Documentation Author

F.Lambercy and P.Bureau for K-Team S.A.
Rue Galilee 9
1400 Yverdon-les-bains
Switzerland

email: info@k-team.com
Url: www.k-team.com

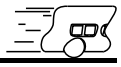
Trademark Acknowledgments:

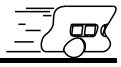
IBM PC: International Business Machine Corp.
Khepera: K-Team and LAMI

Legal Notice:

- The content of this manual is subject to change without notice.
- All effort have been made to ensure the accuracy of the content of this manual. However, should any error be detected, please inform K-Team S.A.
- The above notwithstanding K-Team can assume no responsibility for any error in this manual.

Table Of Contents





1.1 How to Use this Manual

This manual is introducing the Khepera III robot. For a quick start and overview of the robot's functions, please read chapter 1 to 4.

Refer to the following summary if a particular information is needed. If the manual does not cover a particular problem, many more technical documentation is available online from K-Team website (www.k-team.com) and especially a Frequently Asked Question document to solve most common problems and questions.

Unpacking and Inspection: Khepera's package description.

The robot and its accessories: Khepera Hardware overview and main functions and accessories description.

Connections: detailed cables connections for various usage.

Serial Communication mode: detailed description for the Serial communication mode between a computer and the robot.

1.2 Safety Precaution

Check the unit's operating voltage before operation.

It must be identical with that of your local power supply. The operating voltage is indicated on the nameplate at the rear of the power supply.

All connnections (including extension addition or disconnection) must be made when the robot and the interface are switched OFF. Otherwise damages can occur.

Switch OFF the robot if you will not use it for more than a day.

Disconnect the power supply removing it from the wall socket.

Do not manually force any mechanical movement.

Avoid to force, by any mechanical way, the movement of the wheels or any other part.

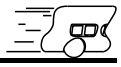
If you have any question or problem concerning the robot, please contact your local Khepera dealer.

1.3 Recycling

Think about the end of life of your robot! Parts of the robot can be recycled and it is important to do so. It is for instance important to keep batteries out of the solid waste stream. When you throw away a battery, it eventually ends up in a landfill or municipal incinerator. These batteries, which contain heavy metals, can contribute to the toxicity levels of landfills or incinerator ash. By recycling the batteries through recycling programs, you can help to create a cleaner and safer environment for generations to come. For those reasons please take care to the recycling of your robot at the end of its life cycle, for instance sending back the robot to the manufacturer or to your local dealer.

Thanks for your contribution to a cleaner environment!

2 Unpacking and Inspection



Open the bag and check each item in the box. You should having the following material.

1. Khepera III robot
2. KoreBotLE (not included in the Khepera III base package)
3. Power Supply with a primary plug
4. Battery Pack
5. Optional KoreConnect

3.1 The Khepera III Robot

3.1.1 Overview

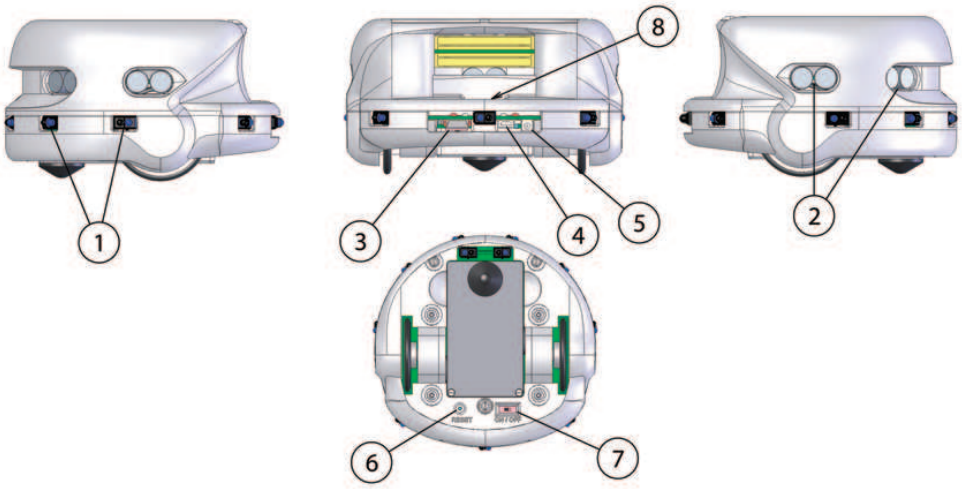


Figure 3.1: Overview of the Khepera III robot.

Make an external inspection of the robot. Note the location of the following parts:

1. Infrared sensors.
2. Ultrasonic sensors.
3. Main serial connector.
4. USB miniAB connector.
5. Power jack connector.
6. Reset.
7. On / Off.
8. Cable unroller connector.

3.1.2 ON-OFF Battery Switch

It allows the user to switch the robot ON or OFF. When ON, the robot is powered. The robot may even be powered without the battery pack using the external power supply connector. See figure 3.1 number 7.

3.1.3 Indication LEDs

The Khepera III has six indication LEDs, two for the battery charger, one for the Power ON, one for the state of the motors controller and two programmables by the end-user (see figure 3.2).

When the Motor controller state LED is on, one of the two motor controller is in error mode. This could append when a motor is blocked or a current limit is detected. In this case the controller must be reset by the dsPIC ('M' command in serial interface mode or call 'initKH3()' in a KoreBotLE program).

For the battery charger, the red one indicates that charge is in progress and the green one signals when the charge is complete. If there's no battery pack when you plug an external power supply, the green one will turn on.

The led 5 and 6 can be set by the user with 'L' command. See annexe A Serial communication protocol.

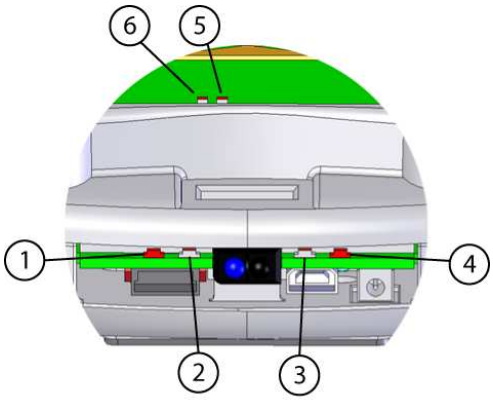


Figure 3.2: LEDs of the Khepera III.

1. Motor Controllers state LED (redD).
2. Power ON LED (green).
3. Charge completed LED (green).
4. Charge in progress LED (red).
5. Programmable LED (green).
6. Programmable LED (red).

