KOA-PICOITX

user manual

K-Team

MOBILE ROBOTICS

version 1.0
april 2010

Documentation Author

Frédéric Lambercy
K-Team S.A.
Rue Galilee 9, Y-Park
1400 Yverdon-les-Bains
Switzerland

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

LEGAL NOTICE:

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

TABLE OF CONTENTS

1. INTRODUCTION

1.1 The Koa-PicoITX

Thank you for buying Koa-PicoITX extension. This computer extension for Koala will give you the opportunity to make advanced embedded applications. Thanks to its powerful processor, the Koa-PicoITX is perfect for image processing, mapping and odometry calculation. The Koa-PicoITX was developed to work with a Koala, but can be used as a standalone mini-computer in your own application.

1.2 Specifications

The Koa-PicoITX is based on an EPIA-P700-10L pico-ITX board from VIA. The main board is connected to a special board which provide different IO, an additional serial port, a power-button, led indication and the possibility to mount up to two KoreMotorLE to drive four motors (for two Pan-Tilt cameras). The case is in anodized aluminum to protect the main board and its extensions.

Model Name	EPIA-P700-10L
Processor	1.0GHz VIA C7® 500MHz VIA EdenTM ULV
Chipset	VIA VX700 Unified Digital Media IGP chipset
RAM	1GB
Hard drive	80GB, Maxtor MobileMax STM980215A
System Memory	DDR2 533/667 SODIMM socket (effective speed to 533MHz)
VGA	Integrated VIA UniChrome™ Pro II 3D/2D AGP graphics with MPEG-2/4 and WMV9 video decoding acceleration
LAN	Gigabit Ethernet port controlled by a VIA VT6122 Gigabit LAN controller
Audio	Line-out, Line-in, & Mic-in controlled by a VIA VT1708B High Definition Audio Codec
BIOS	Award BIOS 4/8Mbit flash ROM
I/O Connectors	2x DB9 serial port male connector 1x VGA connector 1x Gigabit Ethernet connector 3x USB 2.0 ports 3x Audio jacks: (Line-out, Line-in, Mic-in) Up to 4 Motor connectors 2x Terminal Blocks (Supply, IO & I2C bus) 1x KoreBot power supply connector (+12V, +5V & GND)
Operating system	Linux Ubuntu 9.10
System Monitoring & Management	Wake-On LAN, Keyboard Power-on, Timer Power-on System power management, AC power failure recovery Watch Dog Timer
Operating Temperature	0°C ~ 50°C
Operating Humidity	0% ~ 95% (relative humidity; non-condensing)
Dimensions	145 x 125 x 91 mm
Weight	860g

1.3 How to use this handbook

This handbook introduces the Koa-PicoITX extension and how to use it. For a quick start, jump to section 2.3, "First Start-up".

If this handbook does not answer one of the problems you are confronted with, please consult the K-Team web site (www.k-team.com) and, especially the Forum and the FAQs.

- **Unpacking and Inspection:** Koa-PicoITX's package description and first start-up.
- **Hardware installation:** explanation on how to mount the Koa-PicoITX on a Koala and how to mount extension inside the case.
- **Usage:** description of the usage of the PicoITX and its devices.
- **Programming:** description of the programming methods of the Koa-PicoITX.
- **Software reinstallation:** instruction to reinstall all the software needed to program the Koa-PicoITX.
- **Warranty:** legal notice on the Koa-PicoITX Warranty.

1.4 Safety precautions

Here are some recommendations on how to correctly use the Koa-PicoITX extension:

- **Keep the case away from wet area**. Contact with water could cause malfunction and/or breakdown.
- **Store your computer in a stable position**. This will avoid the risks of falls, which could break it or cause damage to a person.
- **Use only the official supply cable which is delivered with the Koa-PicoITX.** Do not try to use another charger; this can cause irreversible damage to the battery.
- **Never leave the Koala and its Pico-ITX powered when it is unused**. When you have finished working with Koala, turn it off. It will save the battery life

1.5 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 Lithium Polymer, 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

