

How-to build ubuntuRTAI 7.10 (Juri Lelli <juri.elli at gmail dot com>)

=====
Partially from

* RTAI & Comedi & Matlab on Ubuntu Gutsy - 2008-02-24 - V3

By Arno Stienen arnoreg at gmail dot com (<https://www.rtai.org/RTAILAB/RTAI-UbuntuGutsy-Matlab.txt>)

* RTAI-Lab tutorial: Scilab, Comedi, and real-time control

Roberto Bucher, Thomas Netter, Simone Mannori. February 28, 2008
(<https://www.rtai.org/RTAILAB/RTAI-Lab-tutorial.pdf>)

* <https://help.ubuntu.com/community/Kernel/Compile>

Following this guide you will be able to create and understand how to personalize ubuntuRTAI 7.10, a live distribution, Ubuntu 7.10 derivative, that makes available a real-time environment (RTAI Linux 3.6) and some software to develop real-time application (NetBeans + Distributed Control Lab).

All the software will be first installed (after have made packages with checkinstall or following the "Debian/Ubuntu" way) on the host system (Ubuntu 7.10 on hd) and than on the guest one within a chroot environment.

As a general rule will build packets with the following values:

0. Maintainer Your name (youremail@address.com);
1. Summary Short description like the one found on the application site;
2. Name Packet's name (see later);
3. Version 0 or the version numer ('7.0.3') or download date for cvs ('cvs20082508').

It is suggested to copy packets on a specific directory for easy finding them later.

=====
1.Preparations
=====

Hard disk install Ubuntu 7.10 to create needed packages for chroot environment install.

Possibly comment out the CD-ROM line in the list with repositories by prefixing it with a '#':

```
sudo nano /etc/apt/sources.list
```

Take ownership of the source directories (replace [yourusername] with your own user name):

```
sudo chown -R [yourusername]:[yourusername] /usr/local/src /usr/src
```

Install the required packages (from a default Ubuntu 7.10 system):

```
sudo apt-get install cvs subversion build-essential automake checkinstall  
x11proto-xext-dev xlibs-static-dev libxext-dev libxt-dev gettext libncurses5-dev  
fakeroot kernel-package swig python-dev libtool libboost-program-options-dev  
libgl0-dev libxmu-dev libxi-dev flex bison
```

These packages will also install automatically:

General:

```
cvsversion --> libapr1 libaprutil1 libpq5 libsvn1
build-essential --> dpkg-dev g++ g++-4.1 libc6-dev libstdc++6-4.1-dev
linux-libc-dev patch
automake --> autoconf autotools-dev m4
checkinstall
```

MesaLib:

```
x11proto-xext-dev --> libxau-dev x11proto-core-dev x11proto-input-dev
xlibs-static-dev --> libfontenc-dev libfreetype6-dev libx11-dev libxdmcp-
dev libxfont-dev x11proto-fonts-dev x11proto-kb-dev xtrans-dev zlib1g-dev
libxext-dev
libxt-dev --> libice-dev libsm-dev
```

EFLTK:

```
gettext
```

Linux Kernel:

```
libncurses5-dev
fakeroot
kernel-package --> intltool-debian po-debconf
```

Comedilib

```
swig
python-dev --> python2.5-dev
libtool
```

Comedi Calibrate:

```
libboost-program-options-dev --> libboost-dev libboost-program-options1.34.1
libgsl0-dev
```

RTAI

```
libxmu-dev --> libxmu-headers
libxi-dev
```

```
=====
2.MesaLib
=====
```

```
cd /usr/local/src
```

Download MesaLib-7.0.3.tar.bz2 from

<http://sourceforge.net/projects/mesa3d> to current directory.

```
tar jxf MesaLib-7.0.3.tar.bz2
cd Mesa-7.0.3
make linux-x86
sudo checkinstall # Use 'mesalib-standalone' as name;
```

```
=====
3.eFLTK
=====
```

(Based on <http://equinox-project.org/cgi-bin/trac.cgi/wiki/UbuntuInstallation>)

Download to folder with current date, compile and install:

```
cd /usr/local/src
svn co https://ede.svn.sourceforge.net/svnroot/ede/trunk/efltk
cd efltk
autoconf
```

