Quick guide on using Partikle/Xtratum

for developing the Real-Time Systems project

Jorge Real, December 2010

1. Introduction

This document aims to be a helpful quick reference for those developing phase 2 of the Real-Time Systems project using the GEME 3000 computers (AKA "boxes") and the real hardware, instead of the simulated versions of the robot and oscillating display. It briefly describes what Partikle and Xtratum are and how to use them in the laboratory. This is a shorter version of the document "*Plataforma de ejecución GEME 3000 y Sistema operativo PaRTiKle/XtratuM*", also available in PoliformaT.

2. Partikle

Partikle is a minimal operating system for embedded, real-time systems. In order to build an Ada application for Partikle, the main subprogram has to be compiled with the script `pgnatmake` , an adaptation of *gnatmake* for Partikle.

Partikle has a high degree of flexibility in terms of execution environments. In particular, Partikle applications can be built to run in one of the following platforms or *targets*:

- **Bare machine**. When Partikle is configured with "bare machine" as target, the application contains the OS and runs on a bare machine (no OS is required). Compiling for a bare machine requires a bootable device (eg. a CD or a USB pen drive) containing the application and a boot manager (eg. *grub*). There are currently two architectures supported, namely x86 and ARM. The boxes have x86 architecture. The embedded kernel in this configuration has a relatively small footprint (some 100 KB). We won't be using this configuration for the projects.
- **Linux**. This configuration is useful to generate a regular Linux application. Real-Time properties are not guaranteed for our application under this configuration, but it may however be useful for testing purposes, allowing full use of Ada.Text_IO. Note that, although your application will suffer interference from the OS, this interference is reduced to the minimum since the Linux configuration used in the boxes includes only a subset of the OS (eg. X11 is excluded).
- **Xtratum domain**. Under this configuration, `pgnatmake` generates the application in a format that can be used as a loadable Xtratum module (see below). Doing so, the application runs concurrently with Linux, but with a higher priority than the OS. This is the preferred method for most of the projects, but take into account that you cannot use Ada.Text_IO under this configuration. Instead, you can use the gnat package Gnat.IO. The text output does not appear immediately in the screen: it has to be extracted by issuing `dmesg`.

If your project requires full IO through the screen and keyboard (eg., the learning robot, or the test programs for obtaining the robot's limits), your only choice is to use the Linux configuration. For the pendulum and Hanoi projects, you should use the Xtratum domain configuration.

Configuring Partikle

This step is required to select the desired Partikle configuration. Once you select one platform and build the compiler, `pgnatmake` will generate the corresponding kind of executable. A configuration remains active until a new configuration is set.

To set a particular `pgnatmake` configuration, you must follow these simple steps in the boxes (make sure you have root privileges):

```
# cd /usr/src/partikle
# make menuconfig
```

This will open a configuration menu from which you must select Architecture (bare machine x86, bare machine ARM, Linux, or Xtratum domain) and make sure that the compiler will support Ada (Language Support section).

Once configured, you should build the compiler for the selected architecture:

```
# make
>> Detected PaRTiKle path: /root/rtos/trunk/partikle/
>> Building PaRTiKle utils: done
>> Building PaRTiKle kernel [linux]: done
>> Building PaRTiKle user libraries: done
>> Add these to your profile environment.
    PRTK=/usr/src/partikle/ export PRTK
    PATH=$PATH:$PRTK/user/bin export PATH
```

Next time you invoke `pgnatmake`, it will generate the appropriate kind of executable from your sources.

Note: running `make distclean` from /usr/src/particle will erase the last compiled version of Partikle.

3. XtratuM

Xtratum is a virtualization mechanism that allows executing several operating systems concurrently with Linux on the same computer. Every OS (or application) executes in a separate space of execution, called **domain** in Xtratum.

Several domains can be loaded and executed concurrently. Domain zero runs with the lowest priority, and that's the one were Linux is executed. We'll load our applications in higher priority domains so that they don't suffer from OS interference. This is a crucial property when our applications have real-time constraints.

Xtratum is already installed in the boxes. In order to use it, you can issue the following commands:

- **`xmctl start`**
  Starts Xtratum by loading the Xtratum core module into memory.

- **`xmctl load`** ***`filename`***
  Loads and starts executing the domain contained in *filename*. The filename must correspond to a previously compiled application, with Partikle configured as Xtratum domain.

- **`xmctl stop`**
  Unloads Xtratum from memory, and all loaded domains, except Linux.

- **`xmctl unload`** ***`filename`***
  Unloads the previously loaded domain specified by *filename*.

- **`xmctl status`**
  Prints the list of domains currently loaded.