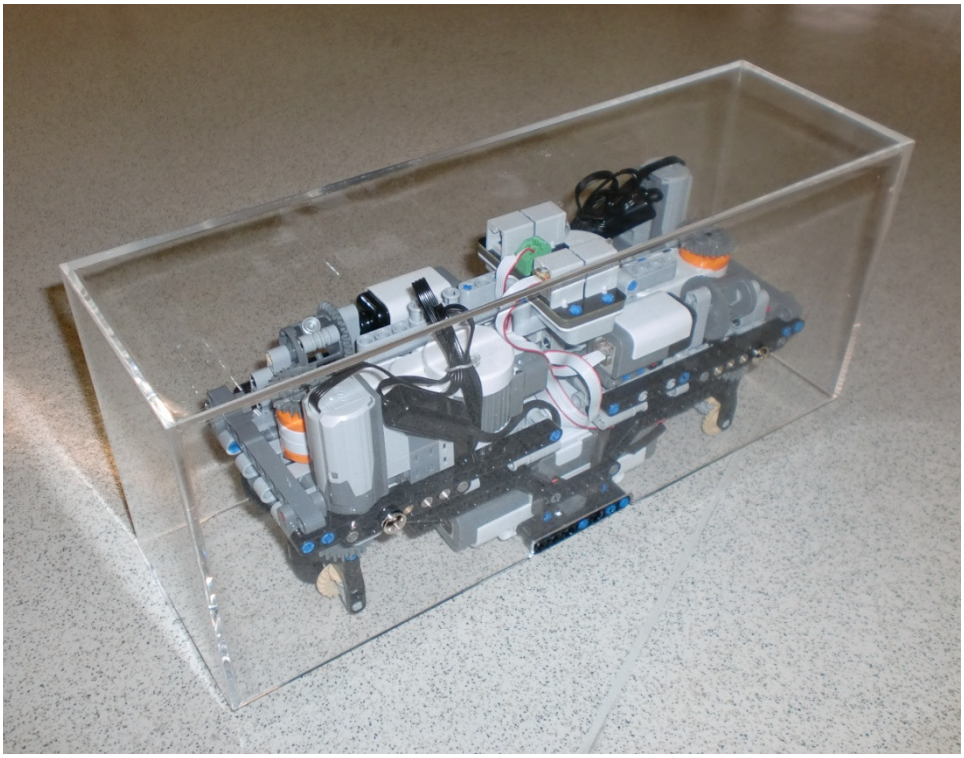
Datasheet and Guide

LEGO® Prototype of ITER Remote Handling Transport Cask



By: Ricardo Oliveira
Contact: ricardo.m.oliv@gmail.com

Lisbon, October 11th, 2011

Contents

1. Overview

The model consists of a prototype of scale 1:25 of the ITER Remote Handling Transporter Cask, made out of LEGO® pieces and third-party components from HiTechnic and mindsensors.com.

It was made to be remote controlled through MATLAB, using Bluetooth as the main interface. It can also be controlled using the local LCD display and buttons, without any personal computer. In alternative to Bluetooth, it can be interfaced with USB, using the same drivers and API.

The complete physical structure, excluding the acrylic piece, is documented in CAD files. The CAD files can be visualized using LEGO Digital Designer (LDD), a free and official LEGO software.

The NXT programmable brick was programmed in NXC (Not eXactly C). The code was edited, compiled and flashed using BricxCC IDE.

MATLAB interface uses RWTH – Mindstorms NXT toolbox developed at RWTH Aanchen University, and it is a free open source product.

The kit includes extra LEGO pieces, for unexpected broken or worn ones, extra screws, USB cable, battery charger and the complete software and source code.

2. Specifications

2.1 Dimensions

- External acrylic box: 33,7 (length) x 10,4 (width) x 14,6 (height) cm
- Clearance between the ground and the acrylic box: 10,2mm
- Distance between wheels: 20,8 (tightest pins position) – 27,2 (farthest pins position) cm
- Wheels: 30,4mm (diameter) x 14mm (thickness)

2.2 How to vary distance between wheels

The cask is made of 4 modules, 2 equal wheel modules, which are placed symmetrically, a central unit that supports both wheel modules and a small module on top to give stiffness to the whole structure (figure 1).

There are 4 possible positions. The distance depends on the position of each wheel module relative to the center module. On figure 1, the green dotes represent the pin holes where the wheel modules can be attached to. The blue arrows represent the longest configuration (27,2mm), while the orange ones represent the shortest one (20,8mm).

To perform this operation, first remove the acrylic box, loosening the four screws on the sides. Then, take off the 4 long gray pins on top (marked with red arrows in figure 1). Avoiding taking apart the central module, carefully disassemble the pins from the bars, highlighted in green, and place them in the desired position.

