minordiscoveries

Recording solutions for Arduino, Raspberry Pi, and Lego RCX/NXT

Using NQC on a Raspberry Pi to Program a Lego Mindstorms RCX Brick

Lego Mindstorms

Seymour Papert is one of the great pioneers of computer science, as well as also being a pioneer of education in computer science and mathematics. He is widely known for his invention, whilst working at MIT, of the Logo programming language, work which he described in his famous book *Mindstorms: Children, Computers and Powerful Ideas*. His MIT research group eventually became the foundation of what is now the renowned MIT Media Lab. The LEGO Company began sponsoring work at the MIT Media Lab in 1985, and this collaboration resulted in the release in 1998 of the LEGO Mindstorms construction kit, which was named after Seymour Papert's seminal book.

In 1998 Lego released the first version of their Mindstorms Robotics Invention System (RIS). This robotics system used standard and technical Lego bricks together with an 8-bit microcontroller (a Hitachi/Renesas H8/3297 microcontroller) packaged into a neat yellow box known as the RCX brick. The RCX brick was programmed via an infrared (IR) serial link from a Windows 95/98 PC, and could drive up to 3 motors and receive data from up to 3 sensors. The Lego Mindstorms RIS product was the fruit of a long-standing research collaboration between the Lego company and the MIT Media Lab. The MIT Media Lab created the forerunner of the RCX brick, which was simply known as "the programmable brick". If you are curious about the MIT programmable brick, then have a look at the Programmable Brick Handbook, available as a PDF from Fred Martin's website at the University of Massachusetts Lowell here (Fred Martin was involved in the Lego Mindstorms work whilst a student at MIT).

The Mindstorms RIS sets are really good, and used kits remain readily available on ebay. The programmable RCX bricks have proven to be very robust, and they remain an excellent system for children to experiment with. In 2006 Lego launched a more sophisticated successor to the Mindstorms RIS, called Mindstorms NXT which contained a more sophisticated programmable brick based on a 32-bit ARM7TDMI-core Atmel AT91SAM7S256 microcontroller. The NXT system has itself now been succeeded by the Mindstorms EV3 system which was released in 2013 and contains a ARM9-based processor running Linux!

While the NXT and EV3 systems offer much more modern technology, the original RCX based Mindstorms system still has a lot to recommend it, particularly for younger roboticists. The RCX system is more akin to standard Lego kits, and I think the RCX is quite a bit easier for younger children to build interesting things with than either the NXT or EV3 systems. There's still a lot of good stuff on the web for the RCX system – see for example the RCX projects on these two websites:

http://www.philohome.com/mindstorms.htm

http://www.marioferrari.org/lego_mindstorm.html

There are also loads of good YouTube videos on building all sorts of RCX projects – see: http://www.youtube.com/watch?v=XsxQYmUzI_o for starters. Of course, YouTube contains a wealth of other RCX projects to inspire you! There's everything there from RCX based lathe's to piano players.....

Just a note of caution on using an external power supply with the version 1.0 RCX brick that I came across at the very useful pbrick.info website – it turns out that the RCX brick expects to be powered by 9-12V **AC** rather than

the more usual DC. I have in fact successfully powered my RCX brick with a 9V DC power supply, but as the `pbrick.info` site warns, there is a risk that using DC will overheat the RCX, and could burn out the electronics! So, if you're planning to use an external power supply with your version 1.0 RCX brick, make sure you're using a suitable AC power supply – the power supply used for recharging the Lego NXT brick's rechargeable battery is suitable for use with the RCX, and these power supplies are readily available on the second hand market.

Programming a Lego RCX using NQC on a Raspberry Pi

A potential difficulty in using an old Mindstorms RIS kit is that the original Lego Mindstorms software is dependent on obsolete versions of Microsoft Windows (95 or 98) to run. The original Lego software provided quite a nice graphical programming system, somewhat similar to MIT Media Lab's Scratch programming system. Fortunately, there are a number of alternative options for programming the RCX, including the open source NQC (Not Quite C) programming language. NQC was originally created by David Baum as a comprehensive C like language for programming the Lego Mindstorms RCX system, and which can also be used for programming some other older Lego robotics systems including the Scout, and Spybotics robots. NQC is now maintained and developed by John Hansen. NQC is a mature and stable system, and is also very well documented, with good tutorials and abundant example NQC programs freely available on the web.

