

Running applications with Xenomai 3.x

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Running a Xenomai 3 application	1
2	Valgrind support	1
3	Available real-time APIs	1

1 Running a Xenomai 3 application

For *Cobalt*, you will need the real-time core built into the target Linux kernel as described in [this document](#).

For *Mercury*, you need no Xenomai-specific kernel support so far, beyond what your host Linux kernel already provides. Your kernel should at least provide high resolution timer support (`CONFIG_HIGH_RES_TIMERS`), and likely complete preemption (`PREEMPT_RT`) if your application requires short and bounded latencies.

Any Xenomai-based application recognizes a set of standard options that may be passed on the command line, described in [this document](#).

In addition, the **Alchemy**, **pSOS**™ and **VxWorks**™ APIs running over the Xenomai core can define the clock resolution to be used, given as a count of nano-seconds, i.e. $HZ=(1000000000 / ns)$, by the `--{alchemy/psos/vxworks}-clock-resolution=<ns>` option.

If your application combines multiple APIs, you may pass several clock-resolution switches to set them all.

The default value depends on the API being considered. For instance, the **VxWorks**™ and **pSOS**™ emulators default to millisecond clock rates. The Alchemy API is tickless by default, i.e. `--alchemy-clock-resolution=1`.



Caution

Specifying a resolution greater than 1 nanosecond requires the low resolution clock support to be available from the Xenomai libraries (see the `--enable-lores-clock` [configuration switch](#)).

2 Valgrind support

Running Xenomai applications over *Valgrind* is currently available to the *Mercury* core only.

When the Valgrind API is available to the application process, the configuration symbol `CONFIG_XENO_VALGRIND_API` is defined at build time, and may be tested for existence by the application code. See the tool documentation at [this address](#).

The Xenomai autoconf script will detect the Valgrind core header on the build system automatically, and define this symbol accordingly (i.e. `/usr/include/valgrind/valgrind.h`).

Note

You may need to install the Valgrind development package on your build system to provide for the core header files. For instance, such package is called *valgrind-devel* on Fedora.

3 Available real-time APIs

Alchemy	This is a re-implementation from scratch of Xenomai's 2.x <i>native</i> API, fully rebased on the new RTOS abstraction interface.
pSOS	pSOS ™ is a registered trademark of Wind River Systems, Inc.
VxWorks	VxWorks ™ is a registered trademark of Wind River Systems, Inc.