Configure (--enable-xft doesn't work on Ubuntu 7.10):

```
./configure --disable-mysql --disable-unixODBC
./emake
```

The efltk.spec file must be modified as is attached (the file could be simply overwritten).

```
sudo checkinstall bash emake install # Use 'efltk' as name;
```

Add '/usr/local/lib' to /etc/ld.so.conf (remember to do this step also in chroot environment when we will create the live cd):

```
sudo nano /etc/ld.so.conf
sudo ldconfig
```

```
=====
4. Download RTAI
=====
```

Download RTAI:

```
cd /usr/local/src
wget --no-check-certificate https://www.rtai.org/RTAI/rtai-3.6.tar.bz2
tar xjf rtai-3.6.tar.bz2
ln -s rtai-3.6 rtai
```

```
=====
5. Linux Kernel 2.6 (Ubuntu 7.10 stock kernel)
=====
```

Download kernel, modules sources and dependencies:

```
cd /usr/src
sudo apt-get build-dep linux-image-2.6.22-14-generic linux-ubuntu-
modules-2.6.22-14-generic
apt-get source linux-image-2.6.22-14-generic linux-ubuntu-
modules-2.6.22-14-generic
```

Also download vanilla kernel sources; they will be used to create a configuration file:

```
wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.24.14.tar.gz
tar xzf linux-2.6.22.14.tar.gz
cd linux-2.6.22.14.tar.gz
```

Patch the vanilla kernel with RTAI's patch:

```
patch -p1 < /usr/local/src/rtai/base/arch/i386/patches/hal-linux-2.6.22-
i386-1.10-12.patch
```

Copy the Ubuntu kernel configuration (press enter at every 'make oldconfig' question):

```
cp /boot/config-2.6.22-14-generic .config
make oldconfig
make menuconfig
```

And config this way:

- Loadable module support ---> Module versioning support ---> disabled
- Processor type and features ---> Symmetric multi-processing support ---> disabled
- Processor type and features ---> Preemption Model
 - > Preemptible Kernel (Low-Latency Desktop)
- Processor type and features ---> Interrupt pipeline ---> enabled

- Power management options (ACPI, APM)
 - > Legacy Power Management API ---> disabled
- Power management options (ACPI, APM) ---> Software Suspend ---> disabled
- Power management options (ACPI, APM)
 - > ACPI (Advanced Configuration and Power Interface) Support
 - > ACPI Support ---> disabled
- Power management options (ACPI, APM)
 - > APM (Advanced Power Management) BIOS Support
 - > APM BIOS Support ---> disabled
- Power management options (ACPI, APM)
 - > CPU Frequency scaling ---> CPU Frequency scaling ---> disabled
- Power management options (ACPI, APM)
 - > Power Management support ---> disabled
- Kernel hacking
 - > Kernel debugging ---> disabled

Create a 'rtai' directory inside kernel sources:

```
cd /usr/src/linux-source-2.6.22-2.6.22/debian/binary-custom.d
mkdir rtai
cd rtai
```

Copy here the configuration file:

```
cp /usr/src/linux-2.6.22.14/.config config.i386
```

And the patch, with a different name:

```
mkdir patchset
cd patchset
cp /usr/local/src/rtai/base/arch/i386/patches/hal-linux-2.6.22-
i386-1.10-12.patch 0001-hal-linux-2.6.22-i386-1.10-12.patch
cd ..
```

Copy two necessary files from 'rt' directory and modify 'vars' with needed values:

```
cp ../rt/rules .
cp ../rt/vars .
nano vars
```

Compile the kernel:

```
cd /usr/src/linux-source-2.6.22-2.6.22
CONCURRENCY_LEVEL=2 AUTOBUILD=1 NOEXTRAS=1 fakeroot debian/rules custom-
binary-rtai
    #first value correspond to the CPUs number; on a dual-core this step
requires half
    #an hour.
```

Modify some files to compile modules:

```
cd /usr/src/linux-ubuntu-modules-2.6.22-2.6.22/debian
nano control
```

Append to the file something like this:

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)
Package: linux-ubuntu-modules-2.6.22-14-rtai
Architecture: i386
Section: base
Priority: optional
```

Provides: ndiswrapper-modules-1.9, apparmor-modules-2.1
Depends: linux-image-2.6.22-14-rtai
Pre-Depends: dpkg (>= 1.10.24)
Description: Ubuntu supplied Linux modules for version 2.6.22 on x86
This package contains modules supplied by Ubuntu for Linux kernel 2.6.22 on x86.

.
You likely do not want to install this package directly. Instead, install the linux-rtai meta-package, which will ensure that upgrades work correctly, and that supporting packages are also installed.

==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)

Edit 'i386.mk', erase all and insert this:

```
nano rules.d/i386.mk
```

==== START COPY FROM LINE BELOW ==== (Do not include this line!)

