Tutorial

Configuring Orange PI PC to receive IR/InfraRed

By AnonymousPi
September 3, 2016 in Reviews, Tutorials, Hardware hacks

AnonymousPi
Posted September 3, 2016
Note: Guide Updated May 2017, as I realise that /dev/input/event3 may not always be the IR receiver device on your armbian installation.

Hi All,

I recently bought an Orange PI PC and the best thing I ever did was install Armbian straight away (and donate). Now that I have a bit of spare time, I wanted to configure my Orange PI PC to do something ridiculous like play Rick Ashley 'Never going to give you up' upon pressing the 'red button' on some generic Chinese IR remote for an LED light strip I have in my living room.

Thanks to Armbian, most of the pieces are in place (such as the SunXI IR package) with the distribution, you just need to glue it all together.

However there are few configuration issues with the default Armbian install on the Orange PI PC that need to be adjusted, otherwise you'll encounter infuriating issues such as:

- No IR device existing or being detected (root cause: sunxi-cir module not loaded)
- No LIRC 'irw' output even after successfully using irrecord (root cause: DRIVER=devinput doesn't work, though it could be my remote), like this poor sod was experiencing.

I should also note that this guide on the terrible Orange PI forums, helped me with my issues.

Step 1) Adjust /etc/lirc/hardware.conf

Updated: This guide was originally written for Armbian based on Debian 'Jessie'. The latest Armbian (as at September 2017) is now based on Ubuntu Xenial. This introduces a new lirc package which yet again comes with a **broken** hardware.conf

For Ubuntu Xenial (September 2017):

The default hardware.conf that comes with Armbian is broken. It's assigning the 'remote' and 'transmitter' to the same device, this breaks everything. Ensure the TRANSMITTER_MODULES="" and TRANSMITTER_DEVICE = ""

```
# /etc/lirc/hardware.conf
#
#Chosen Remote Control
REMOTE="None"
REMOTE_MODULES="sunxi_cir"
REMOTE_DRIVER="default"
REMOTE_DEVICE="/dev/lirc0"
REMOTE_SOCKET="" # FYI - /run/lirc/lircd   will probably be the socket that the system uses
REMOTE_LIRCD_CONF=""
REMOTE_LIRCD_ARGS=""

#Chosen IR Transmitter
TRANSMITTER="None"
TRANSMITTER_MODULES=""
TRANSMITTER_DRIVER=""
TRANSMITTER_DEVICE="/dev/null"
TRANSMITTER_SOCKET=""
TRANSMITTER_LIRCD_CONF=""
TRANSMITTER_LIRCD_ARGS=""

#Disable kernel support.
#Typically, lirc will disable in-kernel support for ir devices in order to
#handle them internally.  Set to false to prevent lirc from disabling this
#in-kernel support.
#DISABLE_KERNEL_SUPPORT="true"

#Enable lircd
START_LIRCD="true"

#Don't start lircmd even if there seems to be a good config file
#START_LIRCMD="false"

#Try to load appropriate kernel modules
LOAD_MODULES="true"

# Default configuration files for your hardware if any
LIRCMD_CONF=""

#Forcing noninteractive reconfiguration
#If lirc is to be reconfigured by an external application
#that doesn't have a debconf frontend available, the noninteractive
#frontend can be invoked and set to parse REMOTE and TRANSMITTER
#It will then populate all other variables without any user input
#If you would like to configure lirc via standard methods, be sure
#to leave this set to "false"
FORCE_NONINTERACTIVE_RECONFIGURATION="false"
START_LIRCMD=""
```

For Debian Jessie (~year 2016):

By default Armbian doesn't have the suxi-cir module enabled at boot-up, but it is available, so you will need to edit hardware.conf to enable this, as well as correct the DRIVER= line and the DEVICE= line, **as the defaults in there are WRONG**