3.1.4 Serial connector

The serial connector contains various serial lines. One serial line which is at TTL level (0V/+5V) is used to connect a personal computer to the Khepera III dsPIC (with a KoreConnect). Used only when there is no KoreBotLE connected. Please see section 4.2.

There is another RS232 (-10V/+10V) used to communicate with the KoreBotLE, ie to see the KoreBotLE's boot message. Please see section 4.3.

It is also possible to use the USB link of the KoreConnect to communicate with the KoreBotLE.

The length of the serial cable should be limited to two meters for proper operation.

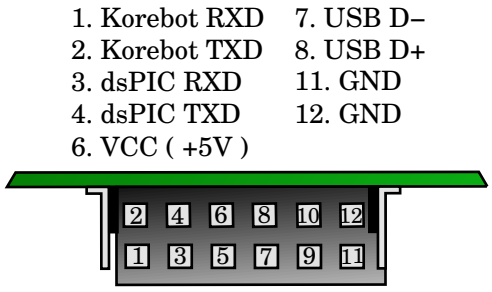


Figure 3.3: Details of the Khepera III serial connector. Pin 5, 9 & 10 are not connected.

3.1.5 Cable unroller connector

The cable unroller connector is a serial and a power connector (see figure 3.4). You can supply your robot and communicate with the robot through it. This connector is usefull if you want to supply your robot with an external power supply (+9V) and keep its mobility with an cable unroller. However, if you want to recharge your battery at the same time, it is mandatory to use the power jack connector.

1. VCC (+9V)
2. dsPIC RXD
3. dsPIC TXD
4. GND
5. GND
6. GND

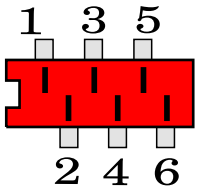


Figure 3.4: Details of the cable unroller connector.

3.1.6 USB connector

The mini-USB connector is usefull only when a KoreBotLE is plugged in the Khepera III. You can connect directly your PC to the Khepera III with an ordinary mini-USB cable. To communicate with the KoreBotLE through the USB connector, please refer to the documentation of the KoreBotLE (found on the web site www.k-team.com).

3.1.7 How to plug a KoreBotLE

The Khepera III has been designed to work together with a KoreBotLE, but as it isn't mandatory, in the default pack there is no KoreBotLE with the robot. You can buy it at the same time, in this case, the KoreBotLE will be mounted into the robot. But if you buy it afterward, you must first unmount the upper body of the Khepera III, then plug the KoreBotLE on the extansion connectors (see figure 3.5). If you have any other extansion (KoreIO, KoreSound, KoreVision, etc...), you must only remove the black metal plate, and plug your extension on the KoreBotLE.

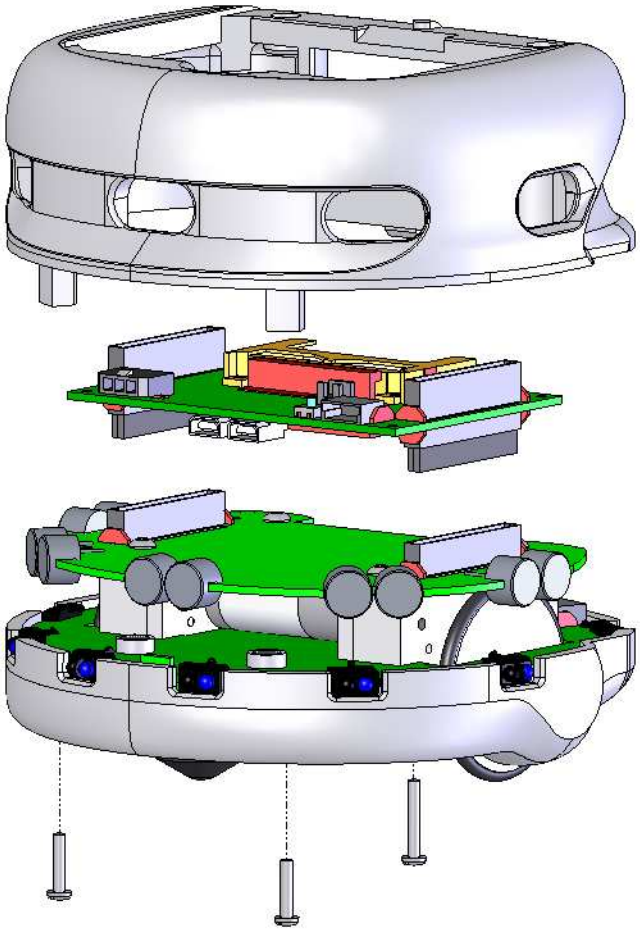


Figure 3.5: Details of the KoreBotLE assembly.

3.2 Motors and motor control

Each wheel is moved by a DC motor coupled with the wheel through a 43.2:1 reduction. The motor itself having a 27:1 reduction and the gear box having a 1.6:1 reduction. The motor has its own embedded incremental encoder, placed on the motor axis, gives 16 pulses per revolution of the motor. This allows a resolution of 691.2 per revolution of the wheel that corresponds to 54 pulses per ten millimeter of path of the robot (diameter of the wheel = 41mm, thus each revolution the robot make 128.8mm). By default, the robot is set in mode encoder resolution x4, in this case for each wheel revolution the controller motor make 2764 measures. The other encoder configuration mode is 2x (1382 measures/turn).

Reminder: 1 revolution = 128.8mm = 2764 measures.

Each motor is driven by its own motor controller implemented in a PIC18F4431. The PIC has direct control on the motor power through a double H bridge and can read the pulses of the incremental encoder.

The motor control block acts as slave devices on the I2C bus. When the KoreBotLE is not connected to the Khepera III robot, the main robot CPU will turns itself into an i2c master. When the KoreBotLE is connectedm, it becomes the i2c master.

Each motor controller switches its motor ON and OFF at a given frequency and during a given time. By this way, the motor react to the time average of the power supply, which can be modified by changing the period the motor is switched ON. This means that only the ratio between ON and OFF periods is modified, as illustrated in figure 3.6. This power control method is called "pulse width modulation" (PWM). The PWM value is defined as the time the motor is switched ON.