A major advantage of the NQC RCX programming system is that it can be easily compiled from source on a Raspberry Pi running the standard Raspbian linux distribution. This means that you can use your Pi to program old RCX, Scout, and Spybotics Lego robots! In the case of the RCX and Scout programmable bricks, communication with your computer is via a Lego IR "tower". These Lego IR towers come in two flavours. Versions 1.0 and 1.5 of the Mindstorms RIS sets came with 9V PP3 battery powered IR towers that connected to a PC via a DE9 9-pin D-type RS232 serial port connector. Version 2.0 of the Mindstorms RIS came with a new IR tower that used USB for both data transfer and power.

Connecting a Serial Lego IR Tower to a Raspberry Pi

The older Serial IR towers can be connected to a Raspberry Pi using either a USB-to-serial adapter, or via a GPIO-to-serial adapter board. I've successfully used the Aten UC232A USB to Serial Converter with my Raspberry Pi and the Lego IR serial tower to program an RCX via NQC. This serial adapter worked out of the box with Raspbian, with no need for any additional driver software. If you plug the serial converter into a USB port on your Pi (or via an attached USB hub) then the device will be detected automatically, and should show up as a new serial device named (something like) `/dev/ttyUSB0` in the `/dev` directory.

As you will have noticed, USB ports are at a premium on the Raspberry Pi, and so I thought I'd see about connecting my RS232 Serial IR tower to my Pi via the GPIO pins instead of using a USB serial converter. Now, it's not advisable to make a direct electrical connection between an RS232 port and the GPIO pins on the Pi, as RS232 works at higher voltage levels than the 3.3V used by the Pi, and a direct connection would damage the Pi. Fortunately, there are now a number of serial RS232 interface boards available for the Raspberry Pi GPIO pins! I can highly recommend the RS232 interface board from AB electronics. This is a high quality board – two particularly nice touches are that the board can screw onto one of the mourning holes on the Pi to secure it, which is useful when connecting to a long and heavy RS232 cable, and secondly that the board comes with a stackable GPIO header, meaning that the non UART pins on the Pi GPIO remain easily accessible for use!

In order to use the GPIO RS232 board, there are a few changes that need to be made to the way that the Pi operates its GPIO UART port. This is straightforward and just involves changing a few of the system files. The necessary changes are well described on the AB electronics website. However, an even easier way of making these system file changes to enable you to use the GPIO for serial communications is to download and run a ready made script from here: https://github.com/lurch/rpi-serial-console. Just follow the detailed instructions on the rpi-serial-console github page. Once you've amended the system files to free up the GPIO serial port, you can communicate directly with a Serial Lego IR tower connected to a Pi via a GPIO interface board by using the `/dev/ttyAMA0` device on the Raspberry Pi.

Connecting a USB Lego IR Tower to a Raspberry Pi

The USB Lego IR tower came with version 2.0 of the Mindstorms RIS, and also with some of the later Lego Education Mindstorms sets. The Linux kernel has had built-in support for the Lego USB IR tower since version 2.6, and this means that the standard Raspbian distribution on the Raspberry Pi can easily work with this device. After plugging the USB IR tower into your Pi, you can verify that the kernel modules have been loaded using the following command:

```
find /lib/modules -name *lego*
```

on my Raspberry Pi this command lists the following three kernel modules related to the USB IR tower:

```
/lib/modules/3.10.18+/kernel/drivers/usb/misc/legousbtower.ko

/lib/modules/3.10.25+/kernel/drivers/usb/misc/legousbtower.ko

/lib/modules/3.10.26+/kernel/drivers/usb/misc/legousbtower.ko
```