2.1 Package Content

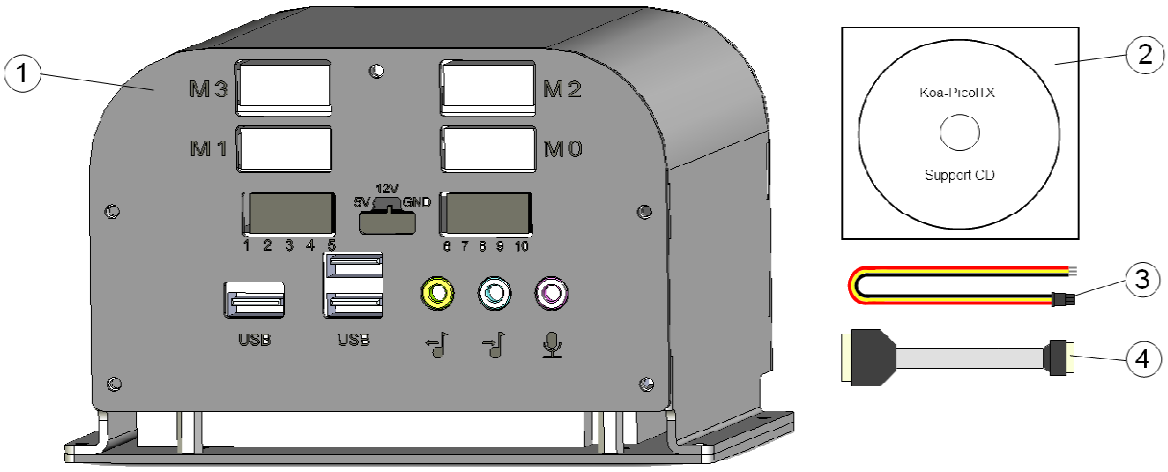

Figure 2.1: Content of the Koa-PicoITX extension

Your package should contain the following items:

1. Koa-PicoITX box
2. Koa-PicoITX Support CD
3. Power Supply cable
4. Serial cable (DB9->DB15)
5. 4x fixation screws

2.2 Koa-PicoITX Overview

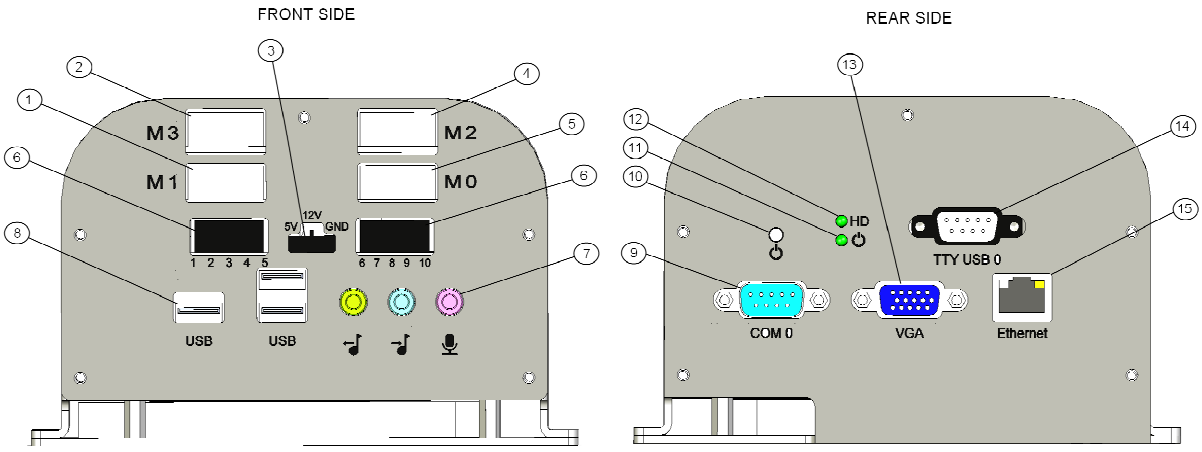

Figure 2.2: Koa-PicoITX overview

1 Motor connector M1
2 Motor connector M3
3 Power supply connector, use the 3 color cable (red = 5V, yellow = 12V, black = GND)
4 Motor connector M2
5 Motor connector M0
6 Terminal block
- *1 : +5V*
- *2 : +3.3V*
- *3 : I^2C clock*
- *4 : I^2C data*
- *5 : GND*
- *6 : GPIO2*
- *7 : GPIO3*
- *8 : GPI4*
- *9 : GPI5*
- *10: GND*