NOTE: be careful with the wires attached, due to their short length.

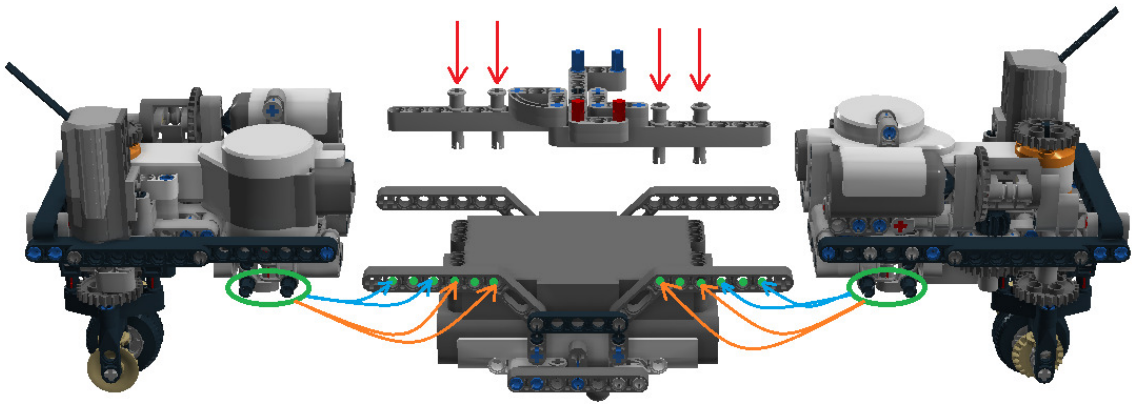


Figure 1 – All 4 modules and the connectivity illustration between wheels and the central modules

2.3 Hardware setup

The electronic components that make part of the prototype are the following:

1x NXT brick

1x battery (2100mAh)

2x NXT servo motor

2x HiTechnic angle sensor

2x Power Functions motor M

2x Converter cable PF-9V

2x Converter cable 9V-NXT

NOTE: The above 3 products attach all together in series in order to fit an ouput port of the NXT brick.

6x NXT standard cables

1x mindsensors.com motor multiplexer (NXTMMX-v2)

The schematic in figure 2 shows the configuration of the hardware.

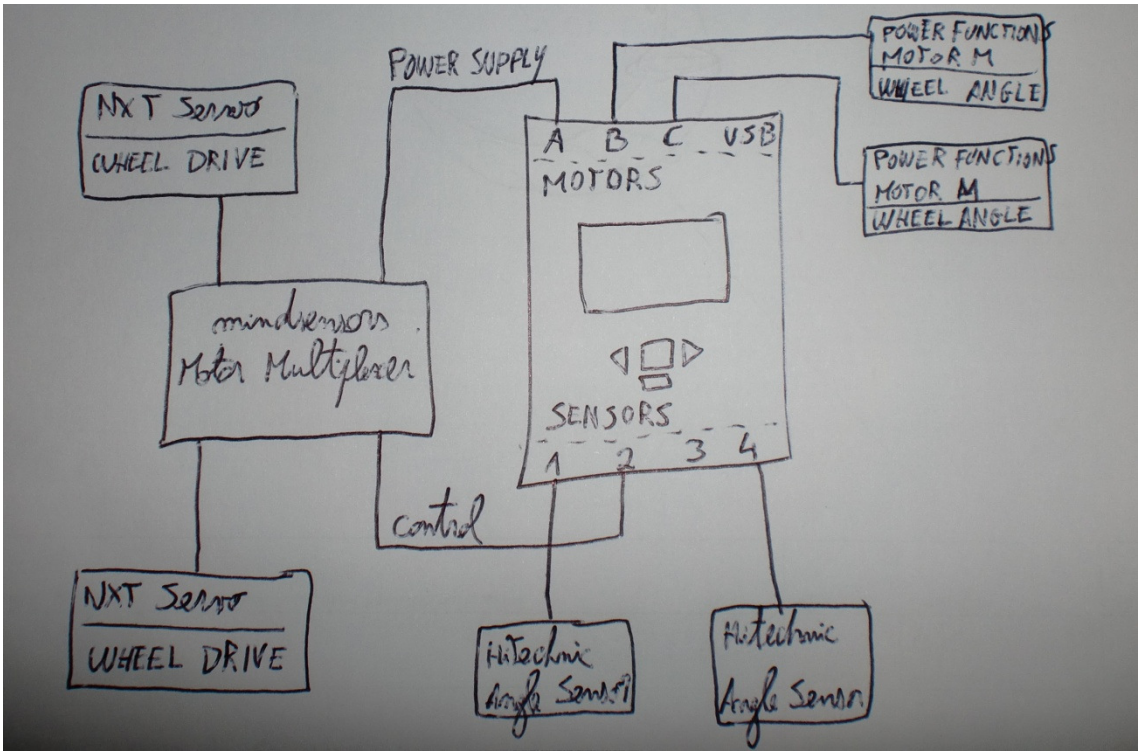


Figure 2 - Hardware schematic

Each wheel module consists of two motors:

