Design and Implementation of Linux Drivers for National Instruments IEEE 1588 Timing and General I/O Cards

Kevin Meyer, Klemen Vodopivec (Cosylab, Ljubljana, Slovenia),
Rok Šabjan, Klemen Žagar (COBIK, Solkan, Slovenia)

Abstract

Cosylab is developing Linux device drivers to support several National Instruments (NI) devices. In particular, drivers have already been developed for the NI PCI-1588, PXI-668 2 (IEEE1588/PTP) devices and the NI PXI-6259 I/O device. These drivers are being used in the development of the latest plasma fusion research reactor, ITER, being built at the Cadarache facility in France. In this paper we discuss design and implementation issues, such as driver API design (device file per device versus device file per functional unit), PCI device enumeration, handling reset, etc. We also present various use-cases demonstrating the capabilities and real-world applications of these drivers.

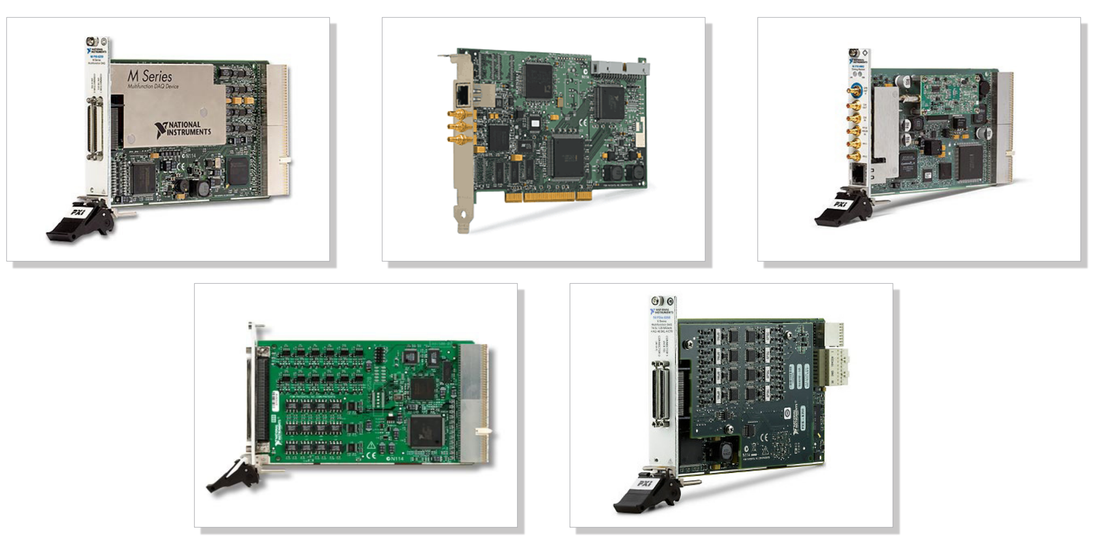
The National Instruments (NI) hardware

ITER selected the NI hardware for its:

- versatility
- high manufacturing quality
- important features such as triggering signals on the backplane
- cost effective

Supported hardware:

- IEEE 1588 Precision Time Protocol (PTP): NI PXI-6682, PCI-1588
- Multifunction input/output (I/O) board: NI PXI-6259, PXI-6528, PXIe-6368



Software

- NI device drivers are targeted for use within LabVIEW (Windows, VxWorks, Pharlap ETS) environment.
- ITER has also adopted Red Hat Enterprise Linux (RHEL) as the operating system (OS).
- Cosylab has developed kernel modules and the user-space libraries for Linux.

API

Communication between user-space library and kernel module via device files (`/dev`)

- One device file per I/O channel
- **Advantage:** granular access to channels (e.g., multiple processes can access the same I/O board's channels)
- **Disadvantage:** performance issues when reading data from multiple channels.
  - **Possible improvement:** implement memory map API for high-performance data transfers.

PCI device enumeration

How to know which physical board corresponds to which device file?

Very important e.g., when adding a board, the device name assignment of already installed devices shouldn't change.

Challenge: physical slot order depends on type of chassis, and is difficult to infer.

Solutions:

- module initialization parameters:
  - serial numbers are given as parameters whose order determines device number assignment
- `udev` mechanism:
  - each board provides serial number information via the Linux `sysfs` file system
  - udev uses this information to determine the device name.
  - Example udev configuration that assigns device with serial 161382B to device file `pxi6259.1`:

```
SUBSYSTEM==”pxi6259”, SYSFS{serial}==”161382B”, NAME=”pxi6259.1%s{suffix}”
```

Code example

Example code using NI PXI-6259 user-mode library to acquire samples from an input channel.

```
// open AI file descriptor
devFD = open(filename, O_RDWR);

// initialise AI configuration
aiConfig = pxi6259_create_ai_conf();

// configure AI channels
pxi6259_add_ai_channel(&aiConfig, channels[i], AI_POLARITY_BIPOLAR, 1, AI_CHANNEL_TYPE_RSE, 0)

// configure number of samples
pxi6259_set_ai_number_of_samples(&aiConfig, nSamples, 0, 0)

// configure AI convert clock
pxi6259_set_ai_convert_clk(&aiConfig, nChannels == 1 ? 16 : 20, 3, AI_CONVERT_SELECT_SI2TC,
                AI_CONVERT_POLARITY_RISING_EDGE)

// configure AI sampling clock
pxi6259_set_ai_sample_clk(&aiConfig, nChannels == 1 ? 16 : 20, 3, AI_SAMPLE_SELECT_SI_TC,
                AI_SAMPLE_POLARITY_ACTIVE_HIGH_OR_RISING_EDGE)

// load AI configuration and let it apply
pxi6259_load_ai_conf(devFD, &aiConfig)

// open file descriptor for AI channel
channelFDs[i] = open(filename, O_RDWR | O_NONBLOCK);

// start AI segment (data acquisition)
pxi6259_start_ai(devFD)

   while (n < nSamples) {
        // read scaled samples
        n += pxi6259_read_ai(channelFDs[i], &buffer[n], nSamples - n);
    }

// stop AI segment
pxi6259_stop_ai(devFD)

// Close channel and device files
close(channelFDs[i]);
close(devFD);
```

Summary

Linux drivers and respective user-mode libraries were developed.

They are available for download via the ITER CODAC Core System public release.

Acknowledgement

COBIK "The Centre of Excellence for Biosensors, Instrumentation and Process Control" - COBIK, Velika pot 22, SI-5250 Solkan, Slovenia

The Centre of Excellence for Biosensors, Instrumentation and Process Control - COBIK is an operation financed by the European Union, European Regional Development Fund and the Republic of Slovenia, Ministry of Higher Education, Science and Technology.

