Appendix K

Real time control system

K.1 Introduction

All hardware control experiments described in this thesis were performed using the RTC system developed by the author. RTC was written in C/C++ and runs under the Linux operating system. Its structure is shown in figure K.1. It consists of a number of Linux kernel modules, kernel modifications, and an X-windows supervisor application. The kernel modules perform the control tasks: A/D sampling, computing the control action and D/A output. The supervisor application graphs A/D sensor values and allows the user to adjust controller parameters online. The interface to the external world is through a "Keithley Data Acquisition and Control" (KDAC) model 575 A/D and D/A system.

Developing kernel modules can be difficult, and using them is inconvenient, but they are able to achieve much more accurate controller scheduling, and also direct access to memory and the I/O space is easier. The controller sampling function is interrupt driven, it is not a standard Linux process because they are unable to achieve reliable real-time scheduling.

Four controller modules were developed for driving different hardware. They are: `crane` which drives a model gantry crane, `invp` which balances an inverted pendulum on a cart, `robot` which steers a toy robot along a track, and `thermal` which controls a model heat-transfer process. The module `invp` is shown in figure K.1.

K.2 Supervisor application

The supervisor application, shown in figure K.2, runs under X-Windows. It communicates with the RTC kernel modules and allows the user to

- Start/stop the controller at a given sampling frequency.
- Display a moving graph of selected sensor values at a user specified magnification.
- Modify controller parameters (such as linear controller gains or FOX learning rates) with sliders, or load them from files.
- Load, save and reset FOX weight tables.
- Start and stop FOX training.

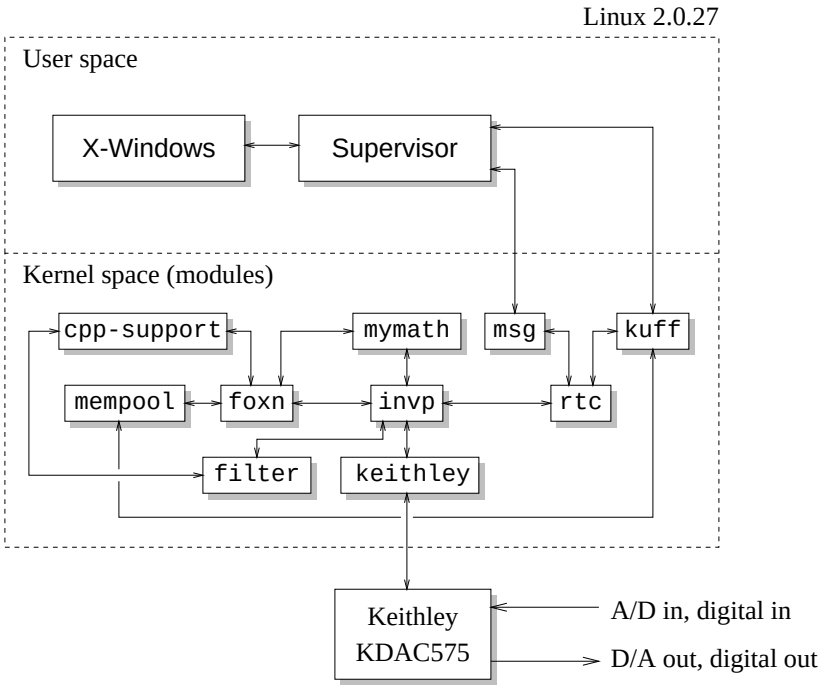

Figure K.1: *The real-time control (RTC) system.*

- Log sensor data to files.
- Select the controller mode (e.g. linear PD or FOX-in-the-loop).

K.3 Kernel modification: `mempool`

`mempool` is a patch to the 2.0.27 Linux kernel which allows a large amount of physical memory to be reserved at boot-time. Pages in this memory pool can be allocated and deallocated by other drivers and kernel modules. It is useful for getting around the size limitation of `kmalloc()`. Also, this memory is safe to `remap_page_range()` into user space, e.g., for `mmap()`. This patch is based on the `bigphysarea` patch by M. Welsh (`mdw@cs.cornell.edu`).

RTC uses `mempool` to allocate large amounts of weight storage and buffers for the FOX algorithm. It is also used by the `kuff` module for allocating buffers that can be shared with users-space processes. To use it, specify the boot option

```
mempool=<number of pages>
```

to specify the number of 4k pages to reserve. If this option is not used then no pages will be reserved. Note that at least one extra page will be reserved for use by a page mapping table. This also adds a file into the `/proc` filesystem called `mempool`, which contains some status information about the memory pool, and also contains a representation of the page map table. The following kernel files are added/modified by the patch:

```
/usr/src/linux/include/linux/mempool.h
/usr/src/linux/mm/mempool.c
```

