EtherLab®

Version 2.1.0

created by

Ingenieurgemeinschaft IgH

Essen

January 2013

Ingenieurgemeinschaft IgH
Heinz-Bäcker-Str. 34
D-45356 Essen
Tel.: +49-201-36014-0
Fax.: +49-201-36014-14
E-mail: igh@igh-essen.com

Contents

List of Figures

1 Introduction

A blockset was created for the real time connection of EtherCAT® components in MATLAB/Simulink®. EtherCAT slaves can be selected from the `etherlab_lib` in MATLAB/Simulink and directly included into the model. After that, the model is translated to C code and compiled to a realtime application, that can be executed in context of a Linux OS. To achieve hard real-time operation, a kernel with realtime preemption [1] is recommended.

2 License

EtherLab is licensed under the GPL, version 3 [4].

The name EtherLab is a registered trademark of Ingenieurgemeinschaft IgH, Heinz-Bäcker-Straße 34, 45356 Essen, Germany.

3 Concept

EtherLab® is a system to create controllers using EtherCAT hardware, that run with the EtherLab's EtherCAT master [2]. The user is able to create models using the MATLAB/Simulink library `etherlab_lib`, that are compiled to a realtime application by traversing Simulink Coder (former Realtime Workshop) and EtherLab.

Figure 1 shows the development and runtime environment and the additional components. The EtherLab technology can be subdivided into the components listed below:

- The `etherlab_lib` for MATLAB/Simulink. It contains all necessary blocks for the creation of Simulink models with EtherCAT slaves.
- The support code, that compiles an executable of the output of Simulink Coder. This happens in background and is not visible to the user.
- The PdServ library [5], that adds non-realtime process data server functionality to the realtime application.

4 Installation

The installation procedure is separated for the development and runtime environments, which do not necessarily have to be on the same system.

In the following sections it is assumed, that EtherLab components are installed in `/vol/opt/etherlab`. If this directory does not exist, it can be created with the commands below, which also allow members of the group `users` to write to that directory:

