ICS ELECTRONICS
division of Systems West Inc.

Better Solutions for GPIB and VXIbus Systems

VXIbus APPLICATION BULLETIN

Plug&Play COMPATIBILITY WITH VISA VTL 3 DRIVERS

INTRODUCTION

VXI is a multivendor industry standard for card-modular instrumentation based on the IEEE-1155 specification. Today there are over 800 VXI products ranging from modular instruments, chassis, wiring products, to test applications software. As the number of VXI products and vendors grows, so does the task of the end user who must select the components for his system. Successfully integrating a multivendor VXI system requires more than hardware products that are electrically and mechanically compatible. The software products that control the hardware must also work together to have a successful system.

Because system-level issues such as software is not covered by the baseline VXI specifications and the IEEE-1155 Standard, the task of integrating modules from different vendors in a system and writing the controlling software is often a difficult and arduous task. The VXIplug&play Systems Alliance is a group of member companies that are working together to define and implement standardized practices that simplifies the integration of multivendor VXI systems. The VXIplug&play Alliance has created a standard system framework concept that allows test programs in any language and operating system to be able to control VXI chassis and instrument modules through Standard Instrument Drivers.

The VXIplug&play Framework identifies the operating system (OS) and applications development environment (ADE) used to generate the test software. The VXIplug&play Alliance just adopted an updated specification for a set of standard VISA Transition Library drivers that the Slot 0 Controllers and Embedded Computer vendors should adhere to assure VXIplug&play compatibility. The VISA (Virtual Instrument Software Architecture) drivers are used between the end user's test application program or general purpose test programs purchased from software vendors and the physical VXI modules or GPIB instruments. These VISA Transition Library (VTL) drivers are described in the VXIplug&play VPP-4.2 Specification Version 3.0 and are referred to as the VTL 3.0 drivers.

ICS's PLUG&PLAY EFFORTS

ICS Electronics Corporation is a VXIplug&play Member and has been aggressively working on a set of VISA Transition Library (VTL) drivers for its series of Slot 0 Controllers and Embedded Computers. ICS's Slot 0 Controllers and Embedded Computers are optimized for applications such as remote control of VXI systems, stand-alone test stand controllers or as servant controllers for controlling groups of VXI modules. In most of these applications, the user's programs are fixed and are stored on the controller's solid-state-drive or are downloaded to the controller from a host computer. ICS's Slot 0 Controllers and Embedded Computers primarily use the DOS operating system although the Embedded Computer can work with MS Windows.

ICS has just announced an initial release of its VISA Transition Library (VTL) drivers for the Version 3.0 of the VPP-4.2 Specification. ICS's initial VTL 3.0 Driver release contains updated versions of all Version 2.0 functions plus enhancements for serial control of the VXI system. The Version 2.0 functions have all been upgraded to reflect the changes and additions specified by Version 3.0. In addition, the VTL 3.0 library includes the viStatusDesc function to provide error descriptions for debugging software. Additional Version 3.0 functions will be added in future releases of the library. Programs written with the Version 3.0 library will be compatible with any future library release or with later versions of the VPP 4.2 Specification.

(over)

SUPPORTED INTERFACE MODES

ICS's initial VTL 3.0 Driver release supports the following interface modes:

VXI	Standard VTL, local VXIbus with internal program
GPIB	Standard VTL, local GPIB bus with internal program
GPIB-VXI	Standard VTL, ICS protocol with external program
SER1-VXI	ICS enhancement, ICS protocol with external program
SER2-VXI	ICS enhancement, ICS protocol with external program

In the standard VXI interface mode, the user's program is stored internally on the Controller's solid-state-drive and the Controller functions just as any other Embedded Computer would. The GPIB mode is similar to the VXI mode but the internal program drives external GPIB devices. In the GPIB-VXI mode, the Slot 0 Controller performs its Resource Manager function and then operates under control of an external computer. Data and instructions are passed between the two computers on a GPIB bus. The SER*n*-VXI mode operation is similar to the GPIB-VXI mode with the exception that inter-computer communications performed over a serial link. n refers to COM port 1 or 2 . The SER*n*-VXI modes are ICS added enhancements that simplify remote control of VXI systems and are not part of the VPP 4.2 Specification. Figures 1, 2 and 3 show the different interface modes.

