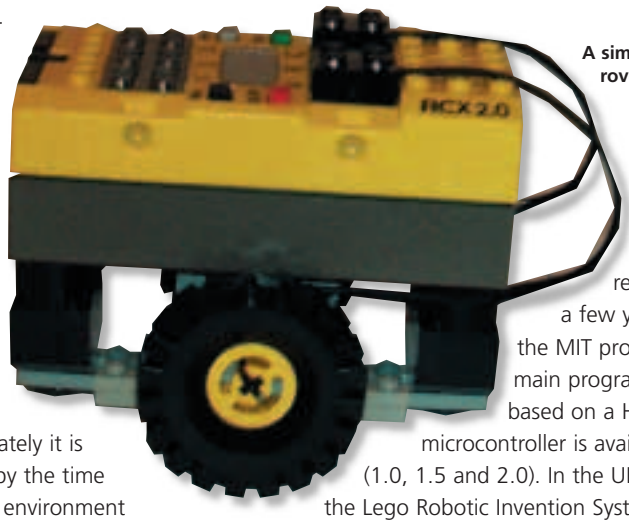
Lego Mindstorm
Dreaming of electric sheep

BRICKS AND PIECES

Robotics is a minefield of a subject due to the many different avenues of exploration. Despite limited time and resources, John Southern bites the bullet and plays with Lego...

You can spend your time building hardware or you can concentrate on the programming. You can even do it all virtually such as the University of West Florida's robot modelling site. Hardware wise most of the kits available depend on the Parallax BasicStamp, which is a PIC microcontroller. Unfortunately it is Windows controlled but by the time this is published a new C environment should be available. The BasicStamp allows simple circuits to be built and controlled giving rise to many third party kits. What I wanted was something ready made so I could play with the software.



A simple rover

A wet Saturday afternoon meant a trip to the local Toys'R'Us to see what we could find to while away the afternoon. Lego Mindstorm cried out but we were cautious. Lego released the Mindstorm a few years ago inspired by the MIT programmable brick. The main programmable block (RCX) based on a Hitachi H8/3292 microcontroller is available in three versions (1.0, 1.5 and 2.0). In the UK you can buy either the Lego Robotic Invention System 1.5 or 2.0. The difference lies not the RCX brick but with the infrared controller. In version 2.0 the controller is USB while 1.5 is serial. The most recent stock in the shop was version 1.5 with a RCX 2.0. The RCX version does not really matter as it can be upgraded and using LegOS can be replaced.

The first hour was spent just opening lots of bags of Lego and playing. Finally deciding to build a robot to follow a path we face our first challenge. Without a Windows machine in the house the supplied software is of no use. We can spend Saturday night installing Windows or turn to the Web for help.

After just a couple of minutes on the Web we were faced with an array of choices. We can use the Lego RCX built-in software and run a Linux-based programming tool or we can download a new programming language into the RCX and again control it from Linux.

Starting with the inbuilt software we can then choose from a range of programming languages such as Forth or NQC (Not Quite C). We opt for NQC as the Forth primer is somewhere upstairs and lazyness has taken over.

The NQC is command line based and the latest version (2.3r1) is a 188K download. The package contains a test file to check that the system is

working and you have everything connected. It is probably worth downloading the NQC package just for this test as it puts your mind at ease over the hardware.

To control the RCX brick we first write our NQC in a simple text editor. We save the file with a .nqc extension and the command

```
nqc test.nqc
```

compiles the code. Now by adding the -d switch we can send the complied code to the RCX brick.