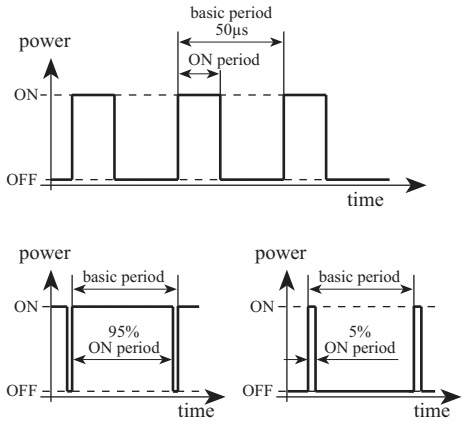


Figure 3.6: The "Pulse Width Modulation"(PWM) motor supply mode is based on a ratio between the ON time and the total time. The basic switching frequency is constant.

Each motor controller can perform the control of the speed or the position of the motor, setting the correct PWM value according to the real speed or position read on the incremental encoders.

Both DC motors can be controlled by a PID controller. Every term of this controller(Proportional, Integral, Derivative) is associated to a constant, setting the weight of the corresponding term: Kp for the proportional, Ki for the integral, Kd for the derivative.

The motor controller can be used in two control modes: the speed or position mode. The active control mode is set according to the kind of command received. If the controller receives a speed control command, it switches to the speed mode. If the controller receives a position control command, the control mode is automatically switched to the position mode. Different control parameters (Kp, Ki and Kd) can be set for each of the two control modes.

3.2.1 Speed

Used in speed mode, the controller has as input a speed value of the wheels, and controls the motor to keep this wheel speed.

The speed value is a division of a constant value by the time between encoder pulsations. In default mode (encoder resolution x4 and postscaler 1:4), a measure is made four times every pulse. Then each revolution of the wheel, 2764 measures are made. The constant value is defined by the maximum time multiplied by 256 (0xFFFF * 256 = 16'776'960). This operation allows a better PID calculation for the lower speed.

$$MotorSpeed = \frac{16'776'960}{Timer5value}$$

To convert into a real time, use the following calculation (this time is the delay between two measures):

$$Time = \frac{\frac{Timer5value}{Postscaler}}{\frac{f_{osc}/4}{Tmr5Prescaler}}$$

Where f_{osc} = 20MHz and Tmr5Prescaler = 8 (default).

To get the real speed in mm/s:

$$RealSpeed = \frac{WheelCircumference}{Time * 2764}$$

Where WheelCircumference is 128.8mm and 2764 correspond to the number of measures per revolution of the wheel.

Here's an example with MotorSpeed = 20'000, Tmr5Prescaler = 8, Postscaler = 4 and encoder resolution = x4

$$Timer5value = \frac{16'776'960}{20'000} = 839$$

$$Time = \frac{\frac{839}{4}}{\frac{20'000'000/4}{8}} = 0.336[ms]$$

$$RealSpeed = \frac{128.8}{0.000336 * 2764} = 138.9[mm/s]$$

$$RealSpeed = \frac{MotorSpeed}{Kspeed}[mm/s]$$

Where Kspeed is 144.01 when default configuration is used.

The maximum reachable speed is 48'000 (= 333 mm/s) in open loop, and 43'000 (= 298 mm/s) in regulation mode. And the minimum is 2'000 (= 13.9 mm/s) with regulation control.

Used in position profile mode ('F' command), the controller has as input a target position of the wheel, an acceleration and a maximal speed. Using this values, the controller accelerates the wheel until the maximal speed is reached, and decelerates in order to reach the target position. This movement follows a trapezoidal speed profile, as described in figure 3.7.

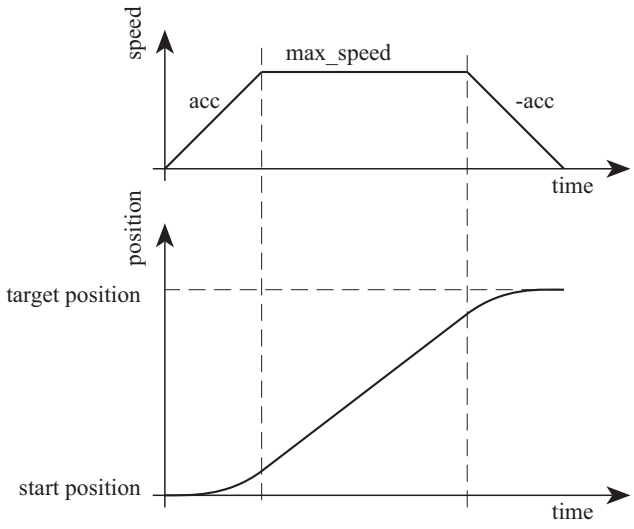


Figure 3.7: Speed profile to reach a target position with a fixed acceleration (acc) and maximal speed (max speed).

The input values and the control mode of this controller can be changed at every moment. The controller will update and execute the new profile in the position mode, or control the wheel speed following the new value in the speed mode.

First configure the speed profile ('J' command) with the needed parameters. Then you can perform a move with the position profile mode ('F' command). See annexe A for more details about communication protocole.

3.3 Infra-red Proximity sensors

Khepera III hase nine sensors placed around the robot and two placed on the bottom. The latters allow experiments like line following. They are positioned and numbered as shown in figure 3.9.

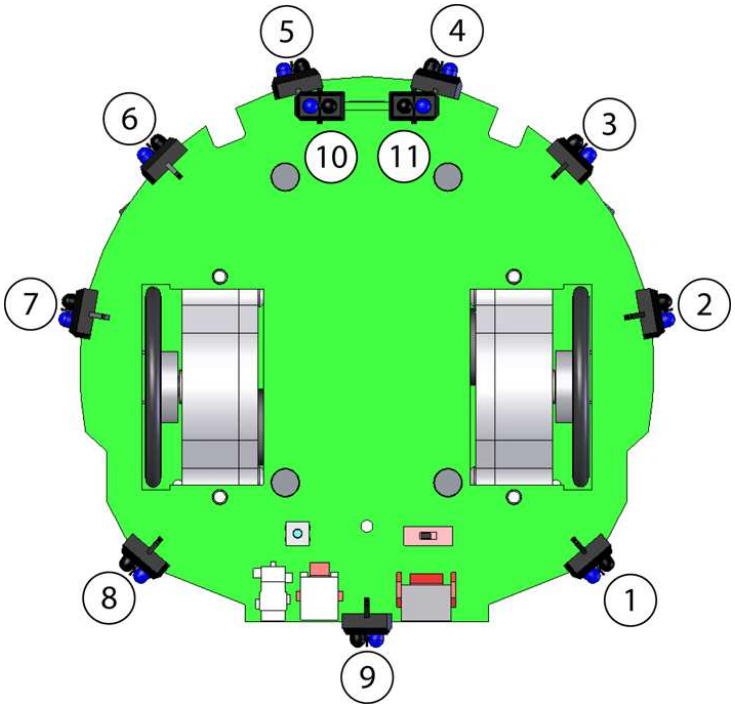


Figure 3.8: Bottom view of the IR sensors

These sensors embed an infra-red light emitter and a receiver. For detailed description, please refer to the manufacturer's datasheet. The eight sensors are TCRT5000, reflective optical sensors from *Vishay Telefunken*.

