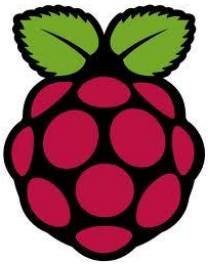
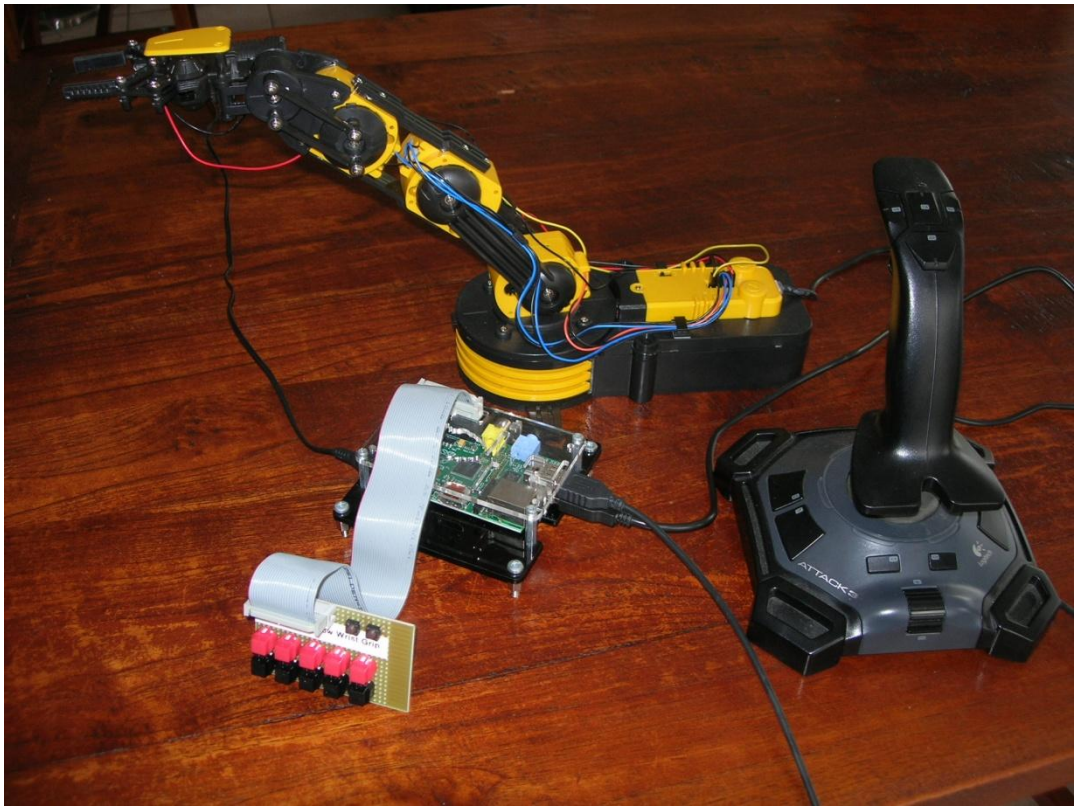


Raspberry PI Robot Arm

Constructors Manual



Bob Rathbone Computer Consultancy

www.bobrathbone.com

2nd of November 2013

Contents

Introduction

This project has been designed to help students get started with a Robot on the Raspberry PI. The principle hardware required to build the Robot Arm consists of the following components:

- A Raspberry PI computer
- A Maplin Robotic Arm
- Optional Joystick (Logitech or similar)
- Optional twelve button interface (Self build)

Raspberry PI computer

The **Raspberry Pi** is a credit-card-sized single-board computer developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools.

