Irdroid-Rpi™
Infrared Transceiver HAT for Rpi

Draft

*All specifications are subject to change

Contents

Draft

Version History

Issue No	Date Released	Description of Change	Owner	Notes
1.0	03/11/2016	Initial document release	GB	

Draft

Overview

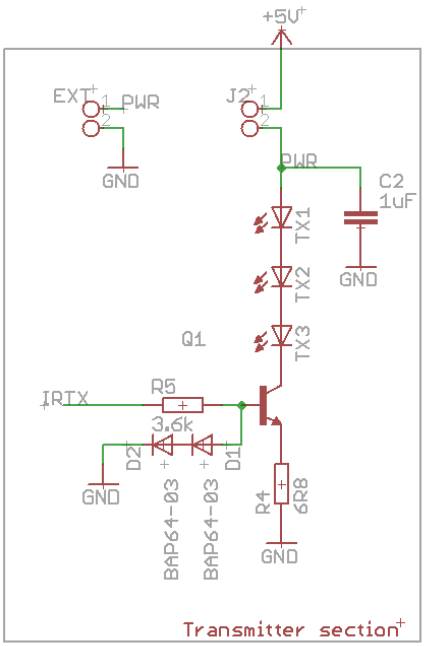
The Irdroid-Rpi™ is a Infrared Transceiver stackable board or the so called HAT for Raspberry Pi . The Irdroid-Rpi™ allows both transmitting and receiving of infrared signals. The Irdroid-Rpi™ Infrared transmitter part is wired with three powerful infrared LEDs which emit infrared signals @ the commonly used infrared wavelength of 940nm. The module is designed and uses optical signal amplification, which combines the Infrared light emitted by the three infrared led's and significantly increases the infrared transmitting range.

The board interface with the Raspberry Pi is a 2x20 pin female header connector. Two GPIOs are used to control the Transmitter and Receiver part. The board also provides an Interface to the Olimex development boards and sensors via the Olimex's UEXT connector. The connector is wired following the Olimex recommendations for wiring a UEXT Host. The UEXT interconnect provides 2 Serial interfaces (1xUART and 1xSPI), 1 I2C interface and 3.3V supply.
The Irdroid-Rpi™ board also has a place for a I2C EEPROM memory which is optional and not installed by default. The Irdroid-Rpi™ board in combination with LIRC for raspberry Pi can be used to control your home infrared consumer electronics like TVs, DVDs, Air-conditioners and even more.

Module Schematic

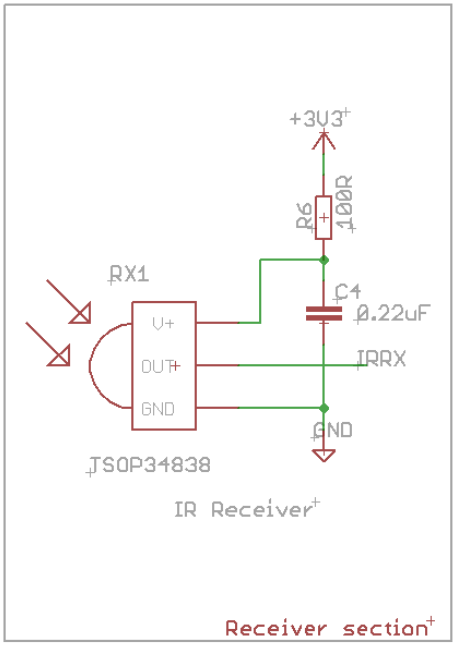
Below are the sections from the schematic responsible for the different actions like, Infrared transmitting, infrared receiving, I2C EEPROM and Olimex UEXT Interconnect section.

Infrared transmitter section



The image on the left shows the infrared transmitter section. The transmitter is powered by the Rpi 5V supply pins. We have envisaged alternative power supply in case the user decides to alter the schematic or change Infrared Leds. The jumper J2 is normally shorted and the Infrared Transmitter section is supplied with 5V DC from the Rpi board. There is a possibility to supply the board with a external 5V power supply by disconnecting jumper J2 and shorting the "EXT" jumper. As mentioned above the schematic allows for optical amplification of the transmitted infrared signal which increases the infrared range. The Infrared transmitter is controlled by the Rpi pin 11 (GPIO17).

Infrared receiver section



The Infrared Receiver section consist of custom infrared receiver IC and it is supplied via the Raspberry Pi 3.3V power supply.

The infrared receiver IC can receive infrared signals @ up to 40khz which is the most common frequency used for consumer electronics infrared remote control.

