
HowTo: Raspberry Pi OpenElec Power Wake and Shutdown Button using GPIO

December 7, 2015August 4, 2017 barry 6 Comments hardware button, linux, openelec, openelec hardware shutdown button, openelec run script at startup, openelec shutdown button, openelec wake up and shutdown with the same button, raspberry pi, reset button, shutdown button

For a little challenge, I want to add a physical on/off switch to OpenElec. This switch should turn the Raspberry Pi on if the Pi is sleeping, and it should initiate a clean shutdown if the Pi is on. This should be a physical button, not just a software remote button.

OpenElec makes this process a little difficult because it uses the squashfs filesystem. Therefore, you don't have access to the same packages and directories as you typically do.

Please be advised, this projects rating is: More Difficult – Involves soldering, use of ssh, command prompt, and coding.

There are four major steps.

1. Wire and connect a physical push button to the Pi. This should be a Normal Open (N.O.) momentary contact push button switch.
2. Set up OpenElec with the appropriate GPIO addons
3. Create a python script to shutdown the Pi when the button is pressed
4. Configure the script to run at startup

Step 1 – Connecting a switch

First, make and connect a simple "wake-up" button. To wake the Raspberry Pi up when it is asleep, all you need to do is short Pin 5 to ground. (Pin 5 is also known as GPIO03). Since Pin 6 is already at ground, you can do this by shorting Pins 5-6.

Find a contact switch

I like to use a simple momentary contact push button like this one from Radio Shack (Click on any picture to enlarge).
push_button

You can find these for a lot cheaper online, but sometimes you just want to complete a project right now! I suppose I'm just too impatient sometimes.

The important thing is that you are looking for a Normal Open (N.O.) Momentary Contact switch. This will short the two pins only when the button is pressed. If you choose a Normal Close (N.C.) button, you may have additional problems (such as the Pi perpetually resetting)

Below is a picture of the button mounted in the case:

power_button_mounted

Header Strip Plug The next thing that you need is a header strip to plug onto to the Pi. You could use a regular female header strip like this one:

header_strip

If you have an old computer case laying around (which many of us do), you can just steal one of the cables connecting a case LED to the mother board. Motherboards have the same spacing as the Raspberry Pi header. For a previous project, I used a former speaker cable:

speaker_jumper

In this case, I used a jumper like the one shown here. I used two of them, one for each lead coming from your button.

jumper_cable

The advantage is that you don't have to do any soldering to the straight pins. Also, it allows your switch to be removable, but the contacts are pretty solid. As long as you use solid wire, place a solid wire lead on each terminal of the switch. Then you can plug the wire directly into the jumper. You want the switch to be able to unplug from the Pi so that you can mount it in a case and remove it if necessary (which always seems the case with my projects).

I recommend covering each lead as well as both contacts together with some head shrink tubing. This gives a little support to the connection and also prevents unintentional shorts. In this example, the leads are shown.

Connecting to the Pi

When you are all done, simply plug your wired switch across pins 5 and 6 on the Pi, as shown below:

open_elec_reset_detail

The reset switch is connected using the green and black jumper cables. Since the switch only provides a short when pressed, it shouldn't matter which lead is connected to which pin.

You'll notice that this is shown with the original Raspberry Pi Model 2, but this should work with all versions of the Pi.

That's it. You now have a button that can wake the Raspberry Pi from sleeping mode. This is a hardware switch, nothing needs to be done in software to wake the Pi from the sleeping state, it will automatically wake when Pin 5 is tied to ground.

Step 2 – Installing the RPi.GPIO tools on OpenElec

OpenElec is a specialized install of Linux and has a few minor "perks" that can make things a little difficult.

