IRAM-COMP-032

Revision: 1
2007-03-06

Contact Author

Institut de RadioAstronomie Millimétrique

Vme Universe Driver

Owner Sebastien Blanchet

Keywords: vme, universe

Approved by: A.Perrigouard	Date: Dec 2005	Signature:

Change Record

REVISION	DATE	AUTHOR	SECTION/PAGE AFFECTED	REMARKS
1	2007-03-06	Blanchet S.	Command line utilities	

Content

1 Introduction

In November 2005, IRAM has begun to install a new receiver generation for the Plateau de Bure Interferometer. These new receivers need obviously new control programs and a new documentation. This document is the documentation reference for all cabin software procedure: installation, technical documentation, daily usage and troubleshooting.

2 Requirements

The software runs on a diskless full-PC VME single board computer in each antenna cabin.

2.1 Network environment

Since the cabin computers are diskless, a server is required to export the filesystem via NFS

Actual situation:

- at Grenoble, pctcp101 is the NFS host server and netsrv1 is the DHCP/PXE server
- at Bure, bure5 is the NFS server, and bure2 is the DHCP/PXE server

For a full description of the installation see the document "*PdB New Generation Antenna Mount Software*" by Alain Perrigouard.

2.2 Hardware requirement

To run the control software, you need:

- VMIVME- 7700 with a Tews TPMC816 PMC card
- 22GHz VME board
- subreflector VME board

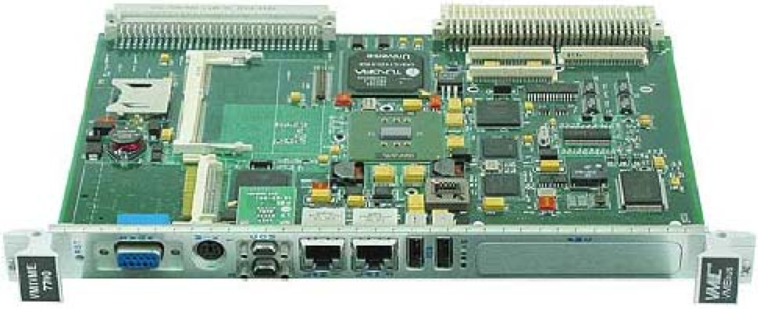

Figure 1: VMIVME-7700 from GEFanuc Automation

Though, if only a software subset is interesting you, you can:

- Use a standard PC running Linux Fedora Core 3 and run the receiver simulator. I develop quite all the software in such a way.
- Since VMIVME-7700 is a full x86-PC, if the VMEbus is not needed. It is possible to use a desktop PC with a PMC carrier board to replace the VMIVME- 7700

2.3 Software requirements

The software has been developed on Linux Fedora Core 4.

- Gcc
- Kernel headers

3 General instructions

3.1 Installation

Get the sources:

```
$ export CVSROOT=:pserver:blanchet@netsrv1.iram.fr:/CVS/PdB
$ mkdir build
$ cd build
$ cvs login
Logging in to :pserver:blanchet@netsrv1.iram.fr:2401/CVS/PdB
CVS password:
$ cvs co LINUX /drivers
```

Install the drivers:
Warning: for the following section, you need the write privilege on the whole filesystem. Therefore it will fail, it the NFS root is exported in read-only mode.

```
$ cd LINUX/drivers/tpmc816
$ make
$ su –c ‘make install’

$  cd  ../vme/vmiwdtf ; make
$ su –c ‘make install’

$ cd ../vme_universe ; make
$ su –c ‘make install’
```

Install the controlling software:

```
$ cd ~build/LINUX/cabin
$ su –c ‘make install_data’
```

4 Drivers

4.1 VME Driver

Vme_universe is the driver to access to the VMEbus. It is a modified version of the original VMIC driver to be compatible with Linux kernel 2.6.x
See the documentation in ~/build/LINUX/drivers/vme/vme_universe/doc

To load the driver:

```
DEVDIRVME=/dev/bus/vme
modprobe vme_universe
mkdir -p $DEVDIRVME
mknod --mode=666 $DEVDIRVME/ctl c 221 8
```

In *dmesg*, you can see the following lines:

```
VME: Board is system controller
VME: Driver compiled for UP system
VME: Installed VME Universe module version: 3.3
```

4.2 Watchdog Driver

The watchdog driver is the original VMIC driver. See the HTML documentation in ~build/*LINUX/drivers/vme/vmiwdtf/doc/*

Load the device driver in the running kernel

```
modprobe vmiwdtf
mknod --mode=666 /dev/watchdog c 10 130
```

In *dmesg* you can see the following line:

```
vmiwdtf: Installed VMIC watchdog timer module version: 1.1
```

4.3 Virtual VME

There is a virtual vme driver: vme_virtual_rtai.
Unlike CAN, it is more difficult to develop VME program with only a simulator.

5 Command line utilities

Several command line tools are provided with the vme_universe driver. They are automatically installed during the installation procedure.

Name	Description
vme_acquire_bus	Acquires and holds the VMEbus
vme_catch_interrupt	Sets up an interrupt handler to retrieve VMEbus interrupts
vme_dma_read	Reads values from the VMEbus using DMA transfers
vme_dma_write	Writes hexadecimal values to a VMEbus address using DMA transfers
vme_endian	Displays or changes configuration of the hardware endian conversion feature
vme_generate_interrupt	Generates interrupts to the VMEbus
vme_peek	Reads values from the VMEbus using memory-mapped registers
vme_poke	Writes hexadecimal values to the VMEbus using memory-mapped registers

vme_release_bus	Releases an acquired and held VMEbus
vme_rmw	Performs a VMEbus read/modify/write cycle
vme_slave_peek	Reads data from local slave RAM using memory-mapped registers
vme_slave_poke	Writes hexadecimal values to local slave RAM using memory-mapped registers
vme_sysreset	Asserts a VMEbus sysreset

For a complete description, use the *man* command.
Example: *man vme_peek*