Also I suggest commenting out Igor's code in the top five lines. A hardware.conf that works:

```
# Cubietruck automatic lirc device detection by Igor Pecovnik
#str=$(cat /proc/bus/input/devices | grep "H: Handlers=sysrq rfkill kbd event" | awk '{print $(NF)}')
#sed -i 's/DEVICE="\/dev\/input.*/DEVICE="\/dev\/input\/'$str'"/g' /etc/lirc/hardware.conf
# /etc/lirc/hardware.conf
#
# Arguments which will be used when launching lircd
LIRCD_ARGS=""

#Don't start lircmd even if there seems to be a good config file
#START_LIRCMD=false

#Don't start irexec, even if a good config file seems to exist.
#START_IREXEC=false

#Try to load appropriate kernel modules
LOAD_MODULES=true

# Run "lircd --driver=help" for a list of supported drivers.
# 'devinput' driver on Orange PI PC causes NO EVENTS TO OCCUR
# via irw for some reason.
DRIVER="default"

# usually /dev/lirc0 is the correct setting for systems using udev
DEVICE="/dev/lirc0"
MODULES="sunxi-cir"

# Default configuration files for your hardware if any
LIRCD_CONF=""
LIRCMD_CONF=""
```

Step 2) Restart lircd service

As lirc is actually already running and installed in Armbian, do the following:

```
root@orangepipc:/etc# /etc/init.d/lirc stop
root@orangepipc:/etc# /etc/init.d/lirc start
```

To reboot the service.

Then perform an 'lsmod' to see if it loaded the sunxi_cir module (because otherwise nothing will work):

```
user@orangepipc:~$ lsmod
Module                  Size  Used by
mali_drm                2732  1
drm                   178255  2 mali_drm
mali                  123208  0
ump                    29379  3 mali
sunxi_cir               1601  0
8189es                1076034  0
```

Step 3) Find out what '/dev/input/eventX' device is your IR receiver

If you do a:

```
ls /dev/input/event*
```

You will most likely get a bunch of possible event devices to choose from, for example:

```
anonymouspi@orangepipc:~$ ls /dev/input/event*
/dev/input/event0  /dev/input/event2  /dev/input/event4
/dev/input/event1  /dev/input/event3  /dev/input/event5
```

For my installation, /dev/input/event3 is the IR receiver, but if you have other devices installed (i.e. USB cameras, keyboards etc.) then the number could be different. For example, executing 'evtest /dev/input/event3' reveals:

```
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "sunxi-ir"
```

A device name of 'sunxi-ir' means that we are using the right device for the purposes of evtest

Step 4) Do a quick test with with 'evtest'(OrangePI PC armbian seems to use /dev/input/event3 for IR input)

Armbian has the 'evtest' program installed, point the IR remote (in my case a LED colour remote) at your Orange PI PC and as root 'evtest /dev/input/event3'.

```
root@orangepipc:/etc# evtest /dev/input/event3
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "sunxi-ir"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 152 (KEY_SCREENLOCK)
  Event type 4 (EV_MSC)
    Event code 4 (MSC_SCAN)
Key repeat handling:
  Repeat type 20 (EV_REP)
    Repeat code 0 (REP_DELAY)
      Value    500
    Repeat code 1 (REP_PERIOD)
      Value    125
Properties:
Testing ... (interrupt to exit)
```

Pressing the remote reveals events like:

```
Testing ... (interrupt to exit)
Event: time 1472917554.113967, type 4 (EV_MSC), code 4 (MSC_SCAN), value 58
Event: time 1472917554.113981, ---------- EV_SYN ------------
Event: time 1472917554.464390, type 4 (EV_MSC), code 4 (MSC_SCAN), value 59
Event: time 1472917554.464398, ---------- EV_SYN ------------
Event: time 1472917554.842832, type 4 (EV_MSC), code 4 (MSC_SCAN), value 45
Event: time 1472917554.842839, ---------- EV_SYN ------------
Event: time 1472917555.345584, type 4 (EV_MSC), code 4 (MSC_SCAN), value 58
```

That was the red, green, blue and white buttons being pressed. This is a good news.