1. The root system is not writeable
2. It does not have a standard system, such as apt-get, for installing new software
3. It doesn't have repositories to install software from the command line, everything must be done through Kodi add-ons

This makes programming with the GPIO a little more difficult. Luckily, the work has already been done and there is an add-on available for OpenElec in the unofficial add-on repository. To install the addon you must do the following:

From the Kodi Menu, go to System

1_system

Then Settings

2_settings

Press down until you get to Add-ons

3_addons

Select Get Add-ons

4_get_addons

Choose the Unofficial repository

5_unofficial_openelec_addons

Choose Libraries

6_addon_libraries

Select RPi.GPIO and hit enter (or select)

7_rpi_gpio

This should bring up the screen below. Navigate down to "Enable" to enable the Add-on.

8_rpi_gpio_enable

Once you have clicked on enable, your screen should look like the following, to show that it has been installed and enabled.

9_rpi_gpio_installed

Notice that the only difference is that your option is now to “Disable” the Add-on, instead of enabling it.

Now, we can do some coding.

Step 3 – Write and install the Shutdown Script

The following commands should be done through SSH. By default, SSH is enabled on OpenElec. You will need to find the IP address of your Pi. If you don’t know it already, you can find it by going to the Main Menu on Kodi, then System -> System Info. As long as you have your Pi plugged into the network, the IP address should be shown on this screen.

I recommend creating a directory to hold your scripts. In my case, I created the directory under the root /storage/ directory. It can be created by doing:

```
mkdir /storage/scripts
```

This will make a scripts directory under the /storage folder. This is a writable and persistent area of the file system (all good things).

Next, we need to write the script. I personally use nano as my editor of choice, but you can use any editor you wish. Please Note: I highly recommend retyping the script. Do not copy and paste. It is possible that if you copy and paste, there will be slight changes in formatting (i.e. comment lines wrapping and no longer being comments) or incorrect characters (particularly quotes) that will keep your script from working!!!

We will call our script shutdown.py (it is written in python). Create and edit the script by doing:

```
nano /storage/scripts/shutdown.py
```

The content of the script should be:

```
#!/usr/bin/python
import sys
sys.path.append('storage/.kodi/addons/python.RPi.GPIO/lib')
import RPi.GPIO as GPIO
import time
import subprocess

# we will use the pin numbering to match the pins on the Pi, instead of the
# GPIO pin outs (makes it easier to keep track of things)

GPIO.setmode(GPIO.BOARD)

# use the same pin that is used for the reset button (one button to rule them all!)
GPIO.setup(5, GPIO.IN, pull_up_down = GPIO.PUD_UP)

oldButtonState1 = True

while True:
    #grab the current button state
    buttonState1 = GPIO.input(5)

  # check to see if button has been pushed
  if buttonState1 != oldButtonState1 and buttonState1 == False:
    subprocess.call("shutdown -h now", shell=True,
      stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    oldButtonState1 = buttonState1

    time.sleep(.1)
```

Once you have created your script, you can test it by typing the following at the command prompt:

```
python /storage/scripts/shutdown.py
```

It may give warnings regarding a pull-up resistor. This is OK. If error's are encountered, resolve them first. If it is able to run, try pushing the button and see if the Pi shuts down.

Once you have the script working the way that you want it, let's make it run at startup.

Configure our script to run at startup

We still have problem with the fact that the core system of openelec is not easily writeable. So…you can't put anything in init.d like usual to have it run at startup. Luckily, the makers of OpenElec created a mechanism for running programs at startup. It is similar to the old autoexec.bat found on DOS machines. Instead, you must place any programs in the /storage/.config/autostart.sh file.

You can edit the autostart.sh file by doing the following:

```
nano autostart.sh
```

The content of my autostart file looks like this:

```
python /storage/scripts/shutdown.py &
```

That's it! Now your script should run at startup. If you already have an autostart.sh file, simply add the line of code at the bottom. Once you restart your pi, you should be able to see shutdown.py running in the background. You can verify this either by pressing the button and seeing if OpenElec shuts down, or doing the following from the command line:

```
ps -A | grep shutdown.py
```

Good Luck!