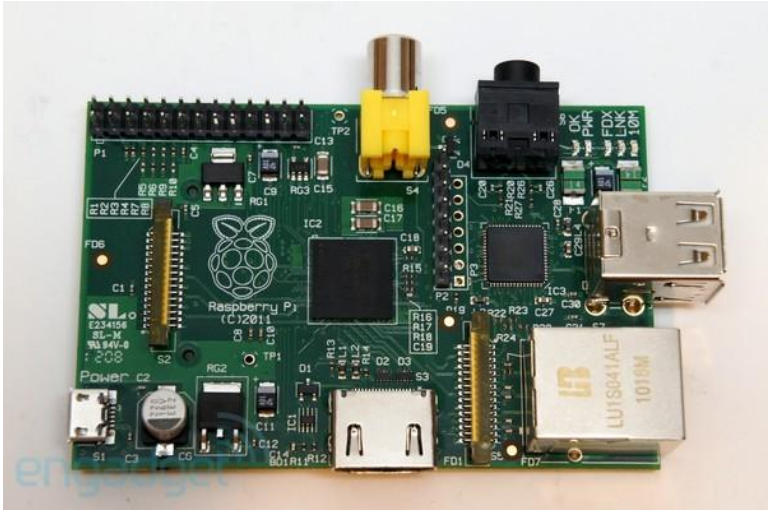


Figure 1 Raspberry PI Computer

More information on the Raspberry PI computer may be found here:
http://en.wikipedia.org/wiki/Raspberry_Pi

If you are new to the Raspberry PI try the following beginners guide. http://elinux.org/RPi_Beginners

Features

The Robot Arm interface provides five separate ways of driving a Maplin Robot Arm. These are:

- A USB Joystick (Logitech or similar)
- A twelve push button interface
- A keyboard interface
- Using an input file containing the required robot commands
- A command line interface using required robot commands

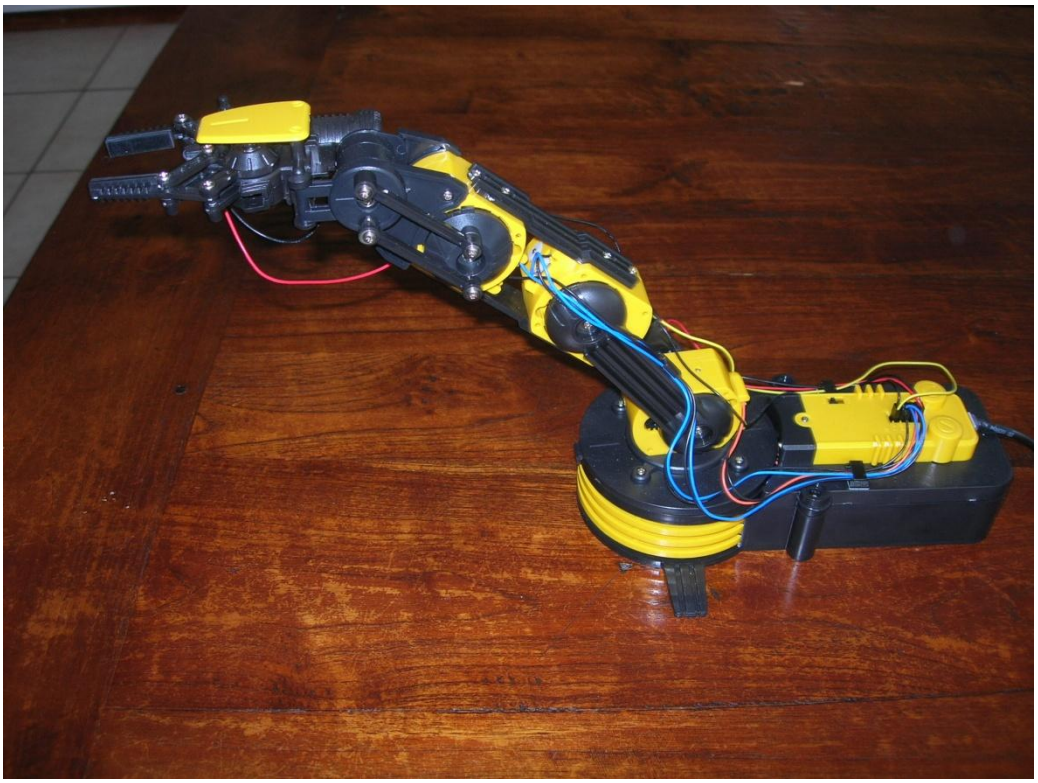


Figure 2 Maplin Robotic Arm

Limitations

The Maplin Robot Arm does not have any positional feedback that the program can make use of. The program has no idea of the current position of the robot arm or gripper. This means that you must observe the Robot Arm position whilst moving it around to position it to say pick up an object. This means that the robot arm can never be programmed to carry out a repetitive task with any accuracy. This is purely a fun project with very limited application.

Parts List

The following table shows the parts list for the Raspberry PI Robot interface.