This sensor device allows two measures:

- The normal ambient light. This measure is made using only the receiver part of the device, without emitting light with the emitter. A new measurement is made every 33 ms. During the 33 ms, the eleven sensors are read in a sequential way every 3 ms. The value returned at a given time is the result of the last measurement made.
- The light reflected by obstacles (= proximity). This measure is made emitting light using the emitter part of the device. The returned value is the difference between the measurement made emitting light and the light measured without light emission (ambient light). A new measurement is made every 33 ms. During the 33 ms, the eleven sensors are read in a sequential way every 3 ms. The value returned at a given time is the result of the last measurement made.

The output of each measurement is an analog value converted to a 12 bit digital value . The following two sections illustrate the meaning of this 12 bit values.

3.3.1 Ambient light measurements

Ambient light measurement is strongly influenced by the robot's environment. Depending on the light source type, color, and distance, ambient light measurement profile might vary. **It is not recommended to use light source with large emission in the infrared range, as this could confuse the IR sensors.**

3.3.2 Reflected light measurements (proximity)

Sensors are mainly meant to detect obstacles around the Khepera. Measurements for reflected light depends on objects reflectivity and on ambient light conditions. Objects color, materials and surfaces do have an influence on the sensors response. Moreover, as any sensor, IR sensors are subject to environmental noise. For all these reasons, graphics below are given for information only and should not be considered as references.

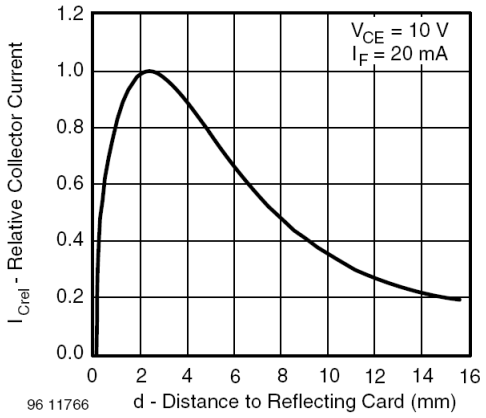


Figure 3.9: Relative collector current vs distance

3.4 Ultrasonic sensors

In its base set, 5 sensors are placed around the robot and are positioned and numbered as shown in figure 3.10.

5 sensors are in fact 5 pairs of ultrasonic devices where each pair is composed of one transmitter and one receiver.

The ultrasonic sensor are powered by a 20 Vdc source. The nominal frequency of these transducers is 40kHz +/- 1kHz.

Parameters such as max number of echo, timeout and active sensors are parametrizable through a configuration api (See 'C' command in chapter A). By default, you have maximum 3 echos, sensor 3 (front) is active, timeout is set to see from 20cm to 4 meters.

The last parameter is if the upper body is mounted or not (default mounted). When the upper body is mounted, there's some noise because of inside rebound echo, which are deleted in software. In fact, you could improve the detection of the nearest obstacles (20cm to 40cm) removing the upper body.

Each measure return the number of founded echos, the distance in cm of each echo, the amplitude of each, and the time (time stamp) when the echo was seen.

The value returned by the command, is the white noise measured before than a Ultrasonic pulse was sent. Then the real amplitude of each echo will be its amplitude minus the white noise. See command 'G' in chapter A for more details about the ultrasonic command.

For more details about the ultrasonic sensor, please check at Midas component, the transceiver is 400ST100 and the receiver is a 400SR100.

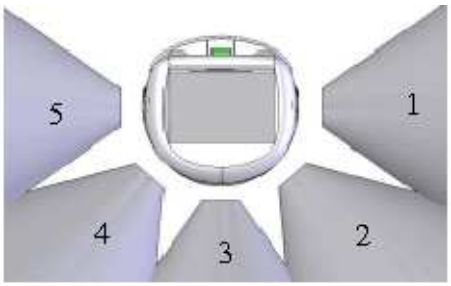


Figure 3.10: Position of the 5 UltraSonic sensors.

3.5 Battery

The Khepera III is equipped with a battery pack composed of two Li-Ion Polymer element. This provides a 7.4V volt battery with a 1400 mAH capacity. Using its embedded power, the robot is able to run completely autonomously during more than four hours, running with a basic configuration. When additional equipment are used, the autonomy is reduced as Khepera's extensions as the KoreBotLE rely on Khepera's batteries as a power source.

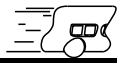
There is no specific power management system on the Khepera. When the batteries voltage falls under 6V, the battery cut himself the power supply to avoid a too important discharge of the cell. Users can implement their own software power management system to handle KoreBotLE to shutdown properly before this case happend. See command 'V' in chapter A for more details about battery management.

3.6 Power Supply

If an external power source is required or during batteries charge, power can be supplied through the power jack connector or through the micro-Match cable unroller connector. See section 4 for detailed description of connections.

Use only the power supply deliver by K-Team. If you want to supply the Khepera III with another device, make sure that the Voltage is +9V and that the power supply can deliver 2A.

SAFETY PRECAUTION: The power supply must be connected to the wall socket after all other connections are already made.



4.1 Configuration for batteries charge

To charge robot's batteries, make sure the following are correct:

- The power supply shall be connected to the robot.
- Warning: the robot battery may be charged with the robot ON or OFF. Be attentive that when the robot is ON, the charging time will be a bit longer.
- The power supply have to be connected on a wall plug.

There are three leds. A green led tells if the robot is powered or not. And two other leds are here if the robot is charging (red led ON) or if the charge is complete (green led ON).

During the charge, the red led is switched ON and the green led is switched OFF. The process is reversed at the end of the charging process. The charging time for an empty battery is about 180 minutes. At this moment the power supply can be unplugged. When charging, the battery can be as hot as 40 degrees.

4.2 Configuration for Robot-Computer Communication in standalone

This configuration allows communicating between the robot and a host computer through a serial link. The host computer is linked to the interface module using a standard RS232 line, while the interface module converts RS232 signal into a TTL level signal to communicate with the robot.

