di ingresso alimentazione 12VDC

Q1 MOSFET tipo N (ad es. IRLZ14)

Lo schema mostrato nella pagina precedente mostra l'uso di un MOSFET di tipo N (Q1) per controllare la tensione di alimentazione di un motore a corrente continua (M1).

Il motore CC (M1) è alimentato da 12VDC esterno (P1) e ha il controllo PWM tramite il pin **pin digital 5** di Arduino. Per controllare la "velocità/intensità" di un dispositivo, è necessario utilizzare un pin digitale PWM Arduino (ad esempio, pin 3 e 5).

Anche l'ingresso di alimentazione 12VDC (P1) e il MOSFET (Q1) devono essere collegati al **pin GND** di Arduino.

Utilizzare i connettori rapidi femmina di UserBot (6) per collegare l'ingresso di alimentazione 12VDC (P1) e il MOSFET (Q1) ai rispettivi pin Arduino.

UserBot *shield* offre una scheda di sviluppo integrata (5) che puoi usare liberamente. Tuttavia, per il diagramma, utilizziamo una piastra esterna (BB) per una migliore comprensione.

⚠ **È fondamentale prestare attenzione alla polarità dei diversi componenti. Una connessione errata può danneggiare i componenti e/o Arduino/UserBot.**

Prepara il codice sorgente

Dopo aver montato i sensori e i dispositivi, è tempo di preparare il codice sorgente per controllarli.

Open Grow fornisce il codice *firmware* di base in un formato open source. Il codice è disponibile per il *download* dalla seguente pagina GitHub:

github.com/OpenWeGrow/UserBot

In questo modo, la prima attività è scaricare il codice. Poiché esistono diversi modi per farlo (ad esempio, attraverso le applicazioni di amministrazione del repository o scaricando direttamente dal sito *web* GitHub), non forniremo dettagli su alcun metodo specifico. Tuttavia, se hai bisogno di aiuto, contatta il nostro supporto tecnico:

www.opengrow.pt @ support@opengrow.pt

NOTA: Gli esempi forniti in questo manuale si basano sul codice disponibile il 14 agosto 2019.

Poiché l'essenza di UserBot *shield* è l'integrazione di Arduino con il sistema GroLab, per preparare il codice, utilizzeremo l'IDE di Arduino. Se in qualsiasi momento si verificano problemi o ostacoli con l'IDE di Arduino, consultare la documentazione ufficiale di Arduino.

Dopo aver scaricato il codice di base, è necessario copiare/spostare i contenuti delle cartelle **'libraries'** ed **'examples'**, nelle rispettive cartelle (stesso nome) all'interno della cartella di installazione di Arduino sul PC.

Assicurati che la scheda Arduino selezionata nell'IDE sia corretta. Per fare ciò, apri l'IDE Arduino e vai su **Strumenti → Scheda** e seleziona la scheda Arduino corretta in base al tuo Arduino fisico.

Il prossimo passo è aprire il codice con l'IDE di Arduino andando su **File → Esempi → OpenGrow → UserBot** e facendo clic per aprirlo.

Prima di continuare con la modifica del codice, elencheremo la struttura del file e spieghiamo:

→ libraries
→ OpenGrow

• ComsTask.cpp • ComsTask.h	**Macchina a stati per gestire le attività di comunicazione.**
• CRC16.cpp • CRC16.h	**Calcolatore CRC di Tim W. Shilling.**
• EEPROM_Utils.cpp • EEPROM_Utils.h	**Funzioni di gestione NVM.**
• GroBot_Variables.h	**Definizioni di *input/output* di UserBot.**
• nRF24L01.h • RF24.cpp • RF24.h • RF24_config.h	***Driver* NRF24L01, di J. Coliz <maniacbug@y-mail.com> con le modifiche apportate da Open Grow.**
• OpenBus.cpp • OpenBus.h	**Manager dei comandi di comunicazione.**
• SerialTask.cpp • SerialTask.h	**Macchina a stati per gestire i comandi UART.**

→ examples
→ OpenGrow
→ UserBot

• SensorsTask.cpp • SensorsTask.h	**Una macchina a stati dedicata per il campionamento dei sensori e il controllo delle uscite.**
• UserBot.ino	**File Arduino (.ino) e il codice di base per eseguire il programma. Questo è il file di configurazione principale per UserBot. Tutti gli *input* e *output* devono essere configurati in questo file.**

→ examples
→ OpenGrow
→ UserBot-DCMotor

(Stessa struttura di file che: **examples → OpenGrow → UserBot**.)