Qty	Part	Description	Supplier
1	Raspberry PI	Raspberry PI credit card computer	Farnell or RS
1	Logitech ATK Joystick	Two axis joystick with at least 8 buttons	Any computer store
1	Maplin Robot Arm	Maplin Robotic Arm with USB interface	Maplin Electronics (UK)
1	4 Gigabyte SD card	For the Raspian Squeeze operating system	Any computer or photographic store
1	Prototype board	Button interface board	Tandy or Farnell Element 14
10	Push to make PCB mount button	Button interface board push buttons (Robot movement buttons)	Tandy or Farnell Element 14
2	Push to make miniature PCB mount button	Button interface board push buttons (light on/off)	Tandy or Farnell Element 14
1	26 way PCB mount male connector	To take the ribbon cable connection to the button interface board	Tandy or Farnell Element 14
1	26 way ribbon cable	Ribbon cable connection to the button interface board	Tandy or Farnell Element 14
	Wiring	Thin wire for PCB wiring prototype board	Any electronics shop

Button Interface Wiring

Wire 3.3v on pin 1 to one side of all of the switches. Wire the other side to the GPIO pin shown in the following table.

Table 1 Button interface wiring list

GPIO pin	Switch	Description
1	All switches	Common +3.3 volt
7	Base clockwise	
8	Base anti-clockwise	
11	Shoulder Up	
10	Shoulder Down	
13	Elbow Up	
12	Elbow Down	
18	Wrist Up	
16	Wrist Down	
21	Gripper Open	
19	Gripper Close	
22	Light On	
23	Light Off	

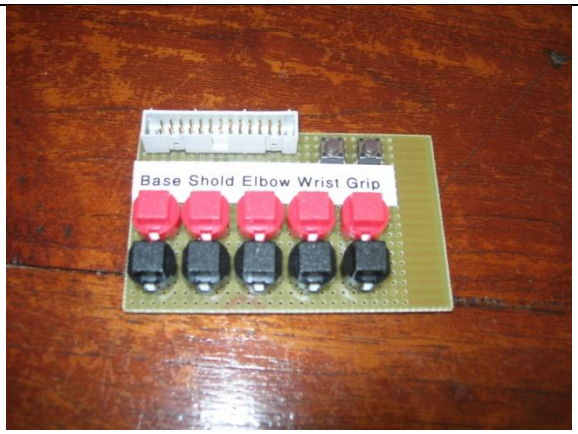


Figure 3 The twelve button interface board

The picture on the left shows the completed twelve button interface board. The larger red buttons are the Up or Open (grip) functions. The larger black buttons are the Down or Close (gripper) functions. The two smaller buttons on the top left are the LED light on and off respectively.

Figure 4 The underside of the twelve button interface

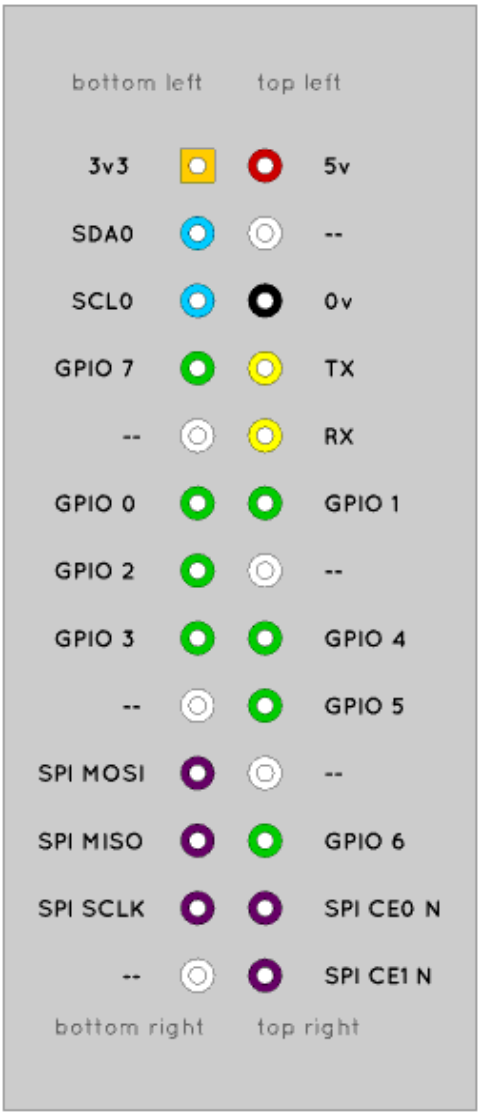
The actual wiring as shown by this view of the underside of the board is relatively simple. It consists of wiring one side of all of the switches to the 3.3v on pin 1. Use a volt meter to test between pin 6 (GND) of the GPIO header and the switches voltage supply is 3.3 volts and not 5 volts. The other side of each switch is wired to the appropriate GPIO pin shown in Table 1 above.

