How-to Install RTAI in Ubuntu 10.04

Jorge Azevedo
jorge.azevedo@gmail.com

v1.0 – December 17, 2012

Abstract

This *how-to* describes one way to install RTAI 3.8.1 on Ubuntu (or *Any*buntu) 10.04 (Lucid Lynx) with Linux kernel 2.6.32.

Disclaimer

The author makes no representations or warranties with respect to the contents or use of this document. Use it at your own risk.

1 Introduction

This guide is intended to help installing RTAI in an Ubuntu based Linux system. As it specifically provides commands to this distribution and version, the process should be quite similar on other Debian based distros. This Ubuntu version comes with 2.6.32.x kernel version, so a 2.6.32 kernel should be used[1].

This guide is an updated version of "How-to Install RTAI in Ubuntu Hardy" by Cristóvão Sousa, who was kind enough to release his original work under a Creative Commons license and send me his source files.

2 Preparation

The first step in installing RTAI is installing the necessary packages from the repositories. These will prepare your system for building a custom kernel. The second step is downloading the necessary source code.

- All the necessary packages for compiling, configuring and generating a package can be installed via the following commands

```
sudo apt-get build-dep linux-image-2.6.32-21-generic
sudo apt-get install libncurses5-dev kernel-package
```

[1] Please note that the latest 64 bit kernel supported by RTAI is version 2.6.23.

- Linux source code is usually located in `/usr/src`, so let's start by going there

```
cd /usr/src
```

- Download and extract Linux kernel 2.6.32.11

```
wget -O - www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.32.11.tar.bz2 | sudo tar -jxf -
```

- Download and extract RTAI 3.8.1

```
wget -O - https://www.rtai.org/RTAI/rtai-3.8.1.tar.bz2 | sudo tar -jxf -
```

- Rename linux source directory to a more descriptive name (optional):

```
sudo mv linux-2.6.32.11 linux-2.6.32.11-rtai-3.8.1
```

- Create symbolic links to the two new folders (you can choose other names if you want):

```
sudo ln -snf linux-2.6.32.11-rtai-3.8.1 linux
sudo ln -snf rtai-3.8.1 rtai
```

3 Kernel Patching and Configuration

- Patch the kernel source with the correspondent RTAI patch:

```
cd /usr/src/linux
sudo patch -p1 -b < /usr/src/rtai/base/arch/x86/patches/hal-linux-2.6.32.11-x86-2.6-03.patch
```

- In order to make the new kernel configuration the most similar to the already installed one you can do the two next steps:

```
sudo cp /boot/config-`uname -r` ./.config
sudo make oldconfig   # then press Enter on all prompts
```

  Press Enter on all the prompts of the last command.

- Now it's time to configure the new kernel:

```
sudo make menuconfig
```

  In the menu there are many parameters you can change if you know what you are doing. The changes needed by RTAI are:

– [ ] Enable loadable module support
  [ ] Module versioning support
– Processor type and features
  [*] Interrupt pipeline
  Processor family = choose yours
  Symmetric multi-processing support = no/yes

If your machine has a single core processor uncheck this last option. If your machine has multicore you can choose to use all or only one[2].

Because power management is a "latency killer" you should turn it off:

– Power management and ACPI options
  [ ] Power Management support
  [ ] CPU Frequency scaling > CPU Frequency scaling

After all changes exit and say yes to save the configuration.

- Optionally you can make a backup for the new configuration file:

```
cd /usr/src/linux ; sudo cp .config /boot/config-2.6.24-rtai-3.6.1
```

4 Kernel Compilation

- First prepare for a clean compilation

```
sudo make-kpkg clean
```

- At this point the kernel source is ready for compilation. The next command will compile the kernel and generate debian packages. You should adjust concurrency level (number of paralell jobs) to your machine. Some suggest that it should be the number of cores plus one for maximum cpu usage. You should also adjust the –append-to-version parameter to fit your needs. In this case, the generated kernel version will be `2.6.32.11-rtai-3.8.1`.

```
sudo CONCURRENCY_LEVEL=5 CLEAN_SOURCE=no fakeroot make-kpkg --initrd \
--append-to-version -rtai-3.8.1 kernel_image kernel_headers
```

This can take from 30 minutes to 3 hours depending on the machine you are using. It can also consume up to 4 GB of disk space. You will see many warning messages, don't worry. If this command ends with an error you should try to fix it and compile again (the internet is your friend).

- Two deb packages are generated in `/usr/src/`, kernel image and source headers. If you wish you can free the disk space used in the compilation with `sudo make-kpkg clean`.

[2] In case you want to simulate a single core machine.