7 Audio jacks connector (Line-IN, Line-OUT, Mic-IN)
8 USB 2.0 connectors
9 COM0 serial connector (/dev/ttyS0)
10 Power button
11 Power LED indicator
12 Hard Drive LED indicator
13 VGA output
14 USB serial port (/dev/ttyUSB0)
15 Gigabit Ethernet connector

2.3 First Start-up

2.3.1 Hardware startup

If you have received the Koa-PicoITX already mounted and connected to a Koala, you can directly jump to the section "Software startup". In other case, please be sure to respect all of the guidelines below.

All manipulations must be executed when Koala is turn off. To avoid any damage, please remove the battery of the Koala during all the process.

To mount the Koa-PicoITX box on a Koala, you must first unmount the black protection plate at the rear of the Robot. Place the Koa-PicoITX box on the four fixations points (see n°2 in page 5 of the "Koala User Manual"); the rear size of the box must be turned at the back of the Robot.

http://ftp.k-team.com/koala/documentation/KoalaSilverUserManual.pdf

Connect the serial cable to the USB serial port (n°14 in the overview) and the RS232 serial port on the rear of the Koala (see n°4 page 4 in the "Koala User Manual"). Place the jumper in the back “DCE” serial line position (see page 8 in the "Koala User Manual") and set the running mode selector of the Robot to the position A (115200bps communication).

Connect a VGA monitor, an USB keyboard and an USB mouse to the Koa-PicoITX.

Connect the power supply cable to the Koala terminal blocks as describe below:

- Yellow => +VBAT (+12V)
- Red => VCC (+5V)
- Black => GND (power ground)

(See page 16 of the Koala User Manual for more details)

And finally, connect the supply cable to the Koa-PicoITX (see connector n°3 in the overview).

Then plug the battery (or the DC/DC converter) in the Robot, turn on the Koala Robot and push the power button of the Koa-PicoITX. The boot of the computer must appear on the screen. Then follow the "Software startup" instructions.

2.3.2 Software startup

At the login, enter the default user name and password as follows:

username: ***picoitx***
password: ***root*** <= this will also be the ***sudo*** password for the following chapters.

You will arrive at the desktop depicted in the figure below. You have the menu bar at the top of the desktop, with direct access to help, file manager, terminal consol, network manager and logout buttons.

You can have more help at http://help.ubuntu.com/.