Caution: Do not wire the switches to the +5 volt supply on pin 2 by mistake or you will irreparably damage the raspberry PI. Check for the correct voltage before continuing to wire the switches.

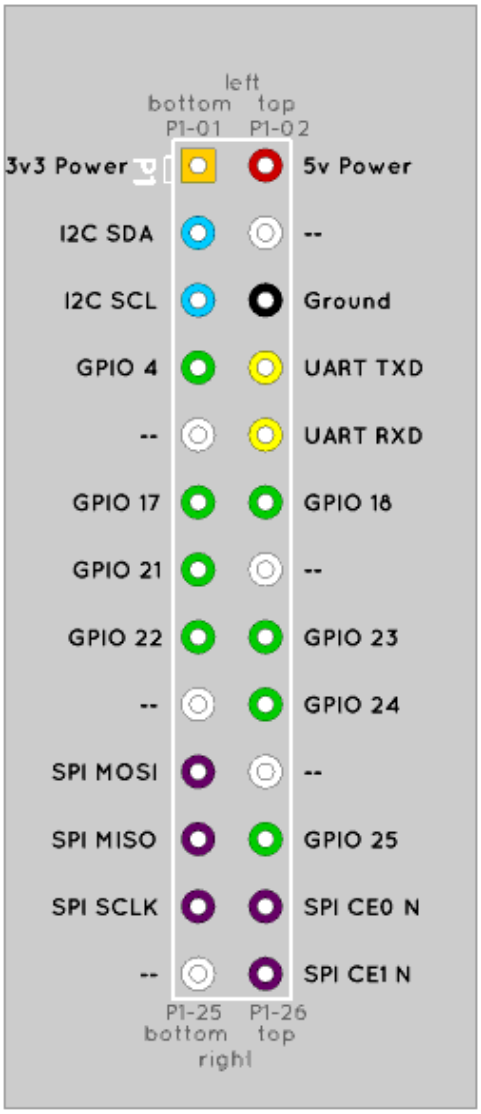
GPIO Hardware Notes

The following shows the pin outs for the GPIO pins. For more information see:
http://elinux.org/RPi_Low-level_peripherals