you may see different output, depending on the details of your own Raspbian installation, but as long as one driver is present you should be fine. You can check for the presence of the Lego USB IR tower device in the `/dev` directory using this command:

```
ls -l /dev/usb
```

on my Raspberry Pi, this results in the following output:

```
total 0 crw-rw-rwT 1 root root 180, 160 Jan 18 23:38 legousbtower0
```

hopefully you'll see something similar when you plug your Lego USB IR tower into your Raspberry Pi. You may see different permissions for the `legousbtower0` file however, and it's important that the you, as the user of this device, have the necessary read and write permissions. You can ensure that this will always be the case by creating the following file: `/etc/udev/rules.d/90-legotower.rules` containing the following code (note you'll need to use the `sudo` command to achieve this):

```
ATTRS{idVendor}=="0694",ATTRS{idProduct}=="0001",MODE="0666",GROUP="lego"
```

now create a user group called `lego` and add yourself to this group (I'm using the default Ras Pi username `pi` here):

```
sudo groupadd lego
sudo usermod -a -G lego pi
```

what you've done here is give read/write access to `legousbtower0` to all users in the `lego` group, and then join yourself (assuming that your userid is the Raspbian default of pi) to that group. With this work done, you'll hopefully have no trouble with permissions when accessing the Lego USB IR tower.

Lego IR Serial and USB Towers Compared

Both types of IR tower work equally well as communications links between the Raspberry Pi and an RCX programmable brick. However, the Serial tower needs its own 9V PP3 battery, whereas the USB tower is powered via the USB bus. The USB tower can also transmit (but not receive) data using the Lego Visible Light Link (VLL) serial protocol used by Lego devices like the MicroScout and the Spybotics robots, which might be useful. Having said that though, the Spybotics kits came with serial VLL cables with the old DE9 9-pin D-type serial connector, so if you get set up with one of these serial ports on your Raspberry Pi in order to use the Serial IR tower, you'll also be able to program Spybotics robots, should you so wish! At the moment, I've not been able to get my Spybotics cable to work with my GPIO serial interface board, but I have got it working with my Atem USB serial converter cable. The Spybotics VLL cable has no internal power source (unlike the Lego Serial IR tower) and I suspect that the Raspberry Pi GPIO pins can't supply sufficient current to power the Spybotics VLL cable – so if you're wanting to program a Spybotics robot using your Pi, then I'd recommend getting hold of a USB serial adapter. I'll also just note here that you can also use NQC to program your RCX brick to send VLL signals to some other old Lego robotics devices, such as the MicroScout – see the instructions here!

Incidentally, there is some potential additional value in getting a Lego Mindstorms IR tower working on your Raspberry Pi, as it seems that these devices can also be used to control yet another robot – the RoboSapien – see the comprehensive write up here: http://www.robotika.sk/mains.php?page=/projects/robsapien/

Compiling NQC on a Raspberry Pi

Having established a means of IR serial communication between your Raspberry Pi and an RCX brick, you now need to install the NQC software package to provide you with a means to program the RCX. There are a few steps to getting this to work, but I found the process remarkably straightforward and trouble free. Hopefully, by following the recipe below, you'll also be able to get up and running without too much trouble. I basically followed the detailed instructions on the pbrick.info site, but slightly adapted them for the Raspberry Pi. I also found some specific instructions for installing NQC on the Pi, apparently adapted from those on the `pbrick.info` site, in this pdf file (this link is now broken, but I've provided a copy of the original PDF file here – rpi_rcx.pdf) within a repository at the University of Bologna!

First of all you need to download the NQC source code from its home on Sourceforge, using the following commands:

```
mkdir nqc-3.1.r6
cd nqc-3.1.r6
wget http://bricxcc.sourceforge.net/nqc/release/nqc-3.1.r6.tgz
tar xfz nqc-3.1.r6.tgz
cd ..
```

The next step is to apply a patch to the NQC source code so that it will work with a USB IR tower. It's important that the downloaded NQC source code is in a directory called `nqc-3.1.r6` as the patch file that you need to apply expects the source code to be in a directory with exactly that name (if you followed the steps above, then you'll be fine).

