

OpenSprinkler Bee (OSBee) Arduino Shield v1.0 User Manual

(Updated May 28, 2014)

Contents

Introduction	2
Hardware Interface	3
Hardware Setup	3
1. Attach OSBee Shield to Arduino	3
2. Select the Power Source	3
3. Wiring Sprinkler Valves	4
Software Setup.....	5
1. Pin Assignment.....	5
2. Arduino Library and Demo Programs	5
3. Using OSBee Shield with other Arduino Shields	6
Specifications	6
Terms and Conditions	6
Open-Source Links	6

Introduction

OpenSprinkler Bee (OSBee) Arduino shield is an open-source Arduino shield for switching battery-operated, latching sprinkler valves. It's an easy and low-cost solution to transform your Arduino into a flexible sprinkler controller with 4 independent zones, for garden and lawn watering, plant and flower irrigation, and other watering applications. By stacking additional Arduino shields, you can enable sensing, web-based control, or wireless control as well.

The OSBee Arduino Shield includes one assembled and tested circuit board. Arduino is **NOT** included.



To get started, you will also need the following, which are **NOT** included in the kit and have to be purchased separately.

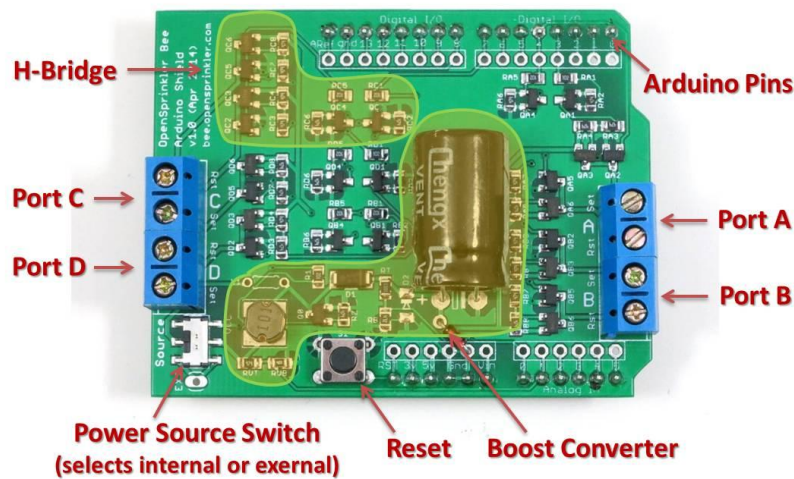
- An Arduino (any version)
- Up to 4 battery-operated, latching valves (**note: DC latching valves only, not AC!**). Examples:
 - [Orbit 58874N Yard Watering Valve](#) (24V, garden-hose thread, or GHT)
 - [Orbit 57861 battery-operated solenoid](#) for inline valves (17V, inline NPT)
 - [Hunter 9V DC solenoid](#) (9V, inline NPT)

*Battery-operated, latching DC valves are different from AC valves in that they usually **have special plugs, or have two wires that are colored differently (i.e has polarity)**. Each valve uses a latching solenoid and only draws power when it opens or closes. No power is drawn when it remains in the same state. The latching solenoid is made of a single coil: apply a positive voltage is on the coil, the valve opens; reverse the polarity of the voltage, the valve closes. A high impulse current (up to a couple amperes) is usually required to reliably open or close the valve.*



Hardware Interface

The image below shows a close-up image of the OSBee Shield and marks the built-in components.



Hardware Setup

1. Attach OSBee Shield to Arduino

WARNING: before plugging in the shield for the first time, please program the Arduino with any given OSBee Shield example. This can help prevent the pre-existing program running on Arduino to potentially cause shoot-through issues on the H-Bridge.

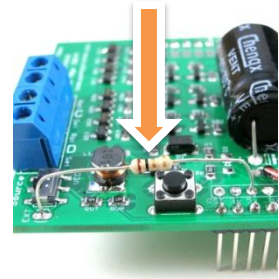
Insert the OSBee Shield into the Arduino. If you have other Arduino shields you'd like to stack, such as a WiFi shield or an Ethernet shield, keep OSBee Shield on the top layer. You need to find out the pins reserved by the other shields (for example, the Ethernet shield reserves pin 10 for Chip Select), and leave the corresponding pins out on the OSBee Shield. Details are explained later in this document.

