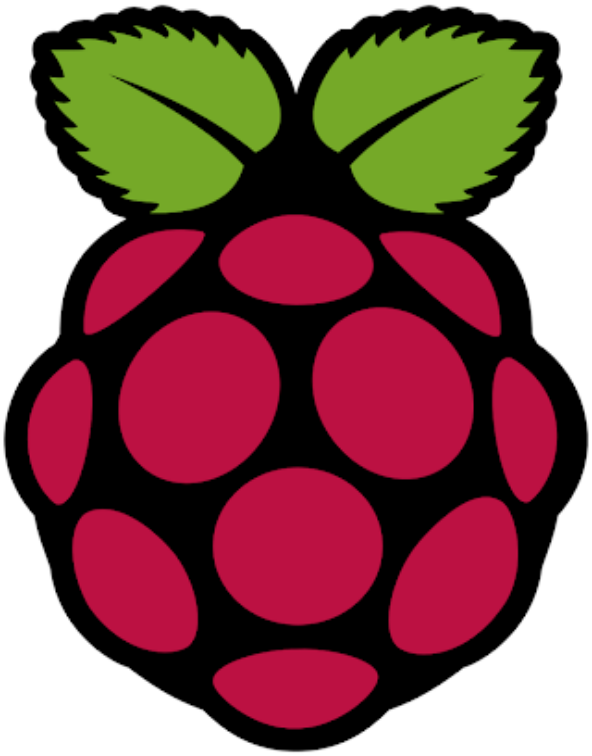
linuxuprising.com

Raspberry Pi: Power On / Off A TV Connected Via HDMI-CEC

7–8 minutes



With the help of cec-client (part of libcec), your Raspberry Pi can control a device that supports CEC, like a TV, connected via HDMI. You could power the TV on or off, switch the active source, and more.

This should work with any Raspberry Pi version or model, including the original Raspberry Pi, as well as the latest Raspberry Pi 4.

A possible use case for this would be to connect to a Raspberry Pi via SSH and send a command to power on or off a TV connected to it via HDMI-CEC. Or you can use the commands to power on TV and make the CEC adapter the active source in a script, so that when you open some application on your Raspberry Pi, the TV that's connected to it via HDMI-CEC powers on and switches to your Raspberry Pi HDMI source. I'm sure you can think of various other use-cases.

CEC, or Consumer Electronics Control, is a feature of HDMI which allows controlling devices connected through HDMI using a remote control. For example, CEC is used to get the play/pause buttons on a remote control to control playback on a device connected via HDMI. Or when you play a video on a Chromecast with the TV off, and the TV automatically powers on and switches to the Chromecast source.

Most modern TVs and AV receivers should support HDMI-CEC. However, it's worth noting that you may need to enable CEC in the TV settings on some models. CEC may have a different name, depending on the device brand. For example, it's called Anynet+ for Samsung TVs, EasyLink or Fun-Link for Philips, SimpLink for LG, and so on.

To be able to power on (and off) a TV connected via HDMI to a Raspberry Pi, the first step is to install cec-client. On Raspbian, or some other Debian or Ubuntu based Linux distribution for Raspberry Pi, **install the `cec-utils` package** (`cec-client` is part of this package):

```
sudo apt install cec-utils
```

On other Linux distributions you'll have to search for `cec-client` or `cec-utils` in the repositories, or build libcec from source.

Now that `cec-utils` is installed, let's **scan the CEC bus for available devices:**

```
echo 'scan' | cec-client -s -d 1
```

In this command `echo 'scan'` sends the scan command to cec-client, `-s` is used so cec-client executes a single command and exists, and `-d 1` sets the log level to 1 (errors only), so it's doesn't pollute your terminal with useless info.

Remember the TV (or other device connected via HDMI-CEC to your Raspberry Pi) device # and address, as we'll use that later on.