Cambia il codice

Una volta preparato il codice di base, è necessario modificarlo in modo che corrisponda ai componenti assemblati.

In sintesi, **per la maggior parte dei casi (se non tutti), è necessario modificare o duplicare solo i seguenti file:**

→ examples
  → OpenGrow
    → UserBot
      - SensorsTask.cpp
      - SensorsTask.h
      - UserBot.ino

Poiché le modifiche al codice richieste dipendono dai componenti assemblati dall'utente e la varietà di combinazioni di sensori e dispositivi è infinita, non è possibile fornire la soluzione esatta per tutti i casi.

Tuttavia, presenteremo il codice necessario per i due diagrammi di esempio mostrati sopra.

Esempio 1

Collegamento di un pulsante, un sensore di temperatura e un LED.

Per interconnettere il codice con il pulsante, il sensore di temperatura e il LED, aprire **UserBot.ino** (**examples → OpenGrow → UserBot**) accedere alla ***funzione setup***. Questa ***funzione*** contiene la configurazione per tutti gli *input* e *output*. È necessario regolare il codice in modo che corrisponda all'elettronica assemblata. Per il diagramma di esempio, il codice deve essere il seguente:

CONFIGURAZIONE DI *INPUT*

```
inputs[INPUT_INDEX0].arduinoPin = A0;    //Button Pin
inputs[INPUT_INDEX1].arduinoPin = A1;    //LM35 Pin
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = BUTTON;
inputs[INPUT_INDEX1].type = DIG_TEMPERATURE;
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURAZIONE DI *OUTPUT*

```
outputs[OUTPUT_INDEX0].arduinoPin = 3;  //LED Pin
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = LED;
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

I diversi tipi di *input* e *output* sono dichiarati nel file **GroBot_Variables.h** (**libraries → Open-Grow**) sotto i commenti **'Possible Input Types'** e **'Possible Output Types'**.

Le modifiche nel primo file sono complete. Il prossimo passo è aprire il file **SensorsTask.cpp** (**examples → OpenGrow → UserBot**).

NOTA: UserBot supporta fino a 10 *inputs* e 10 *outputs*.

Dopo aver aperto il file **SensorsTask.cpp**, vai alla ***funzione SensorsTask*** e cerca il commento **'* IO Config *'.** Sotto il commento troverai un esempio di configurazione dei pin, questa configurazione deve essere adattata ai componenti assemblati. Per il diagramma nell'esempio 1, il codice dovrebbe essere il seguente:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ***********************
    /*Here you need to set your used pins as inputs or outputs*
    pinMode(inputs[INPUT_INDEX0].arduinoPin, INPUT);
    pinMode(inputs[INPUT_INDEX1].arduinoPin, INPUT);

    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Quindi vai alla ***funzione GoSensorsTask*** e osserva l'***switch***. Nel ***case GET_IOS***, è necessario aggiungere il codice per campionare le voci e aggiornarne i valori negli ***array inputs***. Per esempio:

```
if (digitalRead(inputs[INPUT_INDEX0].arduinoPin))
  inputs[INPUT_INDEX0].value = 255;
else
  inputs[INPUT_INDEX0].value = 0;
```

In caso di uscite, è necessario aggiungere il codice per agire su di esse, nel ***case ACT_ON_IOS***. Per esempio:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

Puoi aggiungere più ***cases*** in base alle tue esigenze. Per lo schema dell'esempio 1, il codice all'interno dello ***switch*** deve essere il seguente:


```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = GET_TEMP;
  break;
case GET_TEMP:
  //Sampling Temperature Sensor
  time = getAnalogRead(inputs[INPUT_INDEX1].arduinoPin);
  inputs[INPUT_INDEX1].value += calcTemp(time);
  inputs[INPUT_INDEX1].value = inputs[INPUT_INDEX1].value/2;
  snsState = GET_IOS;
  break;
case GET_IOS:
  //Polling button
  if(digitalRead(inputs[INPUT_INDEX0].arduinoPin))
    inputs[INPUT_INDEX0].value = 255;
  else
    inputs[INPUT_INDEX0].value = 0x00;

  snsState= ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on LED pin
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1))
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = GET_TEMP;
  break;
```