- NXT servo - it controls the velocity of the wheel in rotations per minute (from 50 to 130 rpm). Relative position can be read from the servo with 1º resolution. Note that this resolution applies for the motor outshaft, not for a revolution of the wheel. There is a gear reduction of 20:12 (yellow gears, on figure 3a) that decreases this resolution to 0,6º. Every time the cask starts running, the position is reset to 0.
- PF Motor M – DC motor that act on the angle of the wheel. Together with the Hitechnic angle sensor, they can control this angle with a resolution of 1º. The reduction between the wheel turntable and the actual input axle of the angle sensor is 1:1. Details of the physical implantation can be seen on figure 3b. The angle sensor stores the absolute 0 and is able to calibrate it at any time. Even occurring an external rotation of the wheel with the power off, the absolute 0 is kept.

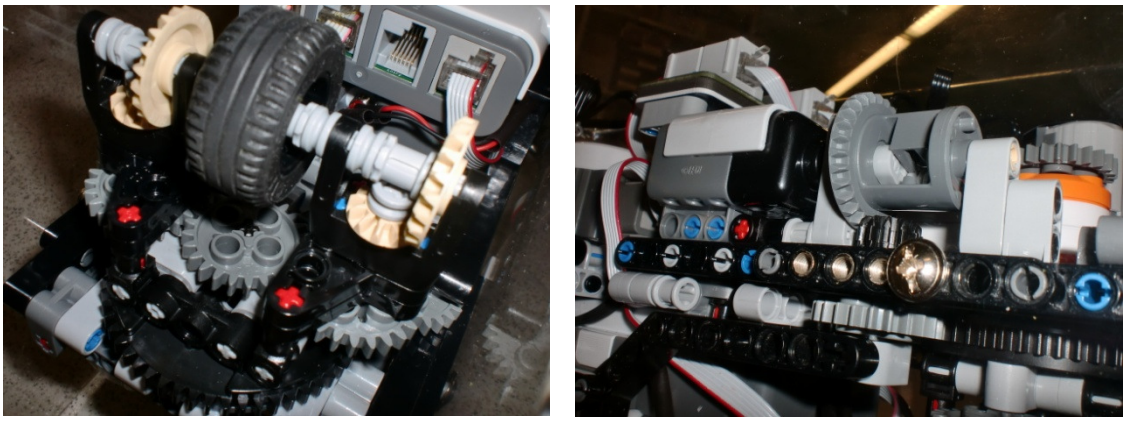


Figure 3 a) Wheel driving mechanism; b) Wheel angle mechanism

The motor multiplexer is powered through output port A (maximum current: 0,8A) of the NXT brick. A standard NXT cable was used, in which one of the ends was cut in order to take the power from the right pins. If this cable is ever broken, figure 4 shows the pinout configuration of the output ports, in order to build a new one.

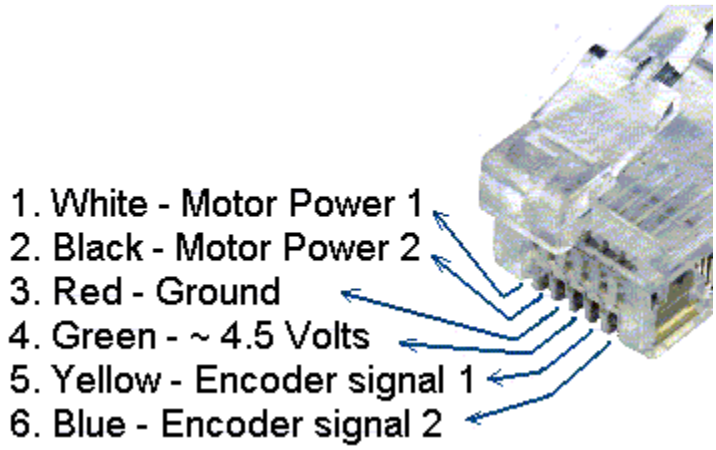


Figure 4 - NXT output ports pinout configuration (from www.josepino.com)

2.4 Sample frequencies

Both NXT servo motors and HiTechnic angle sensors communicate through I2C. Data is collected all at once, every 19ms, ~52Hz. This data includes: battery level (mV), angle of both wheels (degrees), position of both wheels (motor degrees), speed of both wheel (rpm) and the time stamp (milliseconds) associated with all values above specified.