```
build_arch      = i386
```

```
flavours        = rtai
```

==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)

Compile:

```
CONCURRENCY_LEVEL=2 AUTOBUILD=1 fakeroot debian/rules binary-debs
```

Install packets:

```
cd ..
```

```
sudo dpkg -i linux-image-2.6.22-14-rtai_2.6.22-14.52_i386.deb linux-headers-2.6.22-14-rtai_2.6.22-14.52_i386.deb linux-ubuntu-modules-2.6.22-14-rtai_2.6.22-14.37_i386.deb
```

Reboot with the new kernel.

```
=====  
6. Comedilib  
=====
```

Download Comedi source (press enter when asked for cvs password, and ignore any warnings):

```
cd /usr/local/src
```

```
cvs -d :pserver:anonymous@cvs.comedi.org:/cvs/comedi login
```

```
cvs -d :pserver:anonymous@cvs.comedi.org:/cvs/comedi co comedi
```

```
cvs -d :pserver:anonymous@cvs.comedi.org:/cvs/comedi co comedilib
```

```
cvs -d :pserver:anonymous@cvs.comedi.org:/cvs/comedi co comedi_calibrate
```

Make and install comedilib.

```
cd comedilib
```

```
sh autogen.sh
```

```
./configure --sysconfdir=/etc
```

```
make
```

```
sudo checkinstall # use 'comedilib' as name, comedilib.spec attached
```

Make and install comedi_calibrate.

```
cd ../comedi_calibrate
```

```
aclocal
```

```
automake --add-missing
```

```
autoreconf
```

```
./configure
```

```
make
sudo checkinstall # use 'comedi-calibrate' as name
```

```
=====
7. RTAI
=====
```

Make and install RTAI (for now without Comedi-support or RTAI-Lab).

```
cd /usr/local/src
rm rtai
wget http://www.linuxcnc.org/hardy/dists/hardy/base/source/rtai_3.6-
linuxcnc.2.tar.gz
mv rtai-3.6 rtai-3.6_orig
tar xzvf rtai_3.6-linuxcnc.2.tar.gz
cd rtai-3.6/debian
```

Modify 'control' like the following:

```
nano control
```

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)
```

```
Source: rtai
Section: misc
Priority: extra
Maintainer: Juri Lelli (jlelli at tiscali dot it)
Build-Depends: linux-headers-2.6.22-14-rtai
```

```
Package: rtai-modules-2.6.22-14-rtai
Architecture: any
Description: Real Time modules for linux-image-2.6.22-14-rtai
Depends: linux-image-2.6.22-14-rtai
==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)
```

Change 'KERNEL_VER' field as '2.6.22-14-rtai' in 'rules' file:

```
nano rules
```

Compile:

```
cd ..
fakeroot debian/rules binary
```

Install:

```
cd ..
dpkg -i rtai-modules-2.6.22-14-rtai_3.6-linuxcnc.2_i386.deb #don't save
this packet, it will be
#rebuilted
```

Add ':/usr/realtime-2.6.22-14-rtai/bin' to the path variable in
'/etc/environment':

```
sudo nano /etc/environment # the same in chroot environment
```

```
=====
8. Comedi
=====
```

```
cd /usr/local/src/comedi
sh autogen.sh
./configure --enable-kbuild --disable-pcmcia --with-
```

```
rtaidir=/usr/realtime-2.6.22-14-rtai/  
make
```

Create install script:

```
nano install-rtai
```

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)  
#!/bin/bash  
make install  
cp include/linux/comedi.h include/linux/comedilib.h /usr/local/include  
mkdir -p /usr/local/include/linux  
ln -sf /usr/local/include/comedi.h /usr/local/include/linux/comedi.h  
ln -sf /usr/local/include/comedilib.h /usr/local/include/linux/comedilib.h  
==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)
```