Si noti che abbiamo aggiunto più codice relativo alle prestazioni di *output*. Queste modifiche aggiungono il supporto per la variabile di **'speed'** (ad es. per il controllo dell'intensità del LED) e altre funzionalità implementate nel sistema GroLab (come il tempo di raffreddamento per i dispositivi sensibili).

NOTA: Gli *arrays* di *inputs* e *outputs* memorizzano i valori di campionamento del sensore, nonché lo stato dell'uscita. Questi *arrays* sono fondamentali per lo scambio di informazioni con il sistema GroLab.

Se è necessario modificare i ***cases*** di ***switch***, , è necessario modificare l'***enum*** che contiene i ***cases***. Questo ***enum*** è dichiarato nel file **SensorsTask.h** (**examples → OpenGrow → UserBot**).

Dopo aver completato tutti i passaggi precedenti, il codice sarà pronto per gestire i componenti assemblati nel diagramma dell'esempio 1. Il passaggio successivo consiste nel compilare e caricare il codice in Arduino.

Se hai bisogno di aiuto per compilare e caricare il codice, consulta la documentazione ufficiale di Arduino o contatta il centro di supporto Arduino.

Dopo aver caricato il codice nel tuo Arduino, c'è ancora un passaggio necessario per renderlo operativo e pronto per comunicare con il sistema GroLab: **configurare le impostazioni di fabbrica** (numero seriale e canale di comunicazione). Vedere le istruzioni alle **pagine 25 e 26**.

Esempio 2

Collegamento di un motore CC (pompa peristaltica, pompa dell'acqua...).

Per interconnettere il codice con il motore CC, aprire il file **UserBot.ino** (**examples → OpenGrow → UserBot-DCMotor**) e accedere alla ***funzione setup***. Questa ***funzione*** contiene la configurazione per tutti gli ingressi e le uscite. È necessario regolare il codice in modo che corrisponda all'elettronica assemblata. Per il diagramma di esempio, il codice deve essere il seguente:

CONFIGURAZIONE DI *INPUT*

```
inputs[INPUT_INDEX0].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX1].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX2].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX3].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX4].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX5].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX6].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX7].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX8].arduinoPin = 0;     //Unused Input
inputs[INPUT_INDEX9].arduinoPin = 0;     //Unused Input

inputs[INPUT_INDEX0].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX1].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX2].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX3].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX4].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX5].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX6].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX7].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX8].type = OPEN_DEFAULT;    //Unused Input
inputs[INPUT_INDEX9].type = OPEN_DEFAULT;    //Unused Input
```

CONFIGURAZIONE DI *OUTPUT*

```
outputs[OUTPUT_INDEX0].arduinoPin = 5;  //DC Motor as Perista
outputs[OUTPUT_INDEX1].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX2].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX3].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX4].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX5].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX6].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX7].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX8].arduinoPin = 0;  //Unused Output
outputs[OUTPUT_INDEX9].arduinoPin = 0;  //Unused Output

outputs[OUTPUT_INDEX0].type = PERISTALTIC_PUMP;  //DC Motor a
outputs[OUTPUT_INDEX1].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX2].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX3].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX4].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX5].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX6].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX7].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX8].type = OPEN_DEFAULT; //Unused Output
outputs[OUTPUT_INDEX9].type = OPEN_DEFAULT; //Unused Output
```

I diversi tipi di *input* e *output* sono dichiarati nel file **GroBot_Variables.h** (**libraries → Open-Grow**) sotto i commenti **'Possible Input Types'** e **'Possible Output Types'**.

Le modifiche nel primo file sono complete. Il prossimo passo è aprire il file **SensorsTask.cpp** (**examples → OpenGrow → UserBot-DCMotor**).

NOTA: UserBot supporta fino a 10 *inputs* e 10 *outputs*.