Step 5) Configure lirc to map IR input to key presses or events.

Again, Armbian has irrecord installed (great work Igor), but given I'm re-using this remote to configure the output of a LED strip I have, I'll need to map the IR data sent, to something more meaningful. In other use-cases this isn't generally required as lircs provides a database of media remotes which is pre-mapped to Linux commands/keyboard keys.

There's plenty of information on how to use irrecord, command I used was:

```
/etc/init.d/lirc stop
```

...to first stop the service, then:

```
irrecord -H default -d /dev/lirc0 /etc/lirc/lircd.conf
```

... to record my remote and bind to 'keys'.

Step 6) Test with irw

Now that I recorded my configuration file with irrecord:

```
/etc/init.d/lirc start
```

.. to start lird service again

then type 'irw' and check that the key mapping works when I point the remote at the Orange PI PC and press a button:

```
root@orangepipc:/etc# irw
0000000000ff1ae5 00 KEY_R /etc/lirc/lircd.conf
0000000000ff1ae5 01 KEY_R /etc/lirc/lircd.conf
0000000000ff9a65 00 KEY_G /etc/lirc/lircd.conf
0000000000ff9a65 01 KEY_G /etc/lirc/lircd.conf
0000000000ffa25d 00 KEY_B /etc/lirc/lircd.conf
0000000000ffa25d 01 KEY_B /etc/lirc/lircd.conf
0000000000ff22dd 00 KEY_W /etc/lirc/lircd.conf
0000000000ff22dd 01 KEY_W /etc/lirc/lircd.conf
```

Hoo Ray!

Step 7) Create a /etc/lirc/lircrc file to run commands

```
sudo vi /etc/lirc/lircrc
```

I'd actually call mpv here and call the player:

```
# http://www.lirc.org/html/configure.html

begin
  button = KEY_R
  prog = irexec
  config = mpv  /home/root/Rick\\ Astley\\ -\\ Never\\ Gonna\\ Give\\ You\\ Up.m4a & echo "COMMENT RICK ROLLING" &


end
begin
  button = KEY_W
  prog = irexec
  config = killall mpv & echo "SADFACE!" &
end
begin
  button = KEY_B
  prog = irexec
  config = mpv http://sj256.hnux.com &
end
```

You could also create a file for each user of the system if you want, eg: **/root/.lircrc**, **/home/userXXX/.lircrc**
However if you do this, you will need to start the irexec service manually. If you have a /etc/lirc/lircrc file, the irexec service will start automatically at boot - this service is what actually converts the key press to the command.

So there you go, Rickrolling with a simple press of the red (KEY_R) button :-)

Additional References:
[Guide] Android + InfraRed (IR) + Kodi
How to setup Remote Control for Linux

msev
Posted September 6, 2016
Thank you for this tutorial! I shall try to follow your footsteps.

msev
Posted September 6, 2016
Is it possible to bind this lirc to some python script or something? Or how someone could intercept these button press events and then pipe them somewhere else where they could be used to trigger stuff?

Kevin
Posted September 6, 2016
@msdev:

You mean like this: https://pypi.python.org/pypi/python-lirc/1.2.1

You can also pipe irw into python http://www.lirc.org/html/irw.html

msev
Posted September 7, 2016
Thanks Kevin.

PoV
Posted September 12, 2016
I wanted to configure my Orange PI PC to do something ridiculous like play Rick Ashley 'Never going to give you up' upon pressing the 'red button' on some generic crap Chinese IR remote for an LED light strip I have in my living room.