This is an example running this command on my Raspberry Pi that's connected to a Samsung TV via HDMI (with CEC support):

```
$ echo 'scan' | cec-client -s -d 1
opening a connection to the CEC adapter...
requesting CEC bus information ...
CEC bus information
===================
device #0: TV
address:       0.0.0.0
active source: no
vendor:        Samsung
osd string:    TV
CEC version:   1.4
power status:  on
language:      eng
```

```
device #1: Recorder 1
address:       1.0.0.0
active source: no
vendor:        Pulse Eight
osd string:    CECTester
CEC version:   1.4
power status:  on
language:      eng

currently active source: unknown (-1)
```

In this example, device number `0` with the `0.0.0.0` address is my Samsung TV, and device number 1 with the `1.0.0.0` address is my Raspberry Pi device.

Now that we know the device number and address, you can **use the command that follows to power on a TV connected via HDMI-CEC to the Raspberry Pi:**

```
echo 'on <DEVICE #>' | cec-client -s -d 1
```

Or:

```
echo 'on <DEVICE ADDRESS>' | cec-client -s -d 1
```

Both the device number (`0` is the Samsung TV in the example above) and device address (`0.0.0.0` is the Samsung TV device address from my example) should work.

`-d 1` is to limit the log level to errors only, and you can use the command without it, but you'll see a long, probably useless log.

Example:

```
echo 'on 0' | cec-client -s -d 1
```

Or:

```
echo 'on 0.0.0.0' | cec-client -s -d 1
```

You'll also want to run the as command, which makes the CEC adapter the active source (so the TV switches to the Raspberry Pi HDMI source after the TV is powered on):

```
echo 'as' | cec-client -s -d 1
```

Want to turn the TV off (enter standby)? Use:

```
echo 'standby <DEVICE #>' | cec-client -s -d 1
```

Depending on how you use this, you may also need to check the current TV status (is it on or in standby?). This can be done using:

```
echo 'pow <DEVICE #>' | cec-client -s -d 1
```

To view all the commands that cec-client can send to a HDMI-CEC connected device, use `echo h | cec-client -s -d 1`:

```
Available commands:

[tx] {bytes}              transfer bytes over the CEC
line.
[txn] {bytes}             transfer bytes but don't
wait for transmission ACK.
[on] {address}            power on the device with
the given logical address.
[standby] {address}       put the device with the
given address in standby mode.
[la] {logical address}    change the logical address
of the CEC adapter.
```

```
[p] {device} {port}       change the HDMI port
number of the CEC adapter.
[pa] {physical address}   change the physical
address of the CEC adapter.
[as]                      make the CEC adapter the
active source.
[is]                      mark the CEC adapter as
inactive source.
[osd] {addr} {string}     set OSD message on the
specified device.
[ver] {addr}              get the CEC version of the
specified device.
[ven] {addr}              get the vendor ID of the
specified device.
[lang] {addr}             get the menu language of
the specified device.
[pow] {addr}              get the power status of the
specified device.
[name] {addr}             get the OSD name of the
specified device.
[poll] {addr}             poll the specified device.
[lad]                     lists active devices on the
bus
[ad] {addr}               checks whether the
specified device is active.
[at] {type}               checks whether the
specified device type is active.
[sp] {addr}               makes the specified
physical address active.
```

```
[spl] {addr}              makes the specified logical
address active.
[volup]                   send a volume up command to
the amp if present
[voldown]                 send a volume down command
to the amp if present
[mute]                    send a mute/unmute command
to the amp if present
[self]                    show the list of addresses
controlled by libCEC
[scan]                    scan the CEC bus and display
device info
[mon] {1|0}               enable or disable CEC bus
monitoring.
[log] {1 - 31}            change the log level. see
cectypes.h for values.
[ping]                    send a ping command to the
CEC adapter.
[bl]                      to let the adapter enter the
bootloader, to upgrade
the flash rom.
[r]                       reconnect to the CEC
adapter.
[h] or [help]             show this help.
[q] or [quit]             to quit the CEC test client
and switch off all
connected CEC devices.
```