Dopo aver aperto il file **SensorsTask.cpp**, passare alla ***funzione SensorsTask*** cercare il commento **'* IO Config *'.** Sotto il commento troverai un esempio di configurazione dei pin, questa configurazione deve essere adattata ai componenti assemblati. Per il diagramma nell'esempio 2, il codice dovrebbe essere il seguente:

```
SensorsTask::SensorsTask(void)
{
    snsState = INIT_SENSORS;

    //************************  IO Config  ********************
    /*Here you need to set your used pins as inputs or outpu
    pinMode(outputs[OUTPUT_INDEX0].arduinoPin, OUTPUT);
}
```

Quindi vai alla ***funzione GoSensorsTask*** e osserva l'***switch***. Nel ***case ACT_ON_IOS***, è necessario aggiungere il codice per agire sugli *output*. Per esempio:

```
if (outputs[OUTPUT_INDEX0].value > 0)
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
else
  digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
```

NOTA: L'*array* delle *outputs* memorizza gli stati delle uscite. Questo *array* è fondamentale per lo scambio di informazioni con il sistema GroLab.

Puoi aggiungere più ***cases*** in base alle tue esigenze. Per lo schema nell'esempio 2, il codice all'interno dello ***switch*** deve essere il seguente:

```
case INIT_SENSORS:
  //Use this state in the machine to initialize any sensor you may need
  snsState = ACT_ON_IOS;
  break;
case ACT_ON_IOS:
  //Act on DC Motor
  if(outputs[OUTPUT_INDEX0].value>0)
  {
    if((millis() - ticksOut1) > (MILLIS_PER_MINUTE * minutes2BackOffOut1) )
    {
      if(outputs[OUTPUT_INDEX0].speed == 0)
        digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, HIGH);
      else
        analogWrite(outputs[OUTPUT_INDEX0].arduinoPin, outputs[OUTPUT_INDEX0].speed);
    }
  }
  else
  {
    digitalWrite(outputs[OUTPUT_INDEX0].arduinoPin, LOW);
  }
  snsState = INIT_SENSORS;
  break;
```

Si noti che abbiamo aggiunto più codice relativo alle prestazioni di *output*. Queste modifiche aggiungono il supporto per la variabile di **'speed'** e altre funzionalità implementate nel sistema GroLab (come il tempo di raffreddamento per i dispositivi sensibili).

Se è necessario modificare i ***cases*** di ***switch***, , è necessario modificare l'***enum*** che contiene i ***cases***. Questo ***enum*** è dichiarato nel file **SensorsTask.h** (**examples → OpenGrow → UserBot-DCMotor**).

Dopo aver completato tutti i passaggi precedenti, il codice sarà pronto per gestire i componenti assemblati nel diagramma dell'esempio 2. Il passaggio successivo consiste nel compilare e caricare il codice in Arduino.

Se hai bisogno di aiuto per compilare e caricare il codice, consulta la documentazione ufficiale di Arduino o contatta il centro di supporto Arduino.

Configurare Impost.di Fabbrica

Dopo aver caricato il codice nel tuo Arduino, c'è ancora un passaggio necessario per renderlo operativo e pronto per comunicare con il sistema GroLab: **configurare le impostazioni di fabbrica** (numero seriale e canale di comunicazione).

Assicurati che Arduino sia collegato al PC tramite USB e che la porta seriale non sia utilizzata da un'altra applicazione. Apri l'IDE di Arduino e seleziona la sua porta seriale. Accedi al **'Monitor seriale'** di Arduino e imposta la **velocità di trasmissione su 230400**. Invia il comando **'R'** con l'opzione **'Nessun fine riga'**. Attendere un momento fino a quando non viene visualizzata la risposta **RST** sul 'Monitor seriale'. Riavviare Arduino (interruzione di corrente o pulsante di ripristino, se disponibile).

Apri di nuovo il 'Monitor seriale' e invia il comando**'M'** con l'opzione **'Nessun fine riga'**. Quindi invia il numero del canale di comunicazione utilizzato da GroNode*, con l'opzione **'Entrambi (NL & CR)'**.