The data is available to be collected through either Bluetooth or USB. Due to the half-duplex communication, Bluetooth performance is around 10-15Hz, while USB reaches the full speed of 52Hz.

3. Hello World

3.1 Preparing the model

If the model has been modified, check the connections (refer to figure 2).

Turn on the NXT brick by pressing the orange button. Check the battery level, on the top right corner. If the battery icon is flashing on and off, it is under 10% of its capacity.

While running the cask with any of the embedded programs developed, there will be a beep and a “low battery” message flashing every 1,5 seconds warning about low batteries, when it goes under 6,9V.

When charging the batteries, 2 lights will be lit, a red and a green one. The green means that charger is attached; the red means the battery is still charging. Wait until only the green light is lit.

3.2 Running the demo

The orange button turns the NXT on and is used as the “OK” button, once inside the menu. The triangle light gray buttons are used to browse the menu back and forth. The dark gray button is the “Go back” button. While in the main menu, if “Go back” is pressed, a message will pop up asking for confirmation to turn off the NXT brick – orange button will confirm it.

To run the demo, just turn the NXT on and follow: **“My files -> Software files -> embedded_demo -> Run”**.

To exit the program, press the dark gray button for a long period, until it actually exits.

4. Interfacing with the cask

The current chapter will cover how to interface with the cask prototype, using different approaches (on-board control through LCD display, MATLAB, C/C++ and embedded development), by means of USB and Bluetooth.

If MATLAB and Bluetooth are used, the LEGO MINDSTORMS NXT drivers are NOT needed. C/C++ applications with Bluetooth do need LEGO drivers, as well as any USB connection, including MATLAB interface with USB.

For **Windows**, the LEGO drivers (1.1.3) are available under: "**/Installation files/LEGO MINDSTORMS drivers.zip**". For **MAC**, check the LEGO Mindstorms webpage:

- http://mindstorms.lego.com/en-gb/support/files/default.aspx#Driver

For **Linux**, follow these instructions:

- http://www.mindstorms.rwth-aachen.de/trac/browser/tags/version-4.04/RWTHMindstormsNXT/tools/LinuxConnection/LinuxReadme.txt

The above instructions are the requirements to get MATLAB RWTH Mindstorms toolbox ready to run in Linux. This is enough to interface the cask with MATLAB. I believe these instructions also apply for any other kind of interaction between the NXT brick and Linux.

NOTE: When using USB, do not forget to remove the wheel structure close to the USB interface, to avoid damage in both the LEGO structure and the USB cable, by accidentally rotating the wheel (figure 5).

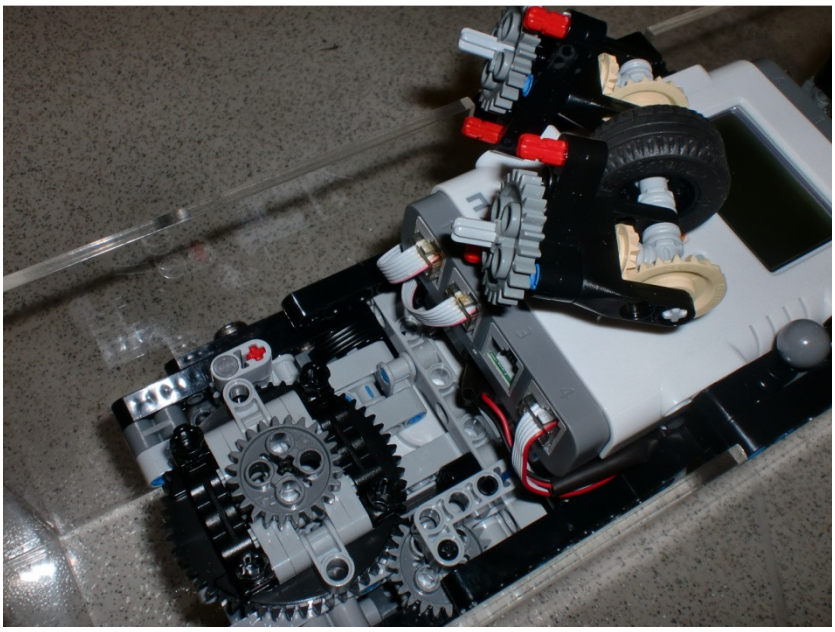


Figure 5 - Wheel removed to avoid USB cable colision

4.1 NXT LCD Display interface

The use of the display interface is intuitive and requires no software setup. Use the buttons to browse the options line by line and change the parameters.

Run "remote_control" program, following the same path as described in chapter 3.