2. Select the Power Source

The power source switch (lower-left corner) is used to select between Arduino's VCC pin (internal) or an external power source, such as a battery. When using an external source, solder a wire from the 'EXT' pin to the source's positive lead, and any 'GND' pin to the source's negative lead (i.e. share the ground). **If your external source is also used to power the Arduino, it's recommended that you connect a 1ohm resistor (included in the kit) in series on the EXT wire to limit the current draw.**

The internal source (VCC) is useful for testing, but keep in mind that some Arduinos (particularly the Pro Mini) have built-in fuse that prevent drawing high current from the VCC pin. During operation, the boost converter on OSBee Shield needs to momentarily draw up to several hundred milliamps of current. To

avoid triggering the fuse, using an external source is recommended. Also, when using the internal source, if you encounter any trouble uploading a program to Arduino (such as an *Inappropriate ioctl for device* error), you can solder the included 10hm resistor between EXT and the +5V pin, then set the power source selector to EXT. See the picture on the right. This will limit the current draw from the VCC pin.

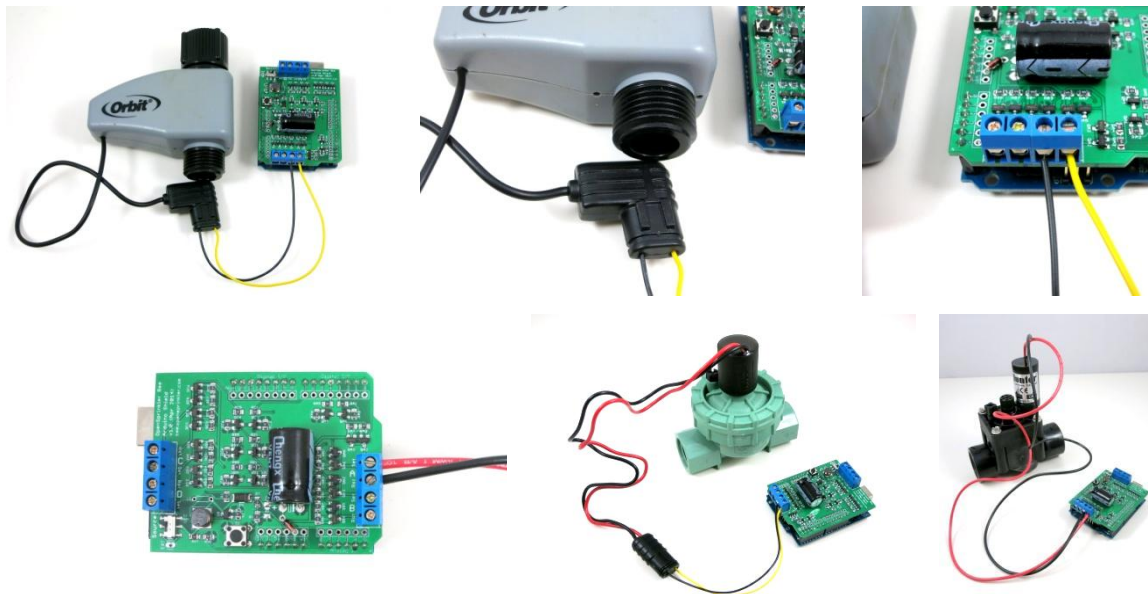


The boost converter can work with any input voltage between 1V DC to 6V DC. Higher input voltage results in faster boosting time but also higher current draw, and vice versa. The current draw / boosting time can also be adjusted in software by using the **OSBee::setDutyCycle()** function – smaller duty cycle reduces the current draw but also slows down the boosting time.

3. Wiring Sprinkler Valves

OSBee Shield supports four independent zones. These are marked ports A, B, C and D. Each port has two pins named **Set** and **Rst (Reset)**. Connect your sprinkler valve's **positive** wire (red colored) to the Set pin, and **negative** (black colored) to the Rst pin. If you accidentally reversed the polarity, you won't damage anything – it will just reverse the logic (i.e. 'opening' the valve actually closes it etc.).

If your sprinkler valve comes with a special plug, such as the Orbit 58874N valve, you can connect it as follow: the top pin on the plug is usually the positive wire, and the bottom pin is the negative wire. You can cut two pieces of wires (22 to 24 AWG recommended), strip the wires; then insert one end to the plug, the other end to the screw terminals on OSBee Shield, as shown in the picture on below.




Software Setup

1. Pin Assignment

OSBee Shield makes use of the following pins on the Arduino:

- PortA: **D3** (set), **D4** (rst); PortB: **D5, D6**; PortC: **D7, D8**; PortD: **D10, A2** (i.e. D16).
- **D9**: boost converter PWM pin.
- **A0**: boost converter feedback pin.
- **A1**: battery voltage feedback pin.