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

Similar to C or C++ the NQC follows very similar syntax:

```
task main()  {
  SetSensor(SENSOR_1, SENSOR_PULSE);
  while(true)  {
    if (SENSOR_1 ==2)  {
      PlaySound(SOUND_FAST_UP);
      ClearSensor(SENSOR_1);
    }
  }
}
```

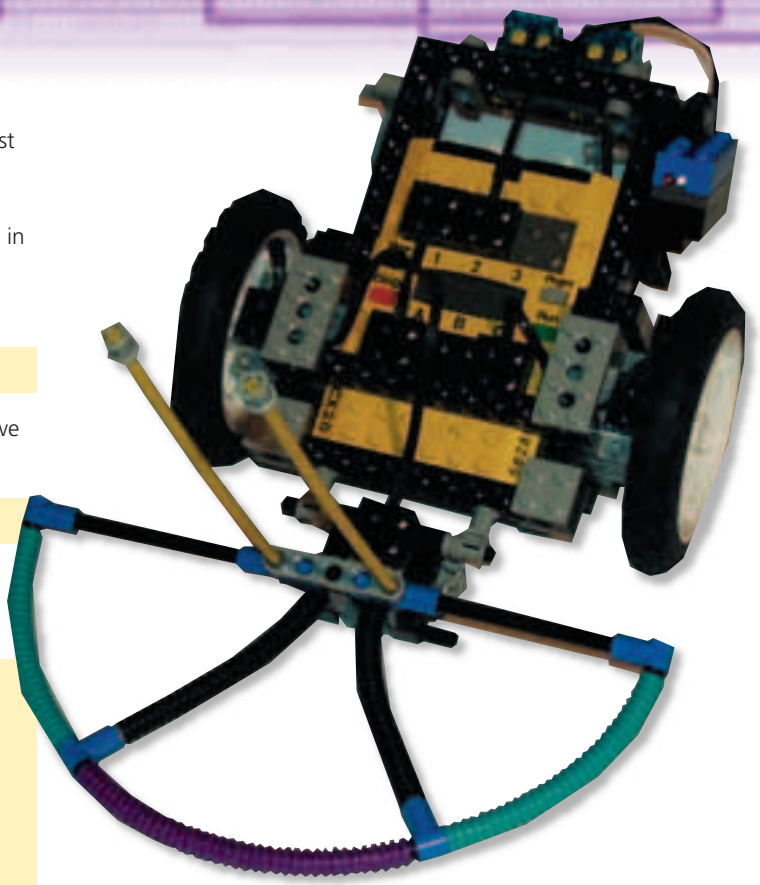
As can be seen from the above example no surprises appear in the coding. The real surprise is that the sensors and output ports (three of each on the RCX brick) are not just digital on/off but analogue and can be used to sense a range from 0 to 1023. This means with just a few logic gates we could expand the number of sensors, but that's for another weekend.

From the above example we can see that there is a sound generator on board. With a little work with the PlayTone command we can get the brick to sing, so long as we are careful to not let the buffer queue overflow (every eighth note we have to pause until the buffer is empty).

Actually more time was spent on building the robots, due to the huge amount of parts in the box and the constant hunt for the correct brick shape. The robots can be initially built by using the easy to follow booklet supplied with the kit – just remember to allow more time to find all the parts.

Having tested that the Linux box could control the RCX brick and had fun making little models dance, sing and even head towards the light (Warning: Cat owners should note that the robot is easily knocked over by an angry feline), we couldn't resist updating the firmware.

Our first choice was the pbFORTH but in practice we settled for LegOS. Both of these routes enable you to replace the firmware inside the RCX brick and thus give you far more control over the unit. To update the firmware we need to use a firmware



Seek the light and avoid the cat

downloader. One was at hand from the NCQ with the -firmware switch.

The LegOS gives much more control and we can now program the tiny LCD screen. We wrote a simple C text file and complied it. First mistake. You need to set the makefiles TARGET environment variable otherwise you will wonder where the .srec files go. Second mistake. Make sure the serial port is correctly set, as the default is ttyS0. Third mistake. Getting the error message "no response from RCX" does not necessarily mean the RCX brick is faulty or out of range. It may be that the IR control unit's battery has finally died.

With the LegOS finally running and after recompiling code to remove errors we find all sorts of extra functions. Motor speeds can be subdivided into 255 units. A brake function enables us to lock a wheel while the off function lets it freewheel.