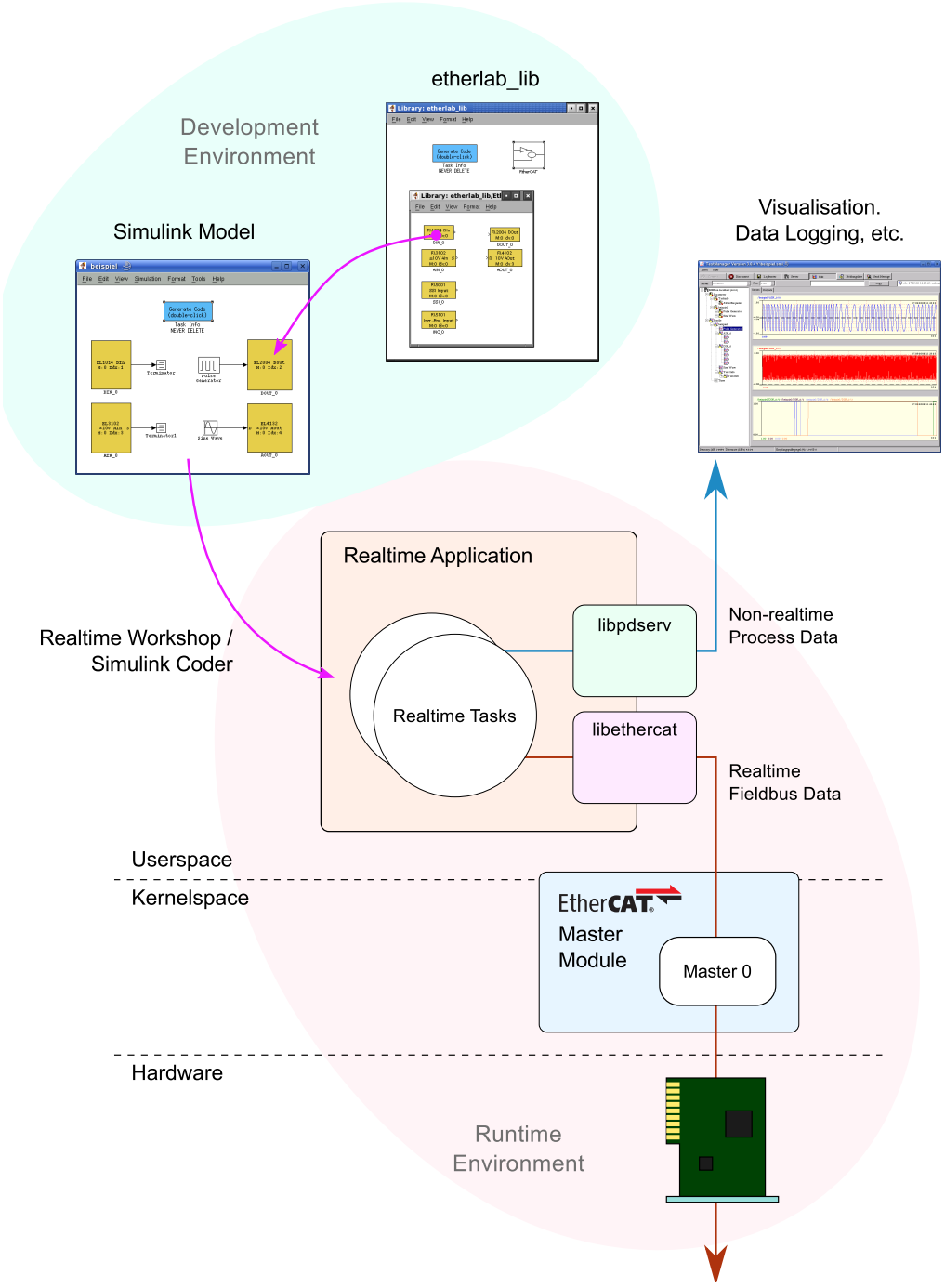

Figure 1: EtherLab Architecture

```
# mkdir -p /vol/opt/etherlab
# chown root:users /vol/opt/etherlab
# chmod 775 /vol/opt/etherlab
```

4.1 Development Environment

4.1.1 Prerequisites

Prerequisites for the installation of EtherLab on the development system:

- Matlab 2009b (or newer)
- Simulink
- Simulink Coder (former Realtime Workshop)
- GNU C Compiler Collection (GCC) [3]
- cmake

4.1.2 Installation

Installation is done after copying the EtherLab tarball from the EtherLab CD (or downloading from etherlab.org, like in the below example) as normal user according to the commands below. The last command installs EtherLab into the directory `/vol/opt/etherlab/rtw`.

```
$ mkdir /vol/opt/etherlab/src
$ cd /vol/opt/etherlab/src
$ wget http://etherlab.org/download/etherlab/etherlab-2.1.0.tar.bz2
$ tar xjf etherlab-2.1.0.tar.bz2
$ mkdir etherlab-2.1.0-build
$ cd etherlab-2.1.0-build/
$ cmake -DPREFIX=/vol/opt/etherlab ../etherlab-2.1.0
$ make
$ make install
```

4.1.3 Library Setup

As a prerequisite for the `setup_etherlab` command (see below) to succeed, the file `toolbox/local/pathdef.m` inside the MATLAB installation directory must be writable for the user running MATLAB. Enter the below commands (as `root`) to make it writable for all users in the group `users` (substitute the variable `MATLABDIR` with the correct path first):

```
# chown :users $MATLABDIR/toolbox/local/pathdef.m
# chmod 664 $MATLABDIR/toolbox/local/pathdef.m
```

To setup the EtherLab library `etherlab_lib` for the use in MATLAB, the below commands must be entered (out of MATLAB).

```
>> cd /vol/opt/etherlab/rtw
>> setup_etherlab
```

Now, the EtherLab library can be used acc. to section 5.