***I canali di comunicazione sono impostati da 1 a 5, con 5 come canale predefinito.**

Invia il comando **‘S’** con l'opzione **‘Nessun fine riga’**, inserisci il numero seriale desiderato (**invia un numero alla volta**) e quando raggiungi l'ultimo numero, invialo con l'opzione **Entrambi (NL & CR).** Il numero seriale deve essere composto da 10 numeri e iniziare con il numero '5', ad esempio '5123456789'.*

Per verificare che la configurazione sia stata applicata correttamente, invia il comando **‘D’** con l'opzione **‘Nessun fine riga’**. Dovresti vedere, attraverso il ‘Monitor seriale’, qualcosa di simile alla seguente immagine:

```
UserBot! Ready
N:       New Module
S:       5123456789
MID:     2
SAdd:    FF
FW:      1.1.0.8
```

Dopo aver completato tutti i passaggi precedenti, Arduino sarà completamente configurato e pronto per comunicare con GroNode tramite UserBot *shield*.

***Il numero di serie "5999999999" è riservato come numero predefinito e non deve essere utilizzato.**

COMUNICAZIONI ②

Connessione con GroNode

Assicurarsi che GroNode sia installato correttamente e accessibile tramite il *software* GroLab. In caso contrario, seguire le istruzioni fornite nel manuale di GroNode.

Dopo aver completato tutte le fasi di installazione della protezione UserBot, aprire il *software* GroLab e connettersi a GroNode. Dopo la connessione, è necessario accedere al pannello **Moduli** tramite il menu di configurazione principale.

Normalmente, 2 o 3 minuti sono sufficienti per GroNode per rilevare un modulo appena installato. Una volta rilevato, apparirà automaticamente nel pannello **Moduli** nella sezione corrispondente al tipo di modulo.

Puoi controllare l'elenco dei moduli dello stesso tipo sul lato destro del pannello **Moduli**. Un GroNode supporta un massimo di 4 moduli per ogni tipo.

Problemi o errori di comunicazione

GroNode è progettato per comunicare con gli altri moduli mediante segnali a radiofrequenza. Il **raggio d'azione massimo** è di **25 metri (82 piedi) in interni** e **100 metri (328 piedi) in campo aperto** a seconda delle condizioni dello spazio.

Accedi al pannello **Moduli** tramite il *software* GroLab e assicurati che i tuoi moduli siano accessibili. Se non sono accessibili o presentano guasti/perdite di comunicazione, è possibile che sia stata superata la distanza tra i moduli e GroNode.

All'interno del pannello **Moduli**, è possibile trovare un'icona di segnale *wireless* che indica se il modulo selezionato è collegato a GroNode e comunica correttamente.

⚠ ATTENZIONE: Alcune pareti di carico e dispositivi elettronici possono interferire con il segnale.

Se il problema persiste, procedere come segue: posizionare il modulo accanto a GroNode e verificare lo stato della comunicazione tramite il *software* GroLab. Se la comunicazione viene ripristinata correttamente, ripetere il processo a diverse distanze fino a quando non viene identificata la massima distanza di comunicazione.

Se il modulo non è in grado di comunicare con GroNode, provare a navigare attraverso i diversi canali di comunicazione fino a trovare i moduli (è necessario attendere 3 minuti tra ogni cambio di canale).

Per cambiare il canale di comunicazione, vai al pannello **Impostazioni** (di GroNode) e quindi alla sezione **Impostazioni Generali**. In questa sezione troverai il campo di configurazione per il canale di comunicazione. Fare clic sul pulsante nell'angolo in basso a destra per attivare la modifica e cambiare il canale di comunicazione nel canale desiderato. Per applicare le modifiche, fai clic sul pulsante verde nell'angolo in basso a destra.

Se, dopo aver seguito i passaggi precedenti, i moduli non vengono ancora visualizzati nel *software* GroLab, contattare il nostro supporto tecnico.

AGGIORNAMENTI ③

Aggiornamento del firmware

Si noti che alcuni aggiornamenti *software* potrebbero richiedere l'aggiornamento del *firmware* dei moduli GroLab.

Tutti gli aggiornamenti offrono importanti miglioramenti che garantiscono le migliori prestazioni del sistema.