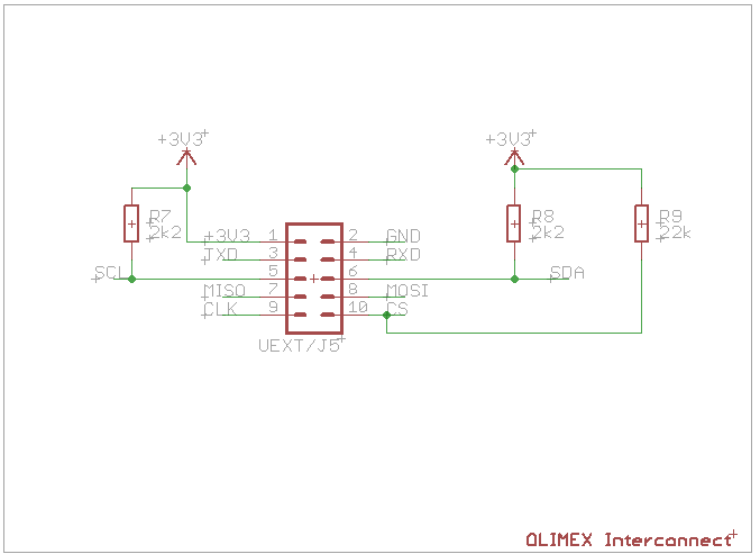
The Infrared receiver IC DATA output pin is wired to the RPI pin 12 (GPIO18).

If required the user could de-solder and use a different Infrared Receiver IC of his choice.

Olimex UEXT Interconnect

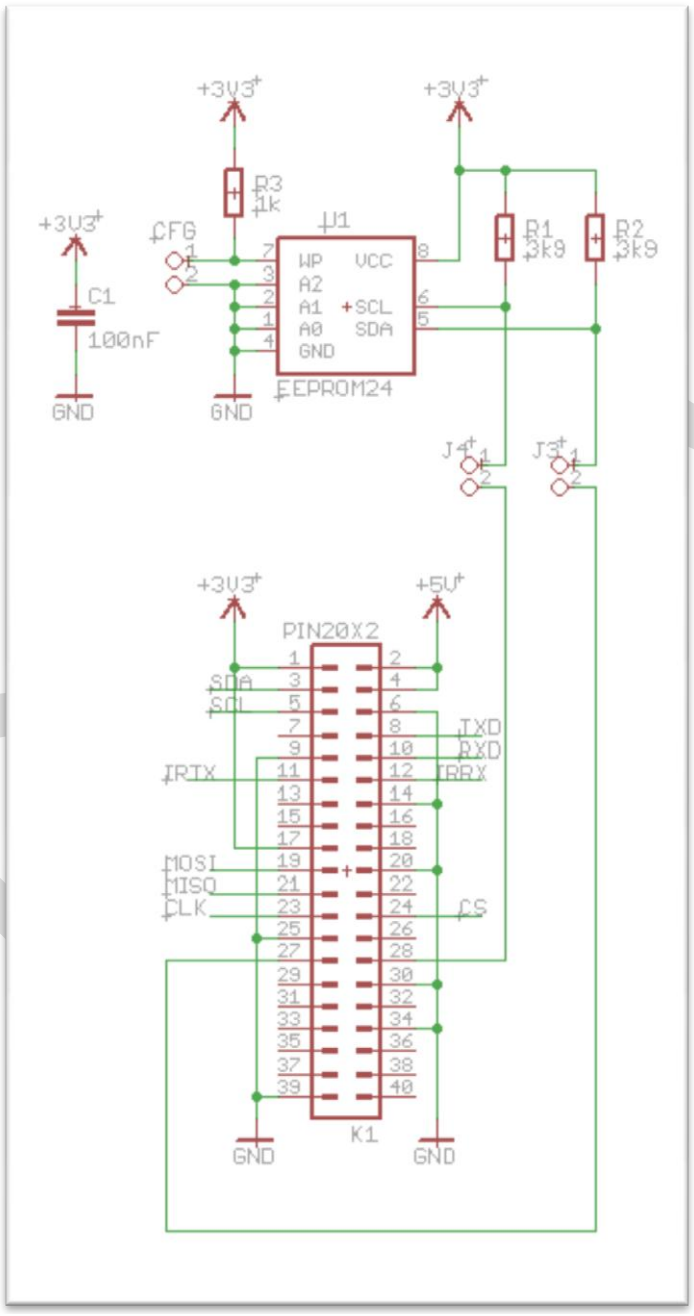
The Irdroid-Rpi™ Infrared Transceiver HAT is equipped with a Olimex UEXT connector that would allow the users to connect various sensors and devices available for purchase from Olimex website - http://www.olimex.com .

The Olimex UEXT is wired for HOST mode which means that the UART RX/TX pins should be crossed at device end of the connection.



I2C EEPROM Section

The Irdroid-Rpi™ is designed with a possibility to solder a I2C eeprom. below is the target schematic, which also indicates the RPi 2x20 pin header pinout.



Software for Irdroid-Rpi

The Irdroid-Rpi Infrared Transceiver HAT is designed for Rpi and the software that is commonly used is LIRC (Linux Infrared Remote control - see http://www.lirc.org). By using LIRC the user gets access to more than 2000+ already digitized infrared remote control codes that are in the LIRC database of supported remotes. The LIRC Software package is available for Almost any distribution of Linux. In this Manual we will describe the case of using the Linux Distribution for Rpi - Raspbian.