5 Kernel Installation

- Go to the packages directory:

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

- Install the kernel image and source headers packages:

```
sudo dpkg -i linux-headers-2.6.32.11-rtai-3.8.1_2.6.32.11-rtai-3.8.1-10.00.Custom_i386.deb
sudo dpkg -i linux-image-2.6.32.11-rtai-3.8.1_2.6.32.11-rtai-3.8.1-10.00.Custom_i386.deb
```

- Although the package installation creates a new grub entry, it doesn't create a new initramfs image and therefore the kernel is unbootable. Notice that all kernels installed in your system can be listed with

```
ls /lib/modules
```

  To generate a new initramfs and update grub run

```
sudo update-initramfs -c -k 2.6.32.11-rtai-3.8.1 && sudo update-grub
```

- Due to a bug in Grub 2, the system is still unbootable. While removing Grub 2 and installing Grub 1 will solve the issue, a more practical solution is to edit directly the grub.cfg entry for the newly compiled kernel and add 16 to end of the word linux and initrd. So, with your favorite editor you can edit grub.cfg

```
sudo vim /etc/grub/grub.cfg
```

  And change the entry so that instead of

```
linux   /boot/vmlinuz-2.6.32.11-rtai-3.8.1 (...)
initrd  /boot/initrd.img-2.6.32.11-rtai-3.8.1
```

  it reads

```
linux16   /boot/vmlinuz-2.6.32.11-rtai-3.8.1 (...)
initrd16  /boot/initrd.img-2.6.32.11-rtai-3.8.1
```

  You should now force save your changes since grub.cfg is a read only file. Please bear in mind that any call to `update-grub` will generate a fresh grub.cfg. The workaround will have to be applied manually every time.

- Now, you must reboot. Choose the new RTAI patched kernel in Grub. If the process fails, the most likely problem is a faulty kernel configuration. You might be interested in consulting section 9.

6 RTAI Configuration and Installation

- Creating a build tree separated from the source tree of RTAI is advised by RTAI people, so do

```
cd /usr/src/rtai
sudo mkdir build
cd build
```

- Configure and compile RTAI:

```
sudo make -f ../makefile menuconfig
```

  - General > Linux source tree = /usr/src/linux-2.6.32.11-rtai-3.8.1
  - Machine (x86) > Number of CPUs (SMP-only) = the right number (if applicable)

  At the end exit and say yes to save the configuration. RTAI will be compiled.

- If there were no errors, install RTAI:

```
sudo make install
```

- Configure RTAI dynamic libraries to be wide available:

```
sudo -s
echo /usr/realtime/lib/ > /etc/ld.so.conf.d/rtai.conf
exit
sudo ldconfig
```

- The RTAI binaries directory can be added automatically to the $PATH variable. To do that, add the line

  PATH="$PATH:/usr/realtime/bin"

  to the end of ~/.profile (user) and/or /root/.profile (root). It also comes in handy to define an alias for sudo so that it inherits the $PATH variable. To do this, add

  alias sudo="sudo env PATH=$PATH"

  to ~/.bashrc.

7 RTAI Test

If everything goes smoothly in the above steps RTAI is readily installed. To be sure execute the latency test:

```
cd /usr/realtime/testsuite/kern/latency/
sudo ./run
```

Press `Ctrl-C` to stop. A typical output from a successfull instalation looks like

```
RTAI Testsuite - KERNEL latency (all data in nanoseconds)
RTH|    lat min|    ovl min|    lat avg|    lat max|    ovl max|   overruns
RTD|      -1566|      -1566|      -1525|       -995|       -995|          0
RTD|      -1571|      -1571|      -1489|       8257|       8257|          0
RTD|      -1569|      -1571|      -1527|        -89|       8257|          0
RTD|      -1566|      -1571|      -1525|       -481|       8257|          0
RTD|      -1566|      -1571|      -1529|      -1049|       8257|          0
RTD|      -1566|      -1571|      -1529|       -939|       8257|          0
```

8 Conclusion

That's all. Your machine is ready for real time. If some problems arise during the process you can try to solve them by searching the internet for the error sentence.

9 Further reading

Here is a list of helpful documents for further help on RTAI.

- **How-to Install RTAI in Ubuntu Hardy** (on which this guide was based) and **RTAI Tutorial** by Cristóvão Sousa contain valuable information about both installation and programming using the RTAI API.
- **RTAI Installation Complete Guide** and **Building Your Way Through RTAI** by João Monteiro approach the same matters as the previous set of documents, but with a more in-depth view of the RTAI API.
- A **Guide to Installing RTAI Linux** by Keith Shortridge is a more extensive look at the installation procedure than the more practical approach developed in this guide.
- **RTAI 3.8 on Ubuntu(9.10)-Linux-kernel : 2.6.31.8** by Manuel Arturo Deza is another installation guide.