Controller Implementation using RT-Linux

Authors: John Lopes & Nader Sabet

Thesis Outline

RT-Linux is a freely available real-time operating system which provides the necessary facilities to implement a real-time control system.

Traditionally, education of real-time operating systems is based on a high-level perspective that ignores critical implementation details.

The objective of this thesis is to investigate real-time controller implementation issues and how RT-Linux can be utilised in an educational environment.

A non-minimal state space representation of a DC motor and the LQR method was used to design a motor speed controller. This enabled the assessment of RT-Linux for the implementation of a non-trivial real-time controller.

Two RT-Linux implementations based on different IPC mechanisms were explored. The first utilised RT-FIFO's, while the second used shared memory.

In addition, SCILAB, a scientific software package was integrated into the system to provide data analysis and interpretation.

3. How can RT-Linux be used for education?

Since RT-Linux is open-source, students may obtain full access to the source code. They may examine and learn from it and even design and implement their own real-time control systems.

1. How does RT-Linux work?

RT-Linux is an extension to the Linux kernel that makes the normal Linux kernel fully preemptable. All real-time tasks which run on a system are managed by the RT-Linux kernel and scheduler. Linux is run as the lowest priority task, executing only when there are no other real-time tasks to be executed.

2. What does RT-Linux have to offer?

- Task priority allocation
- IPC and user defined IPC handlers
- User defined schedulers
- User defined real-time tasks

In addition, RT-Linux promotes a modular software development environment since it is based on Linux’s loadable kernel modules.

Shared Memory Implementation

Shared memory is another mechanism of IPC. It’s major advantage is that it is not restricted to point-to-point communication. Shared memory can be used concurrently by any number of user or kernel processes.

SCILAB was integrated into the system as a user process, reading and writing to shared memory. In this way, students can sample real-time data and perform direct analysis.