Tuttavia, poiché il *firmware* UserBot viene compilato e applicato dall'utente, è necessario seguire il repository di codici UserBot su GitHub. Se vengono inviati aggiornamenti del codice, è necessario applicare tali aggiornamenti al proprio codice. È inoltre necessario ricompilare il codice, caricarlo su Arduino e riapplicare le impostazioni di fabbrica.

ISTRUZIONI DI SICUREZZA ④

Le seguenti linee guida generali sulla sicurezza sono fornite per aiutare a garantire la propria sicurezza personale e proteggere il dispositivo da possibili danni.

•Non tentare di riparare o smontare il dispositivo. Per alcuni dispositivi con una batteria sostituibile dall'utente, leggere e seguire le istruzioni fornite nel manuale di installazione.
•Tenere il dispositivo lontano da radiatori e fonti di calore.
•Tenere l'apparecchiatura lontano da temperature estremamente alte o basse per assicurarsi che venga utilizzata entro l'intervallo operativo specificato.
•Non versare cibo o liquidi sul dispositivo.
•Prima di pulire il dispositivo, scollegarlo dalla presa di corrente. Pulisci il dispositivo con un panno morbido e asciutto. Non usare liquidi.
•Se il dispositivo non funziona normalmente, in particolare in presenza di suoni o odori insoliti, scollegarlo immediatamente e contattare un rivenditore autorizzato o il centro di assistenza Open Grow.
•Per aiutare a prevenire il potenziale rischio di scosse elettriche, non collegare o scollegare i cavi, né mantenere o riconfigurare il dispositivo durante i temporali o durante i periodi di fulmini.
•Controllare la tensione prima di collegare il dispositivo a una presa elettrica per assicurarsi che la tensione e la frequenza richieste corrispondano alla fonte di alimentazione disponibile.
•Inoltre, assicurarsi che i moduli GroLab e i dispositivi ad essi collegati siano classificati elettricamente per funzionare con l'alimentazione disponibile nella propria posizione.
•Non collegare i cavi di alimentazione a una presa se i cavi di alimentazione sono danneggiati.
•Per evitare scosse elettriche, collegare i cavi di alimentazione a prese elettriche dotate di messa a terra.
•Se si utilizza un cavo di prolunga, assicurarsi che l'amperaggio totale dei dispositivi collegati al cavo di prolunga non superi l'amperaggio del cavo di prolunga.

GARANZIA ⑤

Open Grow, LDA, garantisce al consumatore che il prodotto è privo di difetti nei materiali e/o nella fabbricazione. La responsabilità di Open Grow, LDA, è limitata alla riparazione o alla sostituzione di parti difettose. Prima di inviarci parti difettose, contattare il nostro centro di supporto per verificare la procedura.

Puoi trovare il nostro centro assistenza all'indirizzo **opengrow.pt/support** o inviare un'e-mail a **support@opengrow.pt**.

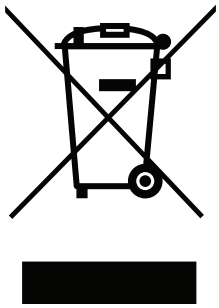
Tutti i prodotti Open Grow, LDA, hanno una garanzia di 2 anni, ad eccezione dei materiali di consumo (sensori e/o attuatori di qualsiasi tipo), se utilizzati correttamente.

La domanda di domanda di garanzia non è trasferibile e può essere effettuata solo dal consumatore originale. Per rendere effettiva la garanzia, il consumatore deve sempre fornire la fattura di acquisto.

ESCLUSIONE DELLA GARANZIA:

L'applicazione della garanzia è esclusa nel caso in cui il guasto delle parti difettose derivi da un uso improprio e/o negligente del prodotto. L'uso improprio e/o negligente deve essere inteso come qualsiasi uso diverso dalla natura del prodotto stesso e/o da quello raccomandato nel manuale di istruzioni, non eseguendo le operazioni di manutenzione raccomandate nel manuale di istruzioni o eseguendo attività diverse da quelle sopra menzionate. e che compromettono la qualità del prodotto, apportano modifiche al di fuori delle officine autorizzate e/o con parti non originali o non approvate.