Using the buttons it is possible to:

- Stop each NXT driving motor at any time (Line 1 – wheel 1; Line 5 – wheel 2)
- Change the angle (degrees) of each wheel (Line 2 – wheel 1; Line 6 – wheel 2)
- Change the velocity (rpm) of each wheel (Line 3 – wheel 1; Line 7 – wheel 2)
- Calibrate the absolute zero of each wheel angle (Line 4 – wheel 1; Line 8 – wheel 2)

Use orange button to apply a change, arrows buttons to change the parameters and the dark gray button to jump from feature to feature.

4.2 MATLAB interface

4.2.1 Installation and first connection

To use MATLAB to interface with the cask, it is needed to install RWTH Mindstorms NXT toolbox (available for Windows, MAC and Linux). All files can be found under: "**Installation files/RWTHMindstormsNXT**" (version 4.0.4). For later versions, check:

- http://www.mindstorms.rwth-aachen.de/trac/wiki/Download

For instructions on how to install the toolbox and get the first Bluetooth connection working, follow **carefully** the step-by-step guide here:

- http://www.mindstorms.rwth-aachen.de/trac/wiki/Download4.04#Step-by-step

Please, ignore the following steps:

- Check NXT Firmware (Checked while developing the prototype)
- Transfer MotorControl to NXT (the embedded program is used to perform motor control operations that the direct USB/BT control is not capable of. However, these operations are not executed on the prototype. Moreover, the prototype needs to be running another embedded program while interfacing with MATLAB – "remote_control")

If problems arise, check the following sources of helpful information:

- http://www.mindstorms.rwth-aachen.de/trac/wiki/FAQ
- http://people.clemson.edu/~nwatts/engr141/instructions/ (with videos)
- http://www.youtube.com/watch?v=wuir-IUyo6k (Portuguese-Brasil installation video, for Windows 7 and MATLAB 2011b)

(!!!) Do NOT underestimate this process. It can be painful to get Bluetooth connection working properly.

4.2.2 Interface functions

Now that everything is installed, the following functions can be used to interface with the cask:

Function	Values	Description	Example
CASK_Init()		It establishes a connection to the NXT, using either USB or Bluetooth, and starts the embedded program "remote_control.rxe".	**CASK_Init**()
[timeStamp batteryLevel angle1 angle2 tacho1 speed1 tacho2 speed2 statusByte] = **CASK_ReadValues**()	**timeStamp** – time in ms, since NXT was turned on last time. **batteryLevel** – battery level in mV **angle1** and **angle2** – angle of wheel 1 in degrees (0-359º) **tacho1** / **tacho2** – relative position (since the NXT was turned on) of the servo connected with wheel 1/2 in degrees **speed1 / speed2** – speed of the servo connected to wheel 1/2 in rpm **statusByte** – status of the reading (0 if succedded)	It reads all values at once from the NXT, plus the time stamp associated with the values. If "statusByte" is not 0, some error happened and the readings should be ignored. The meaning of the status byte can be checked in the document ""**/LEGO documentation/LEGO MINDSTORMS NXT Direct commands.pdf**", page 12.	[timeStamp batteryLevel angle1 angle2 tacho1 speed1 tacho2 speed2 statusByte] = **CASK_ReadValues**()
CASK_SendRequest('Parameter1', value1, 'Parameter2', value2, ...)	Possible parameters: **'Angle1'/'Angle2'** – Angle request for either wheel 1 or 2. Any given value is wrapped to the range [0-359]. **'Speed1'/ 'Speed2'** – Speed request for either wheel 1 or 2. Value must be in the range [-130, -50], {0}, [50, 130].	It sends a request to the NXT to set a new angle, speed or calibrate a new absolute 0º for any of the 2 wheels	**CASK_SendRequest**('Angle2', 90, 'Speed2', 50)

	'CalibrateWheel1'/ 'CalibrateWheel2' – Calibration request for either wheel 1 or 2, to burn a new absolute 0º in the memory of the angle sensor		
`CASK_CloseComm()`		It stops the embedded program running in the NXT and closes the communication with it	`CASK_CloseComm()`

Table 1 – MATLAB API functions to interface with the NXT

A sample code can be found under "**/MATLAB interface/plotReadings.m**", which plots the reading of both angles and speeds, while the embedded program is running on the cask. To try it, simply turn on the NXT, run CASK_Init(), then run the script.

4.3 Extending to other programming languages interfaces

There are several solutions for interfacing with the NXT with other programming languages, already developed. Examples for C/C++ are: NXT++ (http://nxtpp.clustur.com/), libnxt (https://code.google.com/p/libnxt/), aNXT (https://github.com/jgraef/aNXT).

For python, use nxt-python interface: https://code.google.com/p/nxt-python/

It is also possible to develop C/C++ applications to interface with the transporter cask, using the Software Developer Kit that LEGO has released, to interface with the LEGO Mindstorms drivers. These drivers are only available for Windows and MAC.