- ← How To: Raspberry Pi: 5V (UPS) Uninterruptable Power Supply
- HowTo: Adding GPIO to OpenElec running on original series Raspberry Pi (Rev A, B, B+, Zero?) →

6 thoughts on "HowTo: Raspberry Pi OpenElec Power Wake and Shutdown Button using GPIO"

- Pingback: HowTo: Adding GPIO to OpenElec running on original series Raspberry Pi (Rev A, B, B+, Zero?) | barryhubbard.com

- Peter Schuttevaar
  March 13, 2016 at 5:29 pm
  Permalink

  Thank you very much for this useful instruction.

  there is however one little error in the shutdown script. Line 3 is missing a '/'. It should read like

  sys.path.append('/storage/.kodi/addons/python.RPi.GPIO/lib')

  in stead of

sys.path.append('storage/.kodi/addons/python.RPi.GPIO/lib')

Regards,
Peter

Reply

- Frank
  March 20, 2016 at 3:33 pm
  Permalink

  Hi,

  I am new to this, it seems like you build already the solution that i was looking for.

  I want to use this, but i think my soundboard is using these pins (pins 5 (GPIO3-I2C) is it possible to use other pins?
  i don't know if it can harm my soundboard (a suptronics X400) if i just use pin 5 and 6 to build this buton?

  when i start the script now on pin 5 it gives me the error below (buton is not in place a.t.m.):
  OpenELEC:~/scripts # python /storage/scripts/shutdown.py
  /storage/scripts/shutdown.py:14: RuntimeWarning: This channel is already in use, continuing anyway. Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(5, GPIO.IN, pull_up_down = GPIO.PUD_UP)
  /storage/scripts/shutdown.py:14: RuntimeWarning: A physical pull up resistor is fitted on this channel!
  GPIO.setup(5, GPIO.IN, pull_up_down = GPIO.PUD_UP)

  ^CTraceback (most recent call last):
  File "/storage/scripts/shutdown.py", line 28, in
  time.sleep(.1)

  gr, Frank

  Reply

  - barryPost author
    March 29, 2016 at 12:33 pm
    Permalink

    You can use other pins for the shutdown script. Simply modify the lines of code that set up and poll the pin

    GPIO.setup(5, GPIO.IN, pull_up_down = GPIO.PUD_UP)

    and

    buttonState1 = GPIO.input(5)

    To use the pin that you wish.

    Reply

- kirner
  March 24, 2016 at 4:14 pm
  Permalink

Simply Great!
please update your post with the fix of path library as suggested Peter too.

Reply

- fzacca
  May 12, 2016 at 6:34 pm
  Permalink

  That's great! I needed just that script, only for another port, since my HifiBerry DIgi+ uses the GPIO3…works flawlessly thouhg.
  I was wondering how to edit that script in order to execute the backlight commands for my project:
  echo 1 > /sys/class/backlight/rpi_backlight/bl_power
  and
  echo 0 > /sys/class/backlight/rpi_backlight/bl_power
  alternated…you know…press the button once It switches off the backlight, press it again, it goes on…
  Any ideas?

  Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Recent Posts

- HowTo: Change the default Port for apache2
- Raspberry Pi – HowTo Composite Video TRRS Connector
- Hacking a BESTOPE / KanKun / Konke Smart Plug: HowTo Automate Powercycle Network
- Raspberry Pi Project: GPIO LED Stoplight
- Raspberry Pi Project: GPIO LED

Recent Comments

- mpi on HowTo: Raspberry Pi Raspbian Power on / off GPIO button
- mpi on HowTo: Raspberry Pi Raspbian Power on / off GPIO button
- Kory on Hacking a BESTOPE / KanKun / Konke Smart Plug: HowTo Automate Powercycle Network
- fzacca on HowTo: Raspberry Pi OpenElec Power Wake and Shutdown Button using GPIO
- barry on How To: Raspberry Pi: 5V (UPS) Uninterruptable Power Supply

Archives

- January 2016

- December 2015

Categories

- Linux
- Raspberry Pi
- Uncategorized
- Windows

Meta

- Log in
- Entries RSS
- Comments RSS
- WordPress.org