4.2 Runtime Environment

4.2.1 Prerequisites

The EtherLab software itself does not have to be present on the runtime system. In order to compile and run EtherLab-generated realtime applications, the below software has to be available:

- Linux-Kernel $\geq$ 2.6 (RT-Preemption [1] recommended)
- GNU C Compiler Collection (GCC) [3]
- PdServ Library [5] with dependencies:
  - Log4plus Library [6]
  - CommonCpp2 Library [7]
  - YAML Library [8]
- IgH EtherCAT Master 1.5 [2] (optional)

4.2.2 Installation

The realtime application executables created by EtherLab are linked against the PdServ library and optionally the EtherCAT master library[1]. The PdServ library has a few dependencies. There are RPM packages for openSUSE available in the download directory of the EtherLab homepage both for the library itself. The log4cplus, Common C++ and YAML libraries are quite common and usually available as packages for Linux distributions.

To build and compile PdServ, the below commands can be used:

```
$ cd /vol/opt/etherlab/src
$ wget http://etherlab.org/download/pdserv/pdserv-1.1.0.tar.bz2
$ tar xjf pdserv-1.1.0.tar.bz2
$ mkdir pdserv-1.1.0-build
$ cd pdserv-1.1.0-build/
$ cmake -DCMAKE_INSTALL_PREFIX=/vol/opt/etherlab ../pdserv-1.1.0
$ make
$ make install
```

[1] For installation instructions, please refer to the EtherCAT master documentation.

5 The EtherLab Library

The EtherLab library `etherlab_lib` was designed to allow the creation of EtherLab models. It contains blocks for all supported EtherCAT slaves, plus a generic slave block for slaves that are not (yet) in the library. To show the library window (see Figure 2), the commands below have to be entered in MATLAB:

```
1 >> etherlab_lib
```

Figure 2 shows the window containing the `etherlab_lib`, while Figure 3 shows the EtherCAT-specific blockset. Each block has a configuration dialog, that shows up after double-clicking on the block area. There is a "Help" button in all of the configuration dialogs, that provides detailed help concerning the dialog elements.

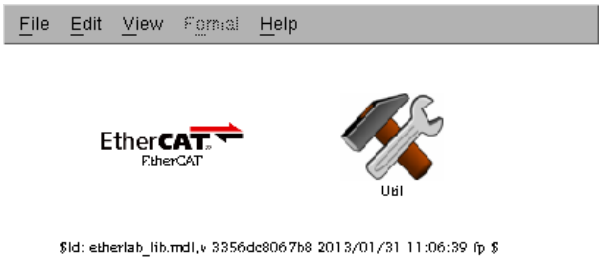

Figure 2: EtherLab Library

On the left side of the EtherCAT subsystem in Figure 3 there are subsystems with EtherCAT slave blocks of different vendors and, on top, the generic slave block that includes an arbitrary EtherCAT slave.

The special blocks on the right side do not correspond to certain slaves:

Master State This block allows to query information from a certain EtherCAT master (like number of slaves, etc.).

Domain State This block allows to query information from a certain process data domain (see EtherCAT master documentation). This is important to get to know, if process data are exchanged correctly.

6 Creation of EtherLab Models

To create a new model using EtherLab blocks, a few model parameters have to be set up (menu *Simulation/Configuration Parameters...*):

- The input field *Realtime Workshop* → *System target file* must contain the file name `etherlab.tlc`. It can be chosen via the *Browse...* button (see Figure 4).