SUPPORTED FUNCTIONS

The following functions are supported in this release:

Resource Management:

viOpenDefaultRM()	Open Default RM Session
viGetDefaultRM()	Get Default RM Session
viFindRsrc()	Instrument Find Resource
viFindNext()	Instrument Find Next
viOpen()	Instrument Open
viClose()	Instrument Close

Characteristic Control Services:

viSetAttribute()	Instrument Set Attribute
viGetAttribute()	Instrument Get Attribute
viStatusDesc()	Get Status Description

Basic I/O Services:

viRead()	Instrument Read Message
viWrite()	Instrument Write Message
viAssertTrigger()	Instrument Trigger
viReadSTB()	Instrument Read Status Byte
viClear()	Instrument Clear

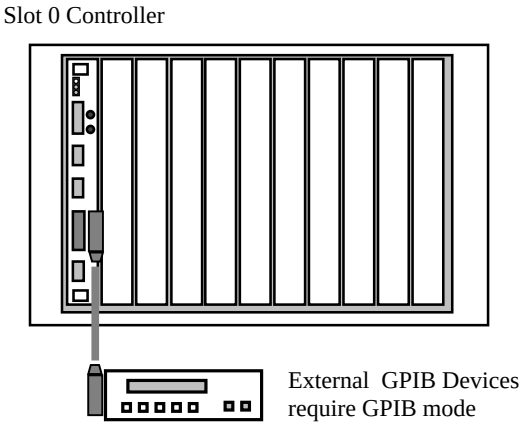

Figure 1 VXI Mode - VXI Chassis Controlled by an internal Embedded Computer.

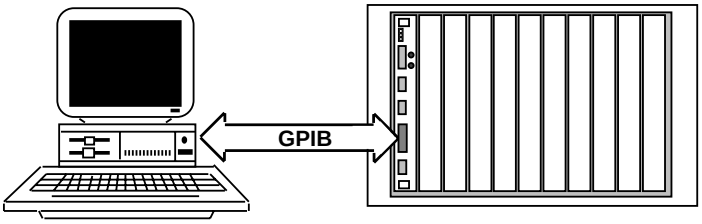

Figure 2 GPIB-VXI Mode - VXI Chassis controlled by an external computer over a GPIB bus.

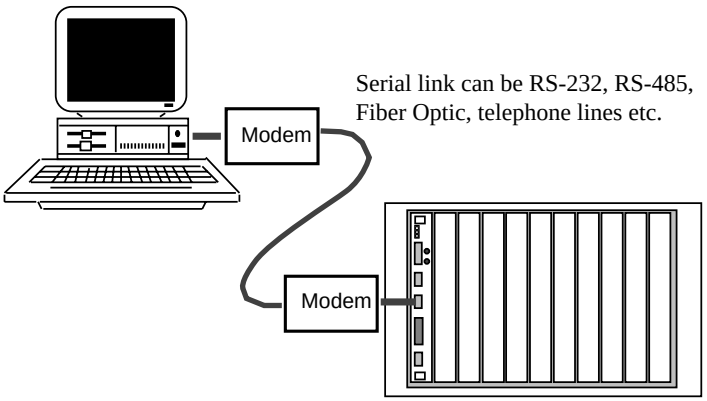

Figure 3 Serial-VXI Mode - VXI Chassis controlled by an external computer over a serial link.

SUPPORTED FUNCTIONS cont'd

Memory I/O Services:

viIn16()	Instrument Input 16 bit
viOut16()	Instrument Output 16 bit
viMapAddress()	Instrument Map Address
viUnmapAddress()	Instrument Unmap Address
viPeek16()	Instrument Peek 16 bit
viPoke16()	Instrument Poke 16 bit

SYSTEM SETUP AND OPERATION

The VXI modules may be setup with either fixed logical addresses or with dynamic addressing. If fixed addressing is used, care must be taken not to duplicate addresses in the system. With dynamic addressing, the Resource Manager function sets the module's logical address. With ICS Slot 0 Controllers, a parameter in the configuration file on the solid-state-drive specifies the first address to be used in that chassis for assigning dynamic addresses.