```
sudo checkinstall --install=no --fstrans=no bash install-rtai # use  
'comedi' as name
```

```
sudo dpkg --force-overwrite -i comedi*.deb  
sudo make dev
```

```
=====  
9. RTAI-Lab  
=====
```

Modify the 'CONFIGURE' field on 'rules' config file:

```
cd /usr/local/src/rtai-3.6/debian  
nano rules  
CONFIGURE := --enable-shm --enable-fpu --enable-compatible --disable-leds  
--enable-rtailab --disable-netrpc --enable-cpus=1 CONFIG_RTAI_RTDM=y --with-  
comedi=/usr/local --enable-comedi-lxrt --with-eftk=/usr/local
```

Compile and install, first remove previous rtai packet.

```
cd ..  
rm ../rtai-modules-2.6.22-14-rtai_3.6-linuxcnc.2_i386.deb  
fakeroot debian/rules binary  
cd ..  
sudo apt-get remove --purge rtai-modules-2.6.22-14-rtai_3.6-  
linuxcnc.2_i386  
sudo dpkg -i rtai-modules-2.6.22-14-rtai_3.6-linuxcnc.2_i386.deb
```

```
=====  
10. Scilab/Scicos  
=====
```

```
cd /usr/local
```

Download the last available version of Scilab source code from www.scilab.org.
Check www.scicos.org
for the presence of a required patch.

```
sudo su  
tar xzvf scilab-4.x.x-src.tar.gz
```

Install this packets:

```
apt-get install g77 sablotron tcl8.4-dev tk8.4-dev xaw3dg xaw3dg-dev  
libpvm3 pvm-dev libgtk2.0-dev
```

Configure and compile:

```
cd scilab-4.x.x
./configure --without-java --with-tcl-library=/usr/lib --with-tcl-include=/usr/include/tcl8.4
make all # NOT make install!!!
ln -s /usr/local/scilab-4.1.2/bin/scilab /usr/local/bin/scilab
```

When we will work in chroot environment we could simply copy 'scilab-4.x.x' directory in '/usr/local' and execute the last command.

```
=====
11. Rtai-Lab add-on
=====
```

```
cd /usr/local
```

Download necessary files from web.dti.supsi.ch/bucher/scilab.html.

```
tar zxvf scilab-4.1.2-rtailab.tgz
cd scilab-4.1.2-rtailab/macros
make install
exit
```

and as normal user:

```
make user
```

```
=====
12. Post Installation
=====
```

If you installed the Comedi packages without going through the compile steps (chroot environment), you'll need to make the dev links. Create and run:

```
nano makedev
```

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)
```

```
#!/bin/bash
for i in `seq 0 15`; do \
    mknod -m 666 /dev/comedi$$i c 98 $$i \
    ; \
done;
```

```
==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)
```

```
chmod +x makedev
sudo ./makedev
```

Measure your system's latency to real-time interrupts:

```
cd /usr/realtime-2.6.22-14-rtai/testsuite/kern/latency/
./run
```

Calibrate RTAI, remember the results of the calibrations.

```
sudo insmod /usr/realtime/modules/rtai_hal.ko
sudo insmod /usr/realtime/modules/rtai_fifos.ko
sudo insmod /usr/realtime/modules/rtai_calibrate.ko
```

```
calibrate -c # => VALUE_FOR_CPU_FREQUENCY
calibrate -r # => SETUP_TIMER_VALUE
calibrate -k # => SETUP_LATENCY_VALUE
```

```
sudo rmmod rtai_hal
sudo rmmod rtai_fifos
sudo rmmod rtai_calibrate
```

Copy the following lines in a rlinsmod script, replacing the variables in FULL_CAPS with the above found values.

```
sudo nano /usr/realtime/bin/rlinsmod
```

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)
```

```
# RLNSMOD - RTAI-Lab Insmod.
```

```
# Inserts RTAI-Lab and Comedi modules in kernel and
```

```
# configures the drivers.
```

```
/sbin/insmod /usr/realtime/modules/rtai_smi.ko
```

```
rtai_cpufreq_arg=VALUE_FOR_CPU_FREQUENCY
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_hal.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_lxrt.ko
```

