Mindstorms/NQC notes

Contents of our Mindstorms sets

- 1 RCX programmable brick
- 1 IR tower
- 1 USB to Serial adepter
- 3 Motors
- 1 lamp
- 3 photo sensors
- 1 angular sensor
- 2 touch sensors

Unfortunately one of the touch sensors and one of the lamps are missing, so one of the sets lack a lamp and one of the touch sensors :-(If you find an extra lamp or touch sensor in your set please tell me!

Each set also contain an approximately equal amount of Lego pieces. However, the sets do not contain the same number of pieces of each type. This means that you may have to trade pieces with other groups if you discover that a certain type of brick you need is missing from your set.

NQC downloads

NQC (Not Quite C) is a C like programming language for the RCX brick.

NQC for Linux, Windows and Mac:

`http://bricxcc.sourceforge.net/nqc/`

Only the command-line application.

Bricx Command Center:

`http://bricxcc.sourceforge.net/`

A Windows-only IDE and NQC bundle.

MacNQC:

`http://homepage.mac.com/rbate/MacNQC/`

A Mac-version of NQC with an IDE.

NQC usage

```
nqc -help
```

Compile and download test.nqc using the default serial port (ttyS0 for Linux and COM0 for Windows):

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

(Linux: remember that you need to have writing rights to /dev/ttyS in order to send anything out the serial port!)

Use serial port COM4 to connect to the IR-tower (Windows):

```
nqc -SCOM4 -d test.nqc
```

Use serial port ttyS1 to connect to the IR-tower (Linux):

```
nqc -S/dev/ttyS1 -d test.nqc
```

A few hints and NQC commands to get you started

The equivalent to the main function in ordinary C is the main task in NQC:

```
task main() {


}
```

Running motor A forwards:

```
OnFwd(OUT_A);
```

Running motor B backwards:

```
OnRev(OUT_B);
```

Turn output C off:

```
Off(OUT_C);
```

The power of an output has eight settings, 0-7:

```
SetPower(OUT_A, 5);
```

Note that the power setting 0 is not the same as Off.

Wait 4 seconds:

```
Wait(400);
```

Before you can start using the sensors you need to set the sensor type with for example:

```
SetSensor(SENSOR_1, SENSOR_LIGHT);
SetSensor(SENSOR_2, SENSOR_TOUCH);
SetSensor(SENSOR_3, SENSOR_ROTATION);
```

Once the sensor type has been set you read the sensor value by accessing the appropriate SENSOR, for example:

```
if (SENSOR_1 < 35) do_something();
```

Use the view button on the RCX in order to figure out sensor ranges and suitable threshold values.

You can use variables, but due to limitations of the RCX you can only use 32 of them and only variables of type int.

NQC tutorial and examples

`http://www.computingscience.nl/people/markov/lego/tutorial.pdf`
`http://www.computingscience.nl/people/markov/lego/examples.zip`

Photos from earlier Mindstorms competitions

`http://www.ituniv.se/~joli/img/robosumo/`
`http://www.ituniv.se/~joli/img/roborace/`

Firmware

The firmware is the operating system of the RCX and is required for it to be possible to program the brick. At the start all of the bricks should have firmware loaded.

When changing batteries do not remove all batteries at the same time! This will cause the RCX brick to "forget" the firmware and you will have to reload it again. Remove a couple of the old batteries first and replace them. When you've inserted a few fresh batteries you can replace the last of the old batteries.

Downloading firmware to the RCX brick:

`http://mindstorms.lego.com/sdk2beta/default.asp`

1. unzip
2. run ATLClient (Windows only)
3. Port / Open
4. Special / Download firmware. This takes a few minutes and may require more than one attempt.

If you have the firmware file (included in the zip file above) NQC is also supposed to be able to download the firmware using

`nqc -firmware <filename>`

or

`nqc -firmfast <filename>`

Attaching standard 0-5V sensors to the RCX

`http://www.plazaearth.com/usr/gasperi/lego.htm`