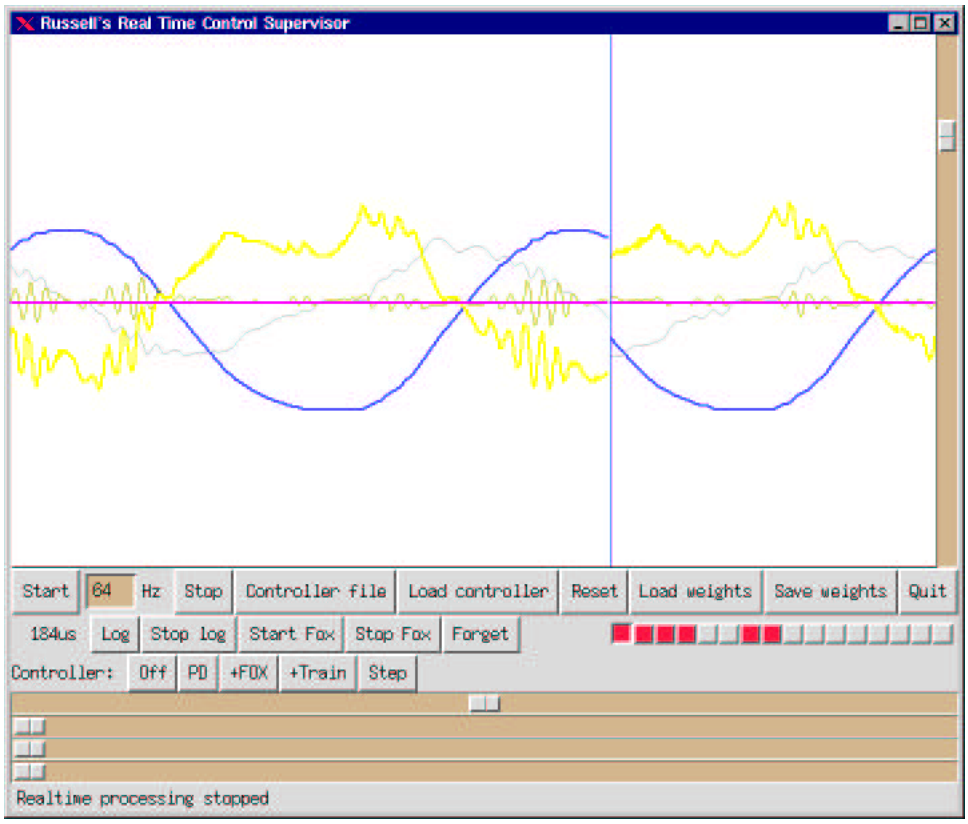

Figure K.2: *The RTC supervisor application.*

K.4 Kernel modules

`invp`, `crane`, `robot`, `thermal` — These modules implement the actual controller algorithms.

`rtc` — The real time control core. This allows a control function to be registered by another kernel module. Communications channels are provided between this module and a user space process (USP). The USP can command the module to start/stop calling the control function at some specified sampling frequency, and it can set controller parameters. Data returned by the controller function is packaged and returned to the USP, along with timing statistics. The PC real-time clock interrupt is used to invoke the control function. The interrupt rate can be set to powers of 2 between 2 Hz and 8192 Hz.

`foxn` — This provides a C++ class called `FoxN` that implements the FOX algorithm.

`keithley` — This provides an interface to the "Keithley Data Acquisition and Control" (KDAC) model 575 A/D and D/A system. Functions are provided to configure/read/write to the A/D, D/A, and digital I/O channels.

`kuff` — This is the **k**ernel-**u**ser **f**ast **f**ifo channel. A fast fifo is a device for one-way communication between the kernel and a user-space process. The fifo is "fast" because the user process gets to manipulate the fifo buffer memory directly. Every `kuff` device has a corresponding file `/dev/kuff[1-4]`. The user space process can not `read()` and `write()` data directly from/to these files. Instead a 4-byte structure is `read()` which gives a length value, which is used to `mmap()` the fifo buffer into the user address space. Note that `select()` can also be called on the device.

`msg` — The `msg` device is a very simple two-way message communications path between the kernel and a user-space process (USP). From USP to kernel: The kernel can set handlers on the `/dev/msg` devices. When the USP `write()`s data to a `/dev/msg` it is sent directly to the handler (if any). From kernel to USP: The kernel can set a message buffer to be read by the USP. When the USP `read()`s data from a `/dev/msg` device, the message buffer is returned (if any). Reads and writes will never block, and will always be atomic.

`cpp-support` — This provides low level support for C++ object files in the kernel. It provides the built-in `new` and `delete` functions.

`filter` — This provides C++ FIR and IIR filter classes for filtering A/D data. Pre-initialized 3rd and 5th order Butterworth filters are provided (with cutoffs of 0.05 and 0.15 times the sampling frequency, respectively).

`mymath` This module provided the following math library functions (which are otherwise unavailable to the kernel): `sin`, `cos`, `exp`, `log`, `ceil`, `rint`, `fabs`, `pow`.