

---

## python-broadlink

A Python module and CLI for controlling Broadlink devices locally. The following devices are supported:

- **Universal remotes:** RM home, RM mini 3, RM plus, RM pro, RM pro+, RM4 mini, RM4 pro, RM4C mini, RM4S, RM4 TV mate
- **Smart plugs:** SP mini, SP mini 3, SP mini+, SP1, SP2, SP2-BR, SP2-CL, SP2-IN, SP2-UK, SP3, SP3-EU, SP3S-EU, SP3S-US, SP4L-AU, SP4L-EU, SP4L-UK, SP4M, SP4M-US, Ankuoo NEO, Ankuoo NEO PRO, Efergy Ego, BG AHC/U-01
- **Switches:** MCB1, SC1, SCB1E, SCB2
- **Outlets:** BG 800, BG 900
- **Power strips:** MP1-1K3S2U, MP1-1K4S, MP2
- **Environment sensors:** A1
- **Alarm kits:** S1C, S2KIT
- **Light bulbs:** LB1, LB26 R1, LB27 R1, SB800TD
- **Curtain motors:** Dooya DT360E-45/20
- **Thermostats:** Hysen HY02B05H
- **Hubs:** S3

### Installation

Use pip3 to install the latest version of this module.

```
1 pip3 install broadlink
```

### Basic functions

First, open Python 3 and import this module.

```
1 python3
```

```
1 import broadlink
```

Now let's try some functions...

### Setup

In order to control the device, you need to connect it to your local network. If you have already configured the device with the Broadlink app, this step is not necessary.

---

1. Put the device into AP Mode.

- Long press the reset button until the blue LED is blinking quickly.
- Long press again until blue LED is blinking slowly.
- Manually connect to the WiFi SSID named BroadlinkProv.

2. Connect the device to your local network with the setup function.

```
1 broadlink.setup('myssid', 'mynetworkpass', 3)
```

Security mode options are (0 = none, 1 = WEP, 2 = WPA1, 3 = WPA2, 4 = WPA1/2)

**Advanced options** You may need to specify a broadcast address if setup is not working.

```
1 broadlink.setup('myssid', 'mynetworkpass', 3, ip_address='192.168.0.255')
   )
```

## Discovery

Use this function to discover devices:

```
1 devices = broadlink.discover()
```

**Advanced options** You may need to specify `local_ip_address` or `discover_ip_address` if discovery does not return any devices.

Using the IP address of your local machine:

```
1 devices = broadlink.discover(local_ip_address='192.168.0.100')
```

Using the broadcast address of your subnet:

```
1 devices = broadlink.discover(discover_ip_address='192.168.0.255')
```

If the device is locked, it may not be discoverable with broadcast. In such cases, you can use the unicast version `broadlink.hello()` for direct discovery:

```
1 device = broadlink.hello('192.168.0.16')
```

If you are a performance freak, use `broadlink.xdiscover()` to create devices instantly:

```
1 for device in broadlink.xdiscover():
2     print(device) # Example action. Do whatever you want here.
```

---

## Authentication

After discovering the device, call the `auth()` method to obtain the authentication key required for further communication:

```
1 device.auth()
```

The next steps depend on the type of device you want to control.

## Universal remotes

### Learning IR codes

Learning IR codes takes place in three steps.

1. Enter learning mode:

```
1 device.enter_learning()
```

2. When the LED blinks, point the remote at the Broadlink device and press the button you want to learn.
3. Get the IR packet.

```
1 packet = device.check_data()
```

### Learning RF codes

Learning RF codes takes place in six steps.

1. Sweep the frequency:

```
1 device.sweep_frequency()
```

2. When the LED blinks, point the remote at the Broadlink device for the first time and long press the button you want to learn.
3. Check if the frequency was successfully identified:

```
1 ok = device.check_frequency()  
2 if ok:  
3     print('Frequency found!')
```

4. Enter learning mode:

---

```
1 device.find_rf_packet()
```

5. When the LED blinks, point the remote at the Broadlink device for the second time and short press the button you want to learn.
6. Get the RF packet:

```
1 packet = device.check_data()
```

**Notes** Universal remotes with product id 0x2712 use the same method for learning IR and RF codes. They don't need to sweep frequency. Just call `device.enter_learning()` and `device.check_data()`.

### Canceling learning

You can exit the learning mode in the middle of the process by calling this method:

```
1 device.cancel_sweep_frequency()
```

### Sending IR/RF packets

```
1 device.send_data(packet)
```

### Fetching sensor data

```
1 data = device.check_sensors()
```

### Switches

#### Setting power state

```
1 device.set_power(True)
2 device.set_power(False)
```

#### Checking power state

```
1 state = device.check_power()
```

---

## Checking energy consumption

```
1 state = device.get_energy()
```

## Power strips

### Setting power state

```
1 device.set_power(1, True) # Example socket. It could be 2 or 3.  
2 device.set_power(1, False)
```

### Checking power state

```
1 state = device.check_power()
```

## Light bulbs

### Fetching data

```
1 state = device.get_state()
```

### Setting state attributes

```
1 devices[0].set_state(pwr=0)  
2 devices[0].set_state(pwr=1)  
3 devices[0].set_state(brightness=75)  
4 devices[0].set_state(bulb_colormode=0)  
5 devices[0].set_state(blue=255)  
6 devices[0].set_state(red=0)  
7 devices[0].set_state(green=128)  
8 devices[0].set_state(bulb_colormode=1)
```

## Environment sensors

### Fetching sensor data

```
1 data = device.check_sensors()
```

---

## Hubs

### Discovering subdevices

```
1 device.get_subdevices()
```

### Fetching data

Use the DID obtained from `get_subdevices()` for the input parameter to query specific sub-device.

```
1 device.get_state(did="000000000000000000000000a043b0d06963")
```

### Setting state attributes

The parameters depend on the type of subdevice that is being controlled. In this example, we are controlling LC-1 switches:

#### Turn on

```
1 device.set_state(did="000000000000000000000000a043b0d0783a", pwr=1)
2 device.set_state(did="000000000000000000000000a043b0d0783a", pwr1=1)
3 device.set_state(did="000000000000000000000000a043b0d0783a", pwr2=1)
```

#### Turn off

```
1 device.set_state(did="000000000000000000000000a043b0d0783a", pwr=0)
2 device.set_state(did="000000000000000000000000a043b0d0783a", pwr1=0)
3 device.set_state(did="000000000000000000000000a043b0d0783a", pwr2=0)
```