Measurements with SPECTRA

Th. Kracht

Hamburger Synchrotronstrahlungslabor HASYLAB at DESY,
Notkestr. 85, 22607 Hamburg, Germany

Abstract: The program package SPECTRA is used at HASYLAB beamlines for experimental control and data acquisition. It runs on various hardware platforms and operating systems providing scientists with a standard interface to the experiments.

1 Introduction

SPECTRA is used for a large variety of measurements at the synchrotron radiation laboratory HASYLAB, including X-ray absorption spectroscopy, diffraction, magnetic scattering, powder diffraction, small angle x-ray scattering, standing waves, VUV spectroscopy, etc.
The program controls the beamline components, the monochromator and the diffractometer. It performs data acquisition and monitors the measurements.
SPECTRA runs on OpenVMS (Alpha and VAX) and LynxOS computers. It communicates with the hardware through several bus systems: CAMAC, VME, IEEE-488 and RS232.
The high level user interface of SPECTRA offers features like: Energy scans, q scans, motor scans, adjustment scans, automatic centering procedures, etc. In addition there is an interpreter that accepts input from a command line or command files. The user interface is independent of the operating system and hardware, with the exception of a few commands.
This is achieved by the device library concept: The functionality of the electronic components is separated from the details of the hardware.
New modules are introduced to the software by inserting them into the device list which contains all relevant information about the hardware. The device list is automatically maintained by SPECTRA.
It is possible to run SPECTRA in a simulation mode on those nodes that do not serve as experiment computers.
This note lists only the main issues. The full documentation can be found on a web page [1].

2 The Software Concept

SPECTRA is mostly run from menus, the command line or command files. But it is also available as a program library. User code may invoke operations in a way which

is very similar to the command line interface. The function gra_command() receives a string which is build according to the command line syntax. It executes the command and returns the completion status. In addition there are query functions which return information from SPECTRA. On the other hand SPECTRA may invoke user code which has been written in a common language like C. This is accomplished by a standard interface to the user-functions.
If the on-line control program runs on a separate host, SPECTRA may be started in a server mode. In this mode SPECTRA accepts commands, executes them and replies with 'DONE', 'ERROR' or a value which has been queried. Any other output is suppressed.
These are the important features of SPECTRA:

- Monochromator control: The adjustment of the setup, including mirrors, is supported by full screen menus. The items that can be accessed through these menus depend on the specific beamline. There is a clear separation between the beamline specific part of the software and those features which are the same for all supported beamlines, like energy scans.

- Interface to the device libraries HCLIB[2]/HVLIB: A device, like a counter, is introduced to SPECTRA by supplying the module type and other information which is needed to access it in a CAMAC or VME crate. A counter has to be cleared, gated and read-out no matter what module type it is or whether it is a CAMAC or VME module. The details are known to the drivers in the device library but they are hidden from the user. All functions of the device libraries are available from the command line of SPECTRA.

- Stepper motor control: Stepping motors can be controlled with the help of a full screen menu. The user may change the motors parameters or perform movements.

- Device menus: There are beamline components which consist of several motors, like absorbers or monochromators. SPECTRA provides menus that allow the user to access a set of motors on one screen.

- Interface to the GPIB bus: SPECTRA has functions that allow the user to exchange data with devices that are connected though the GPIB bus.

- Scan menus: Any scan could be performed by executing a SPECTRA command file. However it is more convenient to use menus to supply the scan parameters, like the scan range, the step width, sample time, etc. These parameters can be entered manually or by qualifiers from the command line.

  Figure 1 gives an example. It shows the on-line display of an EXAFS scan. Four counters have been enabled. Their values are displayed in the upper half of the screen. The absorption of the sample and a reference signal is displayed in the lower part.

- Diffractometer software: Software has been produced that controls four-circle diffractometers.

