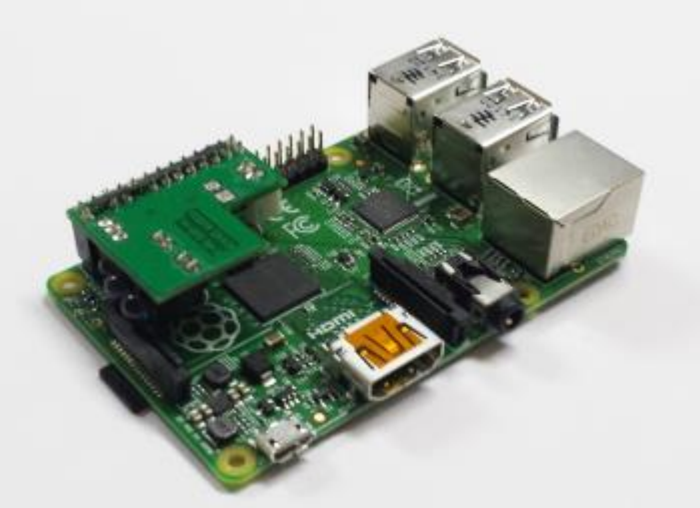
Energenie Raspberry Pi infrared add on board

The energenie IR board is a simple to install piggy-back add-on board that fits on to the row of General Purpose Input Output (GPIO) pins on the R-Pi board models A, B and B+. It works in combination with software such as LIRC to allow you to receive and send infra-red signals under control of your R-Pi.

You can use it to replace or complement an existing handset to control your TV and audio visual equipment.



Here is the layout of the GPIO connector viewed from the top.

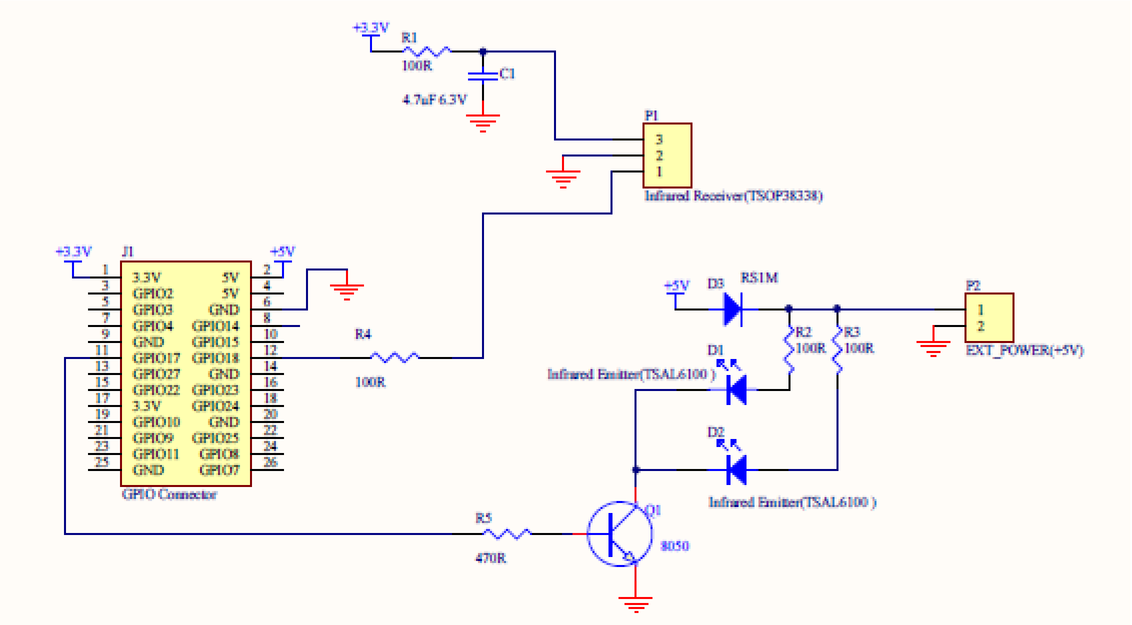
Pin Header Rev 2.0 view from top

	3V3	1	2	5V	
	GPIO2	3	4	5v	
	GPIO3	5	6	Ground	
	GPIO4	7	8	GPIO14	
	Ground	9	10	GPIO15	
IR signal out	GPIO17	11	12	GPIO18	IR Signal in
	GPIO27	13	14	Ground	
	GPIO22	15	16	GPIO23	
	3V3	17	18	GPIO24	
	GPIO10	19	20	Ground	
	GPIO9	21	22	GPIO25	
	GPIO11	23	24	GPIO8	
	Ground	25	26	GPIO7	

The R-Pi's processor uses 3.3V levels and the pins are not 5V tolerant! The maximum permitted current draw from the 3v3 pin is 50mA. To be able to capture the IR signals we use an infrared receiver that can operate at this voltage. We chose the 38kHz type because most remote control standards use this frequency. The GPIO pins that are used are:

PIN12 - GPIO18	input pin for received infrared signal
PIN11 - GPIO17	output pin for transmitted infrared signal

Circuit



Software – LIRC

To use the energenie hardware in a useful way requires a suitable software package to drive it.

