IPA CANopen User Manual

Preliminary version

Tobias Sing

Fraunhofer Institute for
Manufacturing Engineering and Automation

Stuttgart, Germany

Last modified on Wednesday 5th December, 2012

Contents

Chapter 1

Introduction

This manual describes the IPA CANopen library, a C++ framework for communicating with CANopen motor devices. Most users will use the ROS (Robot Operating System) wrapper provided as part of the package. However, the library itself is completely independent from ROS and therefore can also be used in other contexts.

In Chapter 2, a step-by-step guide to installating the library is given. Command-line tools for moving and referencing modules are described in 3. The use of CANopen devices from ROS is explained in 4. Finally, in Chapter 5, we show how to use the library in own C++ projects, and how to adapt it to a user's needs.

Chapter 2

Installation

2.1 CAN device driver

Currently, the library has only been tested with the PCAN-USB CAN interface for USB from Peak System.It has been tested with version 7.5. The Linux user manual is available at: `http://www.peak-system.com/fileadmin/media/linux/files/PCAN%20Driver%20for%20Linux_eng_7.1.pdf`. Briefly, to install the drivers under Linux, proceed as follows:

- Download and unpack the driver: `http://www.peak-system.com/fileadmin/media/linux/files/peak-linux-driver-7.5.tar.gz`.
- `cd peak-linux-driver-x.y`
- `make clean`
- Use the chardev driver: `make NET=NO`
  Note: The option `NET=no` is crucial!
- `sudo make install`
- `/sbin/modprobe pcan`
- Test that the driver is working:
  - `cat /proc/pcan` should look like this, especially `ndev` should be `NA`:

```
*------------ PEAK-System CAN interfaces (www.peak-system.com) -------------
*------------------------- Release_20120319_n (7.5.0) ----------------------
*---------------- [mod] [isa] [pci] [dng] [par] [usb] [pcc] -----------------
*--------------------- 1 interfaces @ major 248 found -----------------------
*n -type- ndev --base-- irq --btr- --read-- --write- --irqs-- -errors- status
32    usb -NA- ffffffff 255 0x001c 0000cc3f 0000edd1 00063ce1 00000005 0x0014
```

  – `./receivetest -f=/dev/pcan32` Turning the CAN device power on and off should trigger some CAN messages which should be shown on screen. **If you do not receive messages using this test, you still have issues with the device driver which need to be solved before proceeding further.**

2.2 ROS-indendent CANopen library

2.2.1 Prerequisites

You will need the following free tools, which are available for all operating systems:

- *CMake* (to manage the build process). It is pre-installed on many *nix operating systems. Otherwise, you need to install it first, e.g. in Ubuntu: `sudo apt-get install cmake`
- *git* (to download the sources from github). In Ubuntu, it can be installed with: `sudo apt-get install git`
- A C++ compiler with good support for the C++11 standard, e.g. *gcc* version 2.6 or higher (default in Ubuntu versions 11.04 or higher).

2.2.2 Installation

- Go to a directory in which you want to create the *canopen* source directory.
- `git clone git://github.com/ipa320/ipa_canopen.git`
- `cd ipa_canopen/ipa_canopen_core`
- Create a build directory and enter it: `mkdir build && cd build`

- Prepare the make files: `cmake ..`
- `make`
  - Optionally, you can make the installation available system-wide: `sudo make install`
- Test if the build was successful:
  - `cd tools`
  - `./homing`
  - This should give the output:

```
Arguments:
(1) device file
(2) CAN deviceID
Example: ./homing /dev/pcan32 12
```

2.3 IPA CANopen ROS package

Currently, this requires that you first perform the two installation steps (PCAN driver, ROS-independent CANopen library) above manually. Then:

- `git clone git://github.com/ipa-tys/ros_canopen.git`
- `rosmake ros_canopen`. Make sure dependencies are installed (e.g. package cob_srvs from stack cob_common).
- Test if the installation was successful:
  - In one terminal: `roscore`
  - In another terminal: `rosrun ros_canopen ros_canopenmasternode`. This should give the output:

```
File not found. Please provide a description file as command line argument
```

Chapter 3

Tools

Chapter 4

CANopen communication in ROS

Chapter 5

Extending the CANopen library