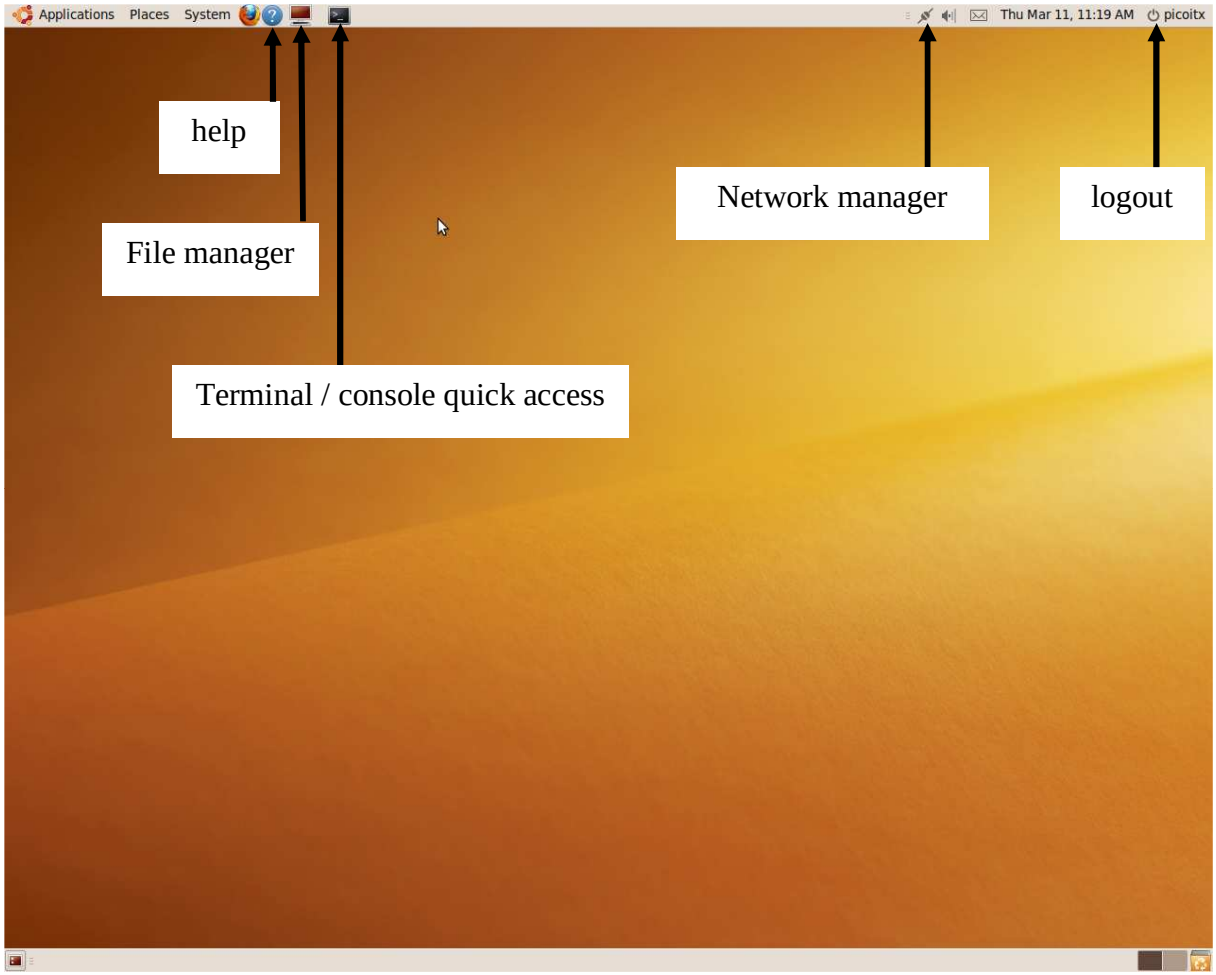

Figure 2.3: Ubuntu desktop

The system should be already fully working with the gcc compiler installed and the ***libkorebot***. See chapter 4, PROGRAMMING" for software development and the next chapter for using the different features.

3. Usage

3.1 Access to the Koala robot

You can communicate to the Koala robot and control it through the serial port.
Open a terminal console and type:

sudo minicom -s

Enter ***root*** as password then go the "Serial port setup" menu and configure as described in figure 2.4.

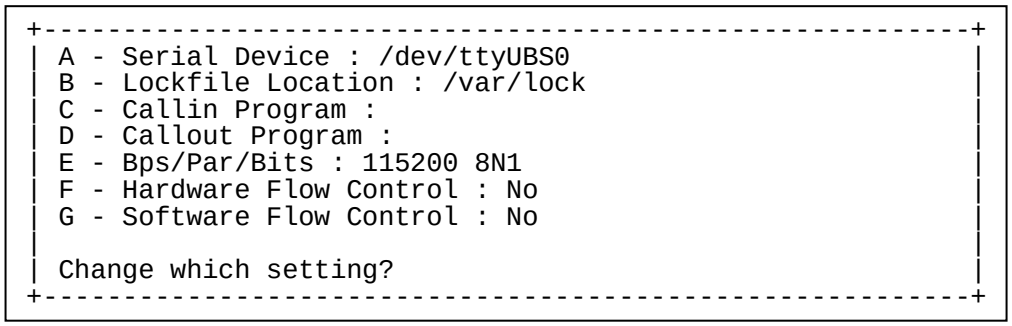

Figure 2.4: Minicom serial parameters

Save the settings with the command "***Save setup as dfl***" of the menu [configuration]. The commands available are listed in the appendix A of the Koala user manual.