To get the required patch file, use a web browser to go to this location: http://sourceforge.net/p/bricxcc/patches/2/. Scroll down the webpage to the "Discussion" section at the bottom, and you'll see a message titled "Linux USB and TCP Support" from user "mesheets" with an attached file called `nqc-01-Linux_usb_and_tcp.diff`. This attached file is the patch file that we need, so go ahead and download it by

clicking on it. Save the patch file into the same location on your Raspberry Pi as you used for the `nqc-3.1.r6` directory you created above (i.e. the patch file should be in the same directory as the `nqc-3.1.r6` directory, and not inside the `nqc-3.1.r6` directory).

Now that you've got all the source code, and the patch file for USB support, you must apply the patch file by giving the following command from the directory that contains both the `nqc-3.1.r6` directory and the `nqc-01-Linux_usb_and_tcp.diff` patch file:

```
patch -p0 < nqc-01-Linux_usb_and_tcp.diff
```

In order to be able to build NQC, you'll need to have both the bison (formerly known as yacc) and the flex tools installed on your Raspberry Pi (these are not present on a default Raspbian installation), so to get these two packages, give the following command:

```
sudo apt-get install bison flex
```

once these two packages are installed, we're ready to build NQC. To compile the NQC package from the source code, first move into the `nqc-3.1.r6` directory. Now build NQC using the following command (note the two distinct sets of quotes around the name of the default serial name):

```
make clean
make DEFAULT_SERIAL_NAME='"/dev/ttyAMA0"'
```

Note that you should choose a `DEFAULT_SERIAL_NAME` according to the type of IR tower you will be using, and if you're using a Serial IR tower, according to how you've attached the Serial IR tower to your Pi. For a Serial IR tower connected via a GPIO adapter card, use `/dev/ttyAMA0` as above, for a Serial IR tower connected via a USB to serial adapter cable, use `/dev/ttyUSB0`, or whatever the name of the `/dev` device file is for your USB adapter. For a USB IR tower, use `/dev/usb/legousbtower0` as the `DEFAULT_USB_NAME`, like so:

```
make DEFAULT_USB_NAME='"/dev/usb/legousbtower0"'
```

Of course, if you happen to want to use both types of IR tower device, then you can specify defaults for both serial and USB devices simultaneously on the command line:

```
make DEFAULT_SERIAL_NAME='"/dev/ttyAMA0"' DEFAULT_USB_NAME='"/dev/usb/legousbtower0"'
```

Alternatively, you could also just edit the NQC `Makefile` and change the values for the `DEFAULT_SERIAL_NAME` and `DEFAULT_USB_NAME` variables there, if you prefer. If you're confused about this, don't worry, as you can always just go ahead and build NQC with the `make` command, without any other arguments, as it's easy to override the default NQC serial/usb IR tower device on the command line when using the nqc program e.g.

```
nqc -S/dev/ttyUSB0 -d test.nqc
```

or

```
nqc -Susb:/dev/usb/legousbtower0 -d test.nqc
```

The `make` command will take a a few minutes to compile all the NQC code. There will be a few warnings, now and again, about unused variables and missing braces, but don't worry about these, as they won't stop the `make` process from succeeding. At the end of the compilation process you should be left with a series of new directories and files in your `nqc-3.1.r6` directory. One of these will be called `bin`, and this should contain an executable file called `nqc`. This `bin/nqc` file is the program that you will use to program and communicate with your RCX.