I like your style kid.

mariuszb
Posted December 4, 2016
Hi all,

According to:
Step 1) Adjust /etc/lirc/hardware.conf

I have a completely different contents of this file,
where I make a mistake?

the contents of my file /etc/lirc/hardware.conf below :

```
count 1
range '0 - 7'
}
}
control.2 {
iface MIXER
name 'MIC2_G boost stage output mixer control'
value 3
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
```

range '0 - 7'
}
}
control.3 {
iface MIXER
name 'LINEIN_G boost stage output mixer control'
value 3
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}
control.4 {
iface MIXER
name 'MIC1 boost AMP gain control'
value 4
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}
control.5 {
iface MIXER
name 'MIC2 boost AMP gain control'
value 4
comment {
access 'read write'
type INTEGER
count 1
range state.audiocodec {
control.1 {
iface MIXER
name 'MIC1_G boost stage output mixer control'
value 3
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}
control.2 {
iface MIXER
name 'MIC2_G boost stage output mixer control'
value 3
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}
control.3 {
iface MIXER
name 'LINEIN_G boost stage output mixer control'
value 3
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}

```
}
control.4 {
iface MIXER
name 'MIC1 boost AMP gain control'
value 4
comment {
access 'read write'
type INTEGER
count 1
range '0 - 7'
}
}
control.5 {
iface MIXER
name 'MIC2 boost AMP gain control'
value 4
comment {
access 'read write'
type INTEGER
count 1
range
```

mariuszb

Posted December 5, 2016

btw. I changed the contents of this file and everything works,

but why after the Armbian installation this file was so different from the target?

zador.blood.stained

Posted December 5, 2016

@mariuszb

Are you sure that this is the default contents? This looks more like filesystem corruption to me, and this file looks like asound.state which is not related to LIRC

mariuszb

Posted December 5, 2016

Certainty is never, but this file seemed ok, only content was strange.
I did not created this file.
Armbian was compiled from sources some four months ago, is working fine
(webmin, apache,ssl,mqtt server and many other things working 24/7)
now IR remote control (Samsung) also work

Maybe an error at compile time? file system error is not indicated

zador.blood.stained

Posted December 6, 2016

Armbian was compiled from sources some four months ago, is working fine
OK, if it's not a prebuilt image from download section then we'll never know why this happened.

mariuszb

Posted December 7, 2016

Thaks for this

Btw.
Did Yoy try play stream from playlist ? (works for me: mpv --playlist = playlist.pls) but..
Do You know how to easy change to next stream from play list using IR ?
(just kill and restart with new stream only metod?)

Last news:
I am fine now,
I solved the problem and I use mpd, mpc for bacground playing music from playlist
eg.

```
begin
  prog = irexec
  button = KEY_CHANNELDOWN
  config = mpc prev & echo "prev music" >/dev/ttyS0
end
```

Hi All,

I recently bought an Orange PI PC and the best thing I ever did was install Armbian straight away (and donate). Now that I have a bit of spare time, I wanted to configure my Orange PI PC to do something ridiculous like play Rick Ashley 'Never going to give you up' upon pressing the 'red button' on some generic crap Chinese IR remote for an LED light strip I have in my living room.

Step 5) Create a /root/.lircrc file to run commands

I'd actually call mpv here and call the player:

```
# http://www.lirc.org/html/configure.html

begin
   button = KEY_R
   prog = irexec
   config = mpv  /home/root/Rick\\ Astley\\ -\\ Never\\ Gonna\\ Give\\ You\\ Up.m4a & echo "COMMENT RICK ROLLING" &


end
begin
   button = KEY_W
   prog = irexec
   config = killall mpv & echo "SADFACE!" &
end
begin
   button = KEY_B
   prog = irexec
   config = mpv http://sj256.hnux.com &
end
```

LinuxUser

Posted January 21, 2017

Hi, I'm having problems with the evtest. Everything works just fine until I reach that point. The evtest shows nothing.
I tried using the command "sudo mode2 -d /dev/lirc0" and it works, so my IR receiver is functioning, but the evtest doesn't detec anything.
I also tried with both the default and the Igor's hardware.conf file. Same result.
I don't know if it is usefull, but the command "irw" also doesn't work for me, it says "connection refused".
I'm using Armbian on an Orange PI PC Plus.