To use the standalone serial communication mode, please make sure the following are correct:

- No KoreBotLE is connected on the Khepera III.
- The robot battery is charged that means the power led is switched on.
- The robot must be connected to a KoreConnect module using the serial cable
- The KoreConnect should be connected to the host computer using a standard RS232 cable. in this mode, the cable has to be connected on the DB9 connector number 2 (see fig 4.1). You can easily purchase such a cable from your host computer dealer.
- Serial port configured as followed : 115200bps, 8 Data bits, 1 stop bit , no parity, no hardware control.

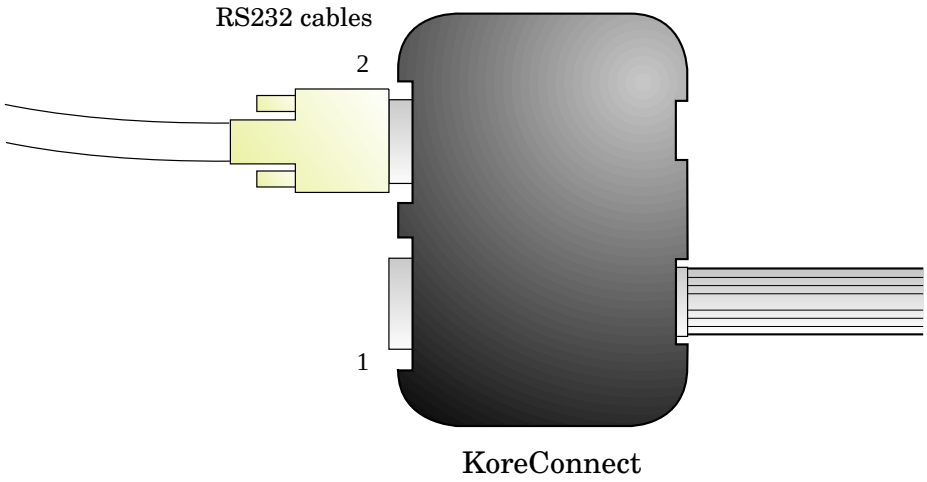


Figure 4.1: Connection when the robot is used without a KoreBotLE.

4.3 Configuration for Robot-Computer Communication with KoreBotLE

4.3.1 Serial link

This configuration allows communicating between the KoreBotLE plugged on the robot and a host computer through a serial link. In this configuration it is not possible to communicate with the robot base itself as explained in the previous chapter. The host computer is linked to the interface module using a standard RS232 line. The adaptation RS232/TTL is made on the KoreBotLE.

To use the serial communication mode, please make sure the following are correct:

- A KoreBotLE is connected on the Khepera III.
- The charged Battery or a power supply is plugged, and the robot is turned ON .
- The robot must be connected to a KoreConnect module using the serial cable
- The KoreConnect should be connected to the host computer using a standard RS232 cable. in this mode, the cable has to be connected on the DB9 connector number 1 (see fig 4.2) You can easily purchase such a cable from your host computer dealer.
- Serial port configured as followed : 115200bps, 8 Data bits, 1 stop bit, no parity, no hardware control.

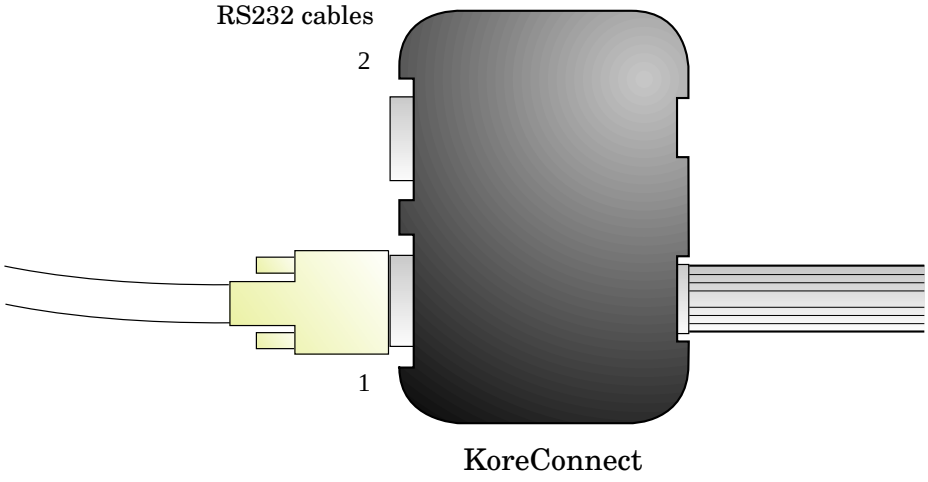


Figure 4.2: Connection when the robot is used with a KoreBotLE.

4.3.2 USB link

This configuration allows communicating between the KoreBotLE plugged on the robot and a host computer through a USB link. There are two ways to connect the USB to the KoreBotLE. With a KoreConnect and standard USB cable or with only a mini-USB cable.

For more information about the USB communication protocole with a KoreBotLE, please refer to the KoreBotLE documentation (found on web site www.k-team.com).

To use the USB communication mode, please make sure the following are correct:

- A KoreBotLE is plugged on the Khepera III.
- The charged battery or a power supply is plugged, and the robot is turned ON .
- The robot must be connected to a KoreConnect module and then a USB cable or a mini-USB cable is plugged directly on the USB miniAB connector of the robot (see figure 3.1).
- The USB cable (standard or mini) should be connected to the host computer. These two type of cable can be easily purchased from any computer dealer.

5.1 Testing the primary serial link

Before any further operations, the serial link between the host computer and the robot should be tested. Please read the following instructions to test the serial communication mode:

- Make sure all connections are correct (see chapter 4 for details).
- The robot should be placed on a flat and safe surface. The battery switch must be OFF.
- A terminal emulator should be running on the host computer. Make sure the terminal is connected to the correct serial port. **The terminal configuration must be set to 115200 Baud, 8 bit, 1 start bit, 1 stop bit, no parity and no hardware control**.

When powered on, the robot should send a message (see fig 5.1) to the terminal emulator.

To test that the KheperaIII is able to receive command, try to send the B command to ask the Khepera for the operating system version and revision.