Raspberry PI GPIO numbering



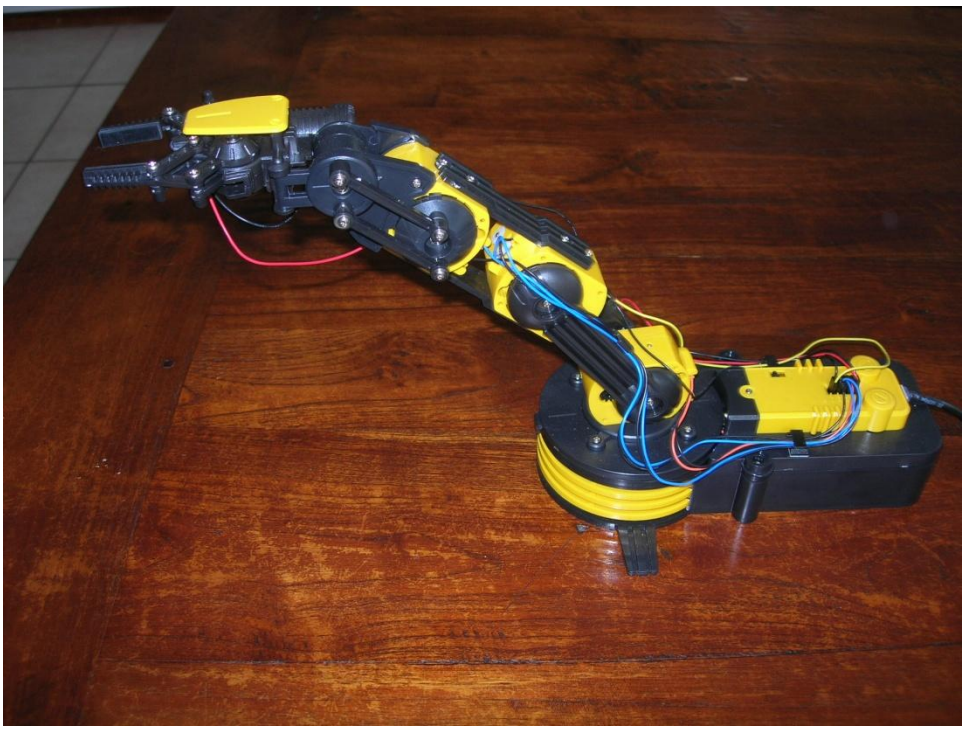
Raspberry GPIO Broadcom numbering



Note: On rev 2 boards GPIO21 is now GPIO27

Building the Maplin Robot Arm

The full instructions for building the Maplin Robot arm comes with the kit you purchase from Maplin Electronics. There is also the necessary Microsoft Windows software on a CD to operate and test the robot arm once it has been built. Make sure it is working with the Maplin Windows software before attempting to use it with the Raspberry PI software.



Build the robot arm as per the Maplin *Assembly and Instruction manual*.

Prepare the Raspberry PI

Before you can start using the robot arm, there are a few things you will have to do to get your Raspberry Pi ready for the robotic arm and joystick. The first thing you will need to do is add the user account you are using on the Raspberry Pi to the 'plugdev' group. Assuming you are using the default 'pi' user account, the command to do this will be:

```
sudo usermod -aG plugdev pi
```

The next step is to add a udev rule to allow the 'pi' user to send commands to the arm. To do this, create a file called **/etc/udev/rules.d/85-robotarm.rules**, using the command:
sudo nano **/etc/udev/rules.d/85-robotarm.rules** with the contents below:

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="1267",
ATTRS{idProduct}=="0000", ACTION=="add",
GROUP="plugdev", MODE="0666"
```

To save the file, press ctrl-x and then press 'Y' followed by Enter.

You'll then need to install the Python USB libraries. The one from the Apt repository is slightly out of date, so you need to install the Apt version first and then upgrade it

using the pip command. The command below will do this:

```
sudo apt-get install python-pip -y
sudo pip install pyusb --upgrade
```

When the Python USB libraries have been installed, reboot your Raspberry Pi so that the changes to udev and the 'pi' user account take effect with the command:

```
sudo shutdown -r now
```

Disable serial interface

Two of the pins used by this design namely pin 8(GPIO8) and 10 (GPIO 15) are configured for the serial interface (UART). This must be disabled for reliable operation by removing all references to *ttyAMA0* in the **/boot/cmdline.txt** and **/etc/inittab** files

Add a hash character at the beginning of lines containing ttyAMA0.
/boot/cmdline.txt

```
#dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

Add the following line to the **/boot/cmdline.txt** file. This is the same as the above line but with all references to ttyAMA0 removed.

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4
elevator=deadline rootwait
```

Edit /etc/inittab to disable the serial interface
/etc/inittab

```
#Spawn a getty on Raspberry Pi serial line
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Now reboot the Raspberry PI to disable the serial interface.

Connect the robot arm

When the Raspberry Pi has been rebooted, connect the arm to a free USB port and turn it on. A USB hub should not be needed here. To check that the Pi has successfully detected the arm, execute the command:

```
dmesg | grep usb | grep 1267
```

I f the arm has been detected successfully you should see output similar to the line below:

```
[ 252.790554] usb 1-1.3: New USB device found, idVendor=1267, idProduct=0000
```

I f you don’t see the result above, turn off the arm using the power switch, unplug from the USB port, plug the arm back in and turn the power back on.

Software installation

This procedure assumes that the Raspberry PI is installed with Debian Wheezy and with a working Internet Connection. There are a few steps to carry out to install the software.

- Download and un-tar the software from the Bob Rathbone web site
- Create a directory **/home/pi/robot**
- Un-zip the program files into to the above directory
- Make ally python files executable

Software download

The software is contained in a compressed tar file called *pi_robot.tar.gz*. This can be downloaded from the following location:

http://www.bobrathbone.com/raspberrypi/source/pi_robot.tar.gz

Either download it to the PC and copy it across to the Raspberry PI or use the *wget* facility if there is an Internet connection on the Raspberry PI.

Create a directory called **/home/pi/robotarm**.

```
# mkdir -p /home/pi/robotarm
```

Copy the *pi_robot.tar.gz* to the **/home/pi/robotarm** directory or use **wget** to download it.

```
# cd /home/pi/robotarm
# wget http://www.bobrathbone.com/raspberrypi/source/pi_robot.tar.gz
```

Un-tar the file with the following command:

```
# tar -xvf pi_robot.tar.gz
```

This will unzip the following files and directory:

log_class.py robot_daemon.py robotd.py

Make sure all of the *.py files in this directory are executable with the following command:

```
# chmod +x *.py
```

Running the robotic arm program

Connect the Robotic Arm to any USB port.