The only disappointment with the kit was to do with the number of pieces included. While it may have taken ages to find parts because they are so numerous, the number of pieces is cleverly limited to only just build the robots in the supplied booklet. Now serious consideration must be put to asking Father Christmas for more pieces. Who knows maybe next time the smaller Lego Scout module may appear.

Info

http://www.enteract.com/~dbaum/lego/nqc/
http://www.legOS.sourceforge.net/

ROBOTS: Extra sensors

QUICK FIXES

The sensors that come with the MindStorm are limited. So John Southern decided to make some of his own

Having finally read the whole of Luis Villa's Mini-HOWTO on Lego MindStorm with Linux, and received lots of emails back from those who asked for the PDF files of the previous article that should have been printed in December, the biggest request was for DIY sensors and actuators, as the range with the MindStorm is considered small.

Sensors consist of two parts: the first connects to the RX brick and the latter is the actual sensor. Like most people I want to do things quick, so I'm prepared to cut corners if the time saved outweighs the flimsiness of the product.

How to hook us easily to the Lego connectors

When making extra sensors I should have produced wonderful connectors by using a lathe, turning down machine screws and mounting these inside a Lego brick to make good connectors. If I had that much spare time I would do something more constructive such as watch TV. A quick look through the spares box finds mini crocodile clips, which clip onto the Lego RCX and hold quite well. Hurrah! The easy part of the sensor, done in a minute.

The heat is on

Now for the actual sensor. The first we build is a temperature sensor in the hope of making the buggy move towards heat. Adding a thermistor to the other ends of the crocodile clips is the quickest way we can make a temperature sensor.

Temperature probe

The only problem that arises from this, is that it needs a temperature offset constant to make it accurate. After two days of playing with this, we found that the crocodile clips give variable contact resistance – so what works one day fails to be accurate the next time we build.

Rather than be forever adjusting the program data each time we build, we can fudge the sensor by putting a variable potentiometer in line with the thermistor. We can now easily adjust the resistance in the connecting cable so the thermistor gives an accurate reading. At this point the buggy sort of works, except that changes in temperature are actually very small and the thermistor requires time to settle on a temperature. This meant it worked very slowly to choose the correct path and we had to drive the buggy, wait, test the temperature and compare this with the previous result to see any change. Another quick fix to make it a waterproof sensor was to cover it in hotmelt glue.

The motion of the ocean

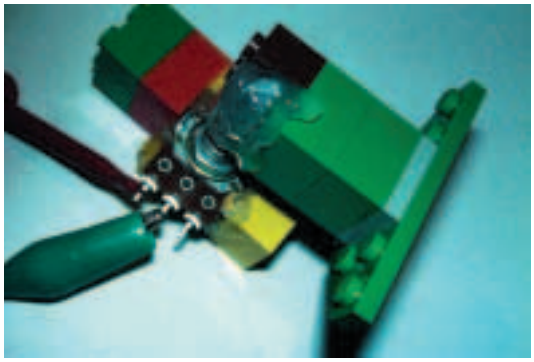
The next quick fix sensor is a mercury tilt switch. This works as a simple contact when turned in a set orientation. These are great fun, as by putting them in series with resistors in parallel we are able to make a motion detector. Changing the orientation proved to be tricky, but hotmelt glue came to the rescue. With one mercury switch we could sense if the buggy was travelling on a level surface or on a slope.

With two mercury switches we have enough data to determine whether it's travelling uphill or downhill. This is done by positioning one sensor so it only contacts when the bugging is climbing and one that makes contact when running level. If sensor one is connected then the buggy is going up hill; if the second sensor is connected the buggy is running level; and if neither sensor is connected the buggy is travelling downhill.

Orienteering

Next up is an angle sensor. Here we simply link a ski on the bottom of the buggy to a variable potentiometer. As the ski changes angle with the terrain so too does the resistance through the potentiometer. This is much easier that our other sensors, and means we can also now work out the gradient.

At this point we were still plagued with cats thinking the buggy was their personal plaything so we will investigate imaging next time.



Hot glue – the quick fix solution. Angle sensor using a potentiometer