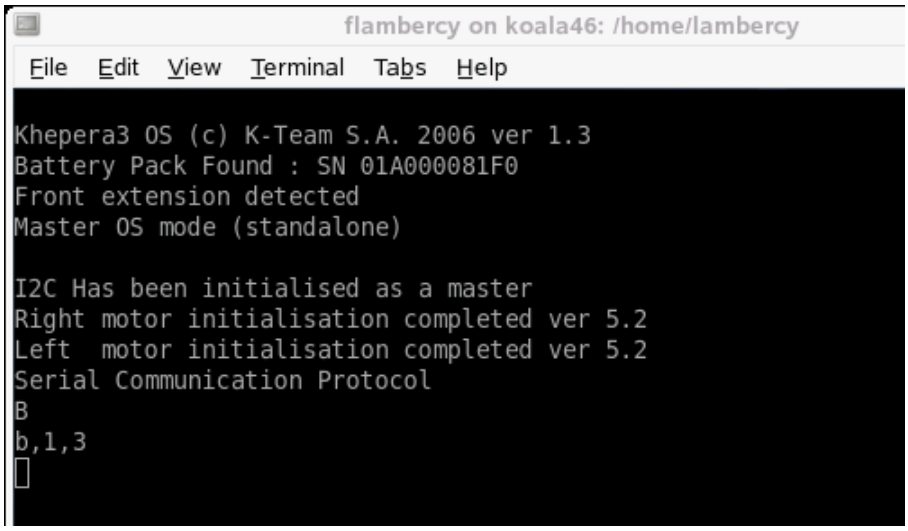


Figure 5.1: Boot message of the Kherpera III.

5.2 Testing the secondary serial link

As the robot is suited to be used with a KoreBotLE it is important to test the serial link between the host computer and the KoreBotLE itself.

- Make sure all connections are correct (see chapter 4 for details).
- The robot should be placed on a flat and safe surface. The battery switch must be OFF.
- A KoreBotLE needs to be plugged on the appropriate socket.
- A terminal emulator should be running on the host computer. Make sure the terminal is connected to the correct serial port. **The terminal configuration must be set to 115200 Baud, 8 bit, 1 start bit, 1 stop bit, no parity and no hardware control**.

When powered on, the KoreBotLE will send the normal linux boot message to the terminal emulator (see fig 5.2).

If any of the serial link is not working properly it is then important to ensure that the serial communication settings has been configured correctly.

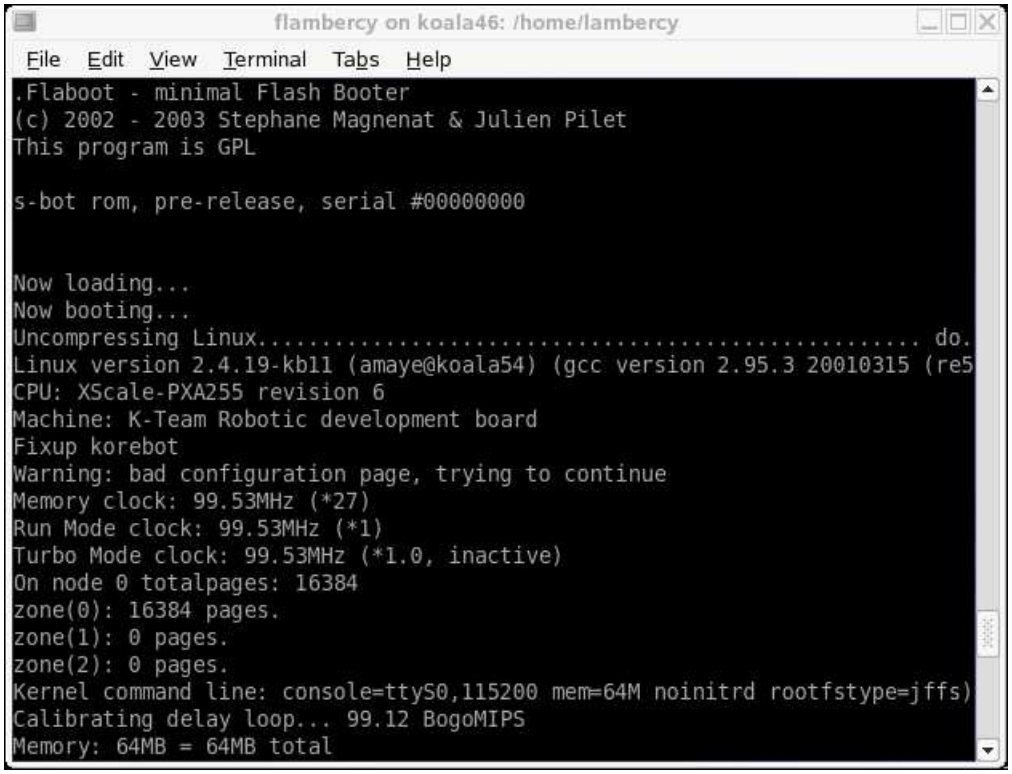


Figure 5.2: Boot message of the KoreBotLE.

5.3 Serial communication protocol

When the robot is used as standalone (without a KoreBotLE), the serial communication protocol is designed to control all Khepera's functions using a RS232 serial line. The serial line configuration (baudrate as well as data, start, stop and parity bits) for the host computer must match the robot's configuration.

The host computer and the Khepera robot are communicating with ASCII messages. Each single interaction is composed by:

- A command, beginning with one ASCII capital letters and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return or a line feed, sent by the host computer to the Khepera robot.
- A response, beginning with the same one ASCII letters of the command but in lower-case and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return and a line feed, sent by the Khepera to the host computer.

During the entire communication, the host computer is acting as a master and the Khepera as slave. All communications are initiated by the master.

Two different types of interactions are possible. The first set of interactions is used to set up the robot configuration from the host computer (set up serial line, changing controllers configuration,...), the second set of interactions is used to control the robot (controlling motors, reading sensors value,...).

A set of commands is available as detailed in appendix.

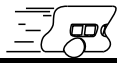
5.3.1 Testing a simple interaction

Testing some basic commands is the best method to understand the serial protocol and tools available on the Khepera. Using a properly configured serial link between the robot and a computer, please follow the instructions bellow:

- Type the capital letter **B** followed by a carriage return or a line feed.
- The robot must respond with **b** followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter **N** followed by a carriage return or a line feed.
- The robot must respond with **n** followed by 11 numbers separated by a comma and terminated by a line feed. These numbers are the values of the robot proximity sensors presented in section 3.1.6.
- Retry the same command (N) putting some obstacles in front of the robot. The response must change.
- Type the protocol command **D,l10000,l-10000** followed by a carriage return or a line feed.

- The robot must start turning on place and respond with **d** and a line feed.
- To stop the robot type the protocol command **D,l0,l0** followed by a carriage return or a line feed.
- Try other commands following the description given in Appendix A.

6 KoreBotLE programming



This section explain how to write a simple program for the KoreBotLE to control the Khepera III base. The Khepera III will act as peripheral of the KoreBotLE.

Please refer to the KoreBotLE user manual (found on www.k-team.com) for further details on KoreBotLE programming. This document only provides a basic example of program compilation and execution.