Connect the joystick to the second USB port.

If you have one, the twelve button keypad to the GPIO header.

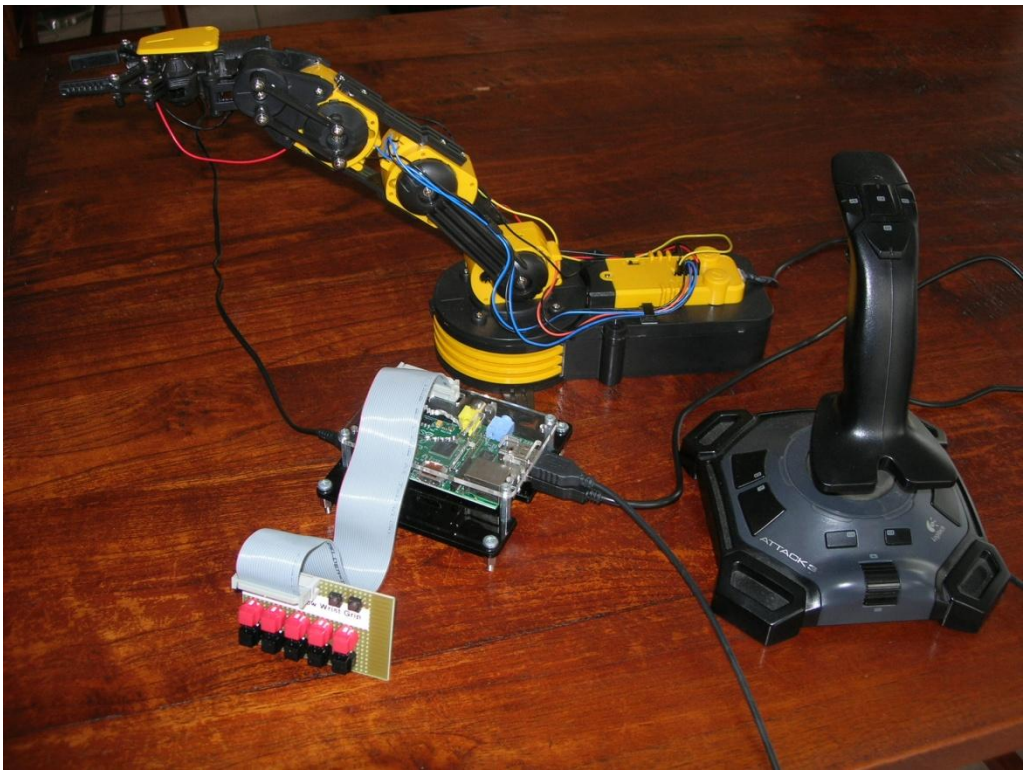


Figure 5 Connecting up the robotic arm

Log in to the Raspberry as user pi and issue the following commands:

```
$ sudo bash
# cd robotarm
# ./robotd.py
```

This will display the following message:

```
Usage: ./robotd.py start|stop|restart|status|version|<command>
Commands: keyboard        - Use keyboard
          execute <file>  - Execute commands in <file>
```

Using the keyboard

Issue the following command:

```
# ./robotd.py keyboard
```

This will display the following:

```
Key Command
--- -------
a   base-anti-clockwise
c   shoulder-down
b   wrist-down
d   shoulder-up
g   wrist-up
f   elbow-up
h   grip-open
k   stop
j   light-on
m   light-off
n   grip-close
v   elbow-down
z   base-clockwise
x   Exit program
Enter command:
```

Now press the appropriate keys on the keyboard to operate the robot arm. Press key 'x' to exit the program.

If this doesn't work go to the troubleshooting section on page 16 to determine the exact problem. There is no point attempting to use the joystick or twelve button keypad until the above commands are working.

Using the Joystick

To use the joystick first start the robot daemon. A daemon is a special type of program which runs in the background and is not connected to a terminal.

```
# ./robotd.py start
```

If it started correctly you can check its status with the following command:

```
# ./robotd.py status
robotd running pid 2437
```

It will display the process ID (pid) of the running **robotd** daemon. The **pid** displayed will be different each time the program is run.