LIRC is a free-for-use software package for Linux based systems that can be installed from the internet and allows you to receive and send infra-red signals via the energenie for many (but not all) commonly used remote controls. Raspberry Pi uses a version of Linux compatible with this package called Raspbian.

Key codes available for common remote controls can be downloaded from the LIRC website as configuration files and installed on your raspberry pi. You may also add to this list if your remote is not present and you can work out how to do it. This guide may help.

Set up can be confusing and difficult so we will attempt an understanding of the software and how to use it as easy as we can get it.

LIRC is fully described at: http://www.lirc.org/

lircd daemon

The most important part of LIRC when you install the package is the lircd daemon that will decode IR signals received by the device drivers and provide the information on a socket. It will also accept commands for IR signals to be sent if the hardware supports this. lircd being a daemon will run unseen in background on your R-Pi and provides an interface to any infrared signal applications you intend to use or create that are software driven.

irsend

The LIRC package contains the `irsend` tool for sending infrared signals to e.g. your TV or CD player.

Irrecord

This program will allow you to record the signals from your remote control and create a configuration file.

Make sure lircd isn't running, you have to kill the process before irrecord will work eg.:

```
sudo killall lircd
irrecord --list-namespace | grep KEY
```

Download and install LIRC on your R-Pi:

Include the xwindow stuff for graphical signal display.

Open up an LXTerminal window from your R-Pi desktop and type in:

```
sudo apt-get install lirc
sudo apt-get install lirc-x
```

Set up the GPIO pins to use

The driver is `lirc` and has 5 parameters: `debug, gpio_out_pin, gpio_in_pin, sense, softcarrier`

The default gpio pins to use when no pins are explicitly set are:

input pin for received infrared signal:	PIN12 - GPIO18
output pin for transmitted infrared signal:	PIN11 - GPIO17

You could change this as follows for a different hardware set up e.g.

```
modprobe lirc_rpi gpio_in_pin=18 gpio_out_pin=17
```

Alternatively edit your `/etc/modules` file to do this and type:

```
lirc_dev
lirc_rpi gpio_in_pin=18 gpio_out_pin=17
```

Edit your `/etc/lirc/hardware.conf` file as follows:

```
sudo su
cd /etc/lirc
sudo /leafpad hardware.conf
```

or

```
sudo /leafpad /etc/lirc/hardware.conf
```

```
########################################################
# /etc/lirc/hardware.conf
#
# Arguments which will be used when launching lircd
LIRCD_ARGS="--uinput"

# 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.
DRIVER="default"
# usually /dev/lirc0 is the correct setting for systems using udev
DEVICE="/dev/lirc0"
MODULES="lirc_rpi"

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

```
########################################################
```

Now restart lircd so it picks up these changes:

```
sudo /etc/init.d/lirc restart
```

Testing the IR receiver

Once all set up, as root, run a quick test. We need to stop the LIRC daemon and start mode2. mode2 shows the pulse/space length of infrared signals.

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

```
sudo modprobe lirc_rpi
sudo mode2 -d /dev/lirc0
```

This adds module to linux kernel then runs program to output mark-space of an ir signal

Point a remote control at your IR receiver and press some buttons. You should see something like this:

```
space 16300
pulse 95
space 28794
pulse 80
space 19395
pulse 83
space 402351
pulse 135
space 7085
pulse 85
space 2903
```

Testing Transmitter using LIRC

To test the Tx use an existing LIRC config file for your remote control or use your IR receiver to generate a new LIRC config file.

You can create a new /etc/lirc/hardware.conf file with the irrecord application that comes with LIRC by typing:

```
# Must stop lirc to free up /dev/lirc0
sudo /etc/init.d/lirc stop

# Create a new remote control configuration file (using /dev/lirc0) and save
the output to ~/lircd.conf
irrecord -d /dev/lirc0 ~/lircd.conf

# Make a backup of the original lircd.conf file
sudo mv /etc/lirc/lircd.conf /etc/lirc/lircd_original.conf

# Copy over your new configuration file
sudo cp ~/lircd.conf /etc/lirc/lircd.conf

# Start up lirc again
sudo /etc/init.d/lirc start
```

Once you've completed a remote configuration file save/add it to `/etc/lirc/lircd.conf`

Use the `irsend` application that comes with LIRC to send commands