You're now ready to install the NQC software into a location on the Pi that will make the `nqc` command line program readily accessible to you from any location on the Pi. Before you give the command to install the software tho', you need to fix the permissions on the NQC `man` page using the following command from within the `nqc-3.1.r6` directory:

```
chmod 644 nqc-man-2.1r1-0.man
```

with that fix in place, you'll be able to see the NQC `man` page after installation, which is achieved by giving the command:

```
sudo make install
```

This will copy the `nqc` executable into `/usr/local/bin/nqc` as well as putting other things where they need to be. You should now have a working NQC installation on your Raspberry Pi! Try typing `nqc -help` on the command line, and see what happens. Hopefully you'll be presented with a version/copyright message, followed by a list of command line options! The first two lines of the nqc command output from my Raspberry Pi:

```
nqc version 3.1 r6 (built Jan 19 2014, 01:17:54)
 Copyright (C) 2005 John Hansen. All Rights Reserved.
```

Loading Firmware onto an RCX from a Raspberry Pi using NQC

The next step is to test out communication between your Pi and RCX brick. In order to load and run programs on your RCX from NQC on your Pi, you need to have some firmware loaded on to the RCX. The RCX firmware is lost from the units memory when its batteries run low, and so if you've bought your RCX second-hand, and you've not used your RCX with its original Lego software, then you'll need reload the firmware now.

The original Lego firmware for the RCX was incorporated into the Windows software that came with the Mindstorms RIS sets. If you have a CDROM containing the original Lego Mindstorms RIS software then you may be able to extract the RCX firmware file from that software if you have access to a computer running Microsoft Windows (note you may well need an old version of Windows such as 95 or 98 to get some of the old RIS software to run). If your RCX set is didn't come with a CDROM, or if the CDROM is lost, or if you just can't get it

to work for some reason, then you can still download the Lego Windows RIS software from the very useful philohome.com site here. As far as I can see though, you'll still need a working Windows installation to extract the RCX firmware from this software bundle however.

Fortunately, the Lego RCX firmware still remains available for direct download from a few places on the web. This means that you needn't delve into the innards of ancient Windows software, unless you really want to. Lego no longer offer any support at all for their old Mindstorms RCX sets, and there is no firmware for the RCX on the current official Lego websites. It seems that Lego are content to allow the remaining RCX hobbyist sites to offer the RCX firmware for download, and it's my opinion that if you're the owner of a Lego RCX brick then you're probably entitled to a copy of the firmware!

Probably the most useful site for RCX firmware is this one: http://pbrick.info/rcx-firmware/, but all the sites in the list below currently have RCX firmware available for download. The `pbrick.info` site has version 3.28 of the RCX firmware that was released with the Lego Mindstorms RIS 2.0 set, and they also have version 3.32, which seems to have been the last version of the RCX firmware ever released by Lego, and which was apparently used in some Lego Education sets. All of these different firmware versions will work fine on any of the RCX hardware versions (1.0, 1.5, and 2.0), so you may as well go ahead and use version 3.32, which is the version that is recommended on the http://pbrick.info/rcx-firmware/ website.

- http://pbrick.info/rcx-firmware/
- http://neuron.eng.wayne.edu/LEGO_ROBOTICS/software_setup_tutorial.html
- http://www.dcs.ed.ac.uk/teaching/cs3/sysdesign/lego/
- http://www.elenafrancesco.org/old/lego/index.html

I've checked all these sites recently, and they're all working as of January 2014. If you're reading this in the distant future then you may find that you need to retrieve the RCX firmware from the Internet Archive!

So now that you've got hold of the Lego RCX firmware, you need to load it onto your RCX. Switch on the RCX, and place it in front of your Lego IR tower (red window facing the window on the front of the tower). If you're using an old Serial IR tower, make sure it's got a 9V battery inside! Now, from whichever directory contains your RCX firmware file, give the following command to load your chosen firmware onto the RCX, using the default IR tower e.g.

```
nqc -firmware firm0332.lgo
```

Depending on how you compiled NQC, and depending on whether or not you are using the default NQC IR tower, you may need to explicit specify the serial link to the RCX on the command line by using the `-s` flag, as shown above. Also, the name of the firmware file in the above command should match that of whichever firmware file you've got, or are trying to use. On giving the above command, you should see a green led light up inside the IR tower. During the firmware upload process, you'll see the counter on the RCX lcd display steadily increasing, and you'll see the following message on the console screen of your Pi:

```
Downloading firmware:.............................
```

After a successful firmware upload, which will take a minute or two to complete, the RCX will give a beep, and the RCX lcd will display the time in hh:mm (which will be 0000 as the RCX does not store the time when it's powered off). See the http://pbrick.info/rcx-firmware/ site for more details, and for pictures of what the RCX lcd should look like. If all went well, you've achieved successful communication between your Pi and the RCX, confirmed that NQC is working, and loaded up the Lego firmware onto the RCX!