You can now operate the joystick. Left and right of the joystick will turn the base clockwise and anti-clockwise. The buttons will move the shoulder, wrist and the gripper. Two more buttons will operate the LED light.

See Table 2 on page 13 for the complete set off commands. Joystick buttons are labelled 0 onwards.

Table 2 Joystick Commands

Command	Joystick movement	Joystick Button
grip-close		0
grip-open		1
wrist-up		9
wrist-down		10
elbow-up		6
elbow-down		5
shoulder-up	back	
shoulder-down	forward	
base-clockwise	left	4
base-anti-clockwise	right	3
light-off		7
light-on		8

Using the twelve button keypad

This is optional. If you did not build the keypad then you can use operate the robotic arm using the joystick only (and of course using the keypad). The following table and illustration show the robotic arm to button mapping.

Table 3 Twelve button board map

Command	Button	Colour
base-clockwise	Base	Red
base-anti-clockwise	Base	Black
shoulder-up	Should	Red
shoulder-down	Should	Black
elbow-up	Elbow Up	Red
elbow-down	Elbow Down	Black
wrist-up	Wrist Up	Red
wrist-down	Wrist Down	Black
grip-open	Grip Up	Red
grip-close	Grip Down	Black
light-off	Top left	Grey
light-on	Top right	Grey

Display the software version

Issue the following command:

```
# ./robotd.py version
Version  0.1
```

Stopping the robot program

```
# ./robotd.py stop
Version  0.1
```

Using an input file

It is possible to execute commands from an input file using the execute option. To use a commands file use the execute command followed by the name of the commands file.

```
# ./robotd.py execute <commands file name>
```

For example to execute the commands in a file called commands.txt, enter the following.

```
# ./robotd.py execute commands.txt
```

Table 4 lists all commands and the maximum travel time and half movement travel time. For example if you wish to rotate the base clockwise from its centre position to complete left the command in the commands text file would be:

```
base-clockwise 10
```

Table 4 File input commands and maximum travel time in seconds

Command	Maximum time In seconds	½ travel time
grip-close	3	1.5
grip-open	3	1.5
wrist-up	9	4.5
wrist-down	9	4.5
elbow-up	13	6.5
elbow-down	12	6.5
shoulder-up	14	7
shoulder-down	14	7
base-clockwise	20	10
base-anti-clockwise	20	10
light-off	n/a	n/a
light-on	n/a	n/a
wait	any	n/a

n/a Not applicable

All instructions are entered into a small text file using **nano** or **vi** (If doing this on a windows PC use either **wordpad** or **notepad**, **word** is not suitable as it contains formatting characters)

Example command file

```
light-on 0
base-clockwise 1.5
wait 2
grip-close 0.5
light-off 0
```

This switches the light on the Robotic arm on, rotates the arm clockwise for 1.5 seconds, wait two seconds, closes the gripper for 0.5 seconds and then switches the light off. The wait command can have any integer or decimal value.

Robot program logging

The Robot program logs to a file called **/var/log/robot.log.** See example log below

```
2013-09-23 17:17:10,663 INFO Robot daemon running pid 2648
2013-09-23 17:17:11,147 INFO Initialized Joystick :Logitech Logitech Attack
3
2013-09-23 17:17:11,451 DEBUG Listening for commands
2013-09-23 17:17:19,197 DEBUG sendCommand (0, 2, 0)
2013-09-23 17:17:19,833 DEBUG sendCommand (0, 0, 0)
2013-09-23 17:17:21,184 DEBUG sendCommand (0, 1, 0)
2013-09-23 17:17:21,918 DEBUG sendCommand (0, 0, 0)
2013-09-23 17:17:22,859 DEBUG sendCommand (64, 0, 0)
2013-09-23 17:17:23,503 DEBUG sendCommand (0, 0, 0)
2013-09-23 17:17:24,342 DEBUG sendCommand (128, 0, 0)
2013-09-23 17:17:24,904 DEBUG sendCommand (0, 0, 0)
2013-09-23 17:17:28,347 DEBUG JOYBUTTONDOWN 6
2013-09-23 17:17:28,360 DEBUG Command elbow-up
2013-09-23 17:17:28,369 DEBUG execute elbow-up
2013-09-23 17:17:28,376 DEBUG Light on =  False
2013-09-23 17:17:28,401 DEBUG sendCommand [16, 0, 0]
2013-09-23 17:17:28,426 DEBUG A light = False
2013-09-23 17:17:28,634 DEBUG JOYBUTTONUP 6
2013-09-23 17:17:28,642 DEBUG light_on False
2013-09-23 17:17:28,649 DEBUG sendCommand [0, 0, 0]
2013-09-23 17:17:29,585 DEBUG JOYBUTTONDOWN 5
2013-09-23 17:17:29,602 DEBUG Command elbow-down
2013-09-23 17:17:29,612 DEBUG execute elbow-down
2013-09-23 17:17:29,618 DEBUG Light on =  False
2013-09-23 17:17:29,642 DEBUG sendCommand [32, 0, 0]
2013-09-23 17:17:29,680 DEBUG A light = False
2013-09-23 17:17:29,887 DEBUG JOYBUTTONUP 5
2013-09-23 17:17:29,894 DEBUG light_on False
2013-09-23 17:17:29,901 DEBUG sendCommand [0, 0, 0]
```