The latest SDK, up to the date of writing of this manual, is present in the software files under "**/Installation files/LEGO MINDSTORMS drivers SDK.zip**". New SDK is unlikely to be released, but it can be found under the LEGO website:

- http://mindstorms.lego.com/en-gb/support/files/Driver.aspx#Advanced

4.3.1 Details about the specific transporter cask application

All that is needed is to interface with a mailbox system that is handled by the LEGO standard firmware (shown in figure 6).

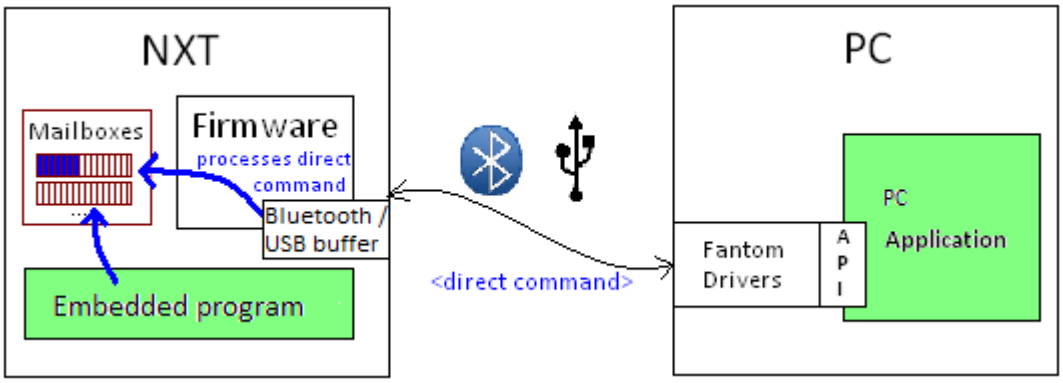


Figure 6 - PC - NXT communication schematic

The direct command type *MessageWrite* and *MessageRead* are all the needed commands to send requests to the cask and retrieve values. These commands read and write to mailboxes that are handled by the firmware. The structure of these direct commands can be consulted in the document "**/LEGO documentation/LEGO MINDSTORMS NXT Direct commands.pdf**". By looking into the MATLAB routines, one can easily find out the number of the mailboxes used and the format of data sent and received.

When using LEGO SDK, *nFANTOM100::iNXT::sendDirectCommand()* is the routine that sends direct commands.

4.4 Embedded development without interface

It is possible to control the transporter cask directly from the embedded software. There is simple API to be used with NXC language that can easily be used in a parallel thread to the one that is running the motors control, remote interface and sensor readings.

The demonstration described in chapter 3 (Hello World) was developed in these conditions. The following code was added to the source code of the program "remote_control":

```
task demo(){
  while (true){
    Wait(2000);
    SendRequest(0, 0, 0, 0); // Angle1, Angle2, Speed1, Speed2
    Wait(1000);
    SendRequest(0, 0, 60, 60);
    Wait(5000);
    SendRequest(90, 90, 60, 60);
    Wait(3000);
    SendRequest(45, 45, -60, -60);
    Wait(4000);
    SendRequest(0, 270, 0, 60);
    Wait(6000);
    SendRequest(270, 270, 60, -60);
    Wait(6000);
    SendRequest(45, 315, -60, -60);
    Wait(2000);
    SendRequest(45, 315, -90, -90);
```

```
    Wait(2000);
    SendRequest(45, 315, -130, -130);
    Wait(3500);
    SendRequest(45, 315, 0, 0);
    Wait(2000);
    SendRequest(270, 90, 0, 0);
    Wait(2000);
    SendRequest(90, 270, 0, 0);
    Wait(5000);
  }
}
```

Details about coding in the embedded platform are later explained in section 4.4.3. Before, a proper IDE has to be installed in order to edit, compile and download NXC programs to the NXT.

4.4.1 Installing BricxCC IDE

Apart from a few other options for editing NXC source code files and compile them, Bricx Command Center is one of the most complete tools for this. However, it is only a Windows compatible IDE.

For MAC or Linux solutions check the NeXT utilities:

- http://bricxcc.sourceforge.net/utilities.html

To install BricxCC (version 3.3.8.10), run the installer present under "**/Installation files/Bricx Command Center/**". After completing the installation, extract the files of the latest test release under the same folder (**test_release20110720.zip**) and copy them to the BricxCC folder where the program was installed, replacing all files.

To install the latest release and the test releases check the URLs:

- http://sourceforge.net/projects/bricxcc/files/bricxcc/ (latest release)
- http://bricxcc.sourceforge.net/test_releases/ (test releases)

4.4.2 Connecting to the NXT, editing and downloading programs

When starting BricxCC, a window will pop up automatically to choose the interface to connect the NXT. Later, in order to perform a connection, click on the button highlighted on figure 7. To close the communication, click on the most right button in line with the highlighted one.