kris777

Posted January 21, 2017
Thanks !
for me, this tutorial was very helpful.
However, the configuration file created :

```
irrecord -H default -d /dev/lirc0 /etc/lirc/lircd.conf
```

but the remote control to work properly.
Already done ...
It is difficult for proper entries: .....and shows that are very important!

```
name       mceusb
bits            16
flags  RC6|CONST_LENGTH
eps             30
aeps            100

header     2667  889
one         444  444
zero        444  444
pre_data_bits      21
pre_data      0x37FF0
gap          105000
toggle_bit        22
rc6_mask    0x100000000
```

after the change of all is OK ...I use the remote control OrigenAE RC153 which uses standard codes RC6 (mce)

original config (lircd.conf) under the remote control mce ..... and created:
mce https://dl.dropboxusercontent.com/u/68191728/LINUX/lircd-mce.conf
OrigenAE https://dl.dropboxusercontent.com/u/68191728/LINUX/lircd-OrigenAE.conf

a list of possible names in the irrecord : https://dl.dropboxusercontent.com/u/68191728/LINUX/key
Useful links (internet radio) http://www.orangepi.org/orangepibbsen/forum.php?mod=viewthread&tid=1389
another sensor and remote control : https://forum.armbian.com/index.php/topic/3288-orangepi-lite-armbian-remote-x10/

Regards !

hardheid
Posted May 1, 2017
Thanks for the write up AnonymousPi!

I am running an OrangePi PC as well with Armbian on top of an Ubuntu Kernel. The default LIRC config, however, is different. I tried making changes, but whatever I do sunxi_cir just does not show with lsmod. Is anyone else having issues?

kris777
Posted May 1, 2017

I have Orange Pi lite / ARMBIAN 5.25 stable Ubuntu 16.04.2 LTS 3.4.113-sun8i
And after the lsmod command:

```
Module                  Size  Used by
mali_drm                2732  1
drm                   178255  2 mali_drm
pcf8591                 3363  0
bmp085                  3487  0
mali                  123146  0
ump                    29379  3 mali
ir_lirc_codec           3650  3
lirc_dev                7834  1 ir_lirc_codec
ir_mce_kbd_decoder      2885  0
snd_usb_audio          80352  2
ir_sanyo_decoder        1480  0
snd_hwdep               5562  1 snd_usb_audio
ir_sony_decoder         1422  0
snd_usbmidi_lib        17545  1 snd_usb_audio
snd_seq_midi            4215  0
snd_seq_midi_event      5340  1 snd_seq_midi
ir_jvc_decoder          1452  0
ir_rc6_decoder          1941  0
snd_rawmidi            17134  2 snd_usbmidi_lib,snd_seq_midi
ir_rc5_decoder          1412  0
ir_nec_decoder          1556  0
g_serial               27617  2
sunxi_cir               1601  0
```

Johann Pascher

Posted May 4, 2017

@LinuxUser I also have the same orangepipc+ and had the same problem.
pleas red my thread about it.

1

6 yr Igor unpinned this topic

AnonymousPi

Posted September 18, 2017

Updated for Ubuntu Xenial issues I faced.

Igor

Posted September 19, 2017

We would appreciate help fixing this upstream. It has been some time since I was playing with LIRC ... And there are some other changes in Debian Stretch which are completely broken in some other way. Haven't check how it is done there yet.

This is the code which should fix this - configuration for Allwinner H3:
https://github.com/armbian/build/blob/master/config/sources/sun8i.conf#L31-L52

AnonymousPi

Posted October 1, 2017

On 19/09/2017 at 7:37 AM, Igor said:

We would appreciate help fixing this upstream. It has been some time since I was playing with LIRC ... And there are some other changes in Debian Stretch which are completely broken in some other way. Haven't check how it is done there yet.

Hi Igor. More than happy to help out, but I am confused as to the purpose of that sun8i.conf config file. What really needs to change is the contents within the distribution '/etc/lirc/hardware.conf' so similar to how lircd.conf is contained here.