Example: the command B will read the bios and protocol versions. If you enter the uppercase ***B***, then press ***return*** this should return: ***b,1.19,1.19***

See chapter 4 PROGRAMMING" for software development.

3.2 Network configuration

The system should connect autonomously at start-up to your wired network if a Ethernet cable is connected and you have a dhcp server without identification.

If this is not the case, with the icon in the figure above you can setup your network connection.

You can have more information about the operation system Ubuntu there:
http://help.ubuntu.com/

and about the network manager there:
https://help.ubuntu.com/9.10/internet/C/networkmanager.html

Network access:

From the main menu at the top, choose "Places" then "Network" to see your network, or "Connect to Server" to connect to a server.

3.3 File transfer

You can also use a terminal to connect or transfer files with the program ***scp***:

scp FILE USER@ REMOTE_IP:/PATH

where

FILE : is the file to transfer
USER: is the username at the remote computer
REMOTE_IP : is the remote computer ip address
PATH: is the path to the file destination on the remote computer

3.4 Remote connection

You can connect to the robot by different ways:

- for transferring files with the ***scp*** command seen above.
- using the RemoteDesktop feature (configure the System/Preferences/Remote Desktop menu).
- with the ***ssh*** protocol command: ***ssh –X picoitx@ROBOT_IP***

  from where you can launch:
  - the file manager: ***nautilus &***
  - any other graphical or console program

3.5 Digital input/output ports usage

In the directory ***~/software/libkorebot_VERSION/src/tests*** , the ***gpio_test*** program is already compiled.

- Run it with the command: ***./gpio_test***

  => You will have a prompt >

- The GPIO 2 and 3 can be used in input or output. The GPIO 4 and 5 are only outputs.
- With the command ***help*** you have the help.
- You configure the direction of the port with the command ***configio*** :

configio IO_PORT_NUMBER DIRECTION

where

IO_PORT_NUMBER: port number 2..5

DIRECTION : 0 = input, 1 = output

- Set the IO to 1 with: ***setio IO_PORT_NUMBER***
- Set the IO to 0 with: ***cleario IO_PORT_NUMBER***
- Read the IO with: ***readio IO_PORT_NUMBER***
- Exit the program with ***quit***

See chapter 4 PROGRAMMING" for software development.

3.6 Motors usage

You can use the motors connectors (M0, M1, chapter 2.2: "Koa-PicoITX Overview") to connect one or two motors (up to 4). See the KoreMotorLE user manual for more details about the connections and motors:

http://ftp.k-team.com/korebot/documentation/KoreMotorLE.UserManual.pdf

In the directory ***~/software/libkorebot_VERSION/src/tests*** , the ***kmotLE_test*** program is already compiled. Run it with the command:

./kmotLE_test MOTOR_NAME

where ***MOTOR_NAME = KoreMotorLE:PriMotor1*** : motor M0
KoreMotorLE:PriMotor2 : motor M1

=> You will have a prompt >

With the command ***help*** you have the help.

The command ***init*** set the default parameters. It is necessary to run in before any other commands because it sets and resets different default settings.

You will have to set the controller (PID) with the command ***setpid***:

- Set it with the command: ***setpid REGULATION_TYPE P I D***

where ***REGULATION_TYPE*** = ***pos*** position control

speed speed control

torque torque control

P : proportional parameter

I : integral parameter

D : derivative parameter

- You can set the speed, the position or the torque with the commands:

  s***etspeed SPEED***

  setpos POS

  settorque TORQUE

- And stop it with ***stop***
- Exit the program with ***quit***

See the source code file ***~/software/libkorebot_VERSION/src/tests/kmotLE_test.c*** or the KoreMotorLE user manual for details:

http://ftp.k-team.com/korebot/documentation/KoreMotorLE.UserManual.pdf

See chapter 4 PROGRAMMING" for software development.