At power turn on or when the system is reset, each Slot Zero Controller runs a Resource Manager which resets and initializes the modules in its chassis and saves the Resource Manager table in a file on the sold-state-drive.

PROGRAMMING WITH VISA

The VISA library provides functions to find and open modules for use. When the Find function is called, a parameter is passed to the function specifying the interface such as VXI, GPIB or SER1-VXI. The Find function returns a string specifying the location and address of the module which is passed to the Open function. The Open function returns a "handle" which is used to communicate with and control the module or device. An instrument must be opened before it can be controlled. Except for the code to find and open the modules, the body of the program is be written with out regard to the location of the modules.

VTL PROGRAMMING NOTES

A. All VTL functions return strings by copying them to the pointer location specified.
B. The *viRead*() function returns 3 SUCCESS codes (zero & positive values). Error codes are negative values.
C. All programs should begin with a call to *viOpenDefaultRM*() and end with a call to *viClose*() using the session value returned.

GPIB-VXI MODE OPERATION

The ICS Electronics GPIB-VXI interface mode requires the program GP_CNTL.EXE to be running on the VXI-5543 connected to the main controller via the GPIB bus. This is a transparent program that executes the commands as they are received. The user can either download programs for the controller to run or use it in a transparent mode. The GPIB address of the VXIbus chassis is specified in the VXI5543.INI file on the solid-state-drive. If secondary addressing is not used (primary only) it should be set to -1. The GPIB address may be set using the EDIT_INI.EXE program. Refer to the VXI-5543 manual for more information on this procedure.

ICS ENHANCEMENTS FOR SERIAL CONTROL

ICS Electronics has added enhancements which are not part of the VISA-2 specification to provide VXIbus control via serial data links. The operation is the same as for the GPIB-VXI mode except that the communication is via a serial port instead of via GPIB. The functions affected are *viOpen*(), *viFindRsrc*() and *viFindNext*(). Two search strings; SER1-VXI and SER2-VXI have been added. Their format is the same as for GPIB-VXI. The returned identifying strings also reflect the same additions. SER1 uses the local COM1 port and SER2 uses the local COM2 port. The program SER_CNTL.EXE must be running on the connected VXI-5543 for proper operation. The communication parameters such as baud rate are specified in the VXI5543.INI file in both units and must be set to the same values. The serial parameters may be set using the EDIT_INI.EXE program. Refer to the VXI-5543 manual for more information on this procedure.

Example:

```
status = viFindRsrc( sesn, “SER1-VXI?*INSTR”,
&vi, &retCnt, instrDesc);
```

The returned string instrDesc will contain “SER1-VXI” and can be passed to *viFindNext*() and *viOpen*(). Refer to the VPP-4.2 specification version 3.0 for more information.

FUTURE FUNCTIONS

Future releases of ICS's VTL 3.0 Library will add support for the FDC and HS488 modes plus the remaining VTL 3.0 functions. FDC stands for the Fast Data Channel method of passing data between a VXI Controller and a VXI module. FDC will replace shared memory in future versions of the VXI specification. HS488 stands for the fast IEEE-488 handshake.

The following VTL 3.0 functions will be added in the future releases:

Formatted I/O Services:

viSetBuf()	Set formatted I/O buffer size
viFlush()	Flush formatted I/O buffer
viPrintf()	Convert, format & send parameters
viVPrintf()	Convert, format & send parameters
viScanf()	Read, convert & format data
viVScanf()	Read, convert & format data

Event Services:

viEnableEvent()	Enable event notification
viDisableEvent()	Disable event notification

FUTURE FUNCTIONS cont'd

viDiscardEvents()	Discard events
viWaitOnEvent()	Wait for event
viInstallHandler()	Install event handler
viUninstallHandler()	Uninstall event handler
viEventHandler()	Event handler prototype

The following 8 bit VTL 3.0 functions will only be supported if there is a need to do so. So far, we have not identified any VXI instruments that use 8 bit (D8) data transfer.

8 Bit Memory I/O Services:

viIn8()	Instrument Input 8 bit
viOut8()	Instrument Output 8 bit
viPeek8()	Instrument Peek 8 bit
viPoke8()	Instrument Poke 8 bit