Any insight in to how to do this appreciated. I am happy to test for stretch.

guidol

Posted November 28, 2017

Today I reinstalled my OPi R1 with stable/legacy for testing IR/lirc with the IR-Port of the NAS-Expansionboard.
I installed the IR-Support via armbian-config (network-section) and it automatically loads the module **sunxi_cir**
Modules are loaded and I get a /dev/input/event2 - but evtest /dev/input/event2 did only produce this output and doesnt recognize the IR-remote:

```
root@opi-zero-r1:~# evtest /dev/input/event2
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "sunxi-ir"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 152 (KEY_SCREENLOCK)
  Event type 4 (EV_MSC)
    Event code 4 (MSC_SCAN)
Key repeat handling:
  Repeat type 20 (EV_REP)
    Repeat code 0 (REP_DELAY)
      Value    500
    Repeat code 1 (REP_PERIOD)
      Value    125
Properties:
Testing ... (interrupt to exit)
```

My OrangePi Zero does load all modules if I **modprobe sunxi_cir** and does respond to the IR-remote:

```
root@orangepizero:/etc/lirc# evtest /dev/input/event2
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "sunxi-ir"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 152 (KEY_SCREENLOCK)
  Event type 4 (EV_MSC)
    Event code 4 (MSC_SCAN)
Key repeat handling:
  Repeat type 20 (EV_REP)
    Repeat code 0 (REP_DELAY)
      Value    500
    Repeat code 1 (REP_PERIOD)
      Value    125
Properties:
Testing ... (interrupt to exit)
Event: time 1511898396.197217, type 4 (EV_MSC), code 4 (MSC_SCAN), value 116
Event: time 1511898396.197227, ---------- SYN_REPORT ------------
Event: time 1511898397.768212, type 4 (EV_MSC), code 4 (MSC_SCAN), value 11a
Event: time 1511898397.768236, ---------- SYN_REPORT ------------
```

I did take a look at the fex-edit in armbian-config for the 2 boards and the section for IR seems to be the same.
Can I take a look in anything other?

on the OPi Zero R1 the follwing lirc-components are installed:

```
root@opi-zero-r1:~# dpkg -l|grep lirc
ii  inputlirc                  23-2                          armhf        Zeroconf LIRC daemon using input event devices
ii  liblircclient0:armhf       0.9.0-0ubuntu6                armhf        infra-red remote control support - client library
ii  lirc                       0.9.0-0ubuntu6                armhf        infra-red remote control support
ii  lirc-x                     0.9.0-0ubuntu6                armhf        infra-red remote control support - X utilities
```

and on the normal OPi Zero this lirc-components are installed:

```
root@orangepizero:/etc/lirc#  dpkg -l|grep lirc
ii  inputlirc                  23-2                          armhf        Zeroconf LIRC daemon using input event devices
ii  liblircclient0:armhf       0.9.0-0ubuntu6                armhf        infra-red remote control support - client library
ii  lirc                       0.9.0-0ubuntu6                armhf        infra-red remote control support
ii  lirc-x                     0.9.0-0ubuntu6                armhf        infra-red remote control support - X utilities
```

seems to be the same 😕

Igor

Posted November 28, 2017

I assume you use identical kernel / armbian version on both?

guidol

Posted November 29, 2017

8 hours ago, Igor said:
I assume you use identical kernel / armbian version on both?
Yes - because I had before the R1 on 4.x Kernel I reinstalled another uSD with 3.4.x legacy kernel:

OPi Zero R1:

ARMBIAN 5.35 user-built Ubuntu 16.04.3 LTS 3.4.113-sun8i
Linux opi-zero-r1 3.4.113-sun8i #4 SMP PREEMPT Wed Nov 22 13:45:28 CET 2017 armv7l armv7l armv7l GNU/Linux

OPi Zero (normal-version):

ARMBIAN 5.35 user-built Ubuntu 16.04.3 LTS 3.4.113-sun8i
Linux orangepizero 3.4.113-sun8i #4 SMP PREEMPT Wed Nov 22 13:45:28 CET 2017 armv7l armv7l armv7l GNU/Linux

⚠ This topic is now closed to further replies.