The first simple executable for KoreBotLE/Khepera III will be built assuming:

- A working arm-linux toolchain is installed on your host computer.
- The NFS link is active and a shared directory (*mySharedDirectory*) is mounted on */mnt/nfs*.

A source file should be first created and edited with a very simple C program such as the following. People familiar with the KoreMotor will notice that the code is quite similar. The controlers have been embedded in the robot but act as two I2C slaves.

```
/* kh3test.c */
#include <korebot/korebot.h>

static knet_dev_t * mot1;
static knet_dev_t * mot2;

int main()
{

  kh3_init();

  /* open various socket and store the handle in their respective pointers */
  dsPic  = knet_open( "Khepera3:dsPic" , KNET_BUS_I2C , 0 , NULL );
  mot1  = knet_open( "Khepera3:mot1" , KNET_BUS_I2C , 0 , NULL );
  mot2  = knet_open( "Khepera3:mot2" , KNET_BUS_I2C , 0 , NULL );

  /* initialize the motor controler 1 */
  kmot_SetMode( mot1 , kMotModeIdle );
  kmot_SetSampleTime( mot1 , 1550 );
  kmot_SetMargin( mot1 , 6 );
  kmot_SetOptions( mot1 , 0x0 , kMotSWOptWindup | kMotSWOptStopMotorBlk
| kMotSWOptDirectionInv );
```

```
  kmot_ResetError( mot1 );
  kmot_SetBlockedTime( mot1 , 10);
  kmot_ConfigurePID( mot1 , kMotRegSpeed , 400 , 0 , 10 );
  kmot_ConfigurePID( mot1 ,kMotRegPos,620,3,10);
  kmot_SetSpeedProfile(mot1 ,30,3);

  /* initialize the motor controler 2 */
  kmot_SetMode( mot2 , kMotModeIdle );
  kmot_SetSampleTime( mot2 , 1550 );
  kmot_SetMargin( mot2 , 6 );
  kmot_SetOptions( mot2 , 0x0 , kMotSWOptWindup | kMotSWOptStopMotorBlk );
  kmot_ResetError( mot2 );
  kmot_SetBlockedTime( mot2 , 10);
  kmot_ConfigurePID( mot2 , kMotRegSpeed , 400 , 0 , 10 );
  kmot_ConfigurePID( mot2 ,kMotRegPos,620,3,10);
  kmot_SetSpeedProfile(mot2 ,30,3);

  /* For ever loop */
  while(1)
  {
/* Wait 5 seconds */
sleep(5);

/* Tell to the motor controller to move the Khepera III forward */
kmot_SetPoint( mot1 , kMotRegSpeed , -motspeed );
kmot_SetPoint( mot2 , kMotRegSpeed , motspeed );

/* Wait 5 seconds */
sleep(5);

/* Tell to the motor controller to stop the Khepera III  */
kmot_SetPoint( mot1 , kMotRegSpeed , 0 );
kmot_SetPoint( mot2 , kMotRegSpeed , 0 );
  }
}
```

This source code can be cross-compiled using *arm-linux-gcc* which supports most standard *gcc* options:

```
arm-linux-gcc kh3test.c -o k3test
```

An kh3test executable file should be created, that can be executed on the KoreBotLE, or eventually another arm-linux machine.

To execute the program, simply copy kh3test to the nfs shared directory, then execute it from a KoreBotLE terminal. If the shared directory is mounted on the default */mnt/nfs*, first change working directory:

```
cd /mnt/nfs
```

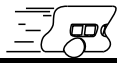
Then check that kh3test is present:

```
ls -l
```

And execute it:

```
./k3test
```

A Communication protocol



This communication protocol allows complete control of the robot functions through a RS232 serial line. The required configuration is presented in section 4.3. The serial line set-up on your host computer must match the one set on the robot according to the chosen running mode.

The protocol is made of commands and responses, all in standard ASCII codes. A command is sent from the host computer to the robot: it is starting with an upper case character and followed, if necessary, with numerical or literal parameters separated with comma and terminated by a line feed. The response is sent by the robot to the host computer: it is starting with the same character that was initiating the command but using lower case, and followed, if necessary, with numerical or literal parameters separated with comma and terminated by a line feed.

When a command ask for a value bigger than 8 bits (bigger than 256) the following prefixes must be added:

- d : for a 16 bits value
- l : for a 32 bits value
- f : for a floating value

Example to set the Speed: D,l20000,l20000

To better understand this protocol, please refer to the following simple test:

- Set the connection configuration presented in section 4.3.
- Start a terminal emulator on your host computer with the serial line set to 115200 Baud, 8 bit data, 1 start bit, 1 stop bits, no parity.
- Type the capital letter B followed by a carriage return or a line feed.
- The robot must respond with b followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter N followed by a carriage return or a line feed.
- The robot must respond with n followed by 11 numbers separated by a comma and terminated by a line feed. These numbers are the values of the proximity sensors presents on the robots.
- Retry the same command (N) putting some obstacles on the front of the robot. The response must change.
- Try other commands...

A Start Braitenberg mode

Command:	A, mode (8 bits)
Answer:	a
Effect:	Start the Braitenberg mode with the Infrared sensor (mode = 0) or with the ultrasonic sensor (mode = 1).
Syntax:	A,0

B Read software version

Command:	B
Answer:	b, version_of_BIOS, revision_of_BIOS
Effect:	Give the software version stored in the robot's EEPROM.

C Configure the current firmware operation

Command:	C, array_index(8bits), array_value(16bits)
Answer:	c
Effect:	Configure the actual firmware state. The configuration array is to be seen as a table. Each line in the table means a parameter and a value may be associated with.
Syntax:	C,0,d3

Index	Default Value	Description:
0	0b00000100	Decide how many US sensor will be active. See the table bellow for supported configurations.
1	3	Decide the maximum echo numbers.
2	0	Not Used.
3	0	Not used.
4	0xFFFF	IR sensor mask, manage the active IR sensor (bit0 = IR0, bit1 = IR1, etc...).
5	12	IR gain for Braintenberg.
6	1	Upper body mounted. If the upper body isn't mounted, change this value to 0 to get better US measure.

The supported US sensors configuration are listed in the following table.