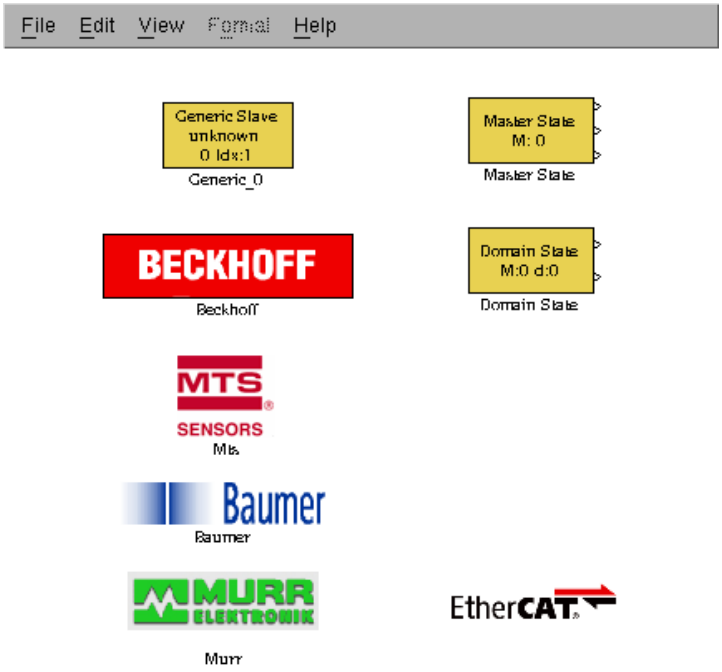

Figure 3: EtherCAT Blockset

- The input field *Solver* → *Fixed-step size* must contain the period time of the realtime cycle, for example *0.01* meaning 100 Hz (see Figure 5).

At this time, the model can be edited as usual. When finished, the shortcut *Ctrl+B* will generate the C code and build the realtime application executable.

If the executable shall be compiled on another host, the option "Generate code only" has to be checked and the following command on the build host can be used to create the executable:

```
1 $ make -C <myapp>_etl_hrt -f <myapp>.mk
```

7 Starting of EtherLab Applications

The simplest way to start the realtime application is the following command, where `myapp` corresponds to the Simulink model name.

```
1 # ./<myapp>
```

At best the application outputs nothing and stays in foreground and can be quitted via pressing *Ctrl-C*. Error messages are printed to the console.

The EtherLab application template creates a few command-line parameters:

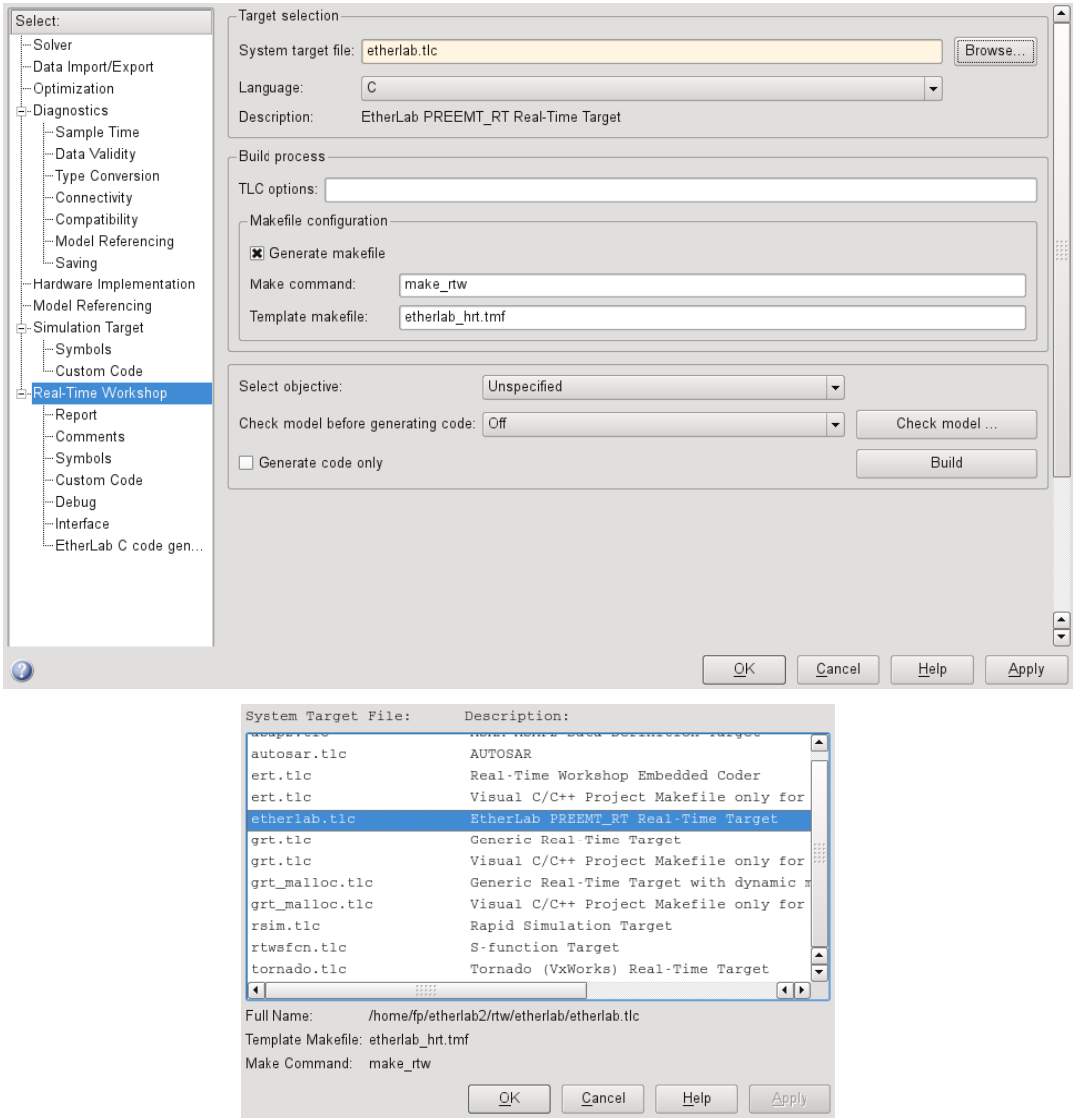

Figure 4: Realtime Workshop Configuration

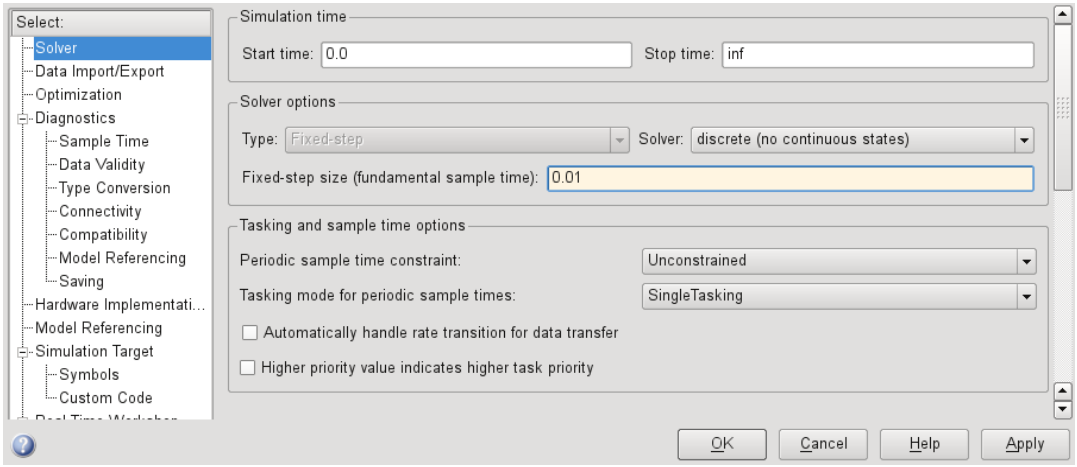

Figure 5: Solver Configuration

```
$ ./myapp -h
Usage: myapp [OPTIONS]
Options:
  --priority        -p <PRIO>  Set task priority. Default: RT.
  --pdserv-config   -c <PATH>  PdServ configuration file.
  --help            -h         Show this help.
```

With the `--pdserv-config` option, that path to a PdServ configuration file can be passed, that allows to configure the MSR protocol, for example.

References

[1] RT PREEMPT HOWTO. https://rt.wiki.kernel.org/index.php/RT_PREEMPT_HOWTO, 2013.

[2] IgH EtherCAT Master for Linux. http://etherlab.org/en/ethercat, 2013.

[3] GCC, the GNU Compiler Collection. http://gcc.gnu.org. 2013.

[4] GNU General Public License, Version 3. http://www.gnu.org/licenses/gpl-3.0.html. 2013.

[5] PdServ Library. http://etherlab.org/en/pdserv/index.php, 2013.

[6] Logging Framework for C++. http://sourceforge.net/p/log4cplus/wiki/Home/, 2013.

[7] GNU Common C++. http://www.gnu.org/software/commoncpp, 2013.

[8] YAML Ain't Markup Language. http://yaml.org, 2013.