Compile and Download Your First NQC Program for RCX!

You're now ready to start programming your RCX using your Pi! Try compiling and loading the simple test program (`test.nqc` in the `nqc-3.1.r6` directory) that comes with NQC. Have a look at the NQC code in the `test.nqc` file, and see if you can work out how to setup your RCX brick to make use of this program. When you're ready, use the following command (adding a suitable -S option if necessary) to compile and upload this program to the RCX:

```
nqc -d test.nqc
```

There are many other example NQC programs available for download from the NQC site.

You can also use the Lego IR tower and NQC to communicate between your Raspberry Pi and RCX brick in real time! The `nqc -raw` command lets you to send individual RCX opcodes from the Pi. These opcodes are essentially individual instructions telling the RCX to, for example, play a sound:

```
nqc -raw 5105
```

or turn on motor A:

```
nqc -raw 2181
```

and turn it off again:

```
nqc -raw 2141
```

By using these "raw" RCX opcodes, you can turn your Raspberry Pi into a sophisticated remote control for your RCX brick, and whatever contraption you build your RCX into! You also now have a nice way to control your RCX, via IR signals from your Pi, from the Internet.....

Despite the age of the Mindstorms RIS and RCX, there's still masses of great material out on the web for this platform, including a lot of very good teaching materials. A good place to get started though for now is the NQC tutorial here: http://bricxcc.sourceforge.net/nqc/doc/NQC_Tutorial.pdf. Another useful introduction is a freely chapter from an old O'Reilly title: The Unofficial Guide to Lego Mindstorms Robots.

Alternatives to NQC on the Raspberry Pi

Lastly, I should just add that NQC isn't the only system for programming the RCX using a Raspberry Pi – an alternative programming system is LeJOS which provides a small Java virtual machine for the RCX. I've not tried using LeJOS on my own Raspberry Pi, but it seems that others have – there are some detailed instructions on getting LeJOS working on the Pi in the rpi_rcx.pdf file from that University of Bologna repository mentioned above – I haven't tried these myself yet though.

Happy robot building/RCX/Ras Pi hacking!

If you have any difficulties with getting any of the above to work, leave a comment, and I'll see if I can help.

This entry was posted in Arduino, Lego Mindstorms RCX, Raspberry Pi and tagged IR, Lego RCX, Mindstorms, NQC, Raspberry Pi, RS232, Spybotics, USB, VLL on January 20, 2014 [https://minordiscoveries.wordpress.com/2014/01/20/using-nqc-on-a-raspberry-pi-to-program-a-lego-mindstorms-rcx-brick/] .

15 thoughts on “Using NQC on a Raspberry Pi to Program a Lego Mindstorms RCX Brick”

Michiel
April 28, 2014 at 8:36 am

Cool post and thanks for the reference links 🙂
I just got myself an RPi so will be trying some of this.
Cheers, Michiel from pbrick.info

lcww1 Post author
April 29, 2014 at 9:34 pm

Hi Michiel,

Thanks for your comment! And thanks for your pbrick.info site which has been very helpful to me; it was nice to see that there’s someone else out there still interested in Spybotics and all the other wonderful old Lego pbricks!

Incidentally, I’ve been looking into making a Spybotics VLL programmer that can be directly connected to the Pi GPIO pins, as I’d like to be able to get the Pi and nqc talking to a Spybot without needing to use a USB/RS232 converter cable. Hopefully this won’t prove to be too difficult.....I’ll do a post about this, if I ever manage to get it set up and working!

