Building a Real-Time Operating System on top of the Adaptive Domain Environment for Operating Systems

Karim Yaghmour
Opersys inc.
www.opersys.com
karym@opersys.com

Abstract

The Adaptive Domain Environment for Operating Systems (Adeos) was designed to offer the capability of sharing the hardware between multiple operating systems. This can in turn be used to create a real-time domain which has priority on all other domains. Leveraging Adeos' capabilities, the real-time operating system would share the hardware with the other execution domains but would have the advantage to be the first in the interrupt pipeline.

Architectural overview[1]

Once Adeos is running on an operating system, it becomes easy to obtain deterministic real-time behavior from the underlying hardware. On the Linux/ix86 host, this amounts to creating a kernel module that upon loading interacts with Adeos to obtain priority status in the interrupt pipe. Once that is done, it can provide all the deterministic facilities necessary for the operation of real-time tasks.

Since it is capable of recognizing and interacting with Adeos, it should not interfere with Adeos' normal behavior and call any system-modifying instructions. This includes the ix86 cli/sti couple. Rather, it should use the stall/unstall primitives provided by Adeos. This will guaranty that Adeos still has a certain degree of control over all that is happening in the system.

It is possible to have a system where Adeos may have the following 4 stages in its pipeline, starting from stage 0 which is the hardware's interrupt issue:

1. Real-Time OS
2. Kernel Debugger
3. Low-latency driver
4. Linux kernel

The same thing may be viewed for any BSD derivative:

1. Real-Time OS
2. Kernel Debugger
3. Low-latency driver
4. BSD Kernel

To offer its services to the real-time tasks, the RTOS may either provide a software- interrupt based facility or enable modules to directly call its services either through call gates or direct calling.

Again this real-time operating system may be built as a kernel module. In future versions of Adeos, it may be foreseeable to enable user-level processes to become domains of their own and have direct access to the hardware as other domains do.

[1] As this is only an architectural brief, please refer to the document entitled "Adaptive Domain Environment for Operating System" by Karim Yaghmour for further details about Adeos.