A detailed information and howtos for lirc can be found @ http://www.lirc.org .

The LIRC software packages has a number of utilities responsible for different actions like infrared transmitting, infrared receiving , serving the data to a socket.

The LIRC Daemon

The LIRC Daemon is a Linux daemon that runs as a process in linux and takes care of the data received via the device drivers and routes that data to a TCPIP Socket. It also listens for commands sent via Socket on a Specific port. The Lirc daemon runs in the background and it provides an interface to the infrared remote control application (LIRC Client). There are many Lirc clients , for Android , IOS, Linux and Windows.

The LIRC irsend utility

The irsend utility is used to transmit infrared commands to your target infrared equipment to be controlled like TVs , STBs , Air-con etc. The utility communicates with the LIRC daemon. The Utility has the following example syntax:

> irsend SEND_ONCE SAMSUNG_TV power

The above command send the command "power" for Device "Samsung_TV" and send it once with no repeat.

The Device should be available in the LIRC configuration file in order for the above command to be executed.

The LIRC irrecord utility

The program will allow you to record infrared commands from your existing remote control and create a LIRC configuration file for this particular remote control.

In order to use it you should make sure that the lircd is not working. Simply issue:

> sudo killall lircd

Draft

> irrecord --list-namespace | grep KEY

Downloading and installing LIRC on Rpi

The LIRC software package can be downloaded and installed using the apt-get command :

> sudo apt-get install lirc
> sudo apt-get install lirc-x

GPIO Pin configuration for LIRC

The default gpio pins for the application are:

input pin for receiving infrared signals: PIN12/GPIO18
output pin for transmitting ir signals is : PIN11/GPIO17

The above default configuration is compatible with Irdroid-Rpi so practically you should not change it. There is a way to change the default pins by issuing the following command :

> modprobe lirc_rpi gpio_in_pin=18 gpio_out_pin=17

You could also edit the /etc/modules and change the pins there :

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
```

Draft

```
# 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=""
5 of 9
############################################################
```

Restart lircd so it picks up these changes:
sudo /etc/init.d/lirc restart

Testing the Irdroid-Rpi Infrared receiver

Once the the configurations described above are completed you can test the Infrared Receiver. To do that you need to stop the LIRC daemon by issuing the following command:

> sudo /etc/init.d/lirc stop
> sudo modprobe lirc_rpi
> sudo mode2 -d /dev/lirc0

the aboe commands first stop the lirc daemon , then it loads the lirc_rpi device driver module to the linux kernel and finally runs the program mode2 to show the mark - space of an infrared singal.

Now you could point a remote control toward the Irdroid-Rpi and press some buttons. You should be able to see similar output in the console as below:

```
space 402351
pulse 135
space 7085
pulse 85
space 2903
```

Testing the Irdroid-Rpi Infrared Transmitter

In order to test the Irdroid-Rpi infrared transmitter part you need to get a lirc configuration file for the target infrared equipment to be controlled . You can try searching in the LIRC database of remotes available at http://lirc.sf.net/remotes . You could also generate a lirc conf file by recording the buttons of your remote control.

You must stop lirc to free up /dev/lirc0 by issuing the following command
> 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

Now you can use the irsend application that comes with the LIRC package for linux and you can issue a command using the syntax below:

> irsend SEND_ONCE Remote_Name Remote_Button

You could also download other remote control configuration files from the database of existing digitized remotes available at http://lirc.sf.net/remotes.

Example LIRC configuration file

The LIRC configuration file is used to store remote control codes either in RAW space pulse format or decoded HEX values, depending on the remote control protocol. Below is a example configuration file for TESY home panel heater unit.

```
# Please make this file available to others
# by sending it to <lirc@bartelmus.de>
#
# this config file was automatically generated
# using lirc-0.9.0(IRToy) on Thu Sep 17 21:47:15 2015
#
# contributed by
#
# brand:                       Tesy
# model no. of remote control:
# devices being controlled by this remote:
#

begin remote

  name  Tesy
  bits           16
  flags SPACE_ENC
  eps            30
  aeps          100

  header       8969  4395
  one           599  1607
  zero          599   489
  ptrail        604
  repeat       8981  2156
  pre_data_bits   16
  pre_data       0xFF
  gap          39670
  repeat_gap   95433
  toggle_bit_mask 0x0
```

```
      begin codes
          power                    0x50AF
          plus                     0x20DF
          minus                    0xE01F
          timer                    0xF807
          temp                     0x08F7
      end codes

end remote
```

As you can see there are specific section for every remote in this file which means that you can have multiple remotes defined in lircd.conf file .

The "begin remote" section defines the section start for a new remote control and the "end remote" section end for the remote . You can have as many remotes as you want defined in a single configuration file.

Draft