CONFORMITÀ ⑥



Questo simbolo sul prodotto o sulla confezione indica che, in conformità con le leggi e le normative locali, questo prodotto non deve essere smaltito insieme ai rifiuti domestici, ma deve essere inviato per il riciclaggio. Si prega di portarlo in un punto di raccolta designato dalle autorità locali dopo aver raggiunto la fine della sua vita utile, alcuni accetteranno prodotti gratuitamente. Riciclando il prodotto e il suo imballaggio in questo modo, contribuisci a preservare l'ambiente e proteggere la salute umana.

Questo simbolo sul prodotto o sulla confezione indica che questo prodotto è conforme alle normative RoHS della Direttiva del Parlamento Europeo e del Consiglio sulle Restrizioni all'Uso di Determinate Sostanze Pericolose nelle Apparecchiature Elettriche ed Elettroniche (2011/65/UE).

Questo simbolo sul prodotto o sulla confezione indica che questo prodotto è conforme alle seguenti direttive e normative:

- (2014/53/EU) Direttiva Sulle Apparecchiature Radio (RED).
- (2011/65/EU) Direttiva RoHS.
- (2014/35/EU) Direttiva Sulla Bassa Tensione (LVD).
- (2014/30/EU) Direttiva Sulla Compatibilità Elettromagnetica (CEM).

	Banda(e) di frequenza	Potenza massima in uscita (EIRP)
2.4 G	2.4 - 2.4835 GHz	100 mW

Open Grow, LDA, si riserva il diritto di aggiornare e/o modificare il contenuto dei suoi prodotti in qualsiasi momento senza preavviso. Consulta i nostri Termini e Condizioni su www.opengrow.pt.

OPENGROW.

ISCRIZIONI

EN Specifications are subject to change without notice. GroLab is a registered trademark of Open Grow, LDA. Other brands and product names are trademarks or registered trademarks of their respective holders.
No part of the specifications may be reproduced in any form or by any means or used to make any derivative such as translation, transformation, or adaptation without permission from Open Grow, LDA.

ES Las especificaciones están sujetas a cambios sin previo aviso. GroLab es una marca registrada de Open Grow, LDA. Otras marcas y nombres de productos son marcas comerciales o marcas comerciales registradas de sus respectivos propietarios.
Ninguna parte de las especificaciones se puede reproducir de ninguna forma ni por ningún medio, ni se puede utilizar para obtener ningún derivado, como traducción, transformación o adaptación sin el permiso de Open Grow, LDA.

FR Les spécifications sont sujettes à changement sans préavis. GroLab est une marque déposée de Open Grow, LDA. Les autres marques et noms de produits sont des marques commerciales ou des marques déposées de leurs propriétaires respectifs.
Aucune partie des spécifications ne peut être reproduite sous quelque forme ou par quelque moyen que ce soit, ni être utilisée pour obtenir des dérivés tels que la traduction, la transformation ou l'adaptation sans l'autorisation de Open Grow, LDA.

PT As especificações estão sujeitas a mudanças sem aviso prévio. GroLab é uma marca registrada da Open Grow, LDA. Outras marcas e nomes de produtos são marcas comerciais ou marcas registradas dos seus respectivos proprietários.
Nenhuma parte das especificações poderá ser reproduzida de qualquer forma ou por qualquer meio, nem pode ser usada para obter qualquer derivado, como tradução, transformação ou adaptação sem a permissão da Open Grow, LDA.

IT Le specifiche sono soggette a modifiche senza preavviso. GroLab è un marchio registrato di Open Grow, LDA. Altri marchi e nomi di prodotti sono marchi o marchi registrati dei rispettivi proprietari.
Nessuna parte delle specifiche può essere riprodotta in qualsiasi forma o con qualsiasi mezzo, né può essere utilizzata per ottenere derivati, come traduzione, trasformazione o adattamento senza l'autorizzazione di Open Grow, LDA.

Open Grow, LDA

Edifício Expobeiras
Prq. Industrial de Coimbrões
3500-618 Viseu
Portugal

(+351) 232 458 475

info@opengrow.pt
support@opengrow.pt

www.opengrow.pt
www.opengrow.pt/support