```
SetupTimeTIMER=SETUP_TIMER_VALUE Latency=SETUP_LATENCY_VALUE
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_fifos.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_sem.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_mbx.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_msg.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_netrpc.ko
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_shm.ko
```

```
/sbin/modprobe comedi
```

```
/sbin/modprobe kcomedilib
```

```
/sbin/insmod /usr/realtime-2.6.22-14-rtai/modules/rtai_comedi.ko
```

```
# Hardware dependent lines below. This example is for a NI PCI 6025 DAQ Card.
```

```
#/sbin/modprobe ni_pcimio
```

```
#/usr/local/sbin/comedi_config -q /dev/comedi0 ni_pcimio
```

```
#/usr/local/bin/comedi_calibrate -q -f /dev/comedi0
```

```
==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)
```

```
sudo chmod +x /usr/realtime/bin/rlinsmod
```

Copy the following lines in a rlrmod script:

```
sudo nano /usr/realtime/bin/rlrmod
```

```
==== START COPY FROM LINE BELOW ==== (Do not include this line!)
```

```
# RLRMOD - RTAI-Lab Rrmod.
```

```
# Removes RTAI-Lab and Comedi modules from kernel.
```

```
#/usr/local/sbin/comedi_config -qr /dev/comedi0
```

```
#/sbin/modprobe -r ni_pcimio
```

```
/sbin/rmmod rtai_comedi
```

```
/sbin/modprobe -r kcomedilib
```

```
/sbin/modprobe -r comedi
```

```
/sbin/rmmod rtai_shm
```

```
/sbin/rmmod rtai_netrpc
```

```
/sbin/rmmod rtai_msg
```

```
/sbin/rmmod rtai_mbx
```

```
/sbin/rmmod rtai_sem
```

```
/sbin/rmmod rtai_fifos
```

```
/sbin/rmmod rtai_lxrt
```

```
/sbin/rmmod rtai_hal
/sbin/rmmod smi_rt
==== STOP COPY AT LINE ABOVE ==== (Do not include this line!)
```

```
sudo chmod +x /usr/realtime/bin/rlrmmod
```

Start rlinsmod after every boot by adding the line '/usr/realtime/bin/rlinsmod' (without the quotes) to your rc.local file. Place this line above the 'exit 0' line.

```
sudo nano /etc/rc.local # also in chroot environment
```

```
=====
End of the First Part
=====
```

```
=====
Building the live cd
=====
```

From a Alex Joni's guide:
http://cvs.linuxcnc.org/cvs/infrastructure/livecd/hardy_i386.livecd?rev=1.3.

```
=====
1. Needed Tools
=====
```

```
sudo apt-get install squashfs-tools mkisofs
sudo modprobe squashfs
```

get CD from <http://releases.ubuntu.com/>

```
mkdir ~/live
mv ubuntu-7.10-desktop-i386.iso ~/live
cd ~/live
```

```
=====
2. Extract CD
=====
```

```
mkdir mnt
sudo mount -o loop ubuntu-7.10-desktop-i386.iso mnt
```

```
mkdir extract-cd
rsync --exclude=/casper/filesystem.squashfs -a mnt/ extract-cd
```

```
mkdir squashfs
sudo mount -t squashfs -o loop mnt/casper/filesystem.squashfs squashfs
```

```
mkdir edit
sudo cp -a squashfs/* edit/
```

```
=====
3. Prepare customization
=====
```

Copy previously created packets (contained in '~/packets') in 'edit/tmp'.

```
cp ~/packets/* edit/tmp
```

```
sudo cp /etc/resolv.conf edit/etc/
sudo cp /etc/hosts edit/etc/
sudo cp edit/etc/apt/sources.list edit/etc/apt/sources.list_orig
```