Cheers,

lcww1

Mark
October 23, 2014 at 6:42 am

Thanks for the instructions, they were easy to follow. I managed to get my Pi talking to an RCX in one evening, so hopefully my Mindstorms kit and my school Mindstorms kit will get a new lease of life.

I am not sure if the issues are to do with different versions of the Pi, but I needed to use "sudo addgroup lego" rather than "sudo group add lego", I needed "sudo make install" rather than just "make install" and I have needed to use "-Susb:/dev/usb/legousbtower0" with each call to nqc because nqc reports that it "Could not open serial port or USB device". Each of these is easy to work around, so I am happy to have found another use for my Pi.

lcww1 Post author
January 16, 2016 at 8:13 pm

Dear Mark – I'm very glad this blog post was useful to you! Thanks also for the feedback, and for picking up the groupadd typo – now fixed!
Apologies for the very late reply!

Regards,
lcww1

Joseph
November 14, 2015 at 4:12 am

A couple of quick notes for my runthrough of this very helpful post (thank you btw).
I had to do "sudo groupadd lego" (not "group add") before I could "chown root:lego /dev/usb/legousbtower0"
The direct link to the patch is
http://sourceforge.net/p/bricxcc/patches/_discuss/thread/00b427dc/b84b/attachment/nqc-01-Linux_usb_and_tcp.diff

lcww1 Post author
January 16, 2016 at 8:08 pm

Dear Joseph – Apologies (again!) for the very late reply – I'm really glad you found this blog post helpful. Many thanks for the feedback and for picking up the type on the groupadd command – I'll fix the blog post accordingly.

Best wishes,

lcww1

Peter
November 26, 2015 at 7:58 pm

The tutorial you linked on how to program in nqc shows a program called Brick Command Center, and I'm not sure where to find that. Thanks!

---

lcww1 Post author
January 16, 2016 at 7:55 pm

Dear Peter – Apologies for very late reply! The Brick Command Center software is free and available for download from: http://bricxcc.sourceforge.net
– it runs well on a Windows PC. I hope that helps?
Best wishes,
lcww1

Mark O.
December 30, 2015 at 3:40 am

When I try to update the firmware I get the following message : Could Not OPen serial or USB device

The IR USB tower is connected and shows up in /dev/usb directory
I used the make DEFAULT_USB_NAME='"/dev/usb/legousbtower0"' to build the bin files.
a lsusb command does list the device.

Any assistance would be appreciated to a newbie with Raspberry.

---

lcww1 Post author
January 16, 2016 at 8:02 pm

Dear Mark – Apologies for the very late reply to your question! I'll do my best to help, but I'll need some more information from you. I assume you've followed my instructions in the "Connecting a USB Lego IR Tower to a Raspberry Pi" section of my blog post? It does sound as though you may have a permissions problem, and I'm wondering whether you've created the lego group and added your Pi userid to the lego group? One quick way to try to circumvent permissions problems is to run all the nqc command line programs with the sudo command as a prefix e.g. $ sudo nqc -Susb:/dev/usb/legousbtower0 -d test.nqc

Let me know if any of the above helps!
Good luck,
lcww1

---

lcww1 Post author
January 16, 2016 at 8:09 pm

Mark – also take a look a the comment from Joseph – I made an error in the groupadd instruction in the blog post, and this may have been the source of your problem! I've fixed the blog now.

BW
lcww1

andy mac
December 2, 2016 at 5:12 pm

Mark O did you ever get it working as I am getting the same error.

lcww1 Post author
January 22, 2017 at 10:43 am

Hi Andy – sorry to hear you're having problems – as per my reply to Mark O, if you can supply me with more information on your probelm then I may be able to help you,

Good luck,

lcww1

martinsmac
June 1, 2016 at 7:50 pm

Thanks so much for this. My old mindstorms set has come back to life!

lcww1 Post author
January 22, 2017 at 10:44 am

Hi Martinsmac! Very glad to hear that this write up has been useful to someone!

Happy robot building!

lcww1