```
irsend SEND_ONCE Remote_Name Remote_Button
```

Depending on what you've saved your remote name as. `SEND_ONCE` command will send a single remote control button signal each time you execute that command. *`Remote_Button`* Will correspond to the remote button signal you wish to send.

Alternatively it is possible to use existing remote control configurations. These existing configurations can be found on LIRC supported remotes index: http://lirc.sourceforge.net/remotes/

Note: not all remote configurations will be available on the LIRC index because not all remotes are supported by LIRC. In this circumstance please be advised to use the previous method of generating an LIRCD config file using `irrecord`.

Example:

In this example the selected remote controller will be the SKY+ HD Remote.

The first step is to find the config file on the LIRC index. This can found on http://lirc.sourceforge.net/remotes/sky/SKY+_DVB-S

Copy the content of the page (the config) or alternatively copy the content on page 7 of this document. then on the terminal access the lircd.conf file by typing the following

```
sudo nano /etc/lirc/lircd.conf
```

Clear the content of this file by holding both Ctrl and K. Paste the content from the LIRC index. Ctrl X to save. Then accept the changes by pressing Y for yes. Then hit enter to save under the same file name. Ensure that the lirc program is running.

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

Now to send a power button signal enter

```
irsend SEND_ONCE SKY+_DVB-S KEY_POWER
```

In order to send different button signals all you must do is alter the last field. Usually it will carry a `KEY` pre-fix.

Here is the SKY+ HD remote config file

```
#
# this config file was automatically generated
# using lirc-0.8.0(userspace) on Fri Jul 28 02:45:39 2006
#
# contributed by Lloyd Williams <binary_frog|chatcircuit.com>
#
# brand:                       SKY
# model no. of remote control: URC 1650-00 B00 - 9RC16P-1014 Sky+ Rev 6
# devices being controlled by this remote: SKY+ DVB-S receiver & PVR
#

begin remote

  name  SKY+_DVB-S
  bits            8
  flags RC6|CONST_LENGTH
  eps            30
  aeps          100

  header       2691   890
  one           427   460
  zero          427   460
  pre_data_bits   17
  pre_data       0x3FF3
  gap          149845
  min_repeat      2
  toggle_bit      0

  rc6_mask    0x100000

      begin codes
          KEY_POWER                0xF3                      #  Was: POWER
          TV_GUIDE                 0x33
          BOX_OFFICE               0x82
          SERVICES                 0x81
          INTERACTIVE              0x0A
          KEY_INFO                 0x34                      #  Was: INFO
          KEY_UP                   0xA7                      #  Was: UP
          KEY_LEFT                 0xA5                      #  Was: LEFT
          KEY_RIGHT                0xA4                      #  Was: RIGHT
          KEY_DOWN                 0xA6                      #  Was: DOWN
          KEY_SELECT               0xA3                      #  Was: SELECT
          KEY_CHANNELUP            0xDF                      #  Was: CH+
          KEY_CHANNELDOWN          0xDE                      #  Was: CH-
          KEY_TEXT                 0xC3                      #  Was: TEXT
          BACK_UP                  0x7C
          KEY_HELP                 0x7E                      #  Was: HELP
          FREV                     0xC2
          KEY_FASTFORWARD          0xD7                      #  Was: FFWD
          KEY_PLAY                 0xC1                      #  Was: PLAY
          KEY_PAUSE                0xDB                      #  Was: PAUSE
          KEY_RECORD               0xBF                      #  Was: RECORD
          KEY_STOP                 0xC0                      #  Was: STOP
          KEY_RED                  0x92                      #  Was: RED
          KEY_GREEN                0x91                      #  Was: GREEN
          KEY_YELLOW               0x90                      #  Was: YELLOW
          KEY_BLUE                 0x8F                      #  Was: BLUE
          KEY_1                    0xFE                      #  Was: 1
          KEY_2                    0xFD                      #  Was: 2
          KEY_3                    0xFC                      #  Was: 3
          KEY_4                    0xFB                      #  Was: 4
          KEY_5                    0xFA                      #  Was: 5
          KEY_6                    0xF9                      #  Was: 6
          KEY_7                    0xF8                      #  Was: 7
          KEY_8                    0xF7                      #  Was: 8
          KEY_9                    0xF6                      #  Was: 9
          KEY_0                    0xFF                      #  Was: 0
          SKY                      0x7F
          KEY_TV                   0x7B                      #  Was: TV
      end codes

end remote
```