  The components are:

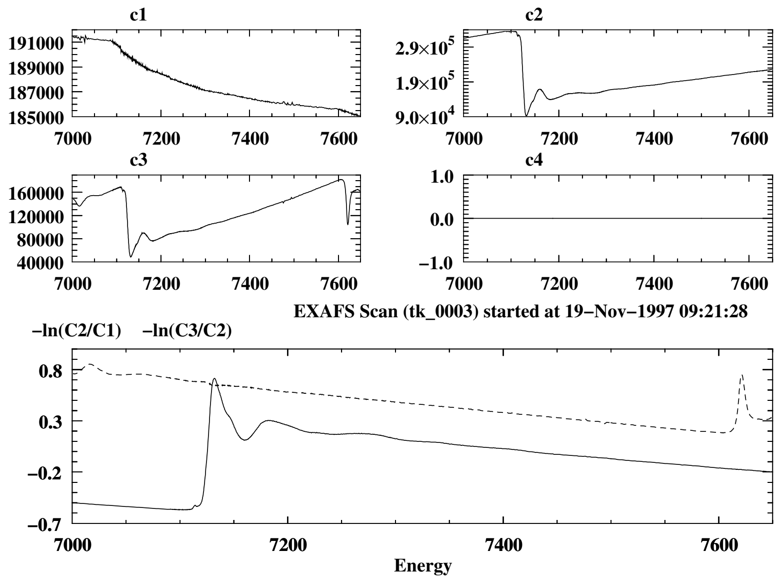

Figure 1: The on-line display of EXAFS scans

- Adjustment of the diffractometer: A full screen menu displays the relevant parameters and allows the user to move single motors. After a successful adjustment it is possible to move the instrument by changing the energy or wavelength and the four diffractometer angles.
- Crystal orientation: There is a menu that allows the centering of a reflection.
- Movements in Q-space: When the adjustment and orientation phase is finished, the instrument can be moved in terms of Qx, Qy and Qz. Q-scans are also supported.

3 The Computer System

Those beamlines of HASYLAB which are operated by DESY use two different computer systems. A part of the experiments is controlled by Q-Bus Microvaxes and CAMAC crates. New experiments are equipped with VME components, including a single board computer (SBC) which runs under the real-time operating system LynxOS. It has been chosen for several reasons:

- Lynx-OS is compatible to UNIX, an operating operating system that has become a standard in the physics community.
- Lynx-OS is a real-time operating system. The most important real-time feature is the preemptive interrupt handling by the kernel which is not a standard UNIX feature.

- GNU products, like gcc, bash, gdbx have already been ported to Lynx-OS making program development very efficient.
- X11 is available. It would be a great advantage to have a high level graphics package like GKS on the SBC, but it should be possible to port the SPECTRA graphics to X11.

SPECTRA runs on the SBC. This allows the program to take advantage of the VME I/O bandwidth, which is more than one mega-word/second.
A Linux PC acts as a file and a boot server for the SBC. Both computers are connected through an Ethernet. The SBC imports the file system by NFS. SPECTRA uses the PC as an X-terminal for the on-line display. Since the PC has a direct access to the file system of the SBC, it is able to export the data without interrupting the measurement process. Data can be transferred by ftp, rcp or copied to a local tape.
The SBC runs in a disk-less mode in order to minimize the maintenance effort. As a consequence file I/O has to be done through the network. The on-board memory serves as a data buffer. It is expected to be sufficiently large to avoid any bottleneck that is due to networking load. However it is possible to install one or more local SCSI disks or tapes, if it becomes necessary.

4 Graphics

One of the goals of SPECTRA is to ensure that the graphical output is identical on every supported platform. As a consequence it uses own the graphics drivers. These are some of the features:

- The recommended drivers are X11 and Postscript. Special care has been taken to ensure that the Postscript output looks the same as the X11 output. In fact also TEK4014 and REGIS drivers are included, but these have very limited capabilities. It is not recommended to use them.
- Fonts are selected using the X11 syntax. So far SPECTRA offers fonts of 17 different families, those fonts that appear in X11 and in Postscript.
- The user may enter special text strings by using the Latex syntax: Umlaute (ä, ö, etc.), special characters, indices etc. SPECTRA supports only a subset of Latex commands. It can be extended, if it becomes necessary.

  The Latex syntax has been selected for describing texts because many users are already accustomed to it.

References

1 http://www.desy.de/hasylab/computing/computing_hasylab.html.

2 The HCLIB was coded by W. Drube, HASYLAB