3.7 Pan-tilt camera usage

You can connect up to two Pan-tilt cameras using the motors connectors (chapter 2.2: "Koa-PicoITX Overview": M0 and M1 for the first one, M2 and M3 and another KoreMotorLE needed). See the Pan-Tilt user manual for more details about the connections:

http://ftp.k-team.com/koala/pan-tilt/PanTiltCameraManual.pdf

In the directory ***~/software/libkorebot_VERSION/src/tests*** , the ***kmotLE_pantilt*** program is already compiled. This program makes the pan-tilt camera moving. Run it with the command:

./kmotLE_pantilt MOT0 MOT1 [NB_CYCLES]

where

MOT0: PAN motor number (usually 1)

MOT1: TILT motor number (usually 2)

[NB_CYCLES]: optional: number of cycles

See chapter 4 PROGRAMMING" for software development.

4. PROGRAMMING

Knowledge needed:

- C programming
- command line compilation (Makefile)

You have two choices for programming:

- directly on the robot and PicoITX (fastest, nothing more to be installed)
- on another computer (better for remote or if the robot is not available)

4.1 Directly on the robot

4.1.1 Template program

The compiler gcc should be already installed:
launching ***gcc*** in a terminal should return "gcc: no input files".

The ***libkorebot***, which is a K-Team library providing facilities for using different robots and modules, should already be installed in ***~/software/libkorebot_VERSION***, where ***VERSION*** is the actual version. The documentation for the library is available here:
http://ftp.k-team.com/korebot/libkorebot-doc/files.html

There is a template program, with which you can start your own program, located in:

~/software/libkorebot_VERSION/template

With the following commands, you can compile then run it:

```
make clean
make
./template
```

For specific documentation and examples on the programming of the Koala, read the next sub-chapter "Using examples", specially the ***koala_test.c*** example.

Remarks:
If you modify the program source code name (currently ***prog-template.c***), you will have to modify its occurrences in the ***Makefile*** file, which is the script file launched when the above ***make*** is executed.

4.1.2 Using examples

You can find many examples using the ***libkorebot*** in the directory of the library:

~/software/libkorebot_VERSION/src/tests

All the examples below are already compiled. You can modify them or take some part to add to your main program. For recompiling them, in the directory ***~/software/libkorebot_VERSION***, execute:

```
make clean
make all
```

You can execute them with the command:

./PROGRAM_NAME

Here after is the list of examples of source code recommended for starting programming with the Koa-PicoITX and its extensions:

- Make the Koala move: ***koala_test.c***

  Here after are desribed the steps to program the robot. The source code is listed in figure 4.1 below.

  - Firstly, a device pointer of type ***knet_dev_t*** must be declared.
  - The library debug level is set ***kb_set_debug_level*** and the library initialised ***kb_init***.
  - Then the device is open with ***knet_open*** .
  - From now, any command can be sent to the Koala. By example ***koa_setSpeed*** for setting the speed of the robot, which will make it move directly. Some main functions are listed below:
    - setting speed:

      int koa_setSpeed (knet_dev_t * dev, short int left, short int right)
    - setting motors position:

      int koa_setPosition(knet_dev_t * dev, long left_pos, long right_pos)
    - reading motors position:

      int koa_readPosition(knet_dev_t * dev, int * position)
    - reading proximity sensors:

      int koa_readProximity (knet_dev_t * dev, int * sens_table)
    - reading battery level:

      int koa_readBattery(knet_dev_t * dev)
    - and many others. For a detailed description check in the file ***~/software/libkorebot_VERSION/src/koala.c***.
  - at the end the ***kb_config_exit*** function is called.

```
// koala_test.c

#include <korebot/korebot.h>

int main( int argc , char * argv[] )
{
  knet_dev_t *koala;
  int rc;
  unsigned char buf[16];
  unsigned char a,b,c,d;

  kb_set_debug_level( 2 );

  if((rc = kb_init( argc , argv )) < 0 )
  return 1;

  koala = knet_open( "Koala:Robot", KNET_BUS_ANY, 0 , NULL );
  if(!koala)
    printf("Open failed\r\n");

  koa_getOSVersion(koala,&a,&b,&c,&d);
  printf("OS revision %d.%d  protocol %d.%d\r\n",a,b,c,d);

  koa_setSpeed(koala,10,10);
  sleep(1);
  koa_setSpeed(koala,-10,-10);
  sleep(1);

  koa_setSpeed(koala,0,0);

  kb_config_exit();

  return 0;
}
```