Mask	Binary	Decimal	Description:
NONE	0b00000000	0	None of the US sensor.
US1ONLY	0b00000001	1	Only the US sensor 1.
US2ONLY	0b00000010	2	Only the US sensor 2.
US3ONLY	0b00000100	4	Only the US sensor 3.
US4ONLY	0b00001000	8	Only the US sensor 4.
US5ONLY	0b00010000	16	Only the US sensor 5.
US1TO2	0b00000011	3	US sensor 1 to 2.
US2TO3	0b00000110	6	US sensor 2 to 3.
US3TO4	0b00001100	12	US sensor 3 to 4.
US4TO5	0b00011000	24	US sensor 4 to 5.
US1TO3	0b00000111	7	US sensor 1 to 3.
US2TO4	0b00001110	14	US sensor 2 to 4.
US3TO5	0b00011100	28	US sensor 3 to 5.
US1TO4	0b00001111	15	US sensor 1 to 4.
US2TO5	0b00011110	30	US sensor 2 to 5.
USALL	0b00011111	31	All US sensors.

D Set speed

Command:	D, speed_motor_left (32bits), speed_motor_right (32bits)
Answer:	d
Effect:	Set the speed of the two motors. See in section 3.2.1 to calculate the equivalent Speed.
Syntax:	D,l20000,l20000

E Read speed

Command:	E
Answer:	e, speed_motor_left, speed_motor_right
Effect:	Read the instantaneous speed of the two motors. See in section 3.2.1 to calculate the equivalent Speed.
Example:	e,20000,20000

F Set Target Profile

Command:	F, target_position_motor_left (32bits), target_position_motor_right (32bits)
Answer:	f
Effect:	Set a position to be reached. The move will be performed with three phase, a acceleration to reach the maximum speed, a constant speed and a deceleration phase before the finish position. The unit is the pulse, each one corresponds to 0,047 mm (with encoder resolution x4 (default)).
Syntax:	F,l1000,l1000

G Get a measure from the US peripheral

Command:	G, US_number (8bits)
Answer:	g, EchoNbr, Dist1, Ampl1,Time1,Dist2,Ampl2, Time2, Dist3, Ampl3, Time3, Dist4, Ampl4, Time4, Dist5, Ampl5, Time5, Noise
Effect:	Retrieve US measures from a given sensor number. The sensor number is supposed to be 1 to 5.
Syntax:	G,3
Example:	g,1,27,2900,100000,0,0,0,0,0,0,0,0,0,0,0,0,1800

H Configure the PID controler

Command:	H, T (16bits), Kp (16bits), Ki (16bits), Kd (16bits)
Answer:	h
Effect:	Set the proportional (Kp), the integral (Ki) and the derivative (Kd) parameters of the T regulator where T can be 0 (speed controler) or 1 (position controler).
Syntax:	H,d1,d200,d5,d10

I Set position

Command:	I, position_motor_left (32bits), position_motor_right (32bits)
Answer:	i
Effect:	Set the 32 bit position counter of the two motors. The unit is the pulse, each one corresponds to 0,047 mm (with encoder resolution x4 (default)).
Syntax:	I,l1000,l1000

J Configure the speed profile controller

Command:	J, max_speed (16bits), acceleration (8bits)
Answer:	j
Effect:	Set the speed and the acceleration for the trapezoidal speed shape of the position controller. The max_speed parameter indicates the maximal speed reached during the displacement. See section 3.2.1 for more details. At the reset, these parameters are set to standard values: max_speed to 20000, acc to 64.
Syntax:	J,d15000,10

K Set Programmable Led

Command:	K, LED (8bits), State (8bits)
Answer:	k
Effect:	Set the state of the two Programmable Leds (see section 3.1.3, Led 5 & 6). LED select which one must be set (0 = led 5, 1 = Led 6). State select the operation (0 = OFF, 1 = ON, 2 = Change state).
Syntax:	K,0,1

L Set speed open loop

Command:	L, speed_motor_left (32bits), speed_motor_right (32bits)
Answer:	l
Effect:	Set the speed of the two motors without a PID controller. Where a value of 1023 correspond to PWM of 100%.
Syntax:	L,l500,l500

M Init Motors

Command:	M
Answer:	m
Effect:	Initialise or reset the motor controllers. Must be used if an error like motor blocked or current limit occured

N Read proximity sensors

Command:	N
Answer:	n,val_sens_back_left, val_sens_left_90, val_sens_left_45, val_sens_front_left, val_sens_front_right, val_sens_right_45, val_sens_right_90, val_sens_back_right, val_sens_back, val_sens_ground_right, val_sens_ground_left, time_stamp
Effect:	Read the 10 bit values of the 11 proximity sensors. The last argument is the relative time stamp of the measure. (section 3.3), from the front sensor situated at the left of the robot, turning clockwise to the back-left sensor.

O Read ambient light sensors

Command:	O
Answer:	n,val_sens_back_left, val_sens_left_90, val_sens_left_45, val_sens_front_left, val_sens_front_right, val_sens_right_45, val_sens_right_90, val_sens_back_right, val_sens_back, val_sens_ground_right, val_sens_ground_left, time_stamp
Effect:	Read the 10 bit values of the 11 light sensors (section3.1.6), from the front left sensor turning clockwise to the back-left sensor. The last argument is the relative time stamp of the measure.

P Set Target Position

Command:	P, target_position_motor_left (32bits), target_position_motor_right (32bits)
Answer:	p
Effect:	Set a position to be reached. The move will be performed without acceleration and decceleration. The unit is the pulse, each one corresponds to 0,047 mm (with encoder resolution x4 (default)).
Syntax:	P,l1000,l1000

Q Unused

R Read position

Command:	R
Answer:	r, position_motor_left, position_motor_right
Effect:	Read the 32 bit position counter of the two motors. The unit is the pulse, each one corresponds to 0,047 mm (with encoder resolution x4 (default)).

S Unused

T Unused

U Enter the serial boot loader mode

Command:	U
Answer:	u
Effect:	Switch to the firmware upgrade mode. Be careful, not to upload a bad hex file into the dsPic memory space. If the dsPic memory is corrupted, an external programmer will be required to reflash the memory.

V Get Battery State

Command:	V, index (8bits)
Answer:	v, battery_measure_unit, battery_measure_decimal
Effect:	0: Read the voltage of the battery (Units: 1V , 0.1mV) . 1: Read the current of the battery (Units: 1A, 0.1mA). 2: Read the battery Average Current (Units: 1A, 0.1mA). 3: Read the battery Absolute Remaining Capacity (Units: 1Ah). 4: Read the Temperature of the battery (Units: 1 C, 0.1́). 5: Read the battery Relative Remaining Capacity (Units: %).

W Unused

X Unused

Y Unused

Z Zero the relative time stamp

Command:	Z
Answer:	z
Effect:	Reset the relative time counter.