```
sudo cp /etc/apt/sources.list edit/etc/apt
```

```
sudo chroot edit
#indented stuff is inside the chroot
mount -t proc none /proc
mount -t sysfs none /sys
mount -t devpts none /dev/pts

export HOME=/root
export LC_ALL=C
```

```
=====
4. Do the changes
=====
```

```
apt-get update
apt-get install x11proto-xext-dev xlibs-static-dev libxext-dev
libxt-dev gettext libncurses5-dev fakeroot kernel-package swig python-dev
libtool libboost-program-options-dev libgsl0-dev libxmu-dev libxi-dev flex bison
g77 sablotron tcl8.4-dev tk8.4-dev xaw3dg xaw3dg-dev libpvm3 pvm-dev libgtk2.0-
dev sun-java6-jdk netbeans

cd /tmp
dpkg -i *.deb

#remove old kernel (use 'dpkg -l | grep linux' for a complete list)
apt-get remove --purge linux-image-*-generic linux-ubuntu-modules-*-
generic
#remove meta packages (use 'dpkg -l | grep linux' for a complete
list)
apt-get remove --purge linux-image-generic
#clean out leftovers (like nvidia-crap)
apt-get autoremove
```

Remember to execute required chroot steps (as indicated before), for step 12 don't execute the 'make user' command; instead the following:

```
mkdir -p /etc/skel/.Scilab/scilab-$(SCILAB_VERSION)
cat scilab >> /etc/skel/.Scilab/scilab-$(SCILAB_VERSION)/.scilab

#cleanup
apt-get clean
rm -rf /tmp/*
rm /etc/resolv.conf
rm /etc/hosts
mv /etc/apt/sources.list_orig /etc/apt/sources.list

#we'll need the initrd.gz from inside the chroot
mkinitramfs -o /initrd.gz 2.6.22-14-rtai

#exit the chroot
umount /proc
umount /sys
umount /dev/pts
exit

#kernel
sudo cp /boot/vmlinuz-2.6.22-14-rtai extract-cd/casper/vmlinuz
sudo mv edit/initrd.gz extract-cd/casper/
```

```
=====
5. Putting the CD back together
```

=====

Regenerate manifest

```
sudo chroot edit dpkg-query -W --showformat='${Package} ${Version}\n' >
extract-cd/casper/filesystem.manifest
```

Rebuild squashfs

```
sudo rm extract-cd/casper/filesystem.squashfs
sudo mksquashfs edit extract-cd/casper/filesystem.squashfs
```

Change CD name

```
sudo nano extract-cd/README.diskdefines
```

Build new md5sum.txt

```
sudo rm extract-cd/md5sum.txt
cd extract-cd
find . -type f | egrep -v '(boot.cat|md5sum.txt)' | xargs sudo md5sum |
sudo tee md5sum.txt
```

Build iso

```
cd extract-cd
sudo mkisofs -r -V "RTAIubuntu7.10" -cache-inodes -J -l -b
isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4
-boot-info-table -o ../ubuntu-7.10-desktop-RTAI-i386.iso .
```

====

Done

====

=====

Special Thanks To

=====

- * Alex Joni on #emc channel on irc.freenode.net, for kernel, packets stuff and live cd rebuilding.
- * Jesse Caulfield (<http://netthink.com/>): that wise me up to the squashfs/unionfs problem.
- * Barbalace Antonio: kernel and RTAI patch stuff.
- * Roberto Bucher (<http://web.dti.supsi.ch/~bucher/>): troubleshooting RTAI-Lab installation.
- * Arno Stienen: troubleshooting eFLTK installation.

Sorry if I forgot someone :-)

=====

eFLTK.spec

=====

```
%define name efltk
%define version 2.0.5
%define release 20080527
%define pakdir %{name}-%{version}-%{release}
%define date `(echo `LC_ALL="C" date +"%a %b %d %Y"`)`
```

Name: efltk

Summary: eFLTK is stable, small and fast cross-platform GUI Toolkit

Version: 2.0.5

```

Release: 20080527
Source: %{name}-%{version}-%{release}.src.tgz
Group: devel
URL: http://ede.sf.net
License: GNU GPL
BuildRoot: %{_tmppath}/%{name}-%{version}-%{release}-root
Requires:

%package -n %{name}-devel
Summary:
Version:
Release:
Group:
#Requires: %{name} = %{version}-%{release}

%description -n %{name}-devel
The efltk-devel package contains the header files and libraries needed
to develop programs that use the eFLTK libraries.