Figure 4.1: koala_test.c source code

For this file, you must change the serial port in the ***libkorebot*** configuration files. Edit it with the following command on the Koa-PicoITX:

sudo gedit /etc/libkorebot/Koala.knc

Change the line

device Robot rs232 /dev/tts/2

to

device Robot rs232 /dev/ttyUSB0

Then execute the program with: ***sudo ./koala_test***

- digital inputs outputs: ***gpio_test.c***
- additional motors: ***kmotLE_test.c***
- pan-tilt camera: ***kmotLE_pantilt.c***

4.2 On another computer

You must have an ix86 compatible computer with the operating system Linux. We do not provide any cross-compiler.

You can install the Ubuntu operating system (http://www.ubuntu.com/) in a dual boot configuration, or install the VirtualBox virtual machine from Sun Microsystems with the Ubuntu OS installed into it: http://www.virtualbox.org/ .

Then you install the compiler and the ***libkorebot*** as described in the annexes. You don't need to install the ***gpio*** and ***i2c*** drivers as they are working only on the PicoITX.

Finally you can compile your program as explained in chapter above 4.1 above"Directly on the robot" then transfer it to your robot (see chapter 3.3 "File transfer").

5. SOFTWARE REINSTALLATION

Usually the Koa-PicoITX comes already installed. But you may need the instructions below if you would like to reinstall the software.

5.1 Ubuntu installation

From an Ubuntu installation cd, do the standard installation.

5.2 Libkorebot installation

- Unpack the library file ***libkorebot_ABC_picoitx_XYZ.tar.bz*** in ***~/software***
  cd ~/software
  tar -xjf libkorebot_ABC_picoitx_XYZ.tar.bz2
- Copy the library into the system:
  sudo cp libkorebot-1.14-kb1/build-x86/lib/libkorebot*.so /usr/lib/
- Copy the config files into the system:
  sudo cp -R libkorebot-1.14-kb1/config/ /etc/libkorebot

5.3 Compiler installation

- Execute just this command in a terminal:
  sudo apt-get install build-essential

- Install linux headers:
  sudo apt-get install linux-headers-$(uname -r)
  sudo apt-get install module-assistant
  sudo module-assistant prepare

5.4 Drivers installation

5.4.1 i2c driver

The i2c driver is usually already included in the Linux OS.

- for an autonomous start-up, add it in the file ***/etc/modules*** the ligne ***i2c_dev***

with command: ***sudo sh -c 'echo i2c_dev >>/etc/modules'***

– add the /dev/i2c device to the dialout group to be used by any user:

create the file ***i2c_chown*** in ***/etc/init.d/***

content of ***i2c_chown***:

```
#! /bin/sh
# add i2c dev into dialout group
chown :dialout /dev/i2c-0
```

- change the file to execution mode:
  sudo chmod +x /etc/init.d/i2c_chown

- create the link to the start-up level 2:
  sudo ln -s /etc/init.d/i2c_chown /etc/rc2.d/S99i2c_chown

- load the module: ***sudo modprobe i2c_dev***
- launch the script: sudo ***/etc/init.d/./i2c_chown***

5.4.2 gpio driver

- for an autonomous start-up, add it in the file ***/etc/modules***
  with command: ***sudo sh -c 'echo via_gpio >>/etc/modules'***

- create the file ***via_gpio*** giving node initialization and device group modification in ***/etc/init.d/*** :

  content of file ***via_gpio*** :

```
#! /bin/sh
#create node
mknod /dev/via_gpio c 241 0 -m660
# add via_gpio into dialout group
chown :dialout /dev/via_gpio
```