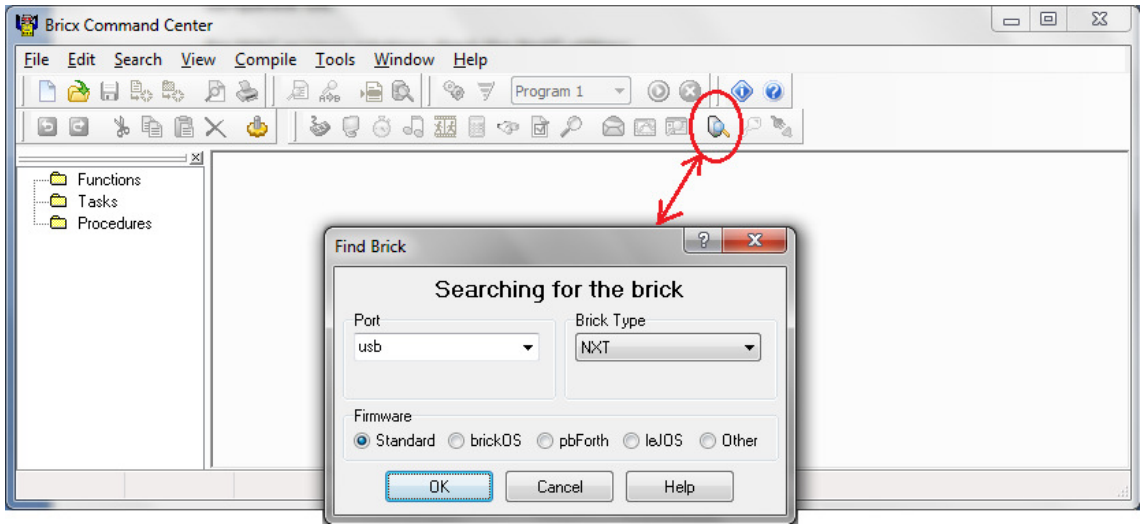


Figure 7 - Connection window on BricxCC

The default interface is USB. It is highly recommended to use USB for the simplicity and fast speed of the connection. For a Bluetooth connection type the following in the *Port* field:

- **BTH**::**NXT**::**MAC**::**COMPORT** (example: BTH::NXT::00:16:53:0E:56:A8::29)

Bluetooth MAC address can be checked browsing the NXT menu: "**Settings -> NXT Version**".

For an easier connection one can add the NXT MAC address to the list of possible ports, by editing the file *nxt.dat* under "**User/AppData/Roaming/JoCar Consulting/BricxCC/3.3**". Choose a name and write down the following line:

- **NAME=BTH::NXT::MAC::COMPORT** (example: ITER= BTH::NXT::00:16:53:0E:56:A8::29)

NOTE: Connections to the NXT may fail if the same interface is being used by MATLAB at the same time. The same applies for the inverse situation.

Once connected, the BricxCC window will look like in figure 8. All is needed is to open the source file "remote_control.nxc", under "**/Embedded files/source**" and edit it as desired. The executable file that is flashed to the NXT has the same name as the source file, with the extension ".rxe". That is abstract to the user, since the executable is generated and flashed, without taking place in the PC.