There are four levels of logging namely DEBUG, INFO, WARNING and ERROR. The log level is configured in the **/var/lib/robotd/loglevel** file. The default is INFO. If you want to increase the logging say to DEBUG carry out the following command as root user and restart the program.

```
# echo DEBUG > /var/lib/robotd/loglevel
# ./robotd.py restart
```

Trouble shooting problems

The Raspberry PI hangs whilst booting up with the Robot arm and Joystick connected

Disconnect the joystick and robotic arm from the Raspberry PI. Reboot the raspberry PI and once fully rebooted, switch on the robotic arm and connect it along with joystick to the USB ports.

Appendix A Robot arm commands

Ignore this section unless you are attempting to understand how the code works.

If you take a look at the program code you may wonder how are commands to operate the robot arm constructed. Commands consist of three bytes which control the robotic arm. These are: *Arm*, *Base* and *LED* light respectively. Base and light commands can be combined with any Arm commands.

Table 5 Robot arm commands

Command	Arm	Base	Light
stop	0	0	x
grip-close	1	y	x
grip-open	2	y	x
wrist-up	4	y	x
wrist-down	8	y	x
elbow-up	16	y	x
elbow-down	32	y	x
shoulder-up	64	x	x
shoulder-down	128	y	x
base-clockwise	x	1	x
base-anti-clockwise	x	2	x
light-off	x	y	0
light-on	x	y	1

x can be either 0 or 1
y can be either 1 or 2

Table 6 Robot arm command mapping (default)

Command	Axis	Value	Joystick Button
stop	n/a	n/a	n/a
grip-close	n/a	n/a	0
grip-open	n/a	n/a	1
wrist-up	n/a	n/a	9
wrist-down	n/a	n/a	10
elbow-up	n/a	n/a	6
elbow-down	n/a	n/a	5
shoulder-up	1	positive	7
shoulder-down	1	negative	8
base-clockwise	0	negative	4
base-anti-clockwise	0	positive	3
light-off	n/a	n/a	7
light-on	n/a	n/a	8

The above mapping is for a Logitech Attack 3 joystick.

Table 7 Twelve button board GPIO mapping

Command	GPIO pin
grip-close	19
grip-open	21
wrist-up	18
wrist-down	16
elbow-up	13
elbow-down	12
shoulder-up	11
shoulder-down	10
base-clockwise	7
base-anti-clockwise	8
light-off	23
light-on	22

Appendix B Licences

The software and documentation for this project is released under the GNU General Public Licence.

The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software. Software that ensures that these rights are retained is called free software. The license was originally written by Richard Stallman of the Free Software Foundation (FSF) for the GNU project.

The GPL grants the recipients of a computer program the rights of the Free Software Definition and uses *copyleft* to ensure the freedoms are preserved whenever the work is distributed, even when the work is changed or added to. The GPL is a *copyleft* license, which means that derived works can only be distributed under the same license terms. This is in distinction to permissive free software licenses, of which the BSD licenses are the standard examples. GPL was the first *copyleft* license for general use.

See http://www.gnu.org/licenses/#GPL for further information on the GNU General Public License.

Acknowledgements

The idea and the basic code for this project came from an article written by Peter Lavelle in Magpi Magazine issue 14.

The software to read the joystick inputs to control the Maplin robotic arm is based on software from www.mybigideas.co.uk (Author unknown).

Glossary

CLI	Command Line Interface
GPIO	General Purpose IO (On the Raspberry PI)
LED	Light Emitting Diode
USB	Universal Serial Bus