- change the file to execution mode:
  sudo chmod +x /etc/init.d/via_gpio

- launch the script: ***sudo /etc/init.d/./via_gpio***

- create the link to the start-up level 2:
  sudo ln -s /etc/init.d/via_gpio /etc/rc2.d/S99via_gpio

- unpack in the created *software* directory the driver file ***gpio-driver_XYZ.tar.bz2***:
  mkdir ~/software
  cd ~/software
  tar -xjf gpio-driver_XYZ.tar.bz2

  - in the directory ***gpiodriver/driver***, clean the old file, recompile it, unload the old module, install the new one and load it with the commands:

    make clean (pay attention to not use sudo for this command, else you will have to resintall the kernel headers files!)
    make
    sudo make install
    sudo make load

6. WARRANTY

K-TEAM warrants that this product is free from defects in materials and workmanship and in conformity with the respective specifications of the product for the minimal legal duration, respectively one year from the date of delivery, under normal use conditions.

Upon discovery of a defect in materials, workmanship or failure to meet the specifications in the Product during the afore mentioned period, Customer must request help on K-Team Internet forum on http://www.k-team.com/kforum/ by detailing:

- The type of the product used (package, version, & serial number).
- The extension modules.
- The programming environment of the robot (standard, version, OS).
- The standard use of Product before the appearance of the problem.
- The description of the problem.

If no answer is received within two working days, Customer can contact K-TEAM support by phone or by electronic mail with the full reference as stated below

If the defect is identified as a “warranty” related problem, K-TEAM shall then, at K-TEAM's sole discretion, either repair such Product or replace it with the equivalent product without charging any technical labour fee and repair parts cost to Customer, under the condition that Customer brings such Product to K-TEAM within the period mentioned before. Repair or replacement under warranty does not entitle to original warranty team extension.

This limited warranty is invalid if the factory-applied serial number has been altered or removed from the Product.

This limited warranty covers only the hardware and software components contained in the Product. It does not cover technical assistance for hardware or software usage and it does not cover any software products contained in the Product.

This limited warranty is non-transferable.

It is likely that the contents of Customer's flash memory will be lost or reformatted in the course of the service and K-TEAM will not be responsible for any damage to or loss of any programs, data or other information stored on any media or any part of the Product serviced hereunder or damage or loss arising from the Product not being available for use before, during or after the period of service provided or any indirect or consequential damages resulting therefore.

If during the repair of the product the contents of the flash memory are altered, deleted or in any way modified, K-Team is not responsible whatever. Customer’s product will be returned to customer configured as originally purchased (subject to availability of software).

Be sure to remove all third parties' hardware, software, features, parts, options, alterations, and attachments not warranted by K-TEAM prior to Product service. K-TEAM is not responsible for any loss or damage to these items.

This warranty is limited as set out herein and does not cover, any consumable items (such as batteries) supplied with the Product; any accessory products which is not contained in the Product; cosmetic damages; damage or loss to any software programs, data, or removable storage media; or damage due to (1) acts of God, accident, misuse, abuse, negligence, commercial use or modifications of the Product; (2) improper operation or maintenance of the Product; (3) connection to improper voltage supply; or (4) attempted repair by any party other than a K-TEAM authorized robot service facility.

This limited warranty does not apply when the malfunction results from the use of the Product in conjunction with any accessories, products or ancillary or peripheral equipment, or where it is determined by K-Team that there is no fault with the Product itself.

K-Team expressly disclaims all other warranties than stated hereinbefore, expressed or implied, including without limitation implied warranties of merchantability and fitness for a particular purpose to the fullest extent permitted by law.

Limitation of Liability: *In not event shall either party be liable to the other for any indirect, special, incidental or consequential damages resulting from performance or failure to perform under the contract, or from the furnishing, performance or use of any goods or service sold or provided pursuant hereto, whether due to a breach of contract, breach of warranty, negligence, or otherwise. Save that nothing herein shall limit either party's liability for death or personal injury arising from its negligence, neither party shall have any liability to the other for indirect or punitive damages or for any claim by any third party except as expressly provided herein.*

K-Team S.A.
Rue Galilee 9
1400 Yverdon-les-Bains
Switzerland