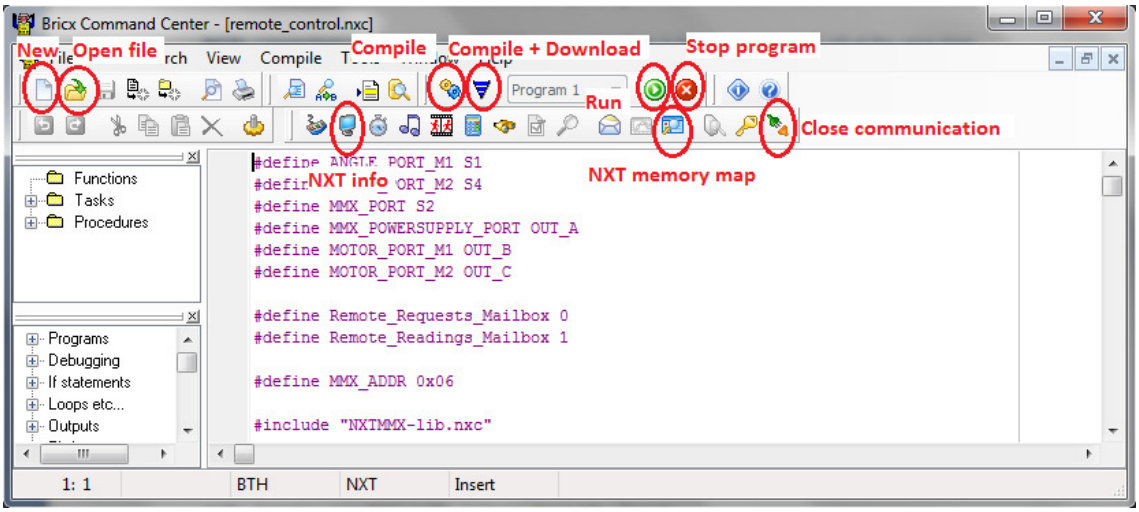


Figure 8 - Brixcc useful buttons

4.4.3 NXC custom API to interface with the cask

As shown before, the structure of the code that needs to be added is the following:

```
task demo(){
      //Send and read values
      …
}

task main(){
      //initializations
      …
      start demo;
      // main loop that controls interfaces, motors and sensors
      …
}
```

task is the keyword used to start a thread. **start** is the keyword to launch a thread. Thus, basically, a thread like "demo" needs to be coded and started from the main thread. Now, let's have a look at the custom NXT API developed to send requests, read values and ask for calibrations.

Functions	Values	Description	Example
SendRequest(long angle1, long angle2, long speed1, long speed2)	**angle1/angle2** - Angle request for either wheel 1 or 2. Values **must be** in the range [0-359]. **speed1/ speed2** – Speed request for either wheel 1 or 2. Value **must be** in the range [-130,	It sends a complete request including angles and speeds of both wheels.	**SendRequest**(angle1, angle2, speed1, speed2)

	-50], {0}, [50, 130].		
ReadStoredValues(long & timeStamp, long & batteryLevel, long & angle1, long & angle2, long & tacho1, long & speed1, long & tacho2, long & speed2)	**timeStamp** – time in ms, since NXT was turned on last time. **batteryLevel** – battery level in mV **angle1** and **angle2** – angle of wheel 1 in degrees (0-359º) **tacho1** / **tacho2** – relative position (since the NXT was turned on) of the servo connected with wheel 1/2 in degrees **speed1 / speed2** – speed of the servo connected to wheel 1/2 in rpm	It reads all values stored in the memory, by the main loop of the program.	**ReadStoredValues**(timeStamp, batteryLevel, angle1, angle2, tacho1, speed1, tacho2, speed2)
ResetSensorHTAngle (byte port, byte mode);		It calibrates a new 0º in the angle sensor specified by "port".	For wheel 1: **ResetSensorHTAngle** (ANGLE_PORT_M1, HTANGLE_MODE _CALIBRATE); For wheel 2: **ResetSensorHTAngle** (ANGLE_PORT_M2, HTANGLE_MODE _CALIBRATE);

Table 2 - Custom NXC API functions to interface with the NXT

NXC guide, tutorial and the full API can be consulted in "**LEGO documentation/NXC programming**". Also, for any doubt related with NXC programming language or any other NXT related issue, it is recommended to ask for help in the online forum:

- http://mindboards.sourceforge.net/

5. Hardware maintenance

In case some parts fall apart or CAD files are needed for any other reason, LEGO Digital Designer, for Windows, can be installed using the installer file under **"/Installation files/LEGO Digital Designer**".

If a MAC version or a later Windows version is preferred, check the website:

- http://ldd.lego.com/download/default.aspx

CAD files of the model are included in the folder "**/CAD files**".

5.1 Where to buy new pieces

If new pieces are needed, the following places are recommended:

1) Official LEGO shop, online:
- http://shop.lego.com/en-PT/
- http://shop.lego.com/en-PT/Pick-A-Brick-ByTheme (individual pieces)
- http://shop.lego.com/en-PT/Robotics-ByCategory (Mindstorms parts)
- http://shop.lego.com/en-PT/Power-Functions-ByTheme (PF M motor and converter cables)

2) ARABOT HI-FI, Lda. - LEGO store in Santarém and also online:
- http://www.portugal-didactico.com/Portugal-Didactico/Bem-vindo.html

It is focused on LEGO kits from the Educational Division, as well as individual parts, plus Mindstorms material (motors, NXT, cables, battery, sensors, and so on).

3) BrickLink - Online collection of stores:

- http://www.bricklink.com/

It is a collection of stores in an "eBay fashion", specialized in all kind of LEGO material. A registration is needed. The majority of sellers require payment through PayPal or IBAN.

4) HiTechnic and mindsensors.com – companies specialized in third-party components for the Mindstorms products:
- http://www.hitechnic.com/
- http://www.mindsensors.com/

The prototype uses thin cables, which are convenient for the little space that they take. However, it is easier to damage them. They can be found in mindsensors.com.

6. Handling unexpected situations

If anything unexpected occurs, it is recommended to expose the situation in the online forum:

- http://mindboards.sourceforge.net/

In any case, be free to contact me at: ricardo.m.oliv@gmail.com