Precaution: make sure you **NEVER** set the pair of pins on the same port high at the same time. For example, setting D3 and D4 high at the same time can cause damage to the Port A H-Bridge.

2. Arduino Library and Demo Programs

An Arduino library for OSBee Shield is provided, with several examples to show how to use the library. One example is a simple testing program that opens each valve for 5 seconds, one after another. The second is a program-based scheduling algorithm – it allows the user to use Serial input to set multiple sprinkler programs. The third is a web-based program – it uses the standard Arduino Ethernet shield to create a simple web interface, and allows the user to click buttons on the webpage to open / close valves. More full-featured programs will be added in the future. Here is a brief summary of the library:

- Include the library by: `#include<OSBee.h>`
- Define an OSBee object such as: `OSBee osb;`
- In the Arduino setup() function, call: `osb.begin();`
- To open a valve, such as Port A, use either `osb.open(0);` or `osb.open('A');`
- To close a valve, such as Port B, use either `osb.close(1);` or `osb.close('B');`
- To read the external source voltage, use `osb.getBattVoltage();`

There are also several parameters you can adjust (optional):

- Set boost voltage (e.g. to 16.0V): `osb.setVoltage(16.0);`
- Set pulse length (e.g. to 25ms): `osb.setPulseLength(25);`
- Set PWM duty cycle (e.g. to 20%): `osb.setDutyCycle(20);`

Generally, the boost voltage is between 9V to 22V, the pulse length between 25ms to 200ms, and the PWM duty cycle between 10 to 50. The pulse length controls how long the H-Bridge will open and should be kept as short as possible. The duty cycle affects the current draw and boosting time. If your valve cannot reliably open / close, first consider increasing the boost voltage to 22.0; if it still doesn't work, gradually increase the pulse length. If the boost converter draws too much current that triggers a reset of the Arduino, consider reducing the duty cycle.

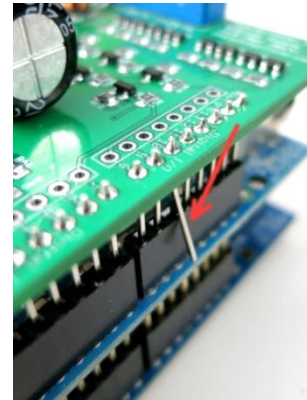
If you are not sure about the correct voltage and timing, don't worry about it. Because the circuit builds up charges into the on-board capacitor and supplies the voltage to your valve from there, you won't

damage your valve even if you used a voltage higher than necessary, or timing longer than necessary. There is no persistent high voltage on the circuit, and the capacitor can only store a limited amount of charge, so it's unlikely you will damage your valve.

Warning: leaving the valve open unintentionally and not being able to close it can lead to water waste and even worse, flood damage. For best practice, periodically perform a 'close' operation on all valves to make sure they are off. Also, before opening a valve, you can use **getBattVoltage** to check the battery voltage and make sure the battery is in good condition to close the valve later.

3. Using OSBee Shield with other Arduino Shields

It's possible to use OSBee Shield with other Arduino Shields, such as the Ethernet Shield, WiFi Shield etc. It's important to find out what pins the other shields require, and avoid any pin conflict. For example, the Ethernet shield uses D10 as Chip Select. Because D10 is also used by OSBee Shield for Port D Set pin, you have to leave this pin out on the OSBee Shield, which will disable Port D functionality. To leave the pin out, you can either cut this pin on OSBee Shield, or bend the pin so that it's not physically inserted into the female pin headers (see picture on the right). You can also cut the PCB trace of D10 (but this will result in permanent change).



Specifications

- **Source Voltage:** 1.8V DC to 6V DC (*selectable from either internal VCC or external*)
- **Boost Voltage:** up to 24V DC.
- **Output Current:** > 10A impulse; > 3A continuous.
- **Number of Zones:** 4 independent.
- **Size:** 70mm x 55mm x 28mm (2.75" x 2.2" x 1.1")
- **Weight:** 28g (1oz) w/o Arduino

Terms and Conditions

OpenSprinkler Bee (OSBee) is an open-source project. The hardware design and software code are made publicly available under the [Creative Commons Attribution-ShareAlike \(CC BY-SA\) 3.0](#) license. The product is open-source for educational purpose. The hardware and software are provided as is. We (Rayshobby LLC) are not responsible for any damage or accident that may occur due to either hardware or software error, or during the assembly, use, and modification of OSBee Shield.

Open-Source Links

- [OSBee Shield Github Repository](#)
- [OSBee homepage](#)