%prep
%setup -q -n %{pakdir}

%build
autoconf
CFLAGS="%optflags -DNDEBUG=1" CXXFLAGS="%optflags -DNDEBUG=1" ./configure
--prefix=%{_prefix} --enable-opengl --enable-utf8 --disable-xft --enable-plugins

# Setup for parallel builds
numprocs=`egrep -c ^cpu[0-9]+ /proc/stat || :`
if [ "$numprocs" = "0" ]; then
    numprocs=1
fi
#make -j$numprocs static
make -j$numprocs

%install
install -d $RPM_BUILD_ROOT/%{_prefix}
install -d $RPM_BUILD_ROOT/%{_prefix}/bin
install -d $RPM_BUILD_ROOT/%{_prefix}/include
install -d $RPM_BUILD_ROOT/%{_prefix}/lib
make install prefix=$RPM_BUILD_ROOT/%{_prefix}

%clean
rm -fr $RPM_BUILD_ROOT

%post -p /sbin/ldconfig

%postun -p /sbin/ldconfig

%files
%defattr(-, root, root)
%{_bindir}/*

%files -n %{name}-devel
%defattr(-, root, root)
%{_libdir}/*.so*
#%{_libdir}/*.theme
#%{_libdir}/*.a
#%{_libdir}/*.la
%{_includedir}/*
%{_prefix}/share/locale/*/LC_MESSAGES/efltk.mo
%doc doc/*

```

```

%changelog
* %{date} Just entered something here... :)
Now efltk.spec file is built with configure script.

=====
comedilib.spec
=====

Summary: Data Acquisition library for the Comedi DAQ driver.
Name: comedilib
Version: 0.9.0
Release: 1
License: LGPL
Group: System Environment/Kernel
URL: http://www.comedi.org/
Source: http://www.comedi.org/comedi/download/comedilib-0.9.0.tar.gz
BuildRoot: /var/tmp/%{name}-buildroot
Prereq: /sbin/ldconfig
BuildRequires: python
provides: comedilib

%description
Comedilib is the library for the Comedi data acquisition driver
for Linux. It allows Linux processes to acquire data from
supported DAQ cards, such as those from National Instruments.

%package devel
Summary:
Group:

%description devel
Comedilib is a library for using Comedi, a driver interface for data
acquisition hardware.

%prep
%setup -q

%build
#called when the rpm is built
CFLAGS="${CFLAGS:-%optflags}" ; export CFLAGS ; \
./configure \
  --prefix=%{_prefix} \
  --mandir=%{_mandir} \
  --datadir=%{_datadir} \
  --sysconfdir=%{_sysconfdir} \
  --disable-dependency-tracking

make

%install
[ -n "$RPM_BUILD_ROOT" -a "$RPM_BUILD_ROOT" != / ] && rm -rf $RPM_BUILD_ROOT

%makeinstall

# Clean out files that should not be part of the rpm.
# This is the recommended way of dealing with it for RH8
#rm -f $RPM_BUILD_ROOT%{_libdir}/*.a
rm -f $RPM_BUILD_ROOT%{_libdir}/*.la

# Move files

```

```
mv $RPM_BUILD_ROOT%{_datadir}/comedilib $RPM_BUILD_ROOT%{_datadir}/comedilib-
devel

%post
/sbin/ldconfig

%postun
/sbin/ldconfig

%clean
[ -n "$RPM_BUILD_ROOT" -a "$RPM_BUILD_ROOT" != / ] && rm -rf $RPM_BUILD_ROOT

%files
%defattr(-,root,root,-)
%doc AUTHORS COPYING README TODO ChangeLog NEWS
%{_libdir}/libcomedi.so
%{_libdir}/libcomedi.so.*
%{_libdir}/python2.2/site-packages/*.so
%{_libdir}/python2.2/site-packages/comedi.py
%{_sbindir}/comedi_*
%{_bindir}/comedi_*
%{_mandir}/man7/*
%{_mandir}/man8/*

%files devel
%defattr(-,root,root,-)
%{_libdir}/libcomedi.a
%{_includedir}/comedi*.h
%{_mandir}/man3/*
%{_datadir}/comedilib-devel/html/*

%changelog
* Tue Jun 03 2002 David Schleef <ds@schleef.org>
- update for new build system

* Wed Feb 21 2002 Tim Ousley <tim.ousley